LOCAL NETWORK CODING ON PACKET ERASURE CHANNELS – FROM SHANNON CAPACITY TO STABILITY REGION

A Preliminary Report

Submitted to the Faculty

of

Purdue University

by

Wei-Cheng Kuo

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

April 2013

Purdue University

West Lafayette, Indiana

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABSTRACT

Kuo, Wei-Cheng Ph.D., Purdue University, April 2013. Local Network Coding on Packet Erasure Channels – From Shannon Capacity to Stability Region. Major Professor: Chih-Chun Wang Professor.

Network Coding (NC) has emerged as a ubiquitous technique of communication networks and has extensive applications in both practical implementations and theoretical developments. While the Avalanche P2P file system from Microsoft, the MORE routing protocol, and the COPE coding architecture from MIT have implemented the idea of NC and exhibited promising performance improvements, a significant part of the success of NC stems from the continuing theoretic development of NC capacity, e.g., the Shannon capacity results for the single-flow multi-cast network and the packet erasure broadcast channel with feedback. However, characterizing the capacity for the practical wireless multi-flow network setting remains a challenging topic in NC. The difficulties of finding the optimal NC strategy over multiple flows hinder the further advancement in this area. Despite the difficulty of characterizing the full capacity for large networks, there are evidences showing that even when using only local operations, NC can still recover substantial NC gain. We believe that a deeper understanding of multi-flow local network coding will play a key role in designing the next-generation high-throughput coding-based wireless network architecture.

This thesis consists of two parts. In the first part, we characterize the full Shannon capacity region of the "*COPE*" principle when applied to a 2-flow wireless butterfly network with broadcast packet erasure channels. The capacity results allow for random overhearing probabilities, arbitrary scheduling policies, network-wide channel state information (CSI) feedback after each transmission, and potential use of non-linear network codes. We propose a theoretical outer bound and a new class of linear network codes, named the

Space-Based Linear Network Coding (SBLNC), that achieves the capacity outer bound. Numerical experiments show that SBLNC provides close-to-optimal throughput even in the scenario with opportunistic routing.

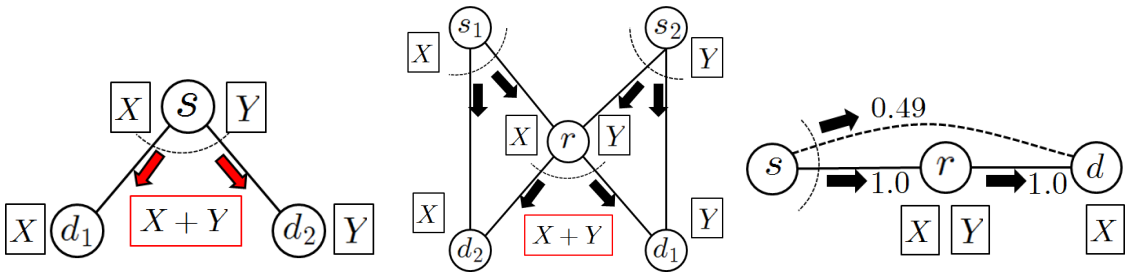
In the second part, we further consider the complete dynamics of stochastic arrivals and queueing and study the corresponding stability region. Based on dynamic packet arrivals, the resulting solution would be one step closer to practical implementation, when compared to the previous block-code-based capacity study. However, the nature of combining packets in NC is not well-defined in existing store-and-forward stability analysis, which provides further challenges for online coding and scheduling implementation. Focusing on broadcast packet erasure channels, an essential component of the wireless butterfly network, we modify the network control algorithm Deficit Maximum Weight (DMW) algorithm and successfully incorporate the nature of combining packets in NC. With the assists of both modified DMW and SBLNC, we characterize the full stability region of 2-use broadcast packet erasure channel with feedback.

For the future work, we plan to extend our current results on 2-receiver multi-input broadcast packet erasure channel to (1) the stability region of the Markovian controlled 2-receiver multi-input broadcast packet erasure channel, which includes the traditional Adaptive Coding and Modulation (ACM) scheme as a special example, and (2) the stability region of the COPE principle 2-flow wireless butterfly network with broadcast packet erasure channels. Last but not least, we also plan to devise practical protocols and testbed simulation to verify the intuition derived from our capacity and stability analysis.

1. INTRODUCTION

As the number of smartphone users growing to the majority of wireless carrier customers, the demand of wireless data rate has increased rapidly and expanded beyond the the traditional wireline service requirements. How to increase the wireless data rate supporting multiple users simultaneously with certain scarce resources of the communication network thus eventually becomes a critical and urgent topic. There are many possible solutions, e.g., ultra wide band communication and the multiple-input multiple-out antenna. Nonetheless, network coding is one of the most promising directions which could potentially provide considerable end-to-end throughput improvement and protect the data privacy of individual users.

Inspired by the butterfly network, Ahlswede *et al.* proposed the concept of network coding in 2000 [1]. Since then, network coding has emerged as a ubiquitous technique of modern data communication networks. The extensive applications of network coding spread from practical implementations to theoretical results. The Avalanche P2P file system from Microsoft removes the need of receiving all individual pieces of the original file as in the BitTorrent system. The MORE protocol from MIT alleviates the use of a scheduler to coordinate the transmission as in previous opportunistic routing protocols. The COPE architecture from MIT incorporate network coding across multiple sessions and demonstrates that the existing TCP/IP network layer transmission still has great potential to further increase the overall throughput by 40% to 200%. All the above implementations are based on the concept of network coding. Furthermore, in the areas of the network security, the data center, and the analog signal processing, network coding has exhibited great potential on augmenting their current performance. Meanwhile, a significant part of the success of network coding stems from the continuing theoretic development of network coding capacity. The seminal work [2] in 2003 utilized network coding as the backbone and prosed “random linear network coding” to achieve the Shannon capacity of single-flow (or single-session)



(a) (b) (c)

Fig. 1.1. The illustration of local network coding gain on (a) the broadcast channel; (b) the COPE principle butterfly wireless network; and (c) the opportunistic routing, where the dashed arcs represent the broadcasting nature and the rectangle represents a packet.

multi-cast networks. And the feedback capacity of the broadcast channel, the long-term open question in the area of Shannon capacity, has also been resolved by network coding for the packet erasure channel case [3, 4].

1.1 Network Coding On Local Networks

However, even though COPE [5] has exhibited the great potential for network coding being applied to multi-user (or multi-flow) wireless data networks, its corresponding theoretical capacity remains largely unsolved. The difficulties of finding the optimal network coding strategy over multiple flows hinder the further advancement in this area. Despite the difficulty of characterizing the full capacity for large networks, there are evidences showing that even when using only local operations, network coding can still recover substantial network coding gain. In the following, we are going to present three examples that can demonstrate substantial network coding gain even on the local operations.

1.1.1 Network Coding On The Broadcast Channel

The first example is the network coding gain on the broadcast packet erasure channel. Figure 1.1(a) illustrates the scenario where the network coding can benefit the throughput. As shown in Figure 1.1(a), the dashed arc represents the node s can broadcast packets to d_1 and d_2 simultaneously with certain probabilities. Assume d_1 would like to convey packet X and d_2 would like to convey packet Y. However, in the first two transmissions by r, unfortunately, d_1 receives Y and d_2 receives X. But then in the next time slot, the node s can transmit the combined packet $X + Y$ and both the destinations can recover the desired packets if it receives the packet $X + Y$. Without network coding, the node s needs to keep transmit X and Y separately until d_1 and d_2 receive the desired packet. Recently, [3] and [4] successfully characterized the full capacity region of the 1-hop broadcast packet erasure channel with ≤ 3 coexisting flows.

1.1.2 Network Coding On The Butterfly Wireless Network

The second example is the network coding gain on the COPE principle butterfly wireless network. Figure 1.1(b) illustrates its scenario and the dashed arc represent the corresponding source node can broadcast packets to the connected end nodes. Suppose source s_1 would like to send a packet X to destination d_1; source s_2 would like to send a packet Y to d_2; and they are allowed to share a common relay r. Also suppose that when s_1 (resp. s_2) sends X (resp. Y) to r, destination d_2 (resp. d_1) can overhear packet X (resp. Y). We further assume that after the first two transmissions, both d_1 and d_2 can use feedback to inform r the overhearing status at d_1 and d_2, respectively. Then instead of transmitting two packets X and Y separately, the relay node r can send the linear combination $[X+Y]$. Each destination d_i can then decode its desired packet by subtracting the overheard packet from the linear combination $[X + Y]$. In the above simple example, the traditional store-and-forward transmission scheme requires at least 4 transmissions (s_1 to r, s_2 to r, r to d_1, and r to d_2). But with the network coding in COPE scheme, it only requires 3 transmissions.

Despite its simple nature, the exact capacity region of the COPE principle remains an open problem even for the simplest case of two coexisting flows. Several attempts have since been made to quantify some suboptimal achievable rate regions of the COPE principle [6–15]. One difficulty of deriving the capacity region is due to the use of feedback in the COPE principle. It is shown in [16] that although feedback could strictly enhance the capacity in a multi-unicast environment, the exact amount of throughput improvement is hard to quantify. [17] proposes one queue-based approach for the general wireline and wireless networks considering both inter-session and intra-session network coding. However, the results in [17] mainly focus on the benefits from the side information to decide either inter-session or intra-session network coding should be applied, which is more related to [18]. [18] circumvents the difficulty of feedback-based analysis by considering a special class of 2-staged coding schemes. Although the results in [18] fully capture the benefits of message side information [16, 17, 19–23] , they capture only partially the feedback benefits, which leads again to a strictly suboptimal achievable rate region.

1.1.3 Network Coding On The Opportunistic Routing

The third example is the network coding gain on the opportunistic routing. Figure 1.1(c) illustrates its scenario and the dashed arc represent the node s can broadcast packets to both r and d with certain probabilities indicated. Here we only have one session from s to d and d would like to receive both packets X and Y. After two transmissions from s, the relay r receives both X and Y while d only receives X. Then the relay r can directly transmit the combined packet $X + Y$ and d can recover Y linear operations. Without network coding, then the opportunistic routing scheme requires a scheduler to inform r what has been received by d. And then r can transmit Y. Keeping track of which packets have been heard or not is a daunting task and network coding drastically simplify it. Recent works [24–26] take the advantage illustrated in this example to remove the need of a central scheduler and experimentally show that with network coding in a 20-node wireless testbed, the unicast

throughput can be 22% higher than the existing opportunistic routing protocols and 95% higher than the current state-of-art best routing protocol for wireless mesh networks.

1.1.4 A Critical Question

All the above schemes can augment the end-to-end throughput in the multi-flow wireless network. An interesting question thus rises: can we combine all of them together? Or can we optimize all of them simultaneously? Furthermore, can we do its analytically Shannon capacity? To answer these questions, we believe that a deeper understanding of multi-flow local network coding will play a key role, which will also benefit designing the next-generation high-throughput coding-based wireless network architecture. The analysis in this thesis is on the packet level as the COPE operating on the current TCP/IP network layer. With the ARQ mechanisms in the data link layer, the packet erasure channel setting thus is a natural choice. Hence the anlysis of Shannon capacity of 2-flow wireless butterfly network with broadcast packet erasure channel turns to our primary objective.

1.2 From Shannon Capacity To Stability Region

The analysis of Shannon capacity is an essential part to establish the possible solutions for the communication networks of interest. However, Shannon capacity is still quite far away from practical implementations. In the analysis of Shannon capacity, the results are derived based on the assumption that the input block code is fixed and the block code length can be infinitely long. This assumption is apparently impractical because of the memory buffer limit in real systems. Furthermore, this assumption would also induce the extremely large decoding delay and control overhead. These flaws deviate the analysis of Shannon capacity from the practical implementations.

Hence we further broaden our attention to the stability region of network coding schemes. The stability region of the network is defined as the set of all end-to-end traffic load that can be supported under the appropriate selection of the network control policy [27]. The analysis of stability region considers the complete dynamics of stochastic arrivals and queueing.

The assumption of dynamic arrivals greatly alleviates the flaws in the block-based Shannon capacity analysis, including the problems of memory buffer limit, decoding delay, and the large control overhead, and promotes the entire analysis one step closer to the practical implementations.

However, there are several difficulties which block the extension from Shannon capacity to the stability region. With fixed block codes as the input in the Shannon capacity, the overall throughput in the end of the transmissions can be analyzed by the law of large number. This does not hold for the case of stochastic arrivals (which means the packets are endlessly injected into the network) and other tools are required to analyze the queue dynamics at each time instance. Tassiulas *et al.* [28] introduced Lyaponov drift to resolve this problem and leaded to the establishment of the network stability analysis research.

Other than the difference between the analysis tools, however, in the existing store-and-forward stability analysis, the stability region is defined on considering all possible scheduling, routing, and resource allocation, but no coding allowed inside the network [27]. The nature of combining packets inside the network provides further challenges for online coding and scheduling implementation. Several attempts have been made to resolve this problem [17, 29]. However, the existing proposed solutions all tends to circumvent the problem of combining packets by converting the network coding scheduling problem back to the existing store-and-forward scheduling problem. This kind of conversions highly relies on case-by-case discussion and lack of the generality to be systematically applied to other network topologies. A general network control algorithm which can incorporate the nature of combining packets inside the network thus is an important subject for the network coding stability analysis.

1.3 Our Contributions

Our contributions consists of two parts. In the first part, we characterize the full Shannon capacity of the COPE principle when applied to a 2-flow wireless butterfly network with broadcast packet erasure channels. The capacity results allow for random overhear-

ing probabilities, arbitrary scheduling policies, network-wide channel state information (CSI) feedback after each transmission, and potential use of non-linear network codes. An information-theoretic outer bound is derived that takes into account the delayed CSI feedback of the underlying broadcast packet erasure channels. We then propose a new class of linear network codes, named the Space-Based Linear Network Coding (SBLNC). SBLNC provides a systematic approach to keep tracking the evolution of knowledge space at each node. We prove that SBLNC can achieve the capacity region of the 2-flow wireless butterfly network without considering the opportunistic routing. Furthermore, numerical experiments show that SBLNC provides close-to-optimal throughput even in the scenario with opportunistic routing.

In the second part of the contributions, we modify a network control algorithm and successfully accommodate the nature of combining packets in network coding, named the Modified Deficit Maximum Weight (Modified DMW) algorithm. The Modified DMW works on the stochastic processing network (SPN), which is a general version of the common queueing network. There are many similarities between SPN and the network coding scheduling problem. We apply SBLNC on the broadcast packet erasure channel and derive the equivalent SPN. With the assist of Modified DMW on the equivalent SPN, we then successfully characterize the stability region of the 2-user broadcast packet erasure channel with feedback.

1.4 Thesis Outline

In the next chapter, we formulate the local network model which incorporate the broadcast PEC with feedback, the COPE principle, and the opportunistic routing all together. The stability region problem is also formulated. In Chapter 3, we describe the central idea of this thesis – Spaced-Based Linear Network Coding. In Chapter 4, we characterize the full Shannon capacity of 2-flow wireless butterfly network with broadcast packet erasure channels. In Chapter 5, we start to discuss the linear network coding stability region and introduce its analogy the stochastic processing network. In Chapter 6, we propose the mod-

ified Deficit Maximum Weight algorithm and fully characterize the stability region of the 2-receiver multi-input broadcast packet erasure channel. In Chapter 7, we conclude this thesis and discuss the possible extensions and applications from SBLNC and modified-DMW.

2. MODEL FORMULATION

In this chapter, we will first formulate the 1-to-M broadcast packet erasure channel as a mathematical model. We then propose a general wireless butterfly model which incorporates the broadcast packet erasure channels with feedback, the COPE principle, and the opportunistic routing all together. A useful probability function which can intuitively describe the probability of interest is defined. We finally discuss the dynamic network coding and scheduling decision in the 1-to-2 broadcast packet erasure channel. We first define a useful notation. For any positive integer M, define $[M] \triangleq \{1, \cdots, M\}$.

2.1 The 1-to-M Broadcast Packet Erasure Channel

Given a finite field $\mathsf{GF}(q)$. A 1-to-M broadcast packet erasure channel (PEC) takes an input packet $X_s \in \mathsf{GF}(q)$ from the source s and outputs an M-dimension vector $\mathbf{Y} = (Y_{s\to d_1}, Y_{s\to d_2}, ..., Y_{s\to M})$, where $Y_{s\to d_i} \in \{X_s, *\}$ for all $i \in [M]$. Figure 2.1(a) illustrates

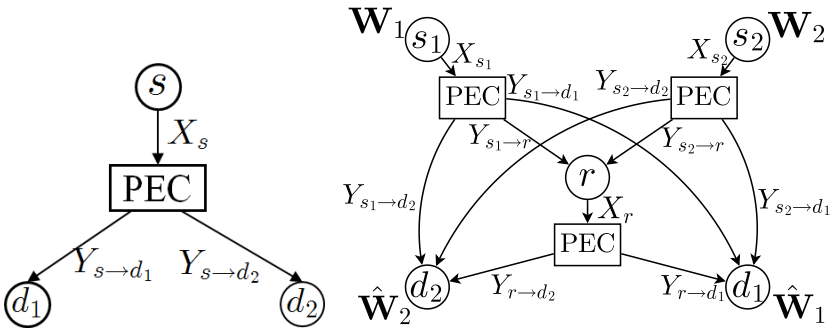


Fig. 2.1. (a) The 1-to-2 broadcast packet erasure channel; and (b) the 2-flow wireless butterfly network with opportunistic routing and packet erasure channels.

a 1-to-2 broadcast packet erasure channel. Here $*$ denotes the erasure symbol. $Y_{s\to d_i} = *$ means that the i-th receiver does not receive the input X_s. We also assume that there is no other type of noise, i.e., the received $Y_{s\to d_i}$ is either X_s or $*$.

We consider only *stationary and memoryless PECs*, i.e., the erasure pattern is independently and identically distributed (i.i.d.) for each channel usage. The characteristics of a memoryless 1-to-M PEC can be fully described by 2^M *successful reception probabilities* $p_{s\to T\overline{[M]\setminus T}}$ indexed by any subset $T \subset [M]$. That is, $p_{s\to T\overline{[M]\setminus T}}$ denotes the probability that a packet X_s sent from source s is heard by and only by the i-th destination for all $i \in T$. For example, suppose $M = 3$, then $p_{s\to\{1,3\}\overline{\{2\}}}$ denotes the probability that a packet X_s is heard by d_1 and d_3 but not by d_2.

For any time t, we use an M-dimensional *channel status vector* $\mathbf{Z}_s(t)$ to represent the channel reception status of the 1-to-M broadcast packet erasure channel:

$$\mathbf{Z}_s(t) = (Z_{s\to d_1}(t), Z_{s\to d_2}(t), ..., Z_{s\to d_m}(t)) \in \{*, 1\}^M$$

where "$*$" and "1" represent erasure and successful reception, respectively. That is, when s transmits a packet $X_s(t) \in \mathsf{GF}(q)$ in time t, the destination d_m receives $Y_{s\to d_m}(t) = X_s(t)$ if $Z_{s\to d_m}(t) = 1$ and receives $Y_{s\to d_m}(t) = *$ if $Z_{s\to d_m}(t) = *$. For simplicity, we use $Y_{s\to d_m}(t) = X_s(t) \circ Z_{s\to d_m}(t)$ as shorthand. Since here we only consider stationary and memoryless PECs, $\mathbf{Z}_s(t)$ is i.i.d. over time.

2.2 The COPE Principle 2-Flow Wireless Butterfly Network With Opportunistic Routing and Broadcast Packet Erasure Channels

Here we are going to construct a local network model which incorporates the network coding gain on (1) the COPE principle, (2) the opportunistic routing, and (3) the broadcast packet erasure channel with feedback. The COPE principle 2-flow wireless butterfly network with opportunistic routing and broadcast packet erasure channels is modeled as follows. We consider a 5-node 2-hop relay network with two source-destination pairs (s_1, d_1) and (s_2, d_2) and a common relay r interconnected by three broadcast PECs. See Fig. 2.1(b) for the illustration. Specifically, source s_i can use a 1-to-3 broadcast PEC to communicate

with $\{d_1, d_2, r\}$ for $i = 1, 2$, and relay r can use a 1-to-2 broadcast PEC to communicate with $\{d_1, d_2\}$. To accommodate the discussion of opportunistic routing, we allow s_i to directly communicate with d_i, see Fig. 2.1(b). When opportunistic routing is not permitted (as in the case when focusing exclusively on the COPE principle and the broadcast packet erasure channels), we simply choose the PEC channel success probabilities $p_{s_i \to \cdot}$ such that the probability that d_i can hear the transmission from s_i is zero.

We assume slotted transmission. Within an overall time budget of n time slots, source s_i would like to convey nR_i packets $\mathbf{W}_i \triangleq (W_{i,1}, \cdots, W_{i,nR_i})$ to destination d_i for all $i \in \{1, 2\}$ where R_i is the rate for flow i. For each $i \in \{1, 2\}$, $j \in [nR_i]$, the information packet $W_{i,j}$ is assumed to be independently and uniformly randomly distributed over $\mathsf{GF}(q)$.

For any time t, we use an 8-dimensional *channel status vector* $\mathbf{Z}(t)$ to represent the channel reception status of the entire network:

$$\begin{aligned}\mathbf{Z}(t) = (&Z_{s_1 \to d_1}(t), Z_{s_1 \to d_2}(t), Z_{s_1 \to r}(t), Z_{s_2 \to d_1}(t),\\ &Z_{s_2 \to d_2}(t), Z_{s_2 \to r}(t), Z_{r \to d_1}(t), Z_{r \to d_2}(t)) \in \{*, 1\}^8\end{aligned}$$

where "$*$" and "1" represent erasure and successful reception, respectively. That is, when s_1 transmits a packet $X_{s_1}(t) \in \mathsf{GF}(q)$ in time t, relay r receives $Y_{s_1 \to r}(t) = X_{s_1}(t)$ if $Z_{s_1 \to r}(t) = 1$ and receives $Y_{s_1 \to r}(t) = *$ if $Z_{s_1 \to r}(t) = *$. For simplicity, we use $Y_{s_1 \to r}(t) = X_{s_1}(t) \circ Z_{s_1 \to r}(t)$ as shorthand.

In this thesis, we consider the node-exclusive interference model. That is, we allow only one node to be scheduled in each time slot. The scheduling decision at time t is denoted by $\sigma(t)$, which takes value in the set $\{s_1, s_2, r\}$. For example, $\sigma(t) = s_1$ means that node s_1 is scheduled for time slot t. For convenience, when s_1 is not scheduled at time t, we simply set $Y_{s_1 \to r}(t) = *$. As a result, the scheduling decision can be incorporated into the following expression of $Y_{s_1 \to r}(t)$:

$$Y_{s_1 \to r}(t) = X_{s_1}(t) \circ Z_{s_1 \to r}(t) \circ 1_{\{\sigma(t) = s_1\}}.$$

Similar notation is used for all other received signals. For example, $Y_{r \to d_2}(t) = X_r(t) \circ Z_{r \to d_2}(t) \circ 1_{\{\sigma(t) = r\}}$ is what d_2 receives from r in time t, where $X_r(t)$ is the packet sent by r in time t.

We assume that the 3 PECs are memoryless and stationary. Namely, we allow arbitrary joint distribution for the 8 coordinates of $\mathbf{Z}(t)$ but assume that $\mathbf{Z}(t)$ is i.i.d. over the time axis t. We also assume $\mathbf{Z}(t)$ is independent of the information messages $\mathbf{W}_1$ and $\mathbf{W}_2$.

For simplicity, we use brackets $[\cdot]_1^t$ to denote the collection from time 1 to t. For example, $[\sigma, \mathbf{Z}, Y_{s_1\to d_2}]_1^t \triangleq \{\sigma(\tau), \mathbf{Z}(\tau), Y_{s_1\to d_2}(\tau) : \forall \tau \in [1,t]\}$. Also, for any $S \subseteq \{s_1, s_2, r\}$ and $T \subseteq \{r, d_1, d_2\}$, we define

$$\mathbf{Y}_{S\to T}(t) \triangleq \{Y_{s\to d}(t) : \forall s \in S, \forall d \in T\}.$$

For example, $\mathbf{Y}_{\{s_1,r\}\to\{d_1,d_2\}}(t)$ is the collection of $Y_{s_1\to d_1}(t)$, $Y_{s_1\to d_2}(t)$, $Y_{r\to d_1}(t)$, and $Y_{r\to d_2}(t)$.

Given the rate vector (R_1, R_2), a joint scheduling and network coding (NC) scheme is defined by n scheduling decision functions

$$\forall t \in [n],\ \sigma(t) = f_{\sigma,t}([\mathbf{Z}]_1^{t-1}), \tag{2.1}$$

$3n$ encoding functions at s_1, s_2, and r, respectively: For all $t \in [n]$

$$X_{s_i}(t) = f_{s_i,t}(\mathbf{W}_i, [\mathbf{Z}]_1^{t-1}), \quad \forall i \in \{1,2\}, \tag{2.2}$$

$$X_r(t) = f_{r,t}([\mathbf{Y}_{\{s_1,s_2\}\to r}, \mathbf{Z}]_1^{t-1}), \tag{2.3}$$

and 2 decoding functions at d_1 and d_2, respectively:

$$\hat{\mathbf{W}}_i = f_{d_i}([\mathbf{Y}_{\{s_1,s_2,r\}\to d_i}, \mathbf{Z}]_1^n), \quad \forall i \in \{1,2\}. \tag{2.4}$$

By (2.1), we allow $\sigma(t)$, the scheduling decision at time t, to be a function of the network-wide reception status vectors before time t. By (2.2), the encoding decision at s_i is a function depending on the information messages and past channel status. Encoding at r depends on what r received in the past and the past channel status vector, see (2.2). In the end, d_i decodes $\mathbf{W}_i$ based on what d_i has received and the past channel status of the entire network.[1] We allow the encoding and decoding functions $f_{s_i,t}$, $f_{r,t}$, and f_{d_i} to be linear or nonlinear.

[1]Since the scheduling decision $\sigma(t)$ is a function of $[\mathbf{Z}]_1^{t-1}$, all the encoding functions in (2.2) and (2.3), and the decoding functions in (2.7) also know implicitly the scheduling decision $\sigma(t)$.

This setting models the scenario in which there is a dedicated, error-free, low-rate control channel that can broadcast the previous network channel status $\mathbf{Z}(t-1)$ causally to all network nodes. The total amount of control information is no larger than 8 bits per time slot, which is much smaller than the actual payload of each packet $\approx 10^4$ bits. As a result, the perfect feedback channel can be easily implemented by piggybacking[2] on the data packets. The scheduling decision $\sigma(t)$ can be computed centrally (by a central controller) or distributively by each individual node since we allow all nodes to have the knowledge of the reception status of the entire network.

Definition 2.2.1 *Fix the distribution of $\mathbf{Z}(t)$. A rate vector (R_1, R_2) is achievable if for any $\epsilon > 0$, there exists a joint scheduling and NC scheme with sufficiently large n and $\mathsf{GF}(q)$ such that*

$$\max_{\forall i \in \{1,2\}} \mathsf{Prob}(\mathbf{W}_i \neq \hat{\mathbf{W}}_i) < \epsilon.$$

The capacity region is defined as the closure of all achievable rate vectors (R_1, R_2).

Remark: In (2.1), the scheduling decision $\sigma(t)$ does not depend on the information messages $\mathbf{W}_i$, which means that we prohibit the use of timing channels [30,31]. Even when we allow the usage of timing channels, we conjecture that the overall capacity improvement with the timing channel techniques is negligible. A heuristic argument is that each successful packet transmission gives $\log_2(q)$ bits of information while the timing information (to transmit or not) gives roughly 1 bit of information. When focusing on sufficiently large $\mathsf{GF}(q)$, additional gain of timing information is thus likely to be absorbed in our timing-information-free capacity characterization. In our setting, r is the only node that can mix packets from two different data flows. Further relaxation such that s_1 and s_2 can hear each other and perform coding accordingly is beyond the scope of this work.

[2]Some pipelining may be necessary to mitigate the propagation delay of the feedback control messages.

2.2.1 A Useful Notation

In our network model, there are 3 broadcast PECs associated with s_1, s_2, and r, respectively. We sometimes term those PECs the s_i-PEC, $i = 1, 2$, and the r-PEC. Since only one node can be scheduled in each time slot, we can assume that the reception events of each PEC are independent from that of the other PECs. As a result, the distribution of the network-wide channel status vector $\mathbf{Z}(t)$ can be described by the probabilities $p_{s_i \to T\overline{\{r,d_1,d_2\}\setminus T}}$ for all $i \in \{1, 2\}$ and for all $T \subseteq \{r, d_1, d_2\}$, and $p_{r \to U\overline{\{d_1,d_2\}\setminus U}}$ for all $U \subseteq \{d_1, d_2\}$. Totally there are $8 + 8 + 4 = 20$ parameters. By allowing some of the coordinates of $\mathbf{Z}(t)$ to be correlated, our setting can also model the scenario in which destinations d_1 and d_2 are situated in the same physical node and thus have perfectly correlated channel success events.

For notational simplicity, we also define the following three *probability* functions $p_{s_i}(\cdot)$, $i = 1, 2$, and $p_r(\cdot)$, one for each of the PECs. The input argument of each function p_s (s being one of $\{s_1, s_2, r\}$) is a collection of the elements in $\{d_1, d_2, r, \overline{d_1}, \overline{d_2}, \overline{r}\}$. The function $p_s(\cdot)$ outputs the probability that the reception event is *compatible* to the specified collection of $\{d_1, d_2, r, \overline{d_1}, \overline{d_2}, \overline{r}\}$. For example,

$$p_{s_1}(d_2\overline{r}) = p_{s_1 \to d_2\overline{d_1 r}} + p_{s_1 \to d_1 d_2 \overline{r}} \tag{2.5}$$

is the probability that the input of the s_1-PEC is successfully received by d_2 but not by r. Herein, d_1 is a *don't-care* receiver and $p_{s_1}(d_2\overline{r})$ thus sums two joint probabilities together (d_1 receives it or not) as described in (2.5). Another example is $p_r(d_2) = p_{r \to d_1 d_2} + p_{r \to \overline{d_1} d_2}$, which is the probability that a packet sent by r is heard by d_2. To slightly abuse the notation, we further allow $p_s(\cdot)$ to take multiple input arguments separated by the comma sign ",". With this new notation, $p_s(\cdot)$ then represents the probability that the reception event is compatible to *at least* one of the input arguments. For example,

$$\begin{aligned} p_{s_1}(d_1\overline{d_2}, r) = {} & p_{s_1 \to d_1\overline{d_2 r}} + p_{s_1 \to d_1\overline{d_2} r} + p_{s_1 \to d_1 d_2 r} \\ & + p_{s_1 \to \overline{d_1} d_2 r} + p_{s_1 \to \overline{d_1 d_2} r}. \end{aligned}$$

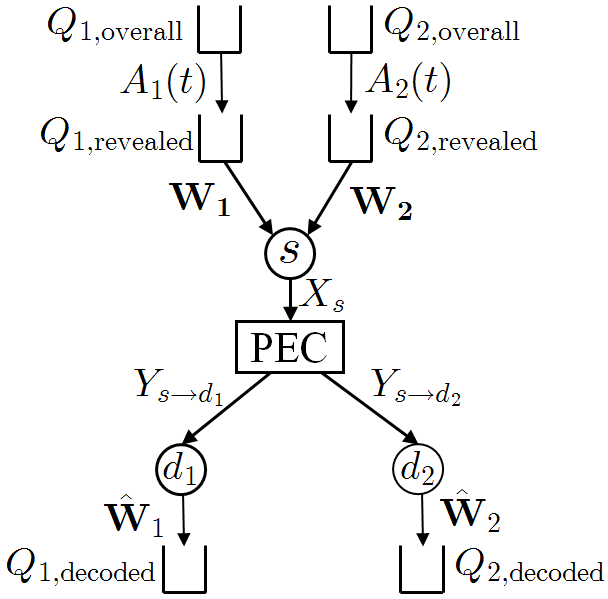


Fig. 2.2. The dynamics of stochastic arrivals in the 2-flow 1-to-2 broadcast packet erasure channel with feedback.

That is, $p_{s_1}(d_1\overline{d_2}, r)$ represents the probability that $(Z_{s_1\to d_1}, Z_{s_1\to d_2}, Z_{s_1\to r})$ equals one of the following 5 vectors $(1,*,*)$, $(1,*,1)$, $(1,1,1)$, $(*,1,1)$, and $(*,*,1)$. Note that these 5 vectors are compatible to either $d_1\overline{d_2}$ or r or both. Another example of this $p_s(\cdot)$ notation is $p_{s_1}(d_1, d_2, r)$, which represents the probability that a packet sent by s_1 is received by *at least* one of the three nodes d_1, d_2, and r.

2.3 Dynamics of Stochastic Arrivals And Queues

In Section 2.2, we described the formulation of Shannon capacity with fixed amount of information packets $\mathbf{W}_i = (W_{i,1}, W_{i,2}, ..., W_{i,nR_i})$. The above models can also be extended to the stability region analysis. We use the 2-flow 1-to-2 broadcast packet erasure channel with feedback as an example.

Figure 2.2 illustrates the dynamics of the stochastic arrivals in the 2-flow 1-to-2 broadcast packet erasure channel. Assuming the time-slotted system, we have two queue $Q_{1,\text{overall}}$ and $Q_{2,\text{overall}}$ which store infinite amount of flow-1 and flow-2 messages in $\mathsf{GF}(q)$, respec-

tively. We also have two arrival processes $A_1(t)$ and $A_2(t)$. For any time t and $i = 1, 2$, $A_i(t)$ denotes how many messages are sequentially transferred from $Q_{i,\text{overall}}$ to $Q_{i,\text{revealed}}$ at time t. And hence at time t, $Q_{i,\text{revealed}}$ contains the first $\sum_{\tau=1}^{t} A_i(\tau)$ messages in $Q_{i,\text{overall}}$. The subscript "revealed" denotes the packet in the queues are "visible" to the networks and required to be transmitted to its destinations. We assume $A_i(t)$ is i.i.d. over time and define the rate $(R_1, R_2) = (\mathsf{E}\{A_1(t)\}, \mathsf{E}\{A_2(t)\})$. We use $\mathbf{W}_i(t) = (W_{i,1}, W_{i,2}, ..., W_{i,\sum_{\tau=1}^{t} A_i(\tau)})$ to denote the collection of the messages in queue $Q_{i,\text{revealed}}$ at time t. We assume $\mathbf{W}_i(t)$ is independent with $\mathbf{Z}_s(t)$ as defined in Section 2.1.

Given the rate vector (R_1, R_2), at any time t, a network control algorithm including coding is defined by the encoding function at s:

$$X_s(t) = f_{s,t}([\mathbf{W}_1, \mathbf{W}_2, \mathbf{Z}_s]_1^{t-1}), \tag{2.6}$$

and 2 decoding functions at d_1 and d_2, respectively:

$$\hat{\mathbf{W}}_i(t) = f_{d_i,t}([\mathbf{Y}_{s\to d_i}, \mathbf{Z}_s]_1^t), \quad \forall i \in \{1, 2\}. \tag{2.7}$$

$\hat{\mathbf{W}}_i(t) = (\hat{W}_{i,1}, \hat{W}_{i,2}, ..., \hat{W}_{i,\sum_{\tau=1}^{t} A_i(\tau)}) \in (\mathsf{GF}(q) \cup \{\mathsf{e}\})^{\sum_{\tau=1}^{t} A_i(\tau)}$ is a $\sum_{\tau=1}^{t} A_i(\tau)$-dimensional vector. For any j, if $\hat{W}_{i,j}$ is in $\mathsf{GF}(q)$ then this message is decoded. Otherwise, $\hat{W}_{i,j} = \mathsf{e}$ means this message is undecoded. For any time t, we use $|Q_{i,\text{decoded}}|$ to denote the number of decoded messages in $\hat{\mathbf{W}}_i(t)$. We then define two queue lengths $Q_{i,\text{undecoded}}(t) = |Q_{i,\text{revealed}}| - |Q_{i,\text{decoded}}|$ at time t for $i = 1, 2$.

Definition 2.3.1 *A rate vector (R_1, R_2) is feasible if there exists a network control algorithm (including coding) such that $Q_{1,\text{undecoded}}(t)$ and $Q_{2,\text{undecoded}}(t)$ are stable*[3]*. The stability region of the 2-flow 1-to-2 broadcast packet erasure channel with feedback is the convex hull of all feasible rate vectors.*

[3]We will discuss the explicit definition of the stability in Section 5.1.

2.4 Chapter Summary

In this chapter, we formulate the model of the 1-to-M broadcast packet erasure channel in Section 2.1. In Section 2.2, we then construct a wireless butterfly network model including the COPE principle, the opportunistic routing, and the broadcast packet erasure channels with feedback. The corresponding Shannon capacity region is also defined in Section 2.2. We extend the problem setting to incorporate the dynamics of the stochastic arrivals and queues in Section 2.3 and finally define the stability region of 2-flow 1-to-2 broadcast packet erasure channel with feedback.

3. SPACE BASED LINEAR NETWORK CODING

In this chapter, we introduce a the central idea of this thesis. We will present a class of network coding scheme named the "Space-Based Linear Network Code (SBLNC)" scheme. An example of SBLNC scheme policies will be provided and will also be used to explain the design motivations of the SBLNC policies. The SBLNC schemes will later be used to achieve(1) the Shannon capacity of the 2-flow wireless butterfly network with packet erasure channels and (2) the stability region of the 2-flow 1-to-2 broadcast packet erasure channel with feedback.

3.1 Definitions

Here we use the 2-flow wireless butterfly network with packet erasure channels in Section 2.2 as an example and construct the corresponding SBLNC scheme. We first provide some basic definitions that will be used when describing an SBLNC scheme.

For $i = 1, 2$, a *flow-i coding vector* $\mathbf{v}^{(i)}$ is an nR_i-dimensional row vector with each coordinate being a scalar in $\mathsf{GF}(q)$. Any linear combination of the message symbols $W_{i,1}$ to W_{i,nR_i} can thus be represented by $\mathbf{v}^{(i)}\mathbf{W}_i^{\mathrm{T}}$ where $\mathbf{W}_i^{\mathrm{T}}$ is the transpose of $\mathbf{W}_i$. We use the superscript "(i)" to emphasize that we are focusing on a flow-i vector.

We define the *flow-i message space* by $\Omega_i \triangleq (\mathsf{GF}(q))^{nR_i}$, an nR_i-dimensional linear space. In the following, we define the following 6 *knowledge spaces* S_r, S_{d_1}, S_{d_2}, T_r, T_{d_1}, and T_{d_2} for the 5-node relay network in Fig. 2.1(b).

The knowledge spaces S_r, S_{d_2}, S_{d_1} are linear subspaces of Ω_1 and represent the knowledge about the flow-1 packets at nodes r, d_2, and d_1, respectively. Symmetrically, the knowledge spaces T_r, T_{d_1}, and T_{d_2} are linear subspaces of Ω_2 and represent the knowledge about the flow-2 packets at nodes r, d_1, and d_2, respectively. In the following, we dis-

cuss the detailed construction of S_r, S_{d_2}, and S_{d_1} and the construction of T_r to T_{d_2} follows symmetrically.[1]

- *In the end of any time t, $S_r(t) \subset \Omega_1$ is the* linear span *of a group of $\mathbf{v}^{(1)}$, denoted by $\mathbf{V}^{(1)}_{s_1\to r}$. The group $\mathbf{V}^{(1)}_{s_1\to r}$ contains the $\mathbf{v}^{(1)}$ vectors sent by s_1 during time 1 to t and have been received successfully by r. Throughout the paper, we use the convention that the linear span of an empty set is a set containing the zero vector, i.e., $\mathsf{span}\{\emptyset\} = \{0\}$. For example, if r has not yet received any packet from s_1, then by convention $S_r(t) = \{0\}$.*

- *In the end of time t, $S_{d_2}(t) \subset \Omega_1$ is the linear span of two groups of $\mathbf{v}^{(1)}$, denoted by $\mathbf{V}^{(1)}_{s_1\to d_2}$ and $\mathbf{V}^{(1)}_{r,N\to d_2}$. The first group $\mathbf{V}^{(1)}_{s_1\to d_2}$ contains the $\mathbf{v}^{(1)}$ vectors corresponding to the packets sent by s_1 during time 1 to t and have been received successfully by d_2. The second group $\mathbf{V}^{(1)}_{r,N\to d_2}$ contains the $\mathbf{v}^{(1)}$ vectors corresponding to the packets sent by r during time 1 to t that* are not mixed with any other flow-2 packets. *The letter "N" in the subscript stands for Not-inter-flow-coded transmission.*

- *In the end of time t, $S_{d_1}(t) \subset \Omega_1$ is the linear span of three groups of $\mathbf{v}^{(1)}$, denoted by $\mathbf{V}^{(1)}_{s_1\to d_1}$, $\mathbf{V}^{(1)}_{r,N\to d_1}$, and $\mathbf{V}^{(1)}_{r,C\to d_1}$. The first group $\mathbf{V}^{(1)}_{s_1\to d_1}$ contains the $\mathbf{v}^{(1)}$ vectors corresponding to the packets sent by s_1 during time 1 to t and have been received successfully by d_1. The second group $\mathbf{V}^{(1)}_{r,N\to d_1}$ contains the $\mathbf{v}^{(1)}$ vectors corresponding to the packets sent by r during time 1 to t that* are not mixed with any other flow-2 packets. *The third group $\mathbf{V}^{(1)}_{r,C\to d_1}$ contains the $\mathbf{v}^{(1)}$ vectors* that can be decoded from the inter-flow coded packets[2] sent by r during time 1 to t. *The letter "C" in the subscript stands for inter-flow-Coded transmission.*

In sum, we use S and T to distinguish whether we are focusing on flow-1 or flow-2 packets, respectively, and we use the subscripts to describe the node of interest. One can easily see that these six knowledge spaces evolve over time since each node may receive

[1]The construction of T_{d_1}(resp. T_{d_2}) follows the construction of S_{d_2}(resp. S_{d_1}).

[2]When the relay r sends a linear combination of both flow-1 and -2 packets.

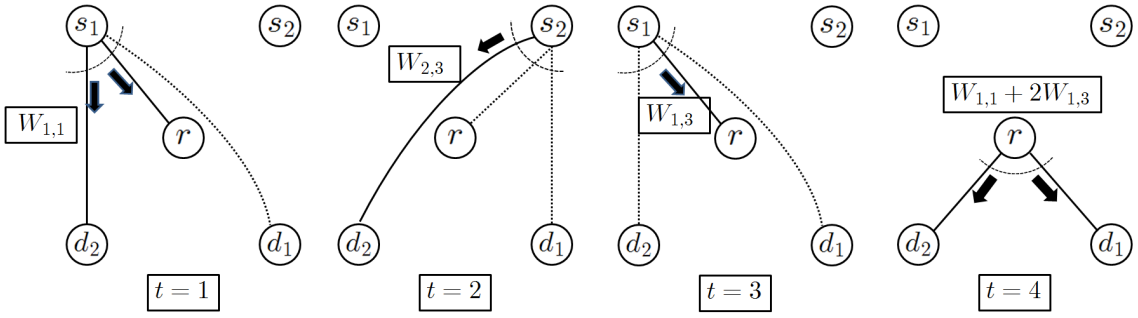


Fig. 3.1. The illustration of the coding procedure in Example 3.1.1. We use a solid line to represent that the corresponding receiver has successfully received the packet and use a dot line to represent the case of erasure.

more and more packets that can be used to obtain/decode new information. We use the following example to illustrate the definitions of S_r to T_{d_2}.

Example 3.1.1 *Consider* $\mathsf{GF}(3)$ *and* $nR_1 = 3$ *and* $nR_2 = 2$*. That is, flow-1 contains 3 message symbols* $W_{1,1}$ *to* $W_{1,3}$ *and flow-2 contains 2 message symbols* $W_{2,1}$ *and* $W_{2,2}$*.* Ω_1 *and* Ω_2 *are thus 3-dimensional and 2-dimensional linear spaces in* $\mathsf{GF}(3)$*, respectively. Consider the first four time slots* $t = 1$ *to* 4 *for our discussion.*

When $t = 1$*, suppose that* s_1 *is scheduled; an uncoded flow-1 message symbol* $W_{1,1}$ *is transmitted; and the packet is heard by and only by* d_2 *and* r*. See Fig. 3.1(a) for illustration, for which we use the solid lines to represent that* d_2 *and* r *have received the packet. We use the dashed line to denote that* d_1 *does not receive the packet. When* $t = 2$*, suppose that* s_2 *is scheduled; an uncoded flow-2 message symbol* $W_{2,1}$ *is transmitted; and the packet is heard by and only by* d_2*, see Fig. 3.1(b). When* $t = 3$*, suppose that* s_1 *is scheduled; an uncoded flow-1 symbol* $W_{1,3}$ *is transmitted; and the packet is heard by and only by* r*. When* $t = 4$*, suppose that* r *is scheduled;* r *sends a linear combination* $[W_{1,1} + 2W_{1,3}]$ *of the two flow-1 packets it has received thus far; and the packet* $[W_{1,1} + 2W_{1,3}]$ *is heard by both* d_1 *and* d_2*. We now describe the six knowledge spaces* S_r *to* T_{d_2} *in the end of* $t = 4$*. By Figs. 3.1(a) and 3.1(d),* d_2 *has received two flow-1 packets* $W_{1,1}$ *and* $[W_{1,1} + 2W_{1,3}]$*, one from* s_1 *and one from* r*. Therefore, by the end of* $t = 4$*, the flow-1 knowledge space at* d_2

Table 3.1
The resulting knowledge spaces at the end of Example 3.1.1.

Flow-1		Flow-2	
$S_{d_1}(4)$	$\mathsf{span}\{(1,0,2)\}$	$T_{d_2}(4)$	$\mathsf{span}\{(1,0)\}$
$S_r(4)$	$\mathsf{span}\{(1,0,0),(0,0,1)\}$	$T_r(4)$	$\{(0,0)\}$
$S_{d_2}(4)$	$\mathsf{span}\{(1,0,0),(1,0,2)\}$	$T_{d_1}(4)$	$\{(0,0)\}$

becomes $S_{d_2}(4) = \mathsf{span}\{(1,0,0),(1,0,2)\}$. *Also, neither* r *nor* d_1 *has received any flow-2 packets by the end of* $t = 4$. *Therefore,* T_r *and* T_{d_1}*, the flow-2 knowledge spaces at* r *and* d_1*, respectively, contain only the zero element. The other knowledge spaces* S_{d_1}*,* S_r*, and* T_{d_2} *in the end of* $t = 4$ *can be derived similarly and they are summarized in Table 3.1.*

The above definitions also lead to the following self-explanatory lemma.

Lemma 3.1.1 *The two destinations* d_1 *and* d_2 *can decode the desired message symbols* $\mathbf{W}_1$ *and* $\mathbf{W}_2$*, respectively, if and only if by the end of time* n

$$S_{d_1}(n) = \Omega_1 \text{ and } T_{d_2}(n) = \Omega_2.$$

For simplicity, we use $S_i(t)$ and $T_i(t)$ to denote the knowledge space $S_{d_i}(t)$ and $T_{d_i}(t)$ for $i = 1, 2$. We also omit the input argument "(t)" if the time index is clear from the context. To conclude this subsection, we introduce the notation of the sum space $(A \oplus B) \triangleq \mathsf{span}\{\mathbf{v} : \forall \mathbf{v} \in A \cup B\}$. Notice that $A \oplus B$ and $A \cup B$ are different. For example, in a 2-dimensional linear space with $\mathsf{GF}(2)$, we assume $A = \mathsf{span}\{(1,0)\}$ and $B = \mathsf{span}\{(1,1)\}$. Then $A \cup B = \{(0,0),(1,0),(2,0),(1,1),(2,2)\}$, but $A \oplus B = \mathsf{span}\{(1,0),(1,1)\} = \{(0,0),(1,0),(2,0),(1,1),(2,2),(2,1),(1,2),(0,1),(0,2)\}$. By simple algebra, we have

Lemma 3.1.2 *For any two linear subspaces* A *and* B *in* Ω*, the following equality always holds.*

$$\mathrm{Rank}(A \oplus B) = \mathrm{Rank}(A) + \mathrm{Rank}(B) - \mathrm{Rank}(A \cap B).$$

3.2 An Instance of SBLNC Policies

In the following, we will introduce a new class of network codes, named the Space-Based Linear Network Code (SBLNC). An SBLNC scheme contains a finite number of *policies*. Each policy Γ contains a linear subspace $A^{(\Gamma)}$, named *the inclusion space/set*, and a finite collection of subspaces $B_l^{(\Gamma)}$ for $l = 1$ to $L^{(\Gamma)}$, named *the exclusion spaces/sets*. For each time slot t, the SBLNC chooses one of the specified policies and uses it to generate the coded packet. For example, say node s is scheduled for transmission and we decide to use a policy Γ for encoding. Then s will first choose arbitrarily a coding vector $\mathbf{v}^{(i)}$ from the set $A^{(\Gamma)} \backslash \left(\bigcup_{l=1}^{L^{(\Gamma)}} B_l^{(\Gamma)} \right)$, and then transmit a linearly encoded packet $X = \mathbf{v}^{(i)} \mathbf{W}_i^{\mathrm{T}}$. That is, the coding vector must be in the inclusion set $A^{(\Gamma)}$ but not in any of the exclusion sets $B_l^{(\Gamma)}$. Obviously, a policy can be used/chosen only when the corresponding set $A^{(\Gamma)} \backslash \left(\bigcup_{l=1}^{L^{(\Gamma)}} B_l^{(\Gamma)} \right)$ is non-empty. For notational simplicity, we say a policy is *feasible* if the corresponding $A^{(\Gamma)} \backslash \left(\bigcup_{l=1}^{L^{(\Gamma)}} B_l^{(\Gamma)} \right)$ is non-empty.

For illustration, consider the following policy for node s_1, named Policy $\Gamma_{s_1,0}$. When Policy $\Gamma_{s_1,0}$ is used/chosen, we let source node s_1 choose arbitrarily a coding vector $\mathbf{v}^{(1)}$ from $\Omega_1 \backslash (S_1 \oplus S_2 \oplus S_r)$ and send the corresponding coded packet $X_{s_1} = \mathbf{v}^{(1)} \mathbf{W}_1^{\mathrm{T}}$. That is, the inclusion set is $A^{(\Gamma_{s_1,0})} = \Omega_1$ and the exclusion set is $B^{(\Gamma_{s_1,0})} = S_1 \oplus S_2 \oplus S_r$.

Continue the example in Section 3.1 for which the knowledge spaces are summarized in Table 3.1. In the beginning of $t = 5$ (or equivalently in the end of $t = 4$), we have $A^{(\Gamma_{s_1,0})} = \Omega_1$ and $B_1^{(\Gamma_{s_1,0})} = S_1 \oplus S_2 \oplus S_r = \{(a, 0, c) : \forall a, c \in \mathsf{GF}(q)\}$. As a result, if we choose Policy $\Gamma_{s_1,0}$ for $t = 5$, any coding vectors of the form (a, b, c) with $b \neq 0$ are in the set $\Omega_1 \backslash (S_1 \oplus S_2 \oplus S_r)$. There are totally 18 such vectors since $\mathsf{GF}(3)$ is used. Source s_1 can then choose arbitrarily from any one of the 18 vectors and send $X = aW_{1,1} + bW_{1,2} + cW_{1,3}$ in time $t = 5$.

In the following, we define 13 policies that will be used to achieve the Shannon capacity of the 2-flow wireless butterfly network with packet erasure channels.

There are 5 policies governing the coding operations at source s_1, which are named Policy $\Gamma_{s_1,j}$ for $j = 0$ to 4. When Policy $\Gamma_{s_1,j}$ is used, s_1 sends $X_{s_1}(t) = \mathbf{v}^{(1)} \mathbf{W}_1^{\mathrm{T}}$ for some

$\mathbf{v}^{(1)}$. That is, source s_1 only mixes/encodes flow-1 packets together. In the following, we describe how to choose the vector $\mathbf{v}^{(1)}$ for each individual policy.

§ *Policy* $\Gamma_{s_1,0}$*:* Choose $\mathbf{v}^{(1)}$ arbitrarily from

$$\Omega_1 \backslash (S_1 \oplus S_2 \oplus S_r). \tag{3.1}$$

§ *Policy* $\Gamma_{s_1,1}$*:* Choose $\mathbf{v}^{(1)}$ arbitrarily from

$$(S_2 \oplus S_r) \backslash \left((S_1 \oplus S_r) \cup (S_1 \oplus S_2) \right). \tag{3.2}$$

§ *Policy* $\Gamma_{s_1,2}$*:* Choose $\mathbf{v}^{(1)}$ arbitrarily from

$$S_2 \backslash (S_1 \oplus S_r). \tag{3.3}$$

§ *Policy* $\Gamma_{s_1,3}$*:* Choose $\mathbf{v}^{(1)}$ arbitrarily from

$$S_r \backslash \left(S_1 \oplus (S_2 \cap S_r) \right). \tag{3.4}$$

§ *Policy* $\Gamma_{s_1,4}$*:* Choose $\mathbf{v}^{(1)}$ arbitrarily from

$$(S_2 \cap S_r) \backslash S_1. \tag{3.5}$$

Policy $\Gamma_{s_2,j}$, $j = 0$ to 4 are symmetric versions of *Policy* $\Gamma_{s_1,j}$ that concern source s_2 and mix/encode flow-2 packets instead. More explicitly, source s_2 sends $X_{s_2} = \mathbf{v}^{(2)}\mathbf{W}_2^{\mathrm{T}}$ for which the coding vector $\mathbf{v}^{(2)}$ is chosen according to the following specification.

§ *Policy* $\Gamma_{s_2,0}$*:* Choose $\mathbf{v}^{(2)}$ arbitrarily from

$$\Omega_2 \backslash (T_1 \oplus T_2 \oplus T_r). \tag{3.6}$$

§ *Policy* $\Gamma_{s_2,1}$*:* Choose $\mathbf{v}^{(2)}$ arbitrarily from

$$(T_1 \oplus T_r) \backslash \left((T_2 \oplus T_r) \cup (T_1 \oplus T_2) \right). \tag{3.7}$$

§ *Policy* $\Gamma_{s_2,2}$*:* Choose $\mathbf{v}^{(2)}$ arbitrarily from

$$T_1 \backslash (T_2 \oplus T_r). \tag{3.8}$$

§ *Policy* $\Gamma_{s_2,3}$*:* Choose $\mathbf{v}^{(2)}$ arbitrarily from

$$T_r \backslash \left(T_2 \oplus (T_1 \cap T_r)\right). \tag{3.9}$$

§ *Policy* $\Gamma_{s_2,4}$*:* Choose $\mathbf{v}^{(2)}$ arbitrarily from

$$(T_1 \cap T_r) \backslash T_2. \tag{3.10}$$

There are 3 policies $\Gamma_{r,j}$, $j = 1, 2, 3$, governing the coding operations at the relay r, which are described as follows.

§ *Policy* $\Gamma_{r,1}$*:* The relay r chooses arbitrarily a vector $\mathbf{v}^{(1)}$ from

$$S_r \backslash \left((S_r \cap S_2) \oplus S_1\right) \tag{3.11}$$

and sends an intra-flow-coded flow-1 packet $X_r = \mathbf{v}^{(1)}\mathbf{W}_1^{\mathrm{T}}$.

§ *Policy* $\Gamma_{r,2}$*:* The relay r chooses arbitrarily a vector $\mathbf{v}^{(2)}$ from

$$T_r \backslash \left((T_r \cap T_1) \oplus T_2\right) \tag{3.12}$$

and sends an intra-flow-coded flow-2 packet $X_r = \mathbf{v}^{(2)}\mathbf{W}_2^{\mathrm{T}}$.

§ *Policy* $\Gamma_{r,3}$ is for the relay node r to send an interflow-coded packet $X_r = \mathbf{v}^{(1)}\mathbf{W}_1^{\mathrm{T}} + \mathbf{v}^{(2)}\mathbf{W}_2^{\mathrm{T}}$, with $\mathbf{v}^{(1)}$ and $\mathbf{v}^{(2)}$ chosen as follows: If $(S_2 \cap S_r) \backslash S_1$ is non-empty, choose $\mathbf{v}^{(1)}$ arbitrarily from

$$(S_2 \cap S_r) \backslash S_1, \tag{3.13}$$

otherwise choose $\mathbf{v}^{(1)} = \mathbf{0}$, a zero vector. If $(T_1 \cap T_r) \backslash T_2$ is non-empty, choose $\mathbf{v}^{(2)}$ arbitrarily from

$$(T_1 \cap T_r) \backslash T_2, \tag{3.14}$$

otherwise choose $\mathbf{v}^{(2)} = \mathbf{0}$.

Continue from Example 3.1.1 in Section 3.1 with the knowledge spaces in the end of $t = 4$ described in Table 3.1. Consider Policy $\Gamma_{s_1,3}$ as defined in (3.4). Since $S_2 \cap S_r = S_r$ in the end of $t = 4$, we have $S_r \backslash \left(S_1 \oplus (S_2 \cap S_r)\right) \subseteq S_r \backslash (S_2 \cap S_r) = \emptyset$ being an empty

set. Thus, in contrast with the fact that Policy $\Gamma_{s_1,0}$ is feasible in the beginning of $t = 5$ as shown in our previous discussion, Policy $\Gamma_{s_1,3}$ is infeasible in the beginning of $t = 5$.

One can repeat the above analysis and verify that out of all 13 policies, only 4 of them are feasible in the beginning of $t = 5$, which are $\Gamma_{s_1,0}$, $\Gamma_{s_1,4}$, $\Gamma_{s_2,0}$, and $\Gamma_{r,3}$. The network code designer can thus apply one of the four policies in $t = 5$.

Suppose the network designer chooses policy $\Gamma_{s_1,0}$ for $t = 5$ and sends a flow-1 coded packet with the coding vector being $\mathbf{v}^{(1)} = (2, 1, 0)$. Also suppose that the packet is received by r but by neither d_1 nor d_2. Then in the end of time $t = 5$, the knowledge space S_r evolves from the original $\mathsf{span}\{(1, 0, 0), (0, 0, 1)\}$ to the new space $\mathsf{span}\{(1, 0, 0), (0, 0, 1), (2, 1, 0)\}$. We now notice that the Policy $\Gamma_{s_1,0}$ is no longer feasible since with the new S_r, the exclusion space of $\Gamma_{s_1,0}$ becomes $S_1 \oplus S_2 \oplus S_r = \mathsf{span}\{(1, 0, 0), (0, 0, 1), (2, 1, 0)\}$ and $\Omega_1 \setminus (S_1 \oplus S_2 \oplus S_r)$ is now empty. On the other hand, the new S_r also lets some previously infeasible policies become feasible. For example, consider Policy $\Gamma_{s_1,3}$. With the new S_r, we have $S_r = \mathsf{span}\{(1, 0, 0), (0, 0, 1), (2, 1, 0)\}$ and $S_1 \oplus (S_2 \cap S_r) = \mathsf{span}\{(1, 0, 0), (1, 0, 2)\}$. Therefore, $S_r \setminus (S_1 \oplus (S_2 \cap S_r)) \neq \emptyset$. Policy $\Gamma_{s_1,3}$ is thus feasible and can be used for transmission in $t = 6$. With similar analysis, one can verify that in the beginning of $t = 6$, we have 5 feasible policies: $\Gamma_{s_1,3}$, $\Gamma_{s_1,4}$, $\Gamma_{s_2,0}$, $\Gamma_{r,1}$, and $\Gamma_{r,3}$.

3.3 The Design Motivations of SBLNC Policies

We conclude this chapter by discussing the design motivations behind the proposed 13 policies. We first consider the relay policies $\Gamma_{r,1}$ to $\Gamma_{r,3}$ due to its conceptual simplicity. We then discuss the source policies $\Gamma_{s_i,0}$ to $\Gamma_{s_i,4}$.

The Relay Policies

We first notice that for all relay r policies $\Gamma_{r,1}$, $\Gamma_{r,2}$, and $\Gamma_{r,3}$, the corresponding inclusion space is either a subspace of S_r or a subspace of T_r. The reason is that for node r to send a coded packet, the encoded packet must already be in S_r or T_r, the knowledge spaces

of r. As a result, the transmitted vector $\mathbf{v}^{(1)}$ (or $\mathbf{v}^{(2)}$) must be drawn from a subset of S_r (or T_r).

It is clear that *a good network code should try to serve two flows simultaneously in order to maximize the throughput.* We now focus on Policy $\Gamma_{r,3}$. First notice that by (3.14), $\mathbf{v}^{(2)}$ is drawn from $(T_1 \cap T_r)$. This means that the value of $\mathbf{v}^{(2)}\mathbf{W}_2^{\mathrm{T}}$ is already known by destination d_1 since $\mathbf{v}^{(2)}$ being in the flow-2 knowledge space T_1 at d_1. Hence whenever d_1 receives the packet $X_r(t) = \mathbf{v}^{(1)}\mathbf{W}_1^{\mathrm{T}} + \mathbf{v}^{(2)}\mathbf{W}_2^{\mathrm{T}}$, it can extract its desired information and recover $\mathbf{v}^{(1)}\mathbf{W}_1^{\mathrm{T}}$ by substracting $\mathbf{v}^{(2)}\mathbf{W}_2^{\mathrm{T}}$. We then note that Policy $\Gamma_{r,3}$ ensures that whenever (3.13) is not empty the selected $\mathbf{v}^{(1)}$ is not in S_1, the flow-1 knowledge space at d_1. Hence upon the reception of such a coded packet, $\mathrm{Rank}(S_1)$ will increase by one. By Lemma 3.1.1 destination d_1 is one step closer to fully decode its desired message $\mathbf{W}_1$. Symmetrically, by (3.13) d_2 has already known the value of $\mathbf{v}^{(1)}\mathbf{W}_1^{\mathrm{T}}$ and thus d_2 can compute the value of $\mathbf{v}^{(2)}\mathbf{W}_2^{\mathrm{T}}$ upon the reception of the inter-flow coded packet generated by Policy $\Gamma_{r,3}$. Since $\mathbf{v}^{(2)}$ is not in T_2, d_2 can decode one extra linear combination of flow-2 packets. Policy $\Gamma_{r,3}$ thus serves both d_1 and d_2 simultaneously.

Although Policy $\Gamma_{r,3}$ can serve both destinations simultaneously, there is a limit on how much information can be sent by $\Gamma_{r,3}$. That is, if we use only Policy $\Gamma_{r,3}$ and nothing else, the information that can be received by d_1 through Policy $\Gamma_{r,3}$ is at most $(S_r \cap S_2)$ since all $\mathbf{v}^{(1)}$ are drawn from $(S_r \cap S_2)$. The largest flow-1 knowledge space that d_1 can possibly attain is thus $S_1 \oplus (S_r \cap S_2)$, where S_1 represents the flow-1 information that d_1 has accumulated by overhearing the transmission directly from its two-hop neighbor s_1, and $(S_r \cap S_2)$ represents the information that can be conveyed by $\Gamma_{r,3}$. Note that it is possible that S_r is not a subspace of $S_1 \oplus (S_r \cap S_2)$, which means that relay r still possesses some flow-1 information that cannot be conveyed to d_1 by $\Gamma_{r,3}$ alone. $\Gamma_{r,1}$ is devised to address this problem. That is, the $\mathbf{v}^{(1)}$ vector chosen from (3.11) is (i) from the knowledge space of r, and (ii) not in $S_1 \oplus (S_r \cap S_2)$, the largest flow-1 knowledge space that d_1 can attain when using exclusively Policy $\Gamma_{r,3}$. Such $\mathbf{v}^{(1)}$ vector thus represents an information packet that is complementary to the inter-flow-coded Policy $\Gamma_{r,3}$.

The Source Policies

Here without loss of generality we focus on source 1 policies. Even though the policies can be chosen arbitrarily whenever they are feasible, in the following discussion, we will explain in a way that they are executed sequentially from Policy $\Gamma_{s_1,0}$ to Policy $\Gamma_{s_1,4}$ to better catch up the intuitions.

Before explaining the source policies (3.1)–(3.5), we first discuss what we expect to achieve during the source 1 transmission. There is one thing we must achieve. Meanwhile, there are two things we desire to achieve. The one we must achieve is $\Omega_1 = (S_1 \oplus S_r)$ at the end of the source 1 transmission since $s_1 \to r \to d_1$ and $s_1 \to d_1$ are the only two routes from s_1 to d_1. For the two things we desire to achieve, the first one is $\Omega_1 = (S_1 \oplus (S_2 \cap S_r))$. The reason is that the condition, $\Omega_1 = (S_1 \oplus (S_2 \cap S_r))$, makes Policy $\Gamma_{r,1}$ always being infeasible, and hence we can exploit the coding benefit from Policy $\Gamma_{r,3}$. The second one is trivially $\Omega_1 = S_1$.

With those things mentioned above in mind, we then examine and explain each source policy. Policy $\Gamma_{s_1,j}$, $j = 0, 1, 2$, are designed for achieving $\Omega_1 = S_1 \oplus S_r$ with $S_1 \oplus S_r$ being a subset of the exclusion sets for these policies. This can be observed step by step. After finishing all possible transmissions from Policy $\Gamma_{s_1,0}$, we have $\Omega_1 = (S_1 \oplus S_2 \oplus S_r)$. After finishing all possible transmissions from Policy $\Gamma_{s_1,0}$ and $\Gamma_{s_1,1}$, it is either $(S_2 \oplus S_r) \subset (S_1 \oplus S_r)$ or $(S_2 \oplus S_r) \subset (S_1 \oplus S_2)$. In the first case, $\Omega_1 = (S_1 \oplus S_2 \oplus S_r) = (S_1 \oplus S_r)$. For the second case, $\Omega_1 = (S_1 \oplus S_2 \oplus S_r) = (S_1 \oplus S_2)$. Then with the help from Policy $\Gamma_{s_1,2}$, $S_2 \subset (S_1 \oplus S_r)$ after finishing all possible transmissions from Policy $\Gamma_{s_1,2}$, the second case in Policy $\Gamma_{s_1,1}$ comes to the result, $\Omega_1 = (S_1 \oplus S_2 \oplus S_r) = (S_1 \oplus S_2) = (S_1 \oplus S_r)$. So far we have examined that Policy $\Gamma_{s_1,j}$, $j = 0, 1, 2$, help us to achieve $\Omega_1 = (S_1 \oplus S_r)$. However, one may ask why we need three policies to achieve this instead of simply one policy $\Omega_1 \backslash (S_1 \oplus S_r)$. This question will be answered right after the discussion of achieving $\Omega_1 = (S_1 \oplus (S_2 \cap S_r))$.

Policy $\Gamma_{s_1,j}$, $j = 0, 1, 2, 3$, are designed for achieving $\Omega_1 = (S_1 \oplus (S_2 \cap S_r))$ with $S_1 \oplus (S_2 \cap S_r)$ being a subset of the exclusion sets for these policies. Similar to the above

observation for $\Omega_1 = S_1 \oplus S_r$, one can verify that after finishing all possible transmissions from Policy $\Gamma_{s_1,0}$ to $\Gamma_{s_1,3}$, we have $\Omega_1 = (S_1 \oplus (S_2 \cap S_r))$. Furthermore, Policy $\Gamma_{s_1,1}$ is carefully designed for achieving $\Omega_1 = (S_1 \oplus (S_2 \cap S_r))$ in the most efficient way. To explain why Policy $\Gamma_{s_1,1}$ is the most efficient policy for achieving $\Omega_1 = (S_1 \oplus (S_2 \cap S_r))$, we first observe

$$\begin{aligned}&\mathrm{Rank}(S_1 \oplus (S_2 \cap S_r)) = \mathrm{Rank}(S_1) + \mathrm{Rank}(S_2 \cap S_r) - \mathrm{Rank}(S_1 \cap S_2 \cap S_r)\\ =&\mathrm{Rank}(S_1) + \mathrm{Rank}(S_2) + \mathrm{Rank}(S_r) - \mathrm{Rank}(S_2 \oplus S_r) - \mathrm{Rank}(S_1 \cap S_2 \cap S_r)\end{aligned}$$

Notice that $S_2 \oplus S_r$ is the inclusion set for Policy $\Gamma_{s_1,1}$; and S_1, S_2, and S_r are subsets of the exclusion sets for Policy $\Gamma_{s_1,1}$. Then with Policy $\Gamma_{s_1,1}$ being chosen, $\mathrm{Rank}(S_1 \oplus (S_2 \cap S_r))$ is expected to increase $p_1(d_1) + p_1(d_2) + p_1(r) - 0 - p_1(d_1 d_2 r)$. In this way, we maximize the positive terms and minimize the negative terms at the same time. This maximum increment of $\mathrm{Rank}(S_1 \oplus (S_2 \cap S_r))$ leads to the most efficient way for achieving $\Omega_1 = (S_1 \oplus (S_2 \cap S_r))$. This also answer the question why we need three policies for achieving $\Omega_1 = (S_1 \oplus S_r)$ instead of simply one policy, $\Omega_1 \backslash (S_1 \oplus S_r)$: Policy $\Gamma_{s_1,0}$ sets up the environment for executing Policy $\Gamma_{s_1,1}$, and Policy $\Gamma_{s_1,2}$ helps Policy $\Gamma_{s_1,1}$ back to achieving $\Omega_1 = (S_1 \oplus S_r)$. Finally, Policy $\Gamma_{s_1,j}$, $j = 0, 1, 2, 3, 4$, are designed for achieving $\Omega_1 = S_1$ with similar observations.

To summarize, these designed source 1 policies help us to achieve $\Omega_1 = S_1 \oplus S_r$, $\Omega_1 = S_1 \oplus (S_2 \oplus S_r)$, and $\Omega_1 = S_1$ simultaneously and in the most efficient way.

3.4 Chapter Summary

In this chapter, we introduce the central idea of this thesis, the space-based linear network coding. We use the 2-flow wireless butterfly network with packet erasure channels as an example and construct the corresponding SBLNC scheme. In Section 3.1, we introduce the necessary definition for constructing SBLNC policies. In Section 3.2, we present an example of SBLNC scheme with 13 designed policies. Later we will use these 13 policies to

achieve the Shannon capacity of the 2-flow wireless butterfly network with packet erasure channels. In Section 3.3, we discuss the design motivations of the SBLNC policies.

4. THE SHANNON CAPACITY OF WIRELESS BUTTERFLY NETWORK

In Section 2.2, we formulate a local network model including the broadcast packet erasure channel with feedback, the COPE principle, and the opportunistic routing. In this chapter, we will first discuss some related work and compare their settings with the one in Section 2.2. We then propose the corresponding outerbound and the inner bound which can be achieved by the SBLNC scheme described in Section 3.2. Finally we demonstrate the numerical results including the capacity region comparison and the sum rate throughput comparison.

4.1 Related Works

Recently, [3] and [4] successfully characterized the full capacity region of the 1-hop broadcast packet erasure channel with ≤ 3 coexisting flows. For comparison, our work focuses on the 2-hop network in Fig. 1.1(b) while [3] and [4] focus on the 1-hop broadcast channel. For the 2-hop network in Fig. 1.1(b), the network designer faces both the *scheduling problem*: which node (out of the two source nodes s_1, s_2, and the relay node r) to transmit at the current time slot, and the *network coding problem*: how to combine the heard/overheard packets and generate the network coded packets. For the 1-hop broadcast channel considered in [3] and [4], there is no scheduling problem since there is only one base station and the base station transmits all the time. As will be seen shortly, for a 2-hop erasure network, the feedback/control messages may propagate through the entire network and affect dynamically the scheduling and coding decisions for all three nodes s_1, s_2, and r, which further complicates the analysis.

Several attempts [17, 18] also have be made to approach the wireless butterfly network with packet erasure channels. Both of [17, 18] take the side-information coding benefit into

Table 4.1
The feature comparison between this thesis and [18]

	Features in [18]	Features in this thesis
The outer bound	(1) Sequential scheduling, (2) Batched feedback, (3) Nonlinear coding functions.	(1) Dynamic scheduling, (2) Per-packet feedback, (3) Nonlinear coding functions.
The inner bound	(1) Sequential scheduling, (2) Batched feedback, (3) Linear coding functions.	(1) Sequential scheduling, (2) Per-packet feedback, (3) Linear coding functions.

concern. But [17] only proposed a suboptimal achievable scheme while [18] characterized the full capacity region. In the following, we will discuss the major differences between [18] and this thesis.

There are three major differences between the setting of this this thesis and in [18]. First, a deterministic sequential scheduling policy was used in [18], which schedules nodes s_1, s_2, and r in strict order. Namely, s_1 transmits first. After s_1 stops, s_2 can begin to transmit. Only after s_2 stops transmission can r start its own transmission. For comparison, our setting allows for dynamically choosing the schedule $\sigma(t)$ for each time slot t. Since we allow the schedule $\sigma(t)$ to depend on the past reception status $[\mathbf{Z}]_1^{t-1}$, the use of $\sigma(t)$ also includes any store-&-forward-based scheduling policies as special cases, such as the back-pressure and the maximal weighted matching schemes (see [26] for references). Our results thus quantify the best achievable rates with jointly designed scheduling and coding policies.

Secondly, in [18] no feedback is allowed when s_1 and s_2 transmit. More specifically, suppose jointly s_1 and s_2 take $t_{s_1} + t_{s_2}$ time slots to finish transmission. Then only in the beginning of time $(t_{s_1}+t_{s_2}+1)$ are we allowed to send the channel status $[\mathbf{Z}]_1^{t_{s_1}+t_{s_2}}$ to r. No further feedback is allowed until time n, the end of overall transmission. For comparison, our setting allows constantly broadcasting network-wide channel status $[\mathbf{Z}]_1^{t-1}$ to s_1, s_2,

and r, as discussed in Section 2.2. This setting thus includes the Automatic Repeat reQuest (ARQ) mechanism as a special case [3, 18]. Broadcasting the control information $[\mathbf{Z}]_1^{t-1}$ to all the network nodes also eliminates the need of estimating/learning the reception status of the neighbors. Thirdly, [18] focuses on an arbitrary number of coexisting flows while this work focuses exclusively on the 2-flow scenario.

Last but not least, in the way of describing the inner bound in [18] and this work, they look similar because of the linear programming expressions and the use of the law of large number. However, they are essentially different. It is extremely difficult for the techniques in [18] to be extended to this work. The major obstacle is that in the setting of this thesis, the linear spaces constructed by the received packets keep evolving over time via per-packet feedback. Hence it is necessary to develop the SBLNC scheme to analysis the space evolution.

The practical COPE implementation contains three major components: (i) Opportunistic listening: Each destination is in a promiscuous monitoring mode and stores all the overheard packets; (ii) Opportunistic coding: The relay node decides which packets to be coded together opportunistically, based on the overhearing patterns of its neighbors; and (iii) Learning the states of the neighbors: Although in the practical COPE implementation reception reports are periodically sent to advertise the overhearing patterns of the next-hop neighbors of the relay, the relay node still needs to extrapolate the overhearing status of its neighbors since there is always a time lag due to the infrequent periodic feedback.

Our setting closely captures the opportunistic listening component of COPE by modeling the wireless packet transmission as a random broadcast PEC. In (2.1)–(2.3), the channel status vector is used to make the coding and scheduling decisions, which captures the opportunistic coding component of COPE. In COPE, the reception reports are broadcast periodically, which is captured by the control information $[\mathbf{Z}]_1^{t-1}$. In sum, our capacity region is a superset of any achievable rates of any COPE-principle-based schemes [5] when focusing on the 5-node 2-hop relay network in Fig. 1.1(b) with the node exclusive interference model.

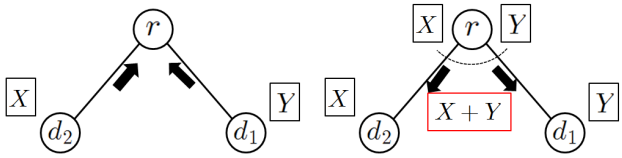


Fig. 4.1. The illustration of the two–way relay channel for which s_1 sends X to s_2 and and s_2 sends Y to s_1. In (b), the common relay can send a linear combination $[X + Y]$ that benefits both destinations simultaneously.

Remark: The setting in Section 2.2 also includes the wireless erasure 2-way relay channel model (Fig. 4.1(a) and 4.1(b)) as a special case. Specifically, if we set the overhearing probabilities: $p_i(d_j) = 1$ for all $i \neq j$, then the capacity region of the setting in Section 2.2 is also the capacity region of the wireless erasure 2-way relay channel in Fig. 4.1.

4.2 Main Results

In this section, we provide our results based on two cases: The case of considering only the COPE principle and the case of combining COPE with the opportunistic routing technique. The main difference is that for the former setting, we assume that no transmission can be heard by its 2-hop neighbors, i.e., $p_i(d_i) = 0$ for all $i = 1, 2$. For the latter setting, we allow $p_i(d_i)$ to be non-zero.

For the case of using exclusively the COPE principle, the full capacity region has been characterized in Section 4.2.1 while for the case of COPE plus opportunistic routing, a pair of outer and inner bounds are provided in Sections 4.2.2 and 4.2.3, respectively.

4.2.1 COPE Principle Relay Network Capacity

Proposition 4.2.1 *Consider any 2-flow wireless butterfly network with packet erasure channels with $p_i(d_i) = 0$ for all $i = 1, 2$. The rate pair (R_1, R_2) is in the capacity region if and only if there exist three non-negative time sharing parameters t_{s_1}, t_{s_2} and t_r such that jointly (R_1, R_2) and (t_{s_1}, t_{s_2}, t_r) satisfy*

$$t_{s_1} + t_{s_2} + t_r \leq 1 \tag{4.1}$$

$$\forall i \in \{1, 2\}, R_i \leq t_{s_i} p_i(r) \tag{4.2}$$

$$\frac{R_1}{p_r(d_1)} + \frac{(R_2 - t_{s_2} p_2(d_1))^+}{p_r(d_1, d_2)} \leq t_r \tag{4.3}$$

$$\frac{(R_1 - t_{s_1} p_1(d_2))^+}{p_r(d_1, d_2)} + \frac{R_2}{p_r(d_2)} \leq t_r \tag{4.4}$$

where $(\cdot)^+ \triangleq \max(0, \cdot)$ is the projection to non-negative reals.

The proof of the achievability part of Proposition 4.2.1 is relegated to Section 4.3.2 and the converse proof is relegated to Appendix A.

The intuition behind (4.1) to (4.4) is as follows. (4.1) is a time sharing bound, which follows from the total time budget being n and the node-exclusive interference model.

Inequality (4.2) is a simple cut-set bound. That is, the message W_i has to be sent from s_i to the common relay r first. Therefore, the rate is upper bounded by the link capacity from s_i to r.

Inequality (4.3) and (4.4) combine the capacity results on message-side-information [18] and the capacity results on channel output feedback for broadcast channels [3, 4]. A very heuristic, not rigorous explanation of (4.3) is as follows. $\frac{R_1}{p_r(d_1)}$ represents how many time slots it takes to send all the flow-1 packets to d_1 as if there is no flow-2. $t_{s_2}p_2(d_1)$ characterizes how much flow-2 information can be "overheard" by d_1, and $(R_2 - t_{s_2}p_2(d_1))^+$ thus represents how much flow-2 information that has not been heard by d_1 but still needs to be sent to d_2. Since those flow-2 packets cannot be "coded" together with any flow-1 packets, they need to be sent separately by themselves in addition to the $\frac{R_1}{p_r(d_1)}$ time slots used to send flow-1 packets. In general, it takes $\frac{\left(R_2 - t_{s_2}p_2(d1)\right)^+}{p_r(d_2)}$ for those packets to arrive at d_2. However, [3] shows that the use of feedback can further reduce the time to $\frac{\left(R_2 - t_{s_2}p_2(d_1)\right)^+}{p_r(d_1,d_2)}$. As a result, (4.3) governs the transmission since the total transmission time of relay r is nt_r time slots. (4.4) is symmetric to (4.3).

4.2.2 Capacity Outer Bound for COPE plus OpR

The capacity results in Proposition 4.2.1 can be generalized as an outer bound for the case when the destination d_i may overhear directly the transmission of s_i, i.e., $p_i(d_i) > 0$.

Proposition 4.2.2 *Consider any 2-flow wireless butterfly network with packet erasure channels in Fig. 2.1(b) with arbitrary channel characteristics. If a rate vector (R_1, R_2) is achievable, there exist three non-negative scalars t_{s_1}, t_{s_2}, and t_r satisfying*

$$t_{s_1} + t_{s_2} + t_r \leq 1 \tag{4.5}$$

$$\forall i \in \{1, 2\}, \quad R_i \leq t_{s_i} p_i(d_i, r) \tag{4.6}$$

$$\frac{(R_1 - t_{s_1} p_1(d_1))^+}{p_r(d_1)} + \frac{(R_2 - t_{s_2} p_2(d_1, d_2))^+}{p_r(d_1, d_2)} \leq t_r \tag{4.7}$$

$$\frac{(R_1 - t_{s_1} p_1(d_1, d_2))^+}{p_r(d_1, d_2)} + \frac{(R_2 - t_{s_2} p_2(d_2))^+}{p_r(d_2)} \leq t_r. \tag{4.8}$$

This proposition can be proven by canonical techniques in the information theory outer bound problems as in [32]. And hence we put the detailed proof in Appendix A for reader's reference.

Remark: One can easily see that when the channel probabilities satisfy $p_i(d_i) = 0$ for all $i = 1, 2$, the outer bound in Proposition 4.2.2 collapses to the capacity region in Proposition 4.2.1. Proposition 4.2.2 is thus a strict generalization of the converse part of Proposition 4.2.1.

4.2.3 Capacity Inner Bound for COPE plus OpR

An inner bound for the general case of $p_i(d_i) \geq 0$ is described as follows.

Proposition 4.2.3 *A rate vector (R_1, R_2) is achievable by a linear network code if there exist 3 non-negative variables t_{s_1}, t_{s_2}, t_r, 10 non-negative variables, $\omega_{s_i}^k$, where $i \in \{1, 2\}$ and $k \in \{0, 1, 2, 3, 4\}$, 4 non-negative variables $\omega_{r,N}^k$, $\omega_{r,C}^k$ for $k = 1, 2$, such that jointly the 17 variables*[1] *and (R_1, R_2) satisfy the following four groups of inequalities:*

[1] In the achieving algorithm in Section 4.3, the t variables correspond to the time slots that each of the sources and the relay is used; and the ω variables correspond to the time slots each policy is used.

Group 1 has 5 inequalities, named the time budget constraints.

$$\forall i = 1, 2, \quad \sum_{k=0}^{4} \omega_{s_i}^k \leq t_{s_i} \tag{4.9}$$

$$\forall i = 1, 2, \quad \omega_{r,N}^1 + \omega_{r,N}^2 + \omega_{r,C}^i \leq t_r \tag{4.10}$$

$$t_{s_1} + t_{s_2} + t_r < 1 \tag{4.11}$$

Group 2 has 12 *inequalities, named* the packet conservation laws at the source nodes. *Consider any* $i, j \in \{1, 2\}$ *satisfying* $i \neq j$. *For each* (i, j) *pair (out of the two choices* $(1, 2)$ *and* $(2, 1)$*), we have the following 6 inequalities.*

$$\omega_{s_i}^0 p_i(d_i, d_j, r) \leq R_i \tag{4.12}$$

$$\omega_{s_i}^1 p_i(d_i, r) \leq \omega_{s_i}^0 p_i(d_j \overline{d_i r}) \tag{4.13}$$

$$\omega_{s_i}^1 p_i(d_i, d_j) \leq \omega_{s_i}^0 p_i(r \overline{d_i d_j}) \tag{4.14}$$

$$\omega_{s_i}^2 p_i(d_i, r) \leq \omega_{s_i}^0 p_i(d_j \overline{d_i r}) - \omega_{s_i}^1 p_i(d_i, r) \tag{4.15}$$

$$\omega_{s_i}^3 p_i(d_i, d_j) \leq \omega_{s_i}^0 p_i(r \overline{d_i d_j}) - \omega_{s_i}^1 p_i(d_j, d_i r) \tag{4.16}$$

$$\begin{aligned} \omega_{s_i}^4 p_i(d_i) \leq\ & \omega_{s_i}^0 p_i(d_j r \overline{d_i}) \\ & + \omega_{s_i}^1 (p_i(d_j) + p_i(r) - p_i(d_i d_j r)) \\ & + \omega_{s_i}^2 p_i(r \overline{d_i}) + \omega_{s_i}^3 p_i(d_j \overline{d_i}) \end{aligned} \tag{4.17}$$

Group 3 has 4 inequalities, named the packet conservation laws at the relay node. *For each* (i, j) *pair with* $i \neq j$*, we have the following 2 inequalities.*

$$\begin{aligned} \omega_{r,N}^i p_r(d_i, d_j) \leq\ & \omega_{s_i}^0 p_i(r \overline{d_i d_j}) - \omega_{s_i}^1 p_i(d_j, d_i r) \\ & - \omega_{s_i}^3 p_i(d_i, d_j) \end{aligned} \tag{4.18}$$

$$\begin{aligned} \omega_{r,C}^i p_r(d_i) \leq\ & \omega_{s_i}^0 p_i(d_j r \overline{d_i}) \\ & + \omega_{s_i}^1 (p_i(d_j) + p_i(r) - p_i(d_i d_j r)) \\ & + \omega_{s_i}^2 p_i(r \overline{d_i}) + \omega_{s_i}^3 p_i(d_j \overline{d_i}) \\ & - \omega_{s_i}^4 p_i(d_i) + \omega_{r,N}^i p_r(d_j \overline{d_i}) \end{aligned} \tag{4.19}$$

Group 4 has 2 inequalities, named the decodability conditions. *Consider* $i = 1, 2$. *For each* i*, we have the following inequality.*

$$\left(\sum_{k=0}^{4} \omega_{s_i}^k\right) p_i(d_i) + \left(\omega_{r,N}^i + \omega_{r,C}^i\right) p_r(d_i) \geq R_i \tag{4.20}$$

An heuristic but not rigorous explanation is as follows. The time budget constraints (4.9)–(4.11) describe the fact that the each transmitting node can only select policies within its own time budget and the overall time budget is one in ratio. The conservation laws (4.12)–(4.19) describe the fact that for one policy being eligible to be selected, it must be a non-empty set, which is equivalent to quantify the space dimensions of the policies. The decodability conditions describe the fact that to be able to reconstruct the required packets at two destinations, a certain amount of packets must be received by two destinations.

Proposition 4.2.3 will be proved by explicit construction of an achievability scheme based on the SBLNC scheme described in the next section. The detailed proof of Proposition 4.2.3 is relegated to Section 4.3.1.

4.3 Capacity Approaching Coding Scheme

In this section, we will first prove the capacity inner bound *Proposition 4.2.3* for the setting of the COPE principle when allowing opportunistic routing. We will then prove that the inner bound coincides with the capacity characterization in Proposition 4.2.1 for the COPE principle without the opportunistic routing component.

4.3.1 Achieving The Inner Bound of Proposition 4.2.3

We prove Proposition 4.2.3 by properly scheduling the 13 policies described in Section 3.2.

Consider any t_{s_1}, t_{s_2}, t_r, $\omega_{s_i}^k$, $i \in \{1, 2\}$ and $k \in \{0, 1, 2, 3, 4\}$, $\omega_{r,\mathrm{N}}^k$, and $\omega_{r,\mathrm{C}}^k$, $k = 1, 2$, satisfying the inequalities (4.9) to (4.20) in Proposition 4.2.3. For any $\epsilon > 0$, we can always

construct another set of t' and ω' variables such that the new t' and ω' variables satisfy (4.9) to (4.19) with strict inequality, and satisfy the following inequality

$$\left(\sum_{k=0}^{4}\omega_{s_i}^k\right)p_i(d_i)+\left(\omega_{r,\mathrm{N}}^i+\omega_{r,\mathrm{C}}^i\right)p_r(d_i)>R_i-\epsilon \tag{4.21}$$

instead of (4.20). Based on the above observation, we will assume that the given t_{s_1}, t_{s_2}, t_r, $\omega_{s_i}^k$, $\omega_{r,\mathrm{N}}^k$, and $\omega_{r,\mathrm{C}}^k$ satisfy (4.9) to (4.19) and (4.21) with strict inequality. In the following, we will construct an SBLNC solution such that the scheme "properly terminates" within the allocated n time slots with close-to-one probability and after the SBLNC scheme stops, each d_i has received at least $n(R_i-\epsilon)$ number of its desired information packets.

We construct the SBLNC scheme as follows. We first schedule the s_1-policies sequentially from $\Gamma_{s_1,0}$ to $\Gamma_{s_1,4}$. Each policy $\Gamma_{s_1,k}$ lasts for $n\cdot\omega_{s_1}^k$ time slots. After finishing $\Gamma_{s_1,k}$ we move on to Policy $\Gamma_{s_1,k+1}$ until finishing all 5 s_1-policies. After finishing the s_1-policies, we move on to the s_2-policies. Again, we choose the s_2-policies sequentially from $\Gamma_{s_2,0}$ to $\Gamma_{s_2,4}$ and each policy lasts for $n\cdot\omega_{s_2}^k$ time slots. After the s_2-policies, we schedule the r-policies sequentially from $\Gamma_{r,1}$ to $\Gamma_{r,3}$. Policies $\Gamma_{r,1}$ and $\Gamma_{r,2}$ last for $n\cdot\omega_{r,\mathrm{N}}^1$ and $n\cdot\omega_{r,\mathrm{N}}^2$ time slots, respectively. Policy $\Gamma_{r,3}$ lasts for $n\cdot\max\{\omega_{r,\mathrm{C}}^1,\omega_{r,\mathrm{C}}^2\}$ time slots. Feedback is critical for the SBLNC scheme as it is used to decide the evolution of the knowledge spaces S_1, S_2,..., T_r, which in turn decides the sets in (3.1)–(3.14).

For the above construction, we first discuss its dependency on the finite field size q. Among the entire designed policies (3.1)–(3.14), observe that $\max L^{(\Gamma)}$ is two, which means there are at most two exclusion spaces for one policy. Thus for each of the designed policies being non-empty, the minimum requirement of q is no less than 2. To prove this statement, assume we have 3 linear spaces A, B, and C with the designed policy $A\backslash(B\cup C)$. For this policy being non-empty, it requires $q^{\mathrm{Rank}(A)}=|A|>|(B\cup C)\cap A|$. Furthermore, one can show that $\mathrm{Rank}(A)>\max\{\mathrm{Rank}(A\cap B),\mathrm{Rank}(A\cap C)\}$ implies $|A|>|A\cap B|+|A\cap C|-1(=q^{\mathrm{Rank}(A\cap B)}+q^{\mathrm{Rank}(A\cap C)}-1)\geq|(B\cup C)\cap A|$ with $q\geq 2$. Thus as shown in [3], the feedback capacity of 2-user broadcast erasure channel can be achieved with $q=2$. The scheme proposed here also works for $q=2$.

To prove the correctness of the above construction, we need to show that the following two statements hold with close-to-one probability: (i) during each time slot, it is always possible to construct the desired coding vectors $\mathbf{v}^{(1)}$ (or $\mathbf{v}^{(2)}$). That is, we never schedule an infeasible policy throughout the operation; (ii) destination d_i can decode $n(R_i - \epsilon)$ of the desired information packets when the scheme terminates[2]. In addition to the above two statements, we will also prove that with close-to-one probability; and (iii) during the first $n \cdot \omega^1_{r,\mathrm{C}}$ (resp. $n \cdot \omega^2_{r,\mathrm{C}}$) time slots of scheduling $\Gamma_{r,3}$, the computed flow-1 vector $\mathbf{v}^{(1)}$ (resp. flow-2 vector $\mathbf{v}^{(2)}$) is not zero.

We first prove (ii) while assuming both (i) and (iii) are true. We notice that all the exclusion spaces of policies $\Gamma_{s_1,0}$ to $\Gamma_{s_1,4}$, and $\Gamma_{r,1}$ contain S_1 as a subset. As a result, all those packets carry some new flow-1 information that has not yet been received by d_1. If d_1 receives any of those packets, the rank of S_1 will increase by one. Similarly, during the first $n\omega^1_{r,\mathrm{C}}$ time slots of Policy $\Gamma_{r,3}$, the computed $\mathbf{v}^{(1)}$ vector does not belong to S_1, see (3.13). As a result, if d_1 receives any of those packets, the rank of S_1 will again increase by one. From the above reasoning, the expected value of $\mathrm{Rank}(S_1)$ in the end of the SBLNC scheme must satisfy

$$\begin{aligned}\mathsf{E}\{\mathrm{Rank}(S_1)\} =& p_1(d_1)\left(\sum_{k=0}^{4} n\omega^k_{s_1}\right)\\ &+ p_r(d_1)(n\omega^1_{r,\mathrm{N}} + n\omega^1_{r,\mathrm{C}}) \qquad (4.22)\\ >& n(R_1 - \epsilon) \qquad (4.23)\end{aligned}$$

where the right-hand side of (4.22) quantifies the expected number of packets received by d_1 during Policies $\Gamma_{s_1,0}$ to $\Gamma_{s_1,4}$, $\Gamma_{r,1}$, and the first $n\omega^1_{r,\mathrm{C}}$ time slots of $\Gamma_{r,3}$. (4.23) follows from (4.21). By the law of large number, $\mathrm{Rank}(S_1) > n(R_1 - \epsilon)$ with close-to-one probability when n is sufficiently large. The above inequality ensures that d_1 can decode $n(R_1 - \epsilon)$ of the flow-1 information packets at the end of the SBLNC scheme. By symmetry, d_2 can also decode $n(R_2 - \epsilon)$ of the flow-2 packets $\mathbf{W}_2$ in the end of time $t = n$. What remains to be shown is to prove that (i) and (iii) hold with close-to-one probability.

[2]As the existence guaranteed in Proposition 3, given t and ω variables satisfying inequality (4.9)–(4.20), inequalities (4.9)–(4.11) guarantee that we can finish transmission within the allocated n time slots.

Next we prove (i) and (iii) by the first order analysis that assumes sufficiently large n. We first consider Policy $\Gamma_{s_1,0}$. For any time t, $\Gamma_{s_1,0}$ is a feasible policy if (3.1) is non-empty, which is equivalent to having

$$\begin{aligned}&\mathrm{Rank}(\Omega_1) - \mathrm{Rank}(\Omega_1 \cap (S_1 \oplus S_2 \oplus S_r)) \\ &= \mathrm{Rank}(\Omega_1) - \mathrm{Rank}(S_1 \oplus S_2 \oplus S_r) > 0. \end{aligned} \tag{4.24}$$

We first note that $\mathrm{Rank}(\Omega_1) = nR_1$ is a constant and does not change over time. Also note that $\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r)$ increases monotonically over time since a node accumulates more "knowledge" over time. As a result, if we can prove that (4.24) holds in the end of the duration of (executing) Policy $\Gamma_{s_1,0}$, then throughout the entire duration of $\Gamma_{s_1,0}$, we can always find some $\mathbf{v}^{(1)}$ belong to (3.1).

To that end, we notice that when we choose $\Gamma_{s_1,0}$ as our coding policy, the coding vector $\mathbf{v}^{(1)}$ is chosen from (3.1). Since $\mathbf{v}^{(1)}$ does not belong to the exclusion space $S_1 \oplus S_2 \oplus S_r$, $\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r)$ *increases by one if and only if at least one of* d_1*,* d_2*, and* r *receives the transmitted packet* $X_{s_1} = \mathbf{v}^{(1)}\mathbf{W}_1^{\mathrm{T}}$. Also note that in the beginning of Policy $\Gamma_{s_1,0}$, $\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r) = 0$. As a result, in the end of the duration of $\Gamma_{s_1,0}$, we have

$$\begin{aligned}&\mathsf{E}\{\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r)\} \\ &= 0 + n \cdot \omega_{s_1}^0 \cdot p_1(d_1, d_2, r) \qquad (4.25) \\ &< nR_1 = \mathrm{Rank}(\Omega_1), \qquad (4.26)\end{aligned}$$

where (4.25) follows from quantifying the expected number of time slots (out of totally $n\omega_{s_1}^0$ time slots) in which at least one of d_1, d_2, and r receives it. (4.26) follows from (4.12).

By the law of large numbers, (4.26) implies that (4.24) holds in the end of the duration of $\Gamma_{s_1,0}$ with close-to-one probability. As a result, with close-to-one probability Policy $\Gamma_{s_1,0}$ remains feasible during the assigned duration of $n \cdot \omega_{s_1}^0$ time slots.

We now consider Policy $\Gamma_{s_1,1}$. For any time t, $\Gamma_{s_1,1}$ is feasible if (3.2) is non-empty, which is equivalent to having

$$\begin{aligned} q^{\mathrm{Rank}(S_2\oplus S_r)} &= |S_2 \oplus S_r| > |(S_2 \oplus S_r) \cap ((S_1 \oplus S_r) \cup (S_1 \oplus S_2))| \\ &= |((S_2 \oplus S_r) \cap (S_1 \oplus S_r)) \cup ((S_2 \oplus S_r) \cap (S_1 \oplus S_2))| \\ \Leftrightarrow\ & \mathrm{Rank}(S_2 \oplus S_r) \\ &> \max\{\mathrm{Rank}((S_2 \oplus S_r) \cap (S_1 \oplus S_r)), \\ &\qquad \mathrm{Rank}((S_2 \oplus S_r) \cap (S_1 \oplus S_2))\} \end{aligned} \tag{4.27}$$

where "$\Leftrightarrow$" holds assuming the underlying finite field $\mathsf{GF}(q)$ satisfying $q \geq 2$. When we choose Policy $\Gamma_{s_1,1}$ as our coding policy, the coding vector $\mathbf{v}^{(1)}$ is chosen from (3.2). Therefore, $\mathbf{v}^{(1)}$ must belong to the inclusion space $S_2 \oplus S_r$, which implies that no matter how many nodes in $\{d_1, d_2, r\}$ receive the packet, $\mathrm{Rank}(S_2 \oplus S_r)$ remains the same. Also note that similar to the case of $\Gamma_{s_1,0}$, $\mathrm{Rank}((S_2 \oplus S_r) \cap (S_1 \oplus S_r))$ and $\mathrm{Rank}((S_2 \oplus S_r) \cap (S_1 \oplus S_2))$ increase monotonically over time. As a result, if we can prove that (4.27) holds in the end of the duration of Policy $\Gamma_{s_1,1}$, then throughout the entire duration of $\Gamma_{s_1,1}$, we can always find some $\mathbf{v}^{(1)}$ belong to (3.2). The remaining task is thus to quantify the three different ranks $\mathrm{Rank}(S_2 \oplus S_r)$, $\mathrm{Rank}((S_2 \oplus S_r) \cap (S_1 \oplus S_r))$, and $\mathrm{Rank}((S_2 \oplus S_r) \cap (S_1 \oplus S_2))$ at the end of (the duration of) $\Gamma_{s_1,1}$. All the following discussions hold with close-to-one probability when focusing on the first order analysis of n.

First consider $\mathrm{Rank}(S_2 \oplus S_r)$. We know that $\mathrm{Rank}(S_2 \oplus S_r)$ remains the same during Policy $\Gamma_{s_1,1}$. Therefore, the value of $\mathrm{Rank}(S_2 \oplus S_r)$ is decided by how much it increases during $\Gamma_{s_1,0}$. Since any $\mathbf{v}^{(1)}$ in Policy $\Gamma_{s_1,0}$ does not belong to $S_2 \oplus S_r$ (see (3.1)), every time one of d_2 and r receives a packet of $\Gamma_{s_1,0}$, $\mathrm{Rank}(S_2 \oplus S_r)$ will increase by one. As a result, in the end of $\Gamma_{s_1,1}$ we have

$$\mathsf{E}\{\mathrm{Rank}(S_2 \oplus S_r)\} = n\omega_{s_1}^0 p_1(d_2, r) + n\omega_{s_1}^1 \cdot 0. \tag{4.28}$$

We now consider the first term $\mathrm{Rank}((S_2 \oplus S_r) \cap (S_1 \oplus S_r))$ in the max operation in (4.27). By Lemma 3.1.2, we can rewrite $\mathrm{Rank}((S_1 \oplus S_r) \cap (S_2 \oplus S_r))$ by

$$\begin{aligned}&\mathrm{Rank}((S_1 \oplus S_r) \cap (S_2 \oplus S_r))\\ &= \mathrm{Rank}(S_2 \oplus S_r) + \mathrm{Rank}(S_1 \oplus S_r) - \mathrm{Rank}(S_1 \oplus S_2 \oplus S_r). \end{aligned} \tag{4.29}$$

The value of $\mathrm{Rank}(S_2 \oplus S_r)$ is quantified in (4.28). Since any $\mathbf{v}^{(1)}$ in Policy $\Gamma_{s_1,0}$ does not belong to $S_1 \oplus S_r$ (see (3.1)) and any $\mathbf{v}^{(1)}$ in Policy $\Gamma_{s_1,1}$ does not belong to $S_1 \oplus S_r$ either (see (3.2)), every time one of d_1 and r receives a packet of $\Gamma_{s_1,0}$ or $\Gamma_{s_1,1}$, $\mathrm{Rank}(S_1 \oplus S_r)$ will increase by one. In the end of $\Gamma_{s_1,1}$ we thus have

$$\mathsf{E}\{\mathrm{Rank}(S_1 \oplus S_r)\} = n\omega_{s_1}^0 \cdot p_1(d_1, r) + n\omega_{s_1}^1 \cdot p_1(d_1, r). \tag{4.30}$$

Similarly, since any $\mathbf{v}^{(1)}$ in Policy $\Gamma_{s_1,0}$ does not belong to $S_1 \oplus S_2 \oplus S_r$ (see (3.1)) and any $\mathbf{v}^{(1)}$ in Policy $\Gamma_{s_1,1}$ belongs to $S_1 \oplus S_2 \oplus S_r$ (see (3.2)), every time one of d_1, d_2, and r receives a packet of $\Gamma_{s_1,0}$, $\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r)$ will increase by one. In the end of $\Gamma_{s_1,1}$ we thus have

$$\mathsf{E}\{\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r)\} = n\omega_{s_1}^0 p_1(d_1, d_2, r) + n\omega_{s_1}^1 \cdot 0. \tag{4.31}$$

By (4.28), (4.29), (4.30), and (4.31), we can verify that (4.13) implies that $\mathrm{Rank}(S_2 \oplus S_r) > \mathrm{Rank}((S_2 \oplus S_r) \cap (S_1 \oplus S_r))$ in the end of Policy $\Gamma_{s_1,1}$. By swapping the roles of d_2 and r, symmetric arguments can be used to prove that (4.14) implies $\mathrm{Rank}(S_2 \oplus S_r) > \mathrm{Rank}((S_2 \oplus S_r) \cap (S_1 \oplus S_2))$ in the end of Policy $\Gamma_{s_1,1}$. Therefore, $\Gamma_{s_1,1}$ is feasible throughout its duration of $n\omega_{s_1}^1$ time slots.

Similar rank-comparison arguments can be used to complete the proof of (i) and (iii). The remaining derivation repeats similar steps described above, and hence is relegated to Appendix B. The proof of Proposition 4.2.3 is thus complete.

4.3.2 Capacity of COPE Principle 2-Flow Wireless Butterfly Network Without Opportunistic Routing

In this subsection we will prove that the capacity outer bound in Proposition 4.2.2 and the capacity inner bound in Proposition 4.2.3 coincide when destination d_i cannot directly

hear from source s_i for $i = 1, 2$. Proposition 4.2.1 thus describes the exact COPE-principle 2-flow wireless butterfly network capacity region without opportunistic routing.

To complete the proof, we note that when $p_i(d_i) = 0$, for $i = 1, 2$, (4.12)–(4.20) of the inner bound in Proposition 4.2.3 is reduced to the following forms:[3]

$$\omega^0_{s_i} p_i(d_j, r) \le R_i, \tag{4.32}$$

$$\omega^1_{s_i} p_i(r) \le \omega^0_{s_i} p_i(d_j \overline{r}), \tag{4.33}$$

$$\omega^1_{s_i} p_i(d_j) \le \omega^0_{s_i} p_i(r \overline{d_j}), \tag{4.34}$$

$$\omega^2_{s_i} p_i(r) \le \omega^0_{s_i} p_i(d_j \overline{r}) - \omega^1_{s_i} p_i(r), \tag{4.35}$$

$$\omega^3_{s_i} p_i(d_j) \le \omega^0_{s_i} p_i(r \overline{d_j}) - \omega^1_{s_i} p_i(d_j), \tag{4.36}$$

$$\begin{aligned} \omega^i_{r,\mathrm{N}} p_r(d_i, d_j) \le{}& \omega^0_{s_i} p_i(r \overline{d_j}) - \omega^1_{s_i} p_i(d_j) \\ & - \omega^3_{s_i} p_i(d_j), \end{aligned} \tag{4.37}$$

$$\begin{aligned} \omega^i_{r,\mathrm{C}} p_r(d_i) \le{}& \omega^0_{s_i} p_i(d_j r) + \omega^1_{s_i} (p_i(d_j) + p_i(r)) \\ & + \omega^2_{s_i} p_i(r) + \omega^3_{s_i} p_i(d_j) + \omega^i_{r,\mathrm{N}} p_r(d_j \overline{d_i}). \end{aligned} \tag{4.38}$$

and (4.20) becomes

$$p_r(d_i)(\omega^i_{r,\mathrm{N}} + \omega^i_{r,\mathrm{C}}) \ge R_i. \tag{4.39}$$

The following lemma proves the tightness of the bounds when there is no 2-hop overhearing, i.e., $p_i(d_i) = 0$ for $i = 1, 2$.

Lemma 4.3.1 *For any 5-tuple $(R_1, R_2, t_1, t_2, t_r)$ satisfying the capacity outer bound (4.1)–(4.4), we can always find 14 companying variables $\omega^j_{s_i}, \omega^i_{r,N}, \omega^i_{r,C}$ for $i = 1, 2$ and $j = 0, 1, 2, 3, 4$, such that jointly the $14 + 5 = 19$ variables satisfy (4.9), (4.10), (4.32) to (4.39).*

[3] Inequality (4.17) becomes trivial since the left-hand side of (4.17) becomes zero and the right-hand side of (4.17) is always non-negative.

Proof Given any $(R_1, R_2, t_{s_1}, t_{s_2}, t_r)$ satisfying (4.1)–(4.4), we construct $\left\{\omega^j_{s_i}, \omega^i_{r,\mathrm{N}}, \omega^i_{r,\mathrm{C}} : i \in \{1,2\}, j \in \{0,1,2,3,4\}\right\}$ in the following way. For each pair $(i,j) = (1,2)$ or $(2,1)$, we define

$$\begin{aligned}
\omega^0_{s_i} =& \frac{R_i}{p_i(d_j, r)},\\
\omega^1_{s_i} =& R_i \left(\min\left\{ \frac{1}{p_i(r)}, \frac{1}{p_i(d_j)} \right\} - \frac{1}{p_i(d_j, r)} \right),\\
\omega^2_{s_i} =& R_i \left(\frac{1}{p_i(r)} - \frac{1}{p_i(d_j)} \right)^+,\\
\omega^3_{s_i} =& \min\left\{ R_i \left(\frac{1}{p_i(d_j)} - \frac{1}{p_i(r)} \right)^+, t_{s_i} - \frac{R_i}{p_i(r)} \right\},\\
\omega^4_{s_i} =& 0,\\
\omega^i_{r,\mathrm{N}} =& \frac{(R_i - t_{s_i} p_i(d_j))^+}{p_r(d_i, d_j)},\\
\omega^i_{r,\mathrm{C}} =& \frac{R_i}{p_r(d_i)} - \frac{(R_i - t_{s_i} p_i(d_j))^+}{p_r(d_i, d_j)}.
\end{aligned}$$

One can verify that the above assignment $\left\{R_1, R_2, t_{s_1}, t_{s_2}, t_r, \omega^j_{s_i}, \omega^i_{r,\mathrm{N}}, \omega^i_{r,\mathrm{C}} : i \in \{1,2\}, j \in \{0,1,2,3,4\}\right\}$ satisfies (4.9), (4.10), (4.32) to (4.39). The detailed verification is relegated to Appendix C. The proof of Lemma 4.3.1 is thus complete. ■

4.4 Numerical Results

In this section, we apply the capacity results to some numerically generated scenarios so that we can explicitly quantify the throughput/capacity improvement of the COPE principle. The detailed simulation setting is described as follows.

Consider one specific setting of the 2-flow wireless butterfly network as depicted in Fig. 4.2. In Fig. 4.2, we specify the success transmission probability between each node pair as the number next to the corresponding arrow and we assume the success events between different node pairs are independent. For example, the probability that a packet sent by s_1 is heard by d_1 is $p_1(d_1) = .2$ and the probability that a packet sent by r is received

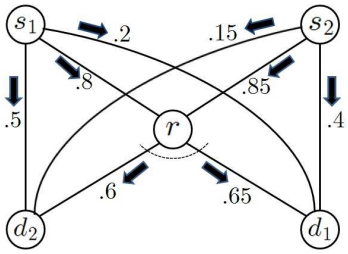


Fig. 4.2. An instance of the 2-flow wireless butterfly network with the success probabilities being indicated next to the corresponding arrows. We also assume that the success events between different node pairs are independent.

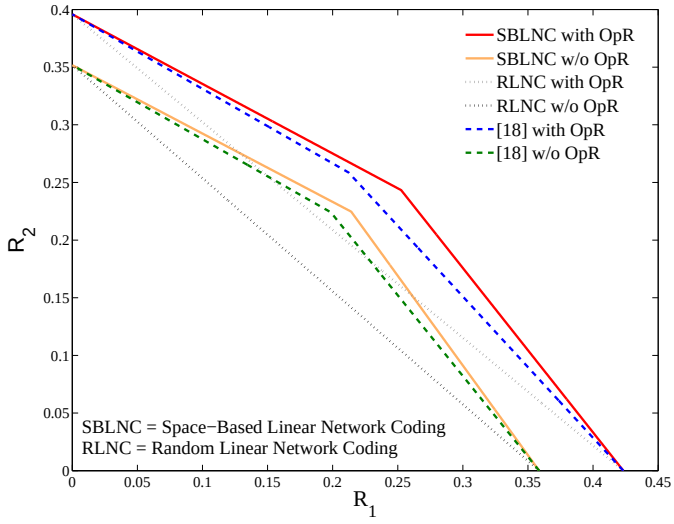
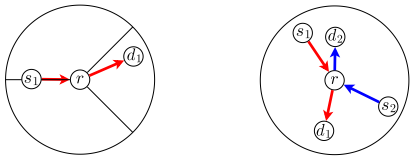


Fig. 4.3. The capacity regions of the scenario in Fig. 4.2. The solid line indicates the throughput region of SBLNC. The dash line indicates the throughput region in [4]. The dot line indicates the throughput region of intra-session network coding (or random linear network coding).



(a) (b)

Fig. 4.4. (a) The relative location of (s_i, d_i). (b) Topology of two (s_i, d_i) pairs.

by d_2 is $p_r(d_2) = .6$. We then compute 6 different capacity rate regions and plot them in Figure 4.3.

The solid line "SBLNC with OpR" represents the ultimate capacity region[4] of this network, for which relay r is allowed to perform inter-flow network coding across both flows and 2-hop overhearing directly from s_1 to d_1 (and from s_2 to d_2) is allowed. The curve "with COPE, w/o OpR" describes the capacity region when the relay r can perform inter-flow coding but there is no 2-hop overhearing. Both the curves "with COPE, with OpR" and "with COPE, w/o OpR" allow for optimal scheduling among s_1, s_2, and r. The dash line " [18] with (or w/o) OpR" represents the throughput region which can be achieved by the results in [18] with either opportunistic routing or not. The dot line "RLNC with (or w/o) OpR" represent the throughput region which can be achieved by simply the intra-session network coding (or random linear network coding [2]) with either opportunistic routing or not.

As can be seen, when there is only one flow in the network (say $R_2 = 0$), then OpR is optimal as was first established in [33]. However, when there are two coexisting flows (when both R_1 and $R_2 > 0$), the COPE principle alone sometimes outperforms OpR due to the strong overhearing between $s_2 \to d_1$ and $s_1 \to d_2$, $p_2(d_1) = 0.4$ and $p_1(d_2) = 0.5$, in this example. On the other hand, SBLNC provide the ultimate throughput when compared to the existing schemes.

[4]Our main results provide a pair of outer and inner bounds for this capacity region. Since the gap between the inner and outer bounds is not visible in the figure (with relative gap less than 0.08%), we use the inner bound (the achievable rate) as the proxy of the capacity region.

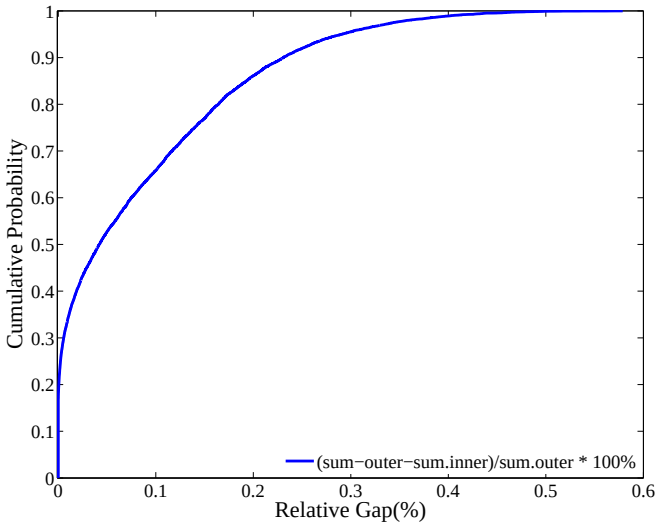


Fig. 4.5. The cumulative distribution of the relative gap between the outer and the inner bounds when there is no fairness constraint. The outer and the inner bounds are described in Propositions 4.2.2 and 4.2.3, respectively.

Table 4.2
Average sum-rates over 10000 random node replacements.

Fairness Constraints	OpR	SBLNC	[18]	RLNC
No	allowed	.6599/.6594	.6472	.6180
	negligible	.4820	.4779	.4116
Proportional	allowed	.6294/.6286	.6101	.5484
	negligible	.4775	.4726	.3854
Min-cut	allowed	.6031/.6026	.5892	.5406
	negligible	.4671	.4626	.3856

We are also interested in quantifying the throughput benefits of COPE and OpR in a randomly placed network. To generate a typical XOR-in-the-air scenario, we first place the relay node in the center of a unit circle. Then we randomly place four nodes (s_1, s_2, d_1, d_2) inside the unit circle. To simulate the need of the relay for each session pair, we force the placement of each pair to be in the opposite 90 degree area. That is, d_i must be located in the opposite 90 degree area of s_i's location for $i = 1, 2$. See Fig. 4.4(a) for illustration. Fig. 4.4(b) illustrates one realization of our random node placement.

We use the Euclidean distance D between any two nodes to decide the overhearing probability when a packet is transmitted. More explicitly, we use the Rayleigh model

$$\mathsf{Prob}(\text{success}) = \int_{T^*}^{\infty} \frac{2x}{\gamma} e^{-\frac{x^2}{\gamma}} dx \quad \text{where } \gamma \triangleq \frac{1}{(4\pi)^2 D^\alpha},$$

where α is the path loss factor, and T^* is the decodable SNR threshold. To reflect the packet delivery ratio measured in practical environments, we choose $\alpha = 2.5$ and $T^* = 0.006$ so that the overhearing probability for a 1-hop neighbor is around 0.7–0.8 while overhearing probability for a 2-hop neighbor is around 0.2–0.3. If no direct overhearing is allowed, we simply hardwire the probability that d_i overhears s_i to be zero. We again assume that the success events between different node pairs are independent.

For practical concerns, we often enforce fairness constraint to avoid the situation that one of the flows occupies all of the allocated resource. Here we propose two kinds of fairness constraint: the min-cut fairness constraint, and the proportional fairness constraint. For the min-cut fairness constraint, we impose an additional constraint $R_i = \beta \min\left(p_i(d_i, r), p_i(d_i) + p_r(d_i)\right)$ for $i = 1, 2$ with a common β, which enforces the individual rate R_i being proportional to the min-cut value from s_i to d_i assuming no other sessions are transmitting and s_i and r are scheduled with the same frequency. For the proportional fairness constraint, we use $\log(R_1) + \log(R_2)$ as the objective function in the linear programming solver. When there is no fairness constraint, we simply maximize the sum rate $R_1 + R_2$ as the objective function in the linear programming solver. Combining the fairness constraint and the linear constraints in Proposition 4.2.2 or Proposition 4.2.3, we derive the optimal rates. We repeat the above experiment for 10000 times and lists the average sum rate $R_1 + R_2$ for each case in Table 4.2.

Table 4.2 lists the sum-rate averaged over 10000 simulations. When allowing 2-hop overhearing ($p_i(d_i) > 0$), then the inner and outer bounds do not always meet. Therefore, for the entry with both COPE and OpR, the number on the left is the average of the sum of the optimal rate $R_1 + R_2$ for the outer bound, denoted by $R_{\text{sum.outer}}$, while the number on the right is the average of the sum of the optimal rate $R_1 + R_2$ for the inner bound, denoted by $R_{\text{sum.inner}}$. When hardwiring the 2-hop overhearing probability to zero, as was proven Section 4.3.2, the sum-rate outer and inner bounds always coincide and hence only one number is shown in that entry. We also list the sum-rate inner bound results in [18]. The capacity of pure routing and pure OpR [33] can be explicitly computed and therefore there is only one number in those entries as well. We first note that in terms of the averaged throughput, the difference between the outer and the inner bounds is around 0.08%. Among all 10000 instances, the largest absolute difference is with $R_{\text{sum.outer}} = 0.6409$ and $R_{\text{sum.inner}} = 0.6375$. The proposed bounds thus effectively bracket the capacity when combining the XOR-in-the-air and the opportunistic routing principle. Jointly using COPE and OpR SBLNC scheme provides 60% throughput improvement over the classic pure routing scheme with optimal scheduling.

Fig. 4.5 focuses on the relative gap per experiment when allowing for both COPE and OpR. Specifically, we compute the relative gap per each experiment, $\left(R_{\text{sum.outer}} - R_{\text{sum.inner}}\right)/R_{\text{sum.outer}}$ when there is no fairness constraint, and then plot the cumulative distribution function (cdf) for the relative gaps. We can see that with more than 80% of the experiments, the relative gap between the outer and inner bounds is smaller than 0.2%.

4.5 Chapter Summary

In this chapter, we discuss the capacity region of the local network formulated in Section 2.2. In Section 4.1, we compare the proposed model with existing results and demonstrate that the proposed model includes all the important features of the broadcast packet erasure channels, the COPE principle, and the opportunistic routing. In Section 4.2, we then characterize the full capacity region of the 2-flow wireless butterfly network without opportunistic routing, and propose the outer bound and the inner bound for the case with opportunistic routing. The SBLNC scheme is used to achieve the inner bound in Section 4.3. In Section 4.4, we use the numerical results to demonstrate how close the SBLNC scheme can approach the outer bound (and hence the optimal throughput) and the throughput provided by the SBLNC scheme strictly outperforms existing results.

5. LINEAR NETWORK CODING SCHEDULING AND THE ANALOGY TO STOCHASTIC PROCESSING NETWORK

Starting from this chapter, we will start the discussion of the stability region. We will first discuss the stability definitions rigorously. Then we discuss why the network coding cross multiple sessions is a critical issue to the existing store-and=forward based network stability analysis. A more general network model, called the stochastic processing network (SPN), is introduced to incorporate the inter-session network coding issue. We further discuss the scheduling algorithm which can stabilize SPN. Finally, we discuss what are the obstacle when applying SPN to the multi-flow wireless network problem of interest.

5.1 Stability Definitions

We have formulated the stability region problem in Section 2.3. Here we are going to formally define the stability. The following definitions and properties hold even when the queue length can possibly be negative (as we will see the "virtual queue length" in the deficit maximum weight scheduling algorithm).

Definition 5.1.1 *A queue length $q(t)$ is stable if*

$$\limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t} \mathsf{E}\{|q_(t)|\} < \infty. \tag{5.1}$$

And the network is stable if all the queues are stable.

Definition 5.1.2 *A queue length $q(t)$ is sublinearly stable/grows sublinearly if for any $\epsilon > 0$ and $\delta > 0$, there exists t_0 such that*

$$\mathsf{Prob}(|q(t)| > \epsilon t) < \delta,\ \forall t > t_0. \tag{5.2}$$

And the network is sublinearly stable if all the queues are sublinearly stable.

5.1.1 Properties

Lemma 5.1.1 *Suppose both queue lengths $q(t)$ and $p(t)$ are stable, then $q(t)+p(t)$ is also stable.*

Proof This follows from the subadditivity of $\limsup$,

$$\limsup_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t}\mathsf{E}\{|q(t)+p(t)|\}\le\limsup_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t}\mathsf{E}\{|q(t)|+|p(t)|\}$$
$$\le\limsup_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t}\mathsf{E}\{|q(t)|\}+\limsup_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t}\mathsf{E}\{|p(t)|\}<\infty.$$

■

Lemma 5.1.2 *Suppose both queue lengths $q(t)$ and $p(t)$ grow sublinearly. Then $q(t)+p(t)$ also grows subnlinearly.*

Proof For any $\epsilon>0$ and $\delta>0$,

$$\begin{aligned}&\mathsf{Prob}(|q(t)+p(t)|>\epsilon t)\le\mathsf{Prob}(|q(t)|+|p(t)|>\epsilon t)\\ \le&\mathsf{Prob}(|q(t)|>\epsilon t/2,\ |p(t)|>\epsilon t/2)\\ \le&\mathsf{Prob}(|q(t)|>\epsilon t/2)+\mathsf{Prob}(|p(t)|>\epsilon t/2).\end{aligned}$$

Since both $q(t)$ and $p(t)$ grow sublinearly, there exists t_1, t_2 such that

$$\begin{aligned}&\mathsf{Prob}(|q(t)+p(t)|>\epsilon t)\le\mathsf{Prob}(|q(t)|>\epsilon t/2)+\mathsf{Prob}(|p(t)|>\epsilon t/2)\\ &<\delta/2+\delta/2,\ \forall t>t_0\stackrel{\triangle}{=}\max\{t_1,t_2\}.\end{aligned}$$

■

Lemma 5.1.3 *Suppose the queue length $q(t)$ is stable with initial condition $q(0)=c<\infty$. If there exists a constant α such that $|q(t)-q(t-1)|<\alpha$ holds with probability 1 for all t, then $q(t)$ grows sublinearly.*

Proof Let $\Delta(t+1) = q(t+1) - q(t)$. We move $q(t)$ to the other side and square both side gives $q(t+1)^2 = q(t)^2 + 2q(t)\Delta(t+1) + \Delta(t+1)^2$. By definition, $|\Delta(t)| < \alpha$ with probability 1 for all t. And taking the expectation

$$\mathsf{E}\{q(t+1)^2\} - \mathsf{E}\{q(t)^2\} \leq 2\mathsf{E}\{|q(t)|\}\alpha + \alpha^2.$$

Since $q(t)$ is stable, $\frac{1}{t}\sum_{\tau=0}^{t-1}\mathsf{E}\{|q(t)|\}$ is bounded, say by U. Iteratively summing up both sides from $\tau = 0$ to $\tau = t-1$,

$$\frac{1}{t}\mathsf{E}\{q(t)^2\} \leq 2\alpha\frac{1}{t}\sum_{\tau=0}^{t-1}\mathsf{E}\{|q(t)|\} + \alpha^2 + \frac{c}{t} \leq 2\alpha U + \alpha^2 + c.$$

For any $\epsilon > 0$ and $\delta > 0$, we use the Markov inequality

$$\mathsf{Prob}(|q(t)| > \epsilon t) \leq \frac{1}{\epsilon^2 t^2}\mathsf{E}\{q(t)^2\} \leq \frac{2\alpha U + \alpha^2 + c}{\epsilon^2 t}.$$

Let t_0 be the smallest t such that $\frac{2\alpha U+\alpha^2+c}{\epsilon^2 t} < \delta$. Then

$$\mathsf{Prob}(|q(t)| > \epsilon t) < \delta,\ \forall t > t_0.$$

■

5.1.2 Achieve The Optimal Throughput By Sublinearly Stability

The stability definition is the one we usually have in the network stability analysis. The definition of sublinearly stable is not as often seen as the former one. We will use the following proposition to prove that the sublinear stability can help us to achieve the optimal throughput.

Proposition 5.1.1 *Suppose the network is sublinearly stable under the arrival rate R and there exists a naive solution such that the network can successfully send out all the packets remaining in the queues (assuming no further new incoming packets). Then the network can achieve the optimal throughput under the arrival rate R.*

Proof We assume the system is time-slotted, and the rate R is in the interior of the sublinearly stability region. We prove this by a frame-based scheme. Let $\{\Delta t_i\}_{i=1}^{\infty}$ be a sequence

of integer numbers denoting the length of the i-th frame. The first frame starts at $t = 1$ and ends at $t = \Delta t_1$. After the first frame ends, we start the first draining procedure at $t = \Delta t_1 + 1$. During the draining procedure, we buffer all the new incoming arrivals without injecting them into the network. At the same time, the network sends out all the packets remaining in the queues. The first draining procedure ends when all the queues in the network are empty, and we use Δt_{d_1} to denote the duration of the first draining procedure. After the first draining procedure (i.e., $t = \Delta t_1 + \Delta t_{d_1} + 1$), we start the second frame and inject all the packets buffered during the first draining procedure, $t = \Delta t_1 + 1$ to $t = \Delta t_1 + \Delta t_{d_1}$, into the network along with the original new arrivals with rate R. The second frame lasts for Δt_2. And then we repeat the draining procedure and so on so forth.

To summarize the above frame-based scheme, the i-th frame starts at $t = \sum_{j=1}^{i-1}(\Delta t_j + \Delta t_{d_j}) + 1$ with all queues being empty. The arrivals during the i-th frame come from two source. The first one is the original arrivals with rate R. The second one is the arrivals buffered at the previous draining procedure. And hence the overall rate during the i-th frame, denoted by R_i, is larger than R. The i-th frame lasts for Δt_i and ends at $t = \sum_{j=1}^{i-1}(\Delta t_j + \Delta t_{d_j}) + \Delta t_i$. And then we send out all the packets remaining in the network, which takes Δt_{d_i} time slots, by buffering all the new incoming packets. Then the $i+1$-th frame starts.

Given any $\delta > 0$ and $\epsilon > 0$. Suppose R_i is in the sublinear stability region for all i. Then all the queues are sublinearly stable during the i-th frame. Since the queue length at the end of the i-th frame is proportional to Δt_{d_i}, there exists t_i such that $\mathsf{Prob}(\Delta t_{d_i} < \epsilon \Delta t_i) > 1-\delta$ for all $\Delta t_i > t_i$. The overall throughput after i-th frame and its draining process, denoted by Γ_i, is

$$\begin{aligned}\Gamma_i =& \frac{R\left(\sum_{j=1}^{i-1}(\Delta t_j + \Delta t_{d_j}) + \Delta t_i\right)}{\sum_{j=1}^{i}(\Delta t_j + \Delta t_{d_j})} \\ =& R\left(1 - \frac{\Delta t_{d_i}}{\sum_{j=1}^{i}(\Delta t_j + \Delta t_{d_j})}\right).\end{aligned}$$

Then

$$\mathsf{Prob}(\Gamma_i > R(1-\epsilon)) > \mathsf{Prob}\left(\Gamma_i > R\left(1 - \frac{\epsilon \Delta t_i}{\sum_{j=1}^{i}(\Delta t_j + \Delta t_{d_j})}\right)\right) > 1 - \delta.$$

Thus the overall throughput can be made arbitrarily close to R with close-to-1 probability.

Hence it remains to show that there exists a sequence $\{\Delta t_i\}_{i=1}^{\infty}$ with $\Delta t_i > t_i$ such that R_i is in the sublinear stability region for all i. We prove this iteratively. $R_1 = R$ is in the interior of the sublinear stability region by the assumption at the beginning of the proof. We assume R_i is in the sublinear stability region. Notice that R_{i+1} comes from (1) the arrivals buffered at the previous draining procedure and (2) the original arrival rate R, and hence can be expressed as

$$R_{i+1} = \frac{R\Delta t_{i+1} + R\Delta t_{d_i}}{\Delta t_{i+1}} = R\left(1 + \frac{\Delta t_{d_i}}{\Delta t_{i+1}}\right).$$

Since R is in the interior of the sublinear stability region, there exists $\epsilon' > 0$ such that $R + \epsilon'$ is in the sublinear stability region. Furthermore, since R_i is in the sublinear stability region, all the queues are sublinearly stable during the i-th frame. By the fact that Δt_{d_i} is proportional to the maximum queue length at the end of the i-th frame, for any $\delta' > 0$ there exists t_i' such that $\mathsf{Prob}(\Delta t_{d_i} > \frac{\epsilon' t}{R}) < \delta'$ for all $t > t_i'$. We then choose $\Delta t_{i+1} = \max\{t_{i+1}, t_i'\} + 1$ and hence $\mathsf{Prob}(\Delta t_{d_i} > \frac{\epsilon' \Delta t_{i+1}}{R}) < \delta'$. That is,

$$\mathsf{Prob}\left(R\frac{\Delta t_{d_i}}{\Delta t_{i+1}} < \epsilon'\right) = \mathsf{Prob}\left(R_{i+1} = R\left(1 + \frac{\Delta t_{d_i}}{\Delta t_{i+1}}\right) < R + \epsilon'\right) > 1 - \delta'.$$

Since δ' can be made arbitrarily small with corresponding Δt_{i+1}, the proof is complete.

■

5.2 The Obstacles Between Store-and-Forward Network Control Algorithm and Network Coding

In the existing store-and-forward network stability analysis, a rate is feasible (namely, this rate is in the stability region) if there exists a network control algorithm which can stabilize all the queues inside the network considering all possible routing, scheduling, and resource allocation. At the same time, as we have seen in previous chapters, applying

network coding within the network can strictly increase the end-to-end throughput. Thus, intuitively speaking, the network coding should be able to enlarge the store-and-forward based stability region.

A standard network model usually consists of queues and links, and the packets can be stored in the queues and forwarded through links. The back-pressure scheduling algorithm [28] evaluates the differential queue length for each link and choose the feasible[1] scheduling decision which maximizes the summation of the differential queue lengths. The back-pressure algorithm has been proven to achieve the optimal throughput in the store-and-forward network model. However, the nature of combining packets in network coding is not a well-defined component in the standard store-and-forward network model. If the network coding is restricted within the same session (i.e., intra-session network coding), the network coding packets can be described as the information flow in the flow model [1]. The flow feature of intra-session thus can be naturally extended to the stability analysis as in [34] without considering the combining packet issue. However, when network coding can be applied across multiple sessions (i.e., inter-session network coding), the feature of information flow in intra-session network coding is no longer legitimate.

5.2.1 An Illustrative Example For The Combining Packet Issue

To illustrate the issue of combining packets more clearly, we again use 2-flow 1-to-2 broadcast packet erasure channel with feedback as an example, and choose the back-pressure algorithm as the scheduling algorithm. To expose the possible inter-session coding benefit, we use a 4-queue scheme as illustrated in Figure 5.1. For each $i, j = 1, 2$ and $j \neq i$, the queue q_i with queue length $Q_i(t)$ stores the uncoded packets from flow i which have not been received by any of d_1 and d_2. And the queue $q_{i,\text{SI}}$ with queue length $Q_{i,\text{SI}}(t)$ stores the uncoded packets from flow i which have been received by d_j but not by d_i. The "SI" in the subscript denotes the packet in the queue is the side information packet for the other destination. Hence suppose W_1 is in $q_{1,\text{SI}}$ and W_2 is in $q_{2,\text{SI}}$, then both destinations can

[1]The "feasible" here means there is no confliction between the scheduled links.

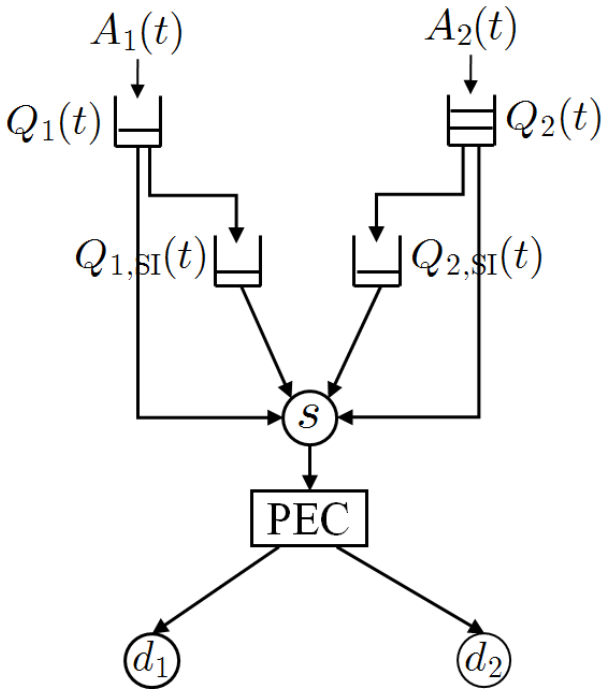


Fig. 5.1. A 4-queue scheme to illustrate the inter-session coding benefit.

decode the inter-session coded packet $W_1 + W_2$ and acquire the desired information packet. At each time slot t, we can choose to schedule one of the four links without inter-session network coding. With inter-session network coding, we can schedule both links, $q_{1,\text{SI}}$ to s and $q_{2,\text{SI}}$ to s, simultaneously and send out the coded packet through the PEC.

In the scenario where the inter-session network coding is prohibited, the back-pressure algorithm simply compares $Q_1(t) + Q_{1,\text{SI}}(t)$ versus $Q_2(t) + Q_{2,\text{SI}}(t)$ to decide which flow packet to be sent uncodedly. On the other hand, in the scenario where the inter-session network coding is legit, there are several possible ways to define the "back-pressure" value for the inter-session coding choice, which schedules $q_{1,\text{SI}}$ and $q_{2,\text{SI}}$ simultaneously. Our first thought will be $Q_{1,\text{SI}}(t) + Q_{2,\text{SI}}(t)$ and then the back-pressure algorithm chooses the maximum back-pressure value among $Q_1(t)$, $Q_2(t)$, and $Q_{1,\text{SI}}(t) + Q_{2,\text{SI}}(t)$ and schedule the corresponding queue(s). However, this creates a problem as one of $Q_{i,\text{SI}}(t)$ may be zero. Suppose $Q_{2,\text{SI}}(t)$ is empty but the back-pressure still chooses $Q_{1,\text{SI}}(t) + Q_{2,\text{SI}}(t)$ and schedules the inter-session coded packet. Then the "expected" inter-session coded packet would simply be a uncoded packet from flow 1, and hence the effective throughput turns to be less than the expected throughput. Suppose we change the definition of the back-pressure value to be $\min\{Q_{1,\text{SI}}, Q_{2,\text{SI}}\}$, then we circumvent this difficulty but the problem is that we now give too little priority to the inter-session network coding operation (inter-session network coding operation can serve both destinations simultaneously and should have higher priority).

As we have illustrated in the above description, the main difficulty is that we can schedule a inter-session coding operation only when both queues, $q_{1,\text{SI}}$ and $q_{2,\text{SI}}$, are non-empty. And this phenomenon is not carefully considered in the back-pressure algorithm for the store-and-forward network scheme. Consequently, a new network model setting which can incorporate the nature of combining packets is required. We also need a new stability analysis for this new network model setting.

5.3 The Stochastic Processing Network

In Section 5.2.1, we illustrated the difficulty of inter-session network coding being applied to the store-and-forward network scheme. To circumvent this difficulty, we introduce another network scheme, called the stochastic processing network (SPN).

SPN is a generalized version of the store-and-forward network. In the store-and-forward network, the packet that leaves the queue directly enters the next queue and one queue can be scheduled as long as there is one packet inside the queue. On the other hand, for SPN the most basic unit of scheduling is the so-called "service activity". Each service activity contains multiple queues. A service activity (SA) can be "chosen/activated/scheduled" only if all queues associated the SA are non-empty. Namely, an SA will take one packet from each associated queue, jointly "process" them, generate a set of new packets, and redistribute them to some output queues. The detailed definition of an SPN will be defined in the next subsection.

This generalization extensively widens the possible applications compared with the store-and-forward network. For example, in the video streaming problem [35], the content in the network consists of different types of data including the voice and the image. Different types of the data require different types of processing, e.g., compressing and decoding, in the network. And at the user-end, these different types of data require to be processed together to be a valid video stream. SPN has also been extended to the problems of the MapReduce scheduling [36] and grid computing [37].

Recall in the example in Section 5.2.1, we can schedule a inter-session coding operation only when both queue, $q_{1,\mathrm{SI}}$ and $q_{2,\mathrm{SI}}$, are non-empty. While this phenomenon is not well-defined in the store-and-forward network, it is resolved by the constraint that a SA in SPN can be activated only if all the queues associated the SA are non-empty. SPN thus inclines to a promising possible solution for the network coding scheduling problem.

However, the analysis of SPN is much more challenging than the regular store-and-forward network. Thus far, the state-of-the-art analysis results only consider *acyclic* SPN

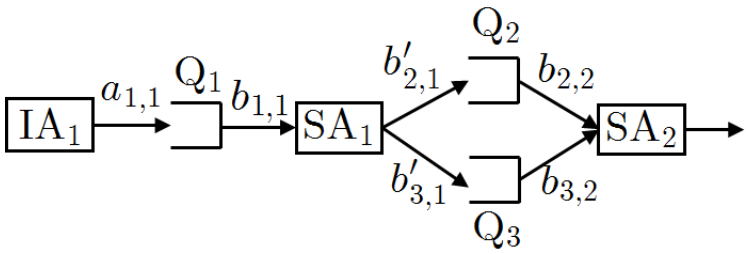


Fig. 5.2. An example of SPN.

with deterministic service[2] while in the network coding scenario, it is critical to consider SPNs that are both cyclic with random service. Detailed discussion of these challenges are provided in Section 5.4.

5.3.1 Definitions And The SPN Model

Assume a time-slotted system. The SPN definition here mainly follows [38]. A SPN consists of three components: the input activities (IA), the service activities (SA), and the queues. Let K be the number of queues, M be the number of IAs, and N be the number of SAs in the SPN. At each time slot t, a set of IAs and SAs can be scheduled (or activated). Each IA denotes a session (or flow) of packets and outputs packets to a collection of queues when activated. When one SA is activated, it takes packets from a set of queues, $\mathfrak{I}_n$, and sends packets to another set of queues, $\mathfrak{O}_n$.

When IA m is activated, it sends $a_{k,m}$ packets to queue k. Let $A \in \mathbb{R}^{K*M}$ be the "input matrix" with $A_{k,m} = -a_{k,m}$, for all m and k. When SA n is activated, it takes $b_{k,n}$ packets from queue $k \in \mathfrak{I}_n$ and send $b'_{k,n}$ packets to queue $k \in \mathfrak{O}_n$. Let $B \in \mathbb{R}^{K*N}$ be the "service matrix" with $B_{k,n} = b_{k,n}$ if $k \in \mathfrak{I}_n$ and $B_{k,n} = -b'_{k,n}$ if $k \in \mathfrak{O}_n$. We assume there is no cycle in SPN.

[2]The existing SPN analyses [38, 39] also consider the network with time-varying channel status (or service). However, the time-varying channel status is assumed to be a prior information for the scheduler at each time t. Thus the channel status is still "deterministic" for the scheduler in this sense. We name it "deterministic service" while "random service" denotes a scenario for which the channel status is not a prior information for the scheduler for any time t.

Let $\mathbf{a}(t) \in \{0,1\}^M$ be the "arrival vector" at time t. If $a_m(t) = 1$, then IA m is activated at time t. We assume $\mathbf{a}(t)$ is a random vector and i.i.d. over time with the average rate $\underline{R} = \mathsf{E}\{\mathbf{a}(t)\}$. Let $\mathbf{x}(t) \in \{0,1\}^N$ be the "service vector" at time t. If $\mathbf{x}_n(t) = 1$, then SA n is activated at time t. Let $\mathbf{s} \in \mathbb{R}^N$ be the vector of average service rate. Figure 5.2 illustrates an example of SPN.

Since there are some constraints between the activations of SAs, e.g., the transmission interference and user's preference, some of SAs can not be scheduled at the same time slot. We say $\mathbf{x}(t)$ is feasible if it satisfies all the constraints and $\mathfrak{X}$ is the set of all feasible $\mathbf{x}(t)$. Let Λ be the convex hull of $\mathfrak{X}$, i.e.[3],

$$\Lambda = \{\mathbf{s} : \exists \mathbf{p} \geq \mathbf{0}, \sum_{\mathbf{x}\in\mathfrak{X}} p_{\mathbf{x}} = 1, s = \sum_{\mathbf{x}\in\mathfrak{X}} (p_{\mathbf{x}}.\mathbf{x})\},$$

and let Λ^o be the interior of Λ.

Definition 5.3.1 *An arrival rate vector* $\mathbf{R}$ *is "feasible" if there exists* $\mathbf{s} \in \Lambda$ *such that* $A\cdot\mathbf{R}+B\cdot\mathbf{s} = 0$*; and* $\mathbf{R}$ *is "strictly feasible" if there exists* $\mathbf{s} \in \Lambda^o$ *such that* $A\cdot\mathbf{R}+B\cdot\mathbf{s} = 0$.

We call the above SPN an "SPN with deterministic processing" since only the arrival process $a(t)$ is random but the service matrix B is deterministic.

5.3.2 Deficit Maximum Weight Algorithm For Stabilizing SPN

Several attempts have been made to characterize the stability region of SPN and to achieve the optimal throughput. Here we focus on the deficit maximum weight (DMW) algorithm [38] due to its succinct analysis and the potential for further extensions. DMW algorithm can stabilize any strictly feasible rate vector and achieve the optimal throughput of SPN described in Section 5.3.1 [38]. We will describe DMW algorithm and conceptually explain how it works.

For each queue k, $Q_k(t)$ denotes its actual queue length at time t. Define a value $D_k(t) \geq 0$ as the "deficit" for the queue k at time t. $D_k(t) = Q_k(t) - q_k(t)$, where $Q_k(t)$

[3]We say a vector is no less than the other vecotr by element-wise comparison.

is the actual queue length, which can be measured directly by counting how many packets are actually in the queue, $q_k(t)$ is the virtual queue length, which started from $q_k(t) = 0$ for $t = 0$, and we will later discuss how to update the virtual queue length $q_k(t)$ over time. Initially, $\mathbf{q}(0) = \mathbf{Q}(0) = \mathbf{D}(0) = 0$. In the following algorithm, we first choose an optimized service activation vector according to the virtual queue length at the current time slot, we then update the virtual queue length for the next time slot. This can be done for each time slots with well-defined initial condition $\mathbf{q}(t) = 0$.

For each time t, we choose the service vector

$$\mathbf{x}^*(t) = \arg\max_{\mathbf{x}\in\mathfrak{X}} \mathbf{d}^T(t) \cdot \mathbf{x},$$

where $\mathbf{d}(t)$ is the back pressure vector defined as $\mathbf{d}(t) = B^T\mathbf{q}(t)$.

Suppose for some queues $k \in \mathfrak{I}_n$, the actual queue length $Q_k(t)$ is smaller than the amount of leaving packets when the SA n is scheduled, we name it "null activity/underflow." When the null activity occurs, the underflow queue generates "fictitious packets" and send it to SA n as if there are enough packets. We then update $\mathbf{q}(t)$ as follows.

$$\mathbf{q}(t+1) = \mathbf{q}(t) - A \cdot \mathbf{a}(t) - B \cdot \mathbf{x}^*(t).$$

We can also rewrite the above equation as

$$q_k(t+1) = q_k(t) - \mu_{out,k}(t) + \mu_{in,k}(t),\ \forall k, \tag{5.3}$$

where

$$\mu_{out,k}(t) = \sum_{n=1}^{N} B^+_{k,n} x^*_n(t),$$
$$\mu_{in,k}(t) = \sum_{n=1}^{N} A^-_{k,m} a_m(t) + \sum_{n=1}^{N} B^-_{k,n} x^*_n(t).$$

and $v^+ = \max\{0, v\}$ and $v^- = \max\{0, -v\}$.

We then update $\mathbf{q}(t)$ and $\mathbf{D}(t)$ as follows.

$$Q_k(t+1) = (Q_k(t) - \mu_{out,k}(t))^+ + \mu_{in,k}(t), \tag{5.4}$$
$$D_k(t+1) = Q_k(t+1) - q_k(t+1).$$

A heuristic but not rigorous explanation about why DMW works is as follows. In the back-pressure algorithm of the store-and-forward network, we always try to stabilize the actual queue length. And hence we choose the schedule decision which maximizes the back-pressure value based on the actual queue length. Notice that the actual queue length is always no less than zero as in (5.4). Thus as long as $\mu_{in,k}(t) \leq \mu_{out,k}(t)$ for most of time slots t, the actual queue length will be conceptually stable. And furthermore, the actual queue length is likely to touch zero frequently. However, in SPN, whenever the actual queue length touches zeros and it is in $\mathfrak{I}_n$ for some activated SA n, the underflow occurs. To avoid the underflow, $\mu_{in,k}(t) \leq \mu_{out,k}(t)$ for most of time slots t is way too aggressive. And hence we turn to stabilize the virtual queue length as updated in (5.3), which choose the schedule decision that maximizes the back-pressure value based on the virtual queue length. Notice that for the virtual queue length being stable, we need to have $\mu_{in,k}(t) = \mu_{out,k}(t)$ for most of time slots t. And in this case, we can avoid the occurrence of the underflow. Notice that $\mu_{in,k}(t) = \mu_{out,k}(t)$ implies $\mu_{in,k}(t) \leq \mu_{out,k}(t)$, and hence we can use DMW to stabilize the actual queue length and to avoid the occurrence of the underflow simultaneously.

5.4 The Obstacle Between Stochastic Processing Network And LNC Scheduling

As discussed in the previous sections, SPN seems to be a promising solution for the inter-session network coding problem. And DMW can help us to stabilize the problem of interest. However, there still exist the obstacle when applying SPN results to the SBLNC scheme.

The obstacle is the random service issue in the wireless network. In a wireless network, when a packet leaves a queue, it can randomly arrive some possible queues with certain probabilities. And before the packets actually arrive the queue and send the feedback back to the source, we have no ideas which queues actually receive the packet. We name this feature as the "random service" in SPN as the SA n might randomly serve several queues in $\mathfrak{O}_n$. We use Figure 5.2 as an example to illustrate this issue. We assume $a_{1,1} = b_{1,1} =$

$b_{2,2} = b_{3,2} = 1$ and IA 1 is activated at every time slot. There is no confliction between SA 1 and SA 2 and hence we can activate them at the same time slot. To model the random service, we assume there are two possible combinations of $(b'_{2,1}, b'_{3,1})$. $(b'_{2,1}, b'_{3,1}) = (1, 0)$ with probability 0.5 and $(b'_{2,1}, b'_{3,1}) = (0, 1)$ with probability 0.5. That is, whenever SA 1 is activated, it takes a packet from Q_1, and with probability 0.5 this packet goes to Q_2. Otherwise, this packet goes to Q_3. Meanwhile, whenever SA 2 is activated, it must take one packet from each of Q_2 and Q_3. Notice that the queue length difference between Q_2 and Q_3 forms a simple random walk. The analysis of the random walk thus shows that the difference between two queues, Q_2 and Q_3, goes unbounded with rate $\sqrt{t}$. And hence there is no scheduling algorithm which can bound both queue sizes.

We will circumvent this obstacle by properly modifying DMW algorithm in the next chapter.

5.5 Chapter Summary

In this chapter, we start to discuss the stability region. The definition of stability and sublinear stability are introduce in Section 5.1. We discuss their properties and prove that with certain conditions, the sublinear stability can achieve the optimal throughput in Section 5.1.2. There exists some difficulties when applying inter-session network coding scheduling problem to the existing store-and-forward network analysis. We carefully illustrate the difficulties in Section 5.2.1. A network model, SPN, which can possibly resolve the difficulties is introduced in Section 5.3. We introduce DMW algorithm, which is the stabilizing algorithm for SPN, in Section 5.3.2. We conclude this chapter by discussing the obstacles when applying SPN to the multi-flow wireless network problem.

6. DEFICIT MAXIMUM WEIGHT-BASED LINEAR NETWORK CODING

In this chapter, we will first device a new linear-space-based conversion method such that we can map the LNC design problem for the dynamic arrival setting to a scheduling problem of an SPN network with random service. Then we will propose a modified DMW scheduling algorithm that can stablize the SPN with random service. Finally, we will use the modified DMW on SPN to derive a LNC solution for dynamic packet arrival such that it is guaranteed to achieve the optimal throughput (since the modified DMW is guaranteed to achieve the largest possible stability region of the SPN with random service.)

6.1 Converting The NC With Dynamic Arrival To An Space-Based LNC Scheduling Problem

In this chapter we exclusively focus on the stability problem of the 2-user broadcast PEC with feedback (or 2-flow 1-to-2 broadcast PEC with feedback).

The full Shannon feedback capacity of the 2-user broadcast PEC was first characterized by [3] in 2009 by a three-stage scheme. We use Figure 2.1(a) to illustrate this three-stage scheme. In the first stage, the source s sends out the flow-1 packets uncodedly until each of them is received by either d_1 or d_2. Suppose a flow-1 packet is received by d_2 but not by d_1, then we put this packet into a queue $q_{1,\text{SI}}$. The second stage is symmetric to the first stage. That is, the source s sends out flow-1 packets uncodedly until each of them is received by either d_1 or d_2. Suppose a flow-1 packet is received by d_1 but not by d_2, then we put this packet into a queue $q_{2,\text{SI}}$. In the last stage, the source s take one packet W_1 from $q_{1,\text{SI}}$ and one packet W_2 from $q_{2,\text{SI}}$, and then send out $W_1 + W_2$. Suppose $q_{1,\text{SI}}$ is empty, then s send a packet W_2 in $q_{2,\text{SI}}$ and vice versa. For $i, j = 1, 2$ and $i \neq j$, since each packet in $q_{i,\text{SI}}$ is overheard by another destination d_j, both destinations d_1 and d_2 can recover $W_1 + W_2$ sent

in Stage 3 correctly. This scheme is devised as a block-code. Namely, we first need to send out *all* uncoded $\mathbf{W}_1$ packets until each is received by at least one destination and then send out*all* $\mathbf{W}_2$ packets until each is received by at least one destination.

When we have dynamic arrival, we actually have 5 different LNC choice one can possibly make in each time slot. Choice 1: Send a new $\mathbf{W}_1$ packet that have not been heard by any of d_1 and d_2; Choice 2: Send a new $\mathbf{W}_2$ packet that have not been heard by any of d_1 and d_2; Choice 3: When both $Q_{1,\text{SI}}$ and $Q_{2,\text{SI}}$ are non-emtpy, send a linear sum of the two; Choice 4: Send a packet from $Q_{1,\text{SI}}$ directly (without mixing with any $Q_{2,\text{SI}}$ packets probably because $Q_{2,\text{SI}}$ is empty); and Choice 5: Send a packet from $Q_{2,\text{SI}}$ directly (without mixing with any $Q_{1,\text{SI}}$ packets probably because $Q_{1,\text{SI}}$ is empty).

A LNC scheme with dynamic arrivals need to balance these 5 choices in an optimal way. On the other hand, from a block-code setting, this 3-stage scheme has been proven to achieve the optimal throughput for the 2-user broadcast PEC with feedback [3]. Following the results in our group [40], we can write the above three-stage scheme as a space-base LNC scheme. This space-based LNC scheme is based on a 5-type coding solution, which will be elaborated in Section 6.1.1.

6.1.1 The 5-Type Coding Scheme

We consider the model formulated in Section 2.3. In the following discussion, we will see that the 5 choices in the above 3-staged scheme can be converted to a scheme of 5 different types. For two linear spaces A and B, $A \oplus B$ is the sum space defined as $A \oplus B = \mathsf{span}\{\mathbf{v} : \forall \mathbf{v} \in A \cup B\}$. For $i = 1, 2$, let $\mathbf{W}_{i,\text{overall}} = (W_{i,1}, W_{i,2}, ...)$ with $W_{i,j} \in \mathsf{GF}(q)$ for all j be an infinite-dimensional vector with each coordinate being a flow-i message in queue $Q_{i,\text{overall}}$. Let $\mathbf{W}_{\text{overall}} \triangleq (W_{1,1}, W_{2,1}, W_{1,2}, W_{2,2}, ...)$ be the joint overall message vector with odd coordinates being a flow-1 message in queue $Q_{1,\text{overall}}$ and even coordinates being a flow-2 message in queue $Q_{2,\text{overall}}$. Let δ_j denote an infinite-dimensional elementary delta row vector with its j-th coordinate being one and all others being zero. We define $\Omega_{\text{overall}} \triangleq \mathsf{span}\{\delta_j : \forall j\}$ as the "overall message space." We also define $\Omega_{1,\text{overall}} \triangleq \mathsf{span}\{\delta_j :$

j is odd.$\}$ as the "overall flow-1 message space," and $\Omega_{2,\text{overall}} \triangleq \mathsf{span}\{\delta_j : j \text{ is even.}\}$ as the "overall flow-2 message space."

For any time t, notice that there are $\sum_{\tau=1}^{t} A_i(\tau)$ flow-i messages in queue $Q_{i,\text{revealed}}$ for $i = 1, 2$. We define the "revealed flow-1 message space at time t" as $\Omega_1(t) \triangleq \mathsf{span}\{\delta_{2j-1} : 1 \leq j \leq \sum_{\tau=1}^{t} A_1(\tau)\}$ and the "revealed flow-2 message space at time t" as $\Omega_2(t) \triangleq \mathsf{span}\{\delta_{2j} : 1 \leq j \leq \sum_{\tau=1}^{t} A_2(\tau)\}$. We define the "revealed message space at time t" as $\Omega(t) \triangleq \Omega_1(t) \oplus \Omega_2(t)$.

We define a coding vector $\mathbf{v}(t)$ to be a infinite-dimensional row vector with each coordinate being a scalar in $\mathsf{GF}(q)$, and the j-th coordinate being zero for any $j > \sum_{i=1,2} \sum_{\tau=1}^{t} A_i(\tau)$ such that $\mathbf{v}(t) \in \Omega_(t)$. Any linear combination of the message symbols in queue $Q_{1,\text{revealed}}$ and $Q_{2,\text{revealed}}$ at time t can thus be represented by $\mathbf{v}(t)\mathbf{W}^T_{\text{overall}}$, where $\mathbf{W}^T_{\text{overall}}$ is the transpose of $\mathbf{W}_{\text{overall}}$. For $i = 1, 2$, we define the knowledge space at d_i at the end of time t as $\Psi_i(t) \triangleq \mathsf{span}\{\mathbf{v}(\tau) : \forall \tau \leq t \text{ and } \mathbf{v}(\tau)\mathbf{W}^T_{\text{overall}} \text{ is received by } d_i\}$. We omit the time index when there is no ambiguity. Let

$$
\begin{aligned}
&A_1 \triangleq \Psi_1;\ A_2 \triangleq \Psi_2 \\
&A_3 \triangleq \Psi_1 \oplus \Omega_1;\ A_4 \triangleq \Psi_2 \oplus \Omega_2;\ A_5 \triangleq \Psi_1 \oplus \Psi_2; \\
&A_6 \triangleq \Psi_1 \oplus \Psi_2 \oplus \Omega_1;\ A_7 \triangleq \Psi_1 \oplus \Psi_2 \oplus \Omega_2;
\end{aligned}
$$

Each coding type is indexed by a 7-bit string $\mathbf{b} = b_1 b_2 ... b_7$ where each b_k indicates the coding vector $\mathbf{v}(t)$ is in A_k or not. For example, $\mathsf{TYPE}_{31} = \mathsf{TYPE}_{0011111}$ is the set of coding vectors, which are in $A_3 \cap A_4 \cap A_5 \cap A_6 \cap A_7$ but not in any of A_1 or A_2. That is,

$$
\begin{aligned}
\mathsf{TYPE}_{31} =& \mathsf{TYPE}_{0011111} \\
\triangleq& (A_3 \cap A_4 \cap A_5 \cap A_6 \cap A_7) \backslash (A_1 \cup A_2) \\
=& (A_3 \cap A_4 \cap A_5) \backslash (A_1 \cup A_2). \qquad (6.1)
\end{aligned}
$$

We say a coding type is "feasible" if the corresponding set is non-empty. At each time slot t, we choose a feasible coding type and randomly select one coding vector $\mathbf{v}(t)$ from the chosen coding type. And then send out the packet $\mathbf{v}(t)\mathbf{W}^T$.

Table 6.1
The relationship between the coding types, the required associated conditions, and the associated SA in the equivalent SPN, Figure 6.1.

Coding type	Required conditions
TYPE_9	Condition 2
TYPE_{18}	Condition 1
TYPE_{31}	Condition 3 and 4
TYPE_{63}	Condition 3
TYPE_{95}	Condition 4

In the case of 2-user broadcast PEC with feedback, we consider the following 5 types TYPE_9, TYPE_{18}, TYPE_{31}, TYPE_{63}, and TYPE_{95}. We also have 4 conditions.

1. $\mathrm{Rank}(\Omega) - \mathrm{Rank}(A_7) > 0.$ (6.2)
2. $\mathrm{Rank}(\Omega) - \mathrm{Rank}(A_6) > 0.$ (6.3)
3. $\mathrm{Rank}(A_5) + \mathrm{Rank}(A_3) - \mathrm{Rank}(A_6) - \mathrm{Rank}(A_1) > 0.$ (6.4)
4. $\mathrm{Rank}(A_5) + \mathrm{Rank}(A_4) - \mathrm{Rank}(A_7) - \mathrm{Rank}(A_2) > 0.$ (6.5)

It is proven in [40] that each coding type is feasible if and only if the associated condition(s) is (are) true. For example, for type 31 to be non-empty, we need condition 3 and 4 are both true. The relationship between the coding types and the conditions are listed in Table 6.1.

6.1.2 Connections Between 3-Stage Scheme And 5-Type LNC Scheme

Having introduced the 5-type LNC scheme, here we demonstrate that the 3-stage scheme and 5-type LNC scheme are eventually equivalent. Since the three-stage scheme has been proven to achieve the optimal throughput [3], it is enough to show that the throughput of the 5-type LNC scheme is no less than the throughput of the three-stage scheme. To show this statement, suppose we have two identical 2-user broadcast PECs with feedback with same arrivals, named PEC 1 and PEC 2. Whenever we send out a packet in the first stage

in PEC 1, we also encode a packet with TYPE_{18} and send it out in PEC 2. Whenever we send out a packet in the second stage in PEC 1, we also encode a packet with TYPE_{9} and send it out in PEC 2. Whenever we send out a packet in the last stage with both $q_{1,\mathrm{SI}}$ and $q_{2,\mathrm{SI}}$ are non-empty, we encode a packet with TYPE_{31} and send it out in PEC 2. If in the last stage $q_{2,2}$ is empty, then we encode a packet with TYPE_{63} and send it out in PEC 2. If in the last stage $q_{1,\mathrm{SI}}$ is empty, then we encode a packet with TYPE_{95} and send it out in PEC 2. Finally, we have the following three claims to complete the statement.

We first claim that for any packet sent in the first stage, its corresponding coding vector, $\mathbf{v}(t)$, is in TYPE_{18}. That is, if the first stage is scheduled in PEC 1, then TYPE_{18} is also feasible in PEC 2. In the first stage, the source s sends out flow-1 packets until each of them is received by any of d_1 and d_2. And hence the packet sent in the first stage is in flow-1 but not known by any of d_1 and d_2. We then check whether $\mathbf{v}(t)$ is in A_i or not for $i = 1, 2, ..., 7$. Since the packet is neither in any of d_1 and d_2, nor a flow-2 packet, $\mathbf{v}(t)$ is in A_3, A_6 but not in A_1, A_2, A_4, A_5, nor A_7, which is in $\mathsf{TYPE}_{0010010} = \mathsf{TYPE}_{18}$. And hence $\mathbf{v}(t)$ is in TYPE_{18}.

We then claim that for any packet sent in the second stage, its corresponding coding vector, $\mathbf{v}(t)$, is in TYPE_{9}. By symmetric analysis as the first claim, the packet sent in the second stage is neither in any of d_1 and d_2, nor a flow-1 packet, and hence $\mathbf{v}(t)$ is in A_4 and A_7, but not in A_1, A_2, A_3, A_5, nor A_6, which is in $\mathsf{TYPE}_{0001001} = \mathsf{TYPE}_{9}$. Thus $\mathbf{v}(t)$ is in TYPE_{18}.

We finally claim that for any packet sent in the last stage, its corresponding coding vector, $\mathbf{v}(t)$, is in one of TYPE_{31}, TYPE_{63}, and TYPE_{95}. There are three cases in the last stage. The first case is both $q_{1,\mathrm{SI}}$ and $q_{2,\mathrm{SI}}$ are non-empty, and then s sends the packet $W_1 + W_2$ where W_i is from queue $q_{i,\mathrm{SI}}$. Since W_1 is in $q_{1,\mathrm{SI}}$, it is known by d_2. On the other hand, W_2 is a flow-2 packet. And hence $\mathbf{v}(t)$ is in the sum space $\Psi_2 \oplus \Omega_2$. Symmetrically, W_2 is in $q_{2,\mathrm{SI}}$ and W_1 is a flow-1 packet. Hence $\mathbf{v}(t)$ is also in the sum space $\Psi_1 \oplus \Omega_1$. Furthermore, since W_1 is known by d_2 and W_2 is known by d_1, $\mathbf{v}(t)$ is also in the sum space $\Psi_1 \oplus \Psi_2$. In summary, $\mathbf{v}(t)$ is not in A_1 nor A_2 but in rest of them, which is in $\mathsf{TYPE}_{0011111} = \mathsf{TYPE}_{31}$.

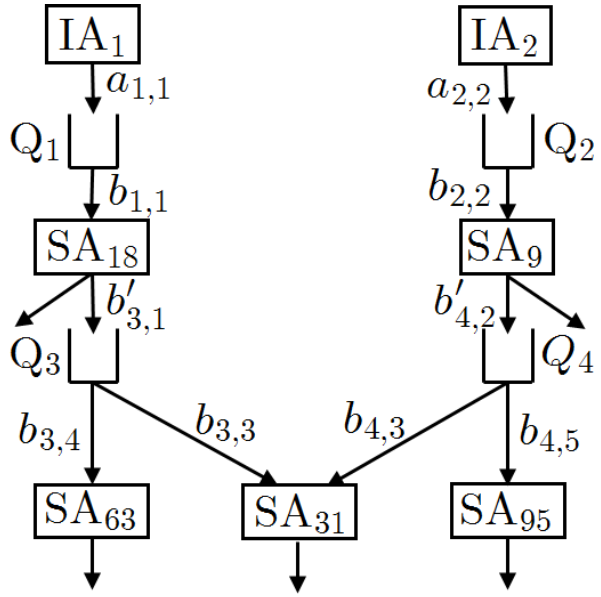


Fig. 6.1. The equivalent SPN of the 5-type space-based LNC scheme

The second case of the last stage is $q_{2,\mathrm{SI}}$ is empty. Then s sends W_1 in queue $q_{1,\mathrm{SI}}$. Notice that W_1 is a flow-1 packet and is known by d_2. Hence the corresponding coding vector, $\mathbf{v}(t)$, is in A_2, A_3, A_4, A_5, A_6, and A_7 but not in A_1, which is in $\mathsf{TYPE}_{0111111} = \mathsf{TYPE}_{63}$. Symmetric to the second case, the third case is $q_{1,\mathrm{SI}}$ is empty. Then s send W_2 in queue $q_{2,\mathrm{SI}}$. And hence the corresponding coding vector is in A_1, A_3, A_4, A_5, A_6, and A_7 but not in A_2, which is in $\mathsf{TYPE}_{1011111} = \mathsf{TYPE}_{95}$.

Combining the above three claims, the throughput of the 5-type LNC scheme is no less than the three-stage scheme in [3], and hence the 5-type LNC scheme achieves the optimal throughput of 2-user broadcast PEC with feedback.

6.1.3 Random Service In Space-Based LNC Scheme

The above 5-type space-based LNC scheme can modeled as an SPN. Figure 6.1 illustrates the equivalent SPN. IA 1 is activated whenever a flow-1 packet is injected into the network, and IA 2 is activated whenever a flow-2 packet is injected into the network.

For example SA_9 corresponds to coding type 9, SA_{18} corresponds to coding type 18. Q_1 corresponds to condition 1, Q_2 corresponds to condition 2, and so on so forth. Since we can schedule type 31 if both Conditions 3 and 4 hold. Similarly, in the SPN network we can schedule SA_{31} if and only if both Q_3 and Q_4 are non-empty. As will be seen shortly after, the service in this SPN is random and we thus need to devise a new algorithm that can stabilize SPNs with random service. The detailed parallelism between the equivalent SPN and the 5-type spaced-based LNC scheme will be provided in Section 6.3 with proper service rate assignments. Each service activity corresponds to a coding type.

6.2 Modified DMW Scheduling For SPN With Random Service

As can be seen that the to properly schedule the LNC solutions, we need to take into account SPNs with random service. In the following, we first propose a modified DMW method that can stabilize SPNs with random service and then we discuss how to design an optimal LNC scheduling solution based on the new stability results on the SPN with random service.

To resolve the random service issue, we first extend the SPN with deterministic service in Section 5.3.1 to the one with random service. We then modify the DMW scheduling algorithm to stabilize the SPN with random service and to achieve the optimal throughput. Even though the following results are motivated by the LNC design problem with dynamic arrival, throughout our discussion, we focus exclusively on the SPN scheduling problem. Its detailed connection to the LNC problem will be discussed in Section 6.3.

6.2.1 The SPN Model With Random Service

We consider two network models, name network model 1 (NM 1) and network model 2 (NM 2).

NM 1 is an SPN model with deterministic service as defined in Section 5.3.1 with arrival vector $\mathbf{a}(t)$ and the arrival rate $\mathbf{R} = \mathsf{E}\{\mathbf{a}(t)\}$. Let K be the number of queues, M be the number of IAs, and N be the number of SAs in NM 1. We use $\overline{B}$ to denote the service

matrix of NM 1 with each entry denoted by $\overline{B_{k,n}}$. And $\mathfrak{X}$ is the set of all feasible activation vector $\mathbf{x}(t)$ in NM 1.

NM 2 is a network model with the same topology (i.e. same relationship between SAs, IAs, and queues), the same arrival vector $\mathbf{a}(t)$, and the same activation set $\mathfrak{X}$ as in NM 1. That is, they share same queues, service activities, input activities, and the links. Furthermore, whenever the input activity IA m is activated in NM 1, the corresponding IA m in NM 2 will also simultaneously be activated. The only difference between NM 1 and NM 2 is that we have the random service for each SA in NM 2. We define the random service rigorously as follows.

Let $B(t)$ be the service matrix in NM 2 at time t. $B(t)$ is a random matrix with each entry is a bounded random variable and is i.i.d. over time. If $k \in \mathfrak{I}_n$, then $B_{k,n}(t)$ is a non-negative random variable. If $k \in \mathfrak{O}_n$, then $B_{k,n}(t)$ is a non-positive random variable. Otherwise, $B_{k,n}(t) = 0$ with probability 1. We further assume that $\mathsf{E}\{B_{k,n}(t)\} = \overline{B_{k,n}}$ for all k and n. We use $\mathbf{b}(t)$ to denote the realization of $B(t)$ and $b_{k,n}(t)$ to denote the realization of $B_{k,n}(t)$.

Remark: Even though we construct the SPN with deterministic service first then extend to the SPN with random service. However, when the problem of interest is an SPN with random service, we will first construct NM 1, the SPN with deterministic service, by taking the average of the random service matrix. We will then use the new NM 1 to help us schedule the random SPN of interest, NM 2.

6.2.2 The Scheduling Algorithm

Let $\mathbf{q}^{(1)}(t)$ to be the virtual queue length in NM 1 (VQ 1) and $\mathbf{q}^{(2)}(t)$ to be the virtual queue length in NM2 (VQ 2). Similarly, $\mathbf{Q}^{(1)}(t)$ and $\mathbf{D}^{(1)}(t)$ are the actual queue length and the deficit in NM 1. And $\mathbf{Q}^{(2)}(t)$ and $\mathbf{D}^{(2)}(t)$ are the actual queue length and the deficit in NM 2. We assume the initial conditions to zero for all the queues, virtual queues, and deficit in both NM 1 and NM 2. We describe the update rules of $\mathbf{q}^{(1)}(t)$, $\mathbf{q}^{(2)}(t)$, $\mathbf{Q}^{(1)}(t)$, $\mathbf{Q}^{(2)}(t)$, $\mathbf{D}^{(1)}(t)$, and $\mathbf{D}^{(2)}(t)$ as follows.

Since NW 1 and NW 2 share the same arrivals process $\mathbf{a}(t)$, we apply the DMW scheduling algorithm on NM 1 and utilize the same scheduling decision $\mathbf{x}^*(t)$ on NM 2. That is, $\mathbf{x}^*(t)$ is the scheduling decision for both NM 1 and NM 2 at time t and

$$\mathbf{x}^*(t) = \arg\max_{\mathbf{x}\in\mathfrak{X}} \mathbf{d}^T(t)\cdot\mathbf{x}, \tag{6.6}$$

where $\mathbf{d} = \overline{B}^T\mathbf{q}^{(1)}(t)$ is the back-pressure vector in NM 1[1]. We update $\mathbf{q}^{(1)}(t)$ as follows.

$$\mathbf{q}^{(1)}(t+1) = \mathbf{q}^{(1)}(t) - A\cdot\mathbf{a}(t) - \overline{B}\cdot\mathbf{x}^*(t). \tag{6.7}$$

(6.7) can also be rewritten as

$$q_k{}^{(1)}(t+1) = q_k{}^{(1)}(t) - \overline{\mu_{out,k}}(t) + \overline{\mu_{in,k}}(t),\ \forall k, \tag{6.8}$$

where

$$\overline{\mu_{out,k}}(t) = \sum_{n=1}^{N}\left(\overline{B_{k,n}}^{+}x_n^*(t)\right),$$
$$\overline{\mu_{in,k}}(t) = \sum_{m=1}^{M}\left(A_{k,m}^{-}a_m(t)\right) + \sum_{n=1}^{N}\left(\overline{B_{k,n}}^{-}x_n^*(t)\right).$$

We also update $\mathbf{q}^{(2)}(t)$ as

$$\mathbf{q}^{(2)}(t+1) = \mathbf{q}^{(2)}(t) - A\cdot\mathbf{a}(t) - B(t)\cdot\mathbf{x}^*(t). \tag{6.9}$$

Similarly, we can rewrite (6.9) as

$$q_k{}^{(2)}(t+1) = q_k{}^{(2)}(t) - \mu_{out,k}(t) + \mu_{in,k}(t),\ \forall k, \tag{6.10}$$

where

$$\mu_{out,k}(t) = \sum_{n=1}^{N}\left(b_{k,n}(t)^{+}x_n^*(t)\right) \tag{6.11}$$
$$\mu_{in,k}(t) = \sum_{m=1}^{M}\left(A_{k,m}^{-}a_m(t)\right) + \sum_{n=1}^{N}\left(b_{k,n}(t)^{-}x_n^*(t)\right).$$

We then update the actual queue length $\mathbf{Q}^{(2)}(t)$ and deficits $\mathbf{D}^{(2)}(t)$ in NM 2. For all k,

$$Q_k{}^{(2)}(t+1) = \left(Q_k{}^{(2)}(t) - \mu_{out,k}(t)\right)^{+} + \mu_{in,k}(t), \tag{6.12}$$

$$D_k{}^{(2)}(t+1) = Q_k{}^{(2)}(t+1) - q_k{}^{(2)}(t+1). \tag{6.13}$$

[1]There is only finite number of service activities and hence $\mathfrak{X}$ is a finite set.

6.2.3 Properties Of The Virtual Queue Length And The Deficit

Lemma 6.2.1 *Given past arrival vectors, VQ 1* $\mathbf{q}^{(1)}(t)$ *is the expectation of VQ 2* $\mathbf{q}^{(2)}(t)$. *That is,* $\mathbf{q}^{(1)}(t) = \mathsf{E}\{\mathbf{q}^{(2)}(t)|\{\mathbf{a}(\tau)\}_{\tau=1}^{t}\}$.

Proof Notice that given the past arrival vectors $[\mathbf{a}(\tau)]_{\tau=1}^{t}$, $\mathbf{q}^{(1)}(t)$ becomes deterministic and so does $\mathbf{x}^*(t)$. This property thus is a consequence of (6.7), (6.9), and can be proven iteratively. ■

Remark: Since $\mathbf{q}^{(1)}(t)$ is the conditional expectation of $\mathbf{q}^{(2)}(t)$, we slightly abuse the notation in the following discussions. We use $\overline{\mathbf{q}}(t) \stackrel{\triangle}{=} \mathbf{q}^{(1)}(t)$ and drop all the superscript "(2)" for the quantities/variables in NM 2[2].

Lemma 6.2.2 $D_k(t)$ *is non-decreasing with* t, *and satisfies*

$$D_k(t+1) = D_k(t) + (\mu_{out,k}(t) - Q_k(t))^+ . \tag{6.14}$$

Proof This property is from [38]. Following the same derivation in [38] and by definitions and updating rules, we have

$$\begin{aligned} D_k(t+1) &= Q_k(t+1) - q_k(t+1) \\ &= (Q_k(t) - \mu_{out,k}(t))^+ - (q_k(t) - \mu_{out,k}(t)) \\ &= Q_k(t) - \mu_{out,k}(t) + (\mu_{out,k}(t) - Q_k(t))^+ - (q_k(t) - \mu_{out,k}(t)) \\ &= D_k(t) + (\mu_{out,k}(t) - Q_k(t))^+ . \end{aligned}$$

■

6.2.4 The Stability Analysis

The following lemmas show that the proposed scheduling algorithm in Section 6.2.2 can stabilize NM 2. Recall that Λ is the convex hull of $\mathfrak{X}$ and Λ^o is the interior of Λ.

[2]Since we never use $\mathbf{Q}^{(1)}(t)$ and $\mathbf{D}^{(1)}(t)$ in the following analysis, we do not apply the notation changes on them.

Lemma 6.2.3 *(The necessary condition for NM 1 stability) For any arrival rate* $\mathbf{R}$*, if NM 1 can be stabilized, then* $A \cdot \mathbf{R} + \overline{B} \cdot y = 0$ *for some* $y \in \Lambda$.

Proof Since NM 1 is an SPN with deterministic service, this follows the results in [38]. ■

Lemma 6.2.4 *(The necessary condition for NM 2 stability) For any arrival rate* $\mathbf{R}$*, if NM 2 can be stabilized, then* $A \cdot \mathbf{R} + \overline{B} \cdot y = 0$ *for some* $y \in \Lambda$.

Proof To prove this lemma, it is enough to show that if $\mathbf{q}(t)$ is stable, then $\overline{\mathbf{q}}(t)$ is stable. By Lemma 6.2.1, if $\mathbf{q}(t)$ is stable, then we take the conditional expectations and derive that $\overline{\mathbf{q}}(t)$ is also stable. ■

Lemma 6.2.5 *(The sufficient condition for the NM 1 stability) If the arrival rate* $\mathbf{R}$ *is strictly feasible as defined in Section 5.3.1, then VQ 1* $\overline{\mathbf{q}}(t)$ *can be stabilized. That is, for all* k*,*

$$\limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t} \mathsf{E}\{|\overline{q_k}(t)|\} < \infty,$$

where the expectation is taken over all possible arrival vectors.

Proof Notice that NM 1 is an SPN with deterministic service, and we apply DMW scheduling algorithm on NM 1. This lemma is a direct consequence of the results in [38]. ■

Lemma 6.2.6 *(The sufficient condition for the NM 2 stability) If the arrival rate* $\mathbf{R}$ *is strictly feasible, then NM 2 can be sublinearly stabilized.*

Proof To prove this lemma, it is enough to show that if $\overline{\mathbf{q}}(t)$ is stable, then $\mathbf{q}(t)$ is sublinearly stable. For any $k \in \{1, 2, ..., K\}$, $\epsilon > 0$, and $\delta > 0$, we square both sides of (6.10)

$$q_k(t+1)^2 - q_k(t)^2 = (\mu_{out,k}(t) - \mu_{in,k}(t))^2 - 2q_k(t)\left(\mu_{out,k}(t) - \mu_{in,k}(t)\right).$$

Conditioning on the arrival vectors, we take the conditional expectation on both sides.

$$\begin{aligned}
&\mathsf{E}\{q_k(t+1)^2|\{\mathbf{a}(\tau)\}_{\tau=1}^t\} - \mathsf{E}\{q_k(t)^2|\{\mathbf{a}(\tau)\}_{\tau=1}^t\} \\
=&\mathsf{E}\{(\mu_{out,k}(t) - \mu_{in,k}(t))^2|\{\mathbf{a}(\tau)\}_{\tau=1}^t\} - 2\mathsf{E}\{q_k(t)\left(\mu_{out,k}(t) - \mu_{in,k}(t)\right)|\{\mathbf{a}(\tau)\}_{\tau=1}^t\} \\
=&\mathsf{E}\{(\mu_{out,k}(t) - \mu_{in,k}(t))^2|\{\mathbf{a}(\tau)\}_{\tau=1}^t\} - 2\overline{q_k}(t)\left(\overline{\mu_{out,k}}(t) - \overline{\mu_{in,k}}(t)\right) \qquad (6.15) \\
\leq& C^2 + 2|\overline{q_k}(t)|U, \qquad (6.16)
\end{aligned}$$

where (6.15) follows from the independence between $q_k(t)$ and $(\overline{\mu_{out,k}}(t) - \overline{\mu_{in,k}}(t))$ conditioned on the arrival vectors[3]; and (6.16) follows from defining C to be the upper bound of $|\mu_{out,k}(t) - \mu_{in,k}(t)|$ and U to be the upper bound[4] of $|\overline{\mu_{out,k}}(t) - \overline{\mu_{in,k}}(t)|$. Now we take the expectation over all arrival vectors,

$$\mathsf{E}\{q_k(t+1)^2\} - \mathsf{E}\{q_k(t)^2\} \leq C^2 + 2U\mathsf{E}\{|\overline{q_k}(t)|\}$$

Iteratively summing up the above equation from $t = 0$ to $t = \tau - 1$,

$$\mathsf{E}\{q_k(\tau)^2\} - \mathsf{E}\{q_k(0)^2\} = \mathsf{E}\{q_k(\tau)^2\} \leq \tau C^2 + 2U\sum_{t=0}^{\tau-1}\mathsf{E}\{|\overline{q_k}(t)|\}$$

Divide both side by τ and notice that $\limsup_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^t \mathsf{E}\{|\overline{q_k}(t)|\} < \infty$ implies $\frac{1}{t}\sum_{\tau=0}^t \mathsf{E}\{|\overline{q_k}(t)|\}$ is bounded, say by L, then

$$\frac{1}{\tau}\mathsf{E}\{q_k(\tau)^2\} \leq C^2 + 2U\frac{1}{\tau}\sum_{t=0}^{\tau-1}\mathsf{E}\{|\overline{q_k}(t)|\} \leq C^2 + 2UL$$

Now we apply Markov inequality with second moment expression.

$$\mathsf{Prob}(|q_k(t)| \geq \epsilon t) \leq \frac{1}{\epsilon^2 t^2}\mathsf{E}\{q_k(t)^2\} \leq \frac{C + 2UL}{\epsilon^2 t}$$

Let t_0 to be the first t such that $\frac{C+2UL}{\epsilon^2 t_0} < \delta$. Then

$$\mathsf{Prob}(|q_k(t)| \geq \epsilon t) < \delta,\ \forall t > t_0.$$

■

[3]Conditioning on the arrival vectors, $\mathbf{x}^*(t)$ is deterministic. And hence the only randomness of $\mu_{in,k}(t)$ and $\mu_{out,k}(t)$ comes from $B(t)$, which is independent of $\mathbf{q}(t)$.

[4]C and U exist because all of $\mu_{out,k}(t)$, $\mu_{in,k}(t)$, $\overline{\mu_{out,k}}(t)$, and $\overline{\mu_{in,k}}(t)$ are bounded by definition.

6.2.5 The Throughput Analysis

Lemma 6.2.3 to Lemma 6.2.6 describe the stability region of NM 2. However, in the proposed scheduling algorithm, there exists the occurrence of null activities which generate fictitious packets (f.p.) as described in Section 5.3.2. To show the proposed scheduling algorithm is able to achieve the optimal throughput, it remains to show that in the long term, the the fictitious packets does not influence the overall throughput. Before we explicitly quantify the fictitious packets throughput, we first analyze the growth rate of the deficit and the rate that the null activity occurs in NM 2. Given a arrival vector $\mathbf{a}(t)$ with strictly feasible rate $\mathbf{R}$ and utilizing the scheduling algorithm proposed in Section 6.2.2, we have the following results.

Lemma 6.2.7 *$D_k(t)$ grows sublinearly.*

The proof of Lemma 6.2.7 is relegated to Appendix D.

Proposition 6.2.1 *In NM 2, let $n_k(t)$ be the number of null activities occur at queue k up to time t. We define the rate of null activities occur at queue t as $r_k(t) \stackrel{\triangle}{=} \frac{1}{t} n_k(t)$. Given any $\epsilon > 0$ and $\delta > 0$. Then $\mathsf{Prob}(r_k(t) > \epsilon) < \delta$ given sufficient large t. That is, the probability that $r_k(t)$ larger than zero can be made arbitrarily small given sufficient large t.*

Proof Notice that the null activity (N.A) occurs at time t if and only if $D_k(t) > D_k(t-1)$. Let c be the minimum increment of $D_k(t)$ whenever N.A. occurs. We use the fact that $D_k(t)$ is non-decreasing and hence

$$n_k(t) = \sum_{\tau=1}^{t} I(D_k(\tau) > D_k(\tau - 1)) \leq \frac{D_k(t)}{c}.$$

Now for any $\epsilon > 0$ and $\delta > 0$,

$$\mathsf{Prob}(r_k(t) > \epsilon) = \mathsf{Prob}(\frac{n_k(t)}{t} > \epsilon) \leq \mathsf{Prob}(D_k(t) > c\epsilon t).$$

From Lemma 6.2.7, there exists t_0 such that

$$\mathsf{Prob}(r_k(t) > \epsilon) \leq \mathsf{Prob}(D_k(t) > c\epsilon t) < \delta, \ \forall t > t_0.$$

The proof is complete. ■

Now we analyze the fictitious packets generated at SA n and evaluate its effect on the overall throughput. Let $\eta_{fp,n}(t)$ be the number of fictitious packets generated by SA n up to time t. We use the following proposition to conclude the throughput analysis.

Proposition 6.2.2 *For each n, $\eta_{fp,n}(t)$ grows sublinearly.*

A heuristic but not rigorous explanation of Proposition 6.2.2 is as follows. Notice that the SPN is assumed to be acyclic. For the fictitious packet generated at the null activity, it can only "pollute" finite number of packets. And hence this result is a consequence of Proposition 6.2.1. The detailed proof of Proposition 6.2.2 is relegated to Appendix E.

Since the NM 2 can be stabilized with strictly feasible rate $\mathbf{R}$, the overall throughput is proportional to t. Then the ratio between the number of fictitious packets and the overall throughput (the influence of the fictitious packets) is proportional to $\frac{\eta_{fp,n}(t)}{t}$. By Proposition 6.2.2, this ratio can be arbitrarily small with sufficiently large t. Hence the scheduling algorithm in Section 6.2.2 achieves the optimal throughput in NM 2.

6.3 The Parallelism Between LNC Scheduling and SPN

We have introduced the 5-type coding scheme in Section 6.1.1 and the stability region of the SPN with random service in Section 6.2. In this section, we will connect these two together and characterize the stability region of the 2-user broadcast PEC with feedback.

We first define the following queue lengths[5].

$$\begin{aligned}
Q_{1,\text{LNC}}(t) &= \text{Rank}(\Omega(t)) - \text{Rank}(A_7(t)),\\
Q_{2,\text{LNC}}(t) &= \text{Rank}(\Omega(t)) - \text{Rank}(A_6(t)),\\
Q_{3,\text{LNC}}(t) &= \text{Rank}(A_5(t)) + \text{Rank}(A_3(t)) - \text{Rank}(A_6(t)) - \text{Rank}(A_1(t)),\\
Q_{3,\text{LNC}}(t) &= \text{Rank}(A_5(t)) + \text{Rank}(A_4(t)) - \text{Rank}(A_7(t)) - \text{Rank}(A_2(t)),\\
Q_{i,\text{undecoded,LNC}}(t) &= \text{Rank}(\Omega_i(t)) - \text{Rank}(\Omega_i(t) \cap \Psi_i(t)),\ i = 1, 2.
\end{aligned}$$

Namely, $Q_{k,\text{LNC}}(t)$ quantifies the gap between the left-hand side and the right-hand side of Condition k in (6.2)–(6.5). Recall that in Table 6.1 and the corresponding discussion,

[5]It is proven in [40] that all of them are non-negative.

we have shown that we can send a coding type if and only if the associated conditions are satisfied with strict inequality. This is captured in our definitions herein. We can schedule an SA (an LNC coding type) if and only if the corresponding queues (the gap of the conditions) are strictly non-empty.

The new queue we defined here is $Q_{i,\text{undecoded,LNC}}(t)$. To understand the physical meaning of $Q_{i,\text{undecoded,LNC}}(t)$, we notice that at any time t, the overall messages for session i is $\Omega_i(t)$. At the same time destination d_i can decode any linearly combined packets in Ψ_i and thus can decode all the desired session-i packets in $\Omega_i(t) \cap \Psi_i(t)$. As a result, the rank difference between the two is how many session-i packets that have arrived at the source s but still cannot be decoded by d_i. We thus term it $Q_{i,\text{undecoded,LNC}}(t)$. The following lemma guides us to connect the stability region definition in Section 2.3 with the 5-type LNC scheme.

Lemma 6.3.1 *If $Q_{i,LNC}(t)$ is sublinearly stable for all i, then $Q_{j,undecoded,LNC}(t)$ is also sublinearly stable for $j = 1, 2$.*

Proof We first claim that in the 5-type LNC scheme, $\text{Rank}(A_6(t)) + \text{Rank}(A_7(t)) - \text{Rank}(\Omega(t)) - \text{Rank}(A_5(t)) = 0$ for all time t. We prove this claim iteratively. For $t = 0$,

$$\begin{aligned}&\text{Rank}(A_6(0)) + \text{Rank}(A_7(0)) - \text{Rank}(\Omega(0)) - \text{Rank}(A_5(0))\\ =&\text{Rank}(\Omega_1) + \text{Rank}(\Omega_2) - \text{Rank}(\Omega_1 \oplus \Omega_2) - 0 = 0\end{aligned}$$

Suppose it is true for t. Note that in time $t + 1$, the LNC designer can choose one of TYPE_i for $i \in \{9, 18, 31, 63, 95\}$. We first notice that if there is any new arrival comes to the network, the increments of $\text{Rank}(A_6(t + 1)) + \text{Rank}(A_7(t + 1))$ and $\text{Rank}(\Omega(t + 1))$ will be the same. For example, if a new session-1 packet arrives. Then $\text{Rank}(A_6)$ will increase by 1 and $\text{Rank}(\Omega(t + 1))$ will increase by 1 as well. We further notice that no matter which coding type we choose, the increments of the positive terms and the negative terms are always the same. For example, if we choose coding type 9, then $\text{Rank}(A_5(t+1))$ (the negative term) and $\text{Rank}(A_6(t + 1))$ (the positive term) will increase by one if any of d_1 and d_2 receive the coding vector while $\text{Rank}(A_7(t + 1))$ (the positive term) remains the same. And hence the statement is still true for $t + 1$.

We now prove $Q_{1,\text{undecoded,LNC}}(t)$ is sublinearly stable and $Q_{2,\text{undecoded,LNC}}(t)$ can be proven by symmetric arguments. By definition,

$$\begin{aligned}
&Q_{1,\text{LNC}}(t) + Q_{3,\text{LNC}}(t) \\
=&\text{Rank}(\Omega(t)) - \text{Rank}(A_7(t)) \\
&+ \text{Rank}(A_5(t)) + \text{Rank}(A_3(t)) - \text{Rank}(A_6(t)) - \text{Rank}(A_1(t)) \\
=&\text{Rank}(A_3(t)) - \text{Rank}(A_1(t)) \\
=&\text{Rank}(\Omega_1(t)) - \text{Rank}(\Omega_1(t) \cap \Psi_1(t)) = Q_{1,\text{undecoded,LNC}}(t)
\end{aligned}$$

Since $Q_{1,\text{LNC}}(t)$ and $Q_{3,\text{LNC}}(t)$ are sublinearly stable, $Q_{1,\text{undecoded,LNC}}(t)$ is also sublinearly stable. ■

Since both $Q_{1,\text{undecoded,LNC}}(t)$ and $Q_{2,\text{undecoded,LNC}}(t)$ are sublinearly stable, the 2-user broadcast PEC with feedback is also sublinearly stable by Proposition 5.1.1.

It remains to characterize the rate region in which $Q_{i,\text{LNC}}(t)$ is sublinearly stable for all i, denoted by Λ_{LNC}. Based on the topology in Figure 6.1, we define SPN_ϵ for any $\epsilon \geq 0$ as follows. Let $Q_{i,\epsilon}(t)$ be the queue length of the queue Q_i in SPN_ϵ at time t for $i = 1, 2, 3, 4$. Let $a_{1,1} = a_{2,2} = 1$. For each time t, the number of IA$_i$ activations, $a_i(t)$, is equal to $A_i(t)$, the arrival process of flow-i messages. We further associate the corresponding random

service distributions with the channel status $\mathbf{Z}_s(t) = (Z_{s\to d_1}(t), Z_{s\to d_2}(t))$ in the 2-user broadcast PEC with feedback of interest.

$$(b_{1,1}(t), b'_{3,1}(t)) = \begin{cases} (1,0) & \text{if } \mathbf{Z}_s(t) = (1,\cdot) \\ (1,1) & \text{if } \mathbf{Z}_s(t) = (0,1) \\ (0,0) & \text{otherwise} \end{cases}$$

$$(b_{2,2}(t), b'_{4,2}(t)) = \begin{cases} (1,0) & \text{if } \mathbf{Z}_s(t) = (\cdot,1) \\ (1,1) & \text{if } \mathbf{Z}_s(t) = (1,0) \\ (0,0) & \text{otherwise} \end{cases}$$

$$(b_{3,3}(t), b_{4,3}(t)) = \begin{cases} (1,\epsilon) & \text{if } \mathbf{Z}_s(t) = (1,0) \\ (\epsilon,1) & \text{if } \mathbf{Z}_s(t) = (0,1) \\ (1,1) & \text{if } \mathbf{Z}_s(t) = (1,1) \\ (0,0) & \text{otherwise} \end{cases}$$

$$b_{3,4}(t) = \begin{cases} 1 & \text{if } \mathbf{Z}_s(t) = (1,\cdot) \\ 0 & \text{otherwise} \end{cases}$$

$$b_{4,5}(t) = \begin{cases} 1 & \text{if } \mathbf{Z}_s(t) = (\cdot,1) \\ 0 & \text{otherwise} \end{cases}$$

We also assume each individual pair listed above is independent with each other, and there is only one SA in SPN_ϵ can be activated at each time t.

In Section 6.2, we have characterized the full stability region of SPN with random service. Hence we also have the stability region of SPN_ϵ, denoted by Λ_ϵ, as defined above. We conclude the stability region of the 5-type LNC 2-user broadcast PEC with feedback by the following three propositions.

Proposition 6.3.1 *For any arrival rate (R_1, R_2), if we can stabilize the 5-type LNC 2-user broadcast PEC with feedback, then we can stabilize the SPN_0 network.*

Proposition 6.3.2 *For any arrival rate (R_1, R_2), if we can stabilize the SPN_ϵ network, then we can stabilize the 5-type LNC 2-user broadcast channel.*

Recall that the stability region of SPN_0 and SPN_ϵ have been fully characterized, we thus have

$$\Lambda_\epsilon \subseteq \Lambda_{\text{LNC}} \subseteq \Lambda_0$$

The final step is to let $\epsilon \to 0$. We thus have

Proposition 6.3.3 *The stability region of the 5-type LNC 2-user broadcast PEC with feedback is the same as the stability region of* SPN_0. *That is,*

$$\Lambda_{LNC} = \Lambda_0.$$

An heuristic but not rigorously explanation is as follows. It might seem that SPN_0 has exactly the same behavior as the 5-type LNC problem. However, this is not true. For the pair $(b_{3,3}(t), b_{4,3}(t))$, it could be $(1,0)$ if $\mathbf{Z}_s(t) = (1,0)$. That is, when $\mathbf{Z}_s(t) = (1,0)$, SA_{31} can be scheduled even if $Q_{4,0}(t) = 0$. But in the 5-type LNC problem, if Condition 4 is not hold (i.e. $Q_{4,\text{LNC}}(t) = 0$), then we can not choose TYPE_{31}. And hence we need to have SPN_ϵ to approach Λ_0 instead of making direct connections. The detailed proof of Proposition 6.3.1 to Proposition 6.3.3 are relegated to Appendix F.

6.4 Chapter Summary

In this chapter, we first discuss the 5-type space-based LNC scheme for achieving the Shannon capacity of 2-user broadcast PEC with feedback in Section 6.1. We also define the space-based LNC notations according to dynamic arrivals, and demonstrate the equivalent SPN is an SPN with random service. And hence we propose a modified deficit maximum weight scheduling algorithm to stabilize the SPN with random service and achieve the optimal throughput in Section 6.2. Finally, we connect all the developed together and characterize the full stability region of 2-user broadcast PEC with feedback in Section 6.3.

7. CONCLUSION AND FUTURE WORK

In this thesis, we propose a space-based LNC scheme to characterize the Shannon capacity of the COPE principle 2-user wireless butterfly network with broadcast packet erasure channels, which incorporates the broadcast packet erasure channels with feedback, the COPE principle, and the opportunistic routing all together. A modified deficit maximum scheduling is proposed to stabilize the SPN with random service. Finally, we combined all the developed tools together to characterize the stability region of the 2-user broadcast PEC with feedback.

In Chapter 1, we discuss the network coding gain for three local wireless network topologies, including the broadcast packet erasure channel with feedback, the COPE principle wireless butterfly network, and the opportunistic routing. All of them exhibit significant end-to-end throughput improvement when network coding can be utilized. In Chapter 2, we propose a local network topology which incorporates all the three local wireless network together, name the "COPE principle 2-user wireless butterfly network with broadcast packet erasure channels." The space-based LNC scheme established in Chapter 3 provides a intuitive and systematical approach to exploit the possible joint inter-session and inter-session network coding gain in the network. With the help of space-based LNC scheme, in Chapter 4, we characterize the Shannon capacity of the COPE principle 2-user wireless butterfly network with broadcast packet erasure channel and demonstrate significant throughput improvement compared with existing solutions. We further extend the block-code-based model to the dynamic arrival model, so called the stability analysis. This extension promotes the proposed scheme one step closer to practical implementations. However, there exist obstacles between the LNC scheme to the stability analysis, as discussed in Chapter 5. With existing results in stochastic processing networks, we can find some connection between SPN and the LNC problem. However, the random service issue is not well-considered in the existing results. We develop modified deficit maximum weight al-

gorithm to stabilize the SPN with random service. With 5-type space-based LNC scheme and modified DMW, we characterize the stability region of the 2-user broadcast PEC with feedback in Chapter 6.

Even though we build up the analysis tool based on the local wireless network, those tools reveal great insight about the possible joint intra-session and inter-session linear network coding gain. The future work can branch out to three categories, the near objective, the mid-range objective, and the long-term objective.

7.1 The Stability Of 2-User Multi-Input Broadcast PEC With Feedback

Recently, another work in our group [40] also utilizes the similar SBLNC concept to characterize the LNC feedback capacity of 2-receiver multi-input broadcast PECs. [40] proposes a space-based LNC scheme to achieve the LNC capacity region and numerically demonstrates the matching between Shannon capacity and LNC capacity under this specified channel. The space-based LNC scheme in [40] consists of "coding types." Each coding type can be associated with one SBLNC policy described in Chapter 3. That is, at each time t, the source of the PEC can choose one coding type to encode the transmitted packet. However, unlike the sequential scheduling scheme in our achieving algorithm in Section 4.3, there is no fixed sequence of policy (coding type) scheduling in [40]. Instead, the key to achieve the LNC capacity region is to properly adjust the coding-type frequency, so called "tunneling approach."

Extending our results in this thesis to the 2-use multi-input broadcast PEC with feedback is clearly a near objective. There are 18 coding types in this problem while we discuss the 5-type scheme in Chapter 6. However, Figure 7.1 exhibit the obstacle. Notice that there exist a cycle, Queue 3–Type 2–Queue 4–Type 1, in the equivalent SPN. In Section 5.3.1, we assume there is no cycle in SPN. However, in the multi-flow wireless network with network coding, it is likely to have cycles in the network as we will see in later. Even though the cycle issue does not affect the stability analysis in [38], but the problem is in the throughput analysis. In DMW algorithm in Section 5.3.2, the fictitious packet is generated

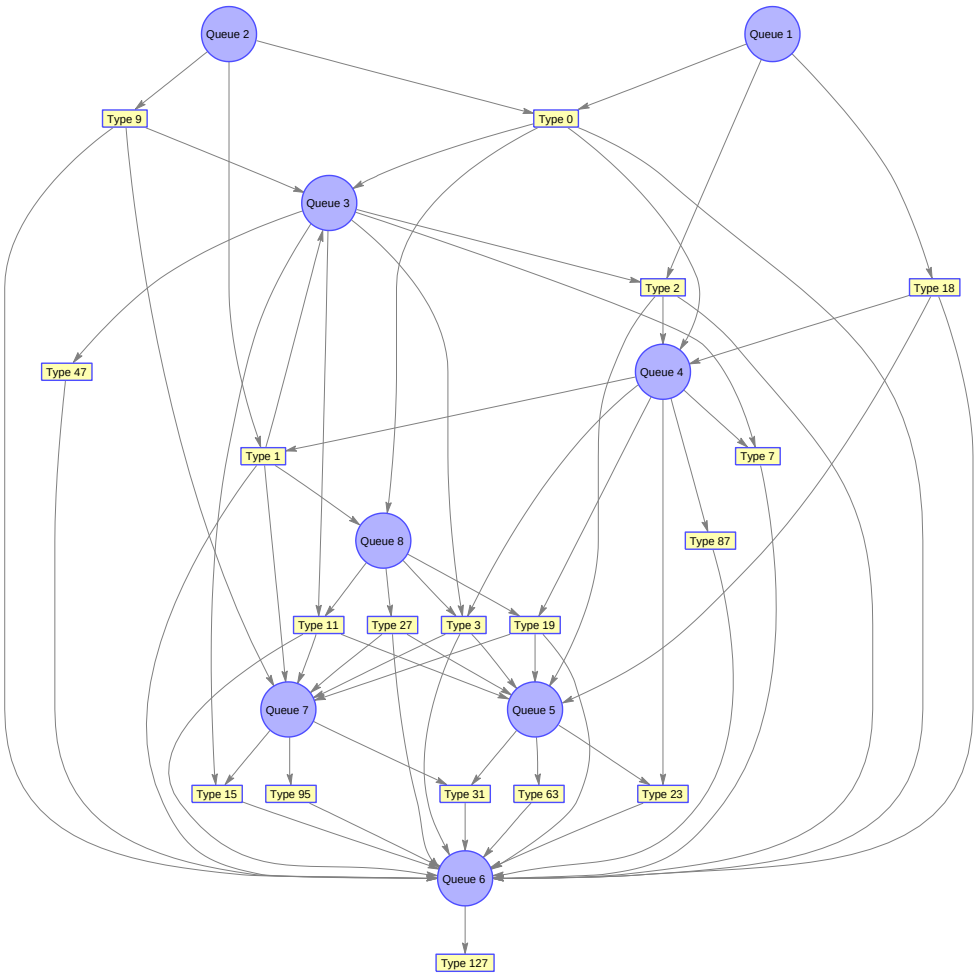


Fig. 7.1. The equivalent SPN of 18 coding types for 2-use multi-input broadcast PEC with feedback.

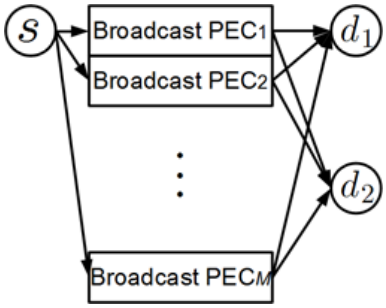


Fig. 7.2. The illustration of multi-user multi-input broadcast packet erasure channel.

whenever the underflow occurs. If unfortunately, the fictitious packet goes to one cycle, this fictitious packet will further generate new fictitious packets endlessly. And thus the throughput analysis in [38] is no longer valid.

This obstacle can be possibly be resolved by the random service property. Unlike SPN with deterministic service, it is possible for the packet which enters the cycle leaves the cycle after certain amount of "looping." By properly analyzing the amount of "looping," it gives a possible approach to circumvent this cycle issue.

7.2 The Mid-Range And Long-Term Objectives

Another possible extension is the stability region of the Markovian controlled 2-receiver multi-input broadcast packet erasure channel. Consider an ergodic finite-state Markov chain $\{S_t : \forall t = 1, 2, ...\}$ with the state space $\mathcal{S}$ and a sequence of user controllable actions $\{\mathsf{Act}_t \in \mathcal{A} : \forall t = 1, 2, ...\}$ taken from a finite set $\mathcal{A}$. The reception status of the Markovian controlled 2-receiver multi-input broadcast PEC at time t is then depends on S_t and Act_t but not on $S_{t'}$ or $\mathsf{Act}_{t'}$ for any $t' \neq t$. Since the probability distribution of the reception status is no longer invariant over time, the Markovian controlled 2-receiver multi-input broadcast PEC is a generalization of the canonical 2-receiver multi-input broadcast PEC as illustrated in Figure 7.2. With this generalization, the analysis results can closely capture many important practical applications.

For example, consider a setting of cognitive radio. When there exist external transmissions, the probability of success reception becomes lower due to stronger interference. And the probability of success reception is higher when there is no external transmissions. This kind of interference can be modeled by the Markov state S_t. At the same time, the classic Gilbert-Elliott channel model for burst noise is also a special case of Markovian setting.

On the other hand, the setting of user controllable action Act_t also provides a variety of flexibility to capture some practical scnearios. For example, consider an adaptive coding and modulation with two schemes. The first scheme provides higher transmission rate but lower success probability while the second scheme provides lower transmission rate but higher success probability. This setting can be closely captured by Act_t. [40] provides an space-based LNC scheme to characterize the LNC capacity region. Extending the existing capacity analysis to the stability analysis of the Markovian setting is our mid-range goal. To incorporate the Markovian state and the user-controllable actions, the SPN setting is required to further consider the these two issues.

The wireless butterfly network which incorporates broadcast PEC with feedback, the COPE principle, and the opportunistic all together is definitely a objective for the extension of current results. Our long-term objective is to devise practical protocols and testbed simulation to verify the intuition derived from our capacity and stability analysis.

APPENDICES

A. THE CONVERSE OF THE CAPACITY

In this appendix, we prove Proposition 4.2.2. For any joint scheduling and NC scheme, we choose t_{s_i} (resp. t_r) as the normalized *expected number of time slots for which s_i (resp. r) is scheduled*. Namely,

$$t_{s_i} \triangleq \frac{1}{n}\mathsf{E}\left\{\sum_{\tau=1}^{n} 1_{\{\sigma(\tau)=s_i\}}\right\} \text{ and } t_r \triangleq \frac{1}{n}\mathsf{E}\left\{\sum_{\tau=1}^{n} 1_{\{\sigma(\tau)=r\}}\right\}.$$

By definition, t_{s_1}, t_{s_2}, and t_r must satisfy (4.5).

In the subsequent proofs, the logarithm is taken with base q. We prove (4.6) first. To that end, we notice that

$$I(\mathbf{W}_1;\hat{\mathbf{W}}_1) \le I(\mathbf{W}_1;[\mathbf{Y}_{\{s_1,s_2,r\}\to d_1},\mathbf{Z}]_1^n) \tag{A.1}$$

$$=I(\mathbf{W}_1;[\mathbf{Z}]_1^n) + I(\mathbf{W}_1;[\mathbf{Y}_{\{s_1,s_2,r\}\to d_1}]_1^n|[\mathbf{Z}]_1^n) \tag{A.2}$$

$$\le I(\mathbf{W}_1;[\mathbf{Y}_{\{s_1,s_2\}\to\{d_1,r\}}]_1^n|[\mathbf{Z}]_1^n) \tag{A.3}$$

$$=I(\mathbf{W}_1;[\mathbf{Y}_{s_1\to\{d_1,r\}}]_1^n|[\mathbf{Z}]_1^n) \tag{A.4}$$

$$\le H([\mathbf{Y}_{s_1\to\{d_1,r\}}]_1^n|[\mathbf{Z}]_1^n)$$

$$=\sum_{t=1}^{n} H(\mathbf{Y}_{s_1\to\{d_1,r\}}(t)|[\mathbf{Z}]_1^n,[\mathbf{Y}_{s_1\to\{d_1,r\}}]_1^{t-1}) \tag{A.5}$$

$$\le \sum_{t=1}^{n} \mathsf{E}\left\{1_{\{Z_{s_1\to d_1}(t)=1 \text{ or } Z_{s_1\to r}(t)=1\}} \circ 1_{\{\sigma(t)=s_1\}}\right\} \tag{A.6}$$

$$=nt_{s_1}p_1(d_1,r) \tag{A.7}$$

where (A.1) follow from (2.7); (A.2) follows from the chain rule; (A.3) follows from (2.3), the data processing inequality, and the fact that $\mathbf{Z}$ is independent of $\mathbf{W}_1$; (A.4) follows from that conditioning on $\mathbf{Z}$ (and σ since σ is a function of $\mathbf{Z}$) $\mathbf{Y}_{s_2\to\{d_1,r\}}$ is a function of $\mathbf{W}_2$ and is thus independent of $\mathbf{W}_1$; (A.5) follows from the chain rule; (A.6) follows from that only when $1_{\{Z_{s_1\to d_1}(t)=1 \text{ or } Z_{s_1\to r}(t)=1\}} = 1$ and $1_{\{\sigma(t)=s_1\}} = 1$

will we have a non-zero entropy value $H(\mathbf{Y}_{s_1\to\{d_1,r\}}(t)|[\mathbf{Z}]_1^n, [\mathbf{Y}_{s_1\to\{d_1,r\}}]_1^{t-1})$, and when $H(\mathbf{Y}_{s_1\to\{d_1,r\}}(t)|[\mathbf{Z}]_1^n, [\mathbf{Y}_{s_1\to\{d_1,r\}}]_1^{t-1}) > 0$, it is upper bounded by 1 since the base of the logarithm is q; (A.7) follows from Wald's lemma.

On the other hand, Fano's inequality gives us

$$I(\mathbf{W}_1; \hat{\mathbf{W}}_1) \geq nR_1(1-\epsilon) - H(\epsilon). \tag{A.8}$$

Combining (A.7) and (A.8), we have

$$R_1(1-\epsilon) - \frac{H(\epsilon)}{n} \leq t_{s_1} p_1(d_1, r). \tag{A.9}$$

Letting $\epsilon \to 0$, (A.9) implies (4.6) for the case of $i = 1$. With symmetric arguments, we can derive (4.6) for $i = 2$.

We prove (4.8) by similar techniques as used in [16, 41]. Specifically, we create a new network from the original network by adding an auxiliary pipe that sends all information available at d_2 directly to d_1. Later we will show that even with the additional information, the achievable rates R_1 and R_2 are still upper bounded by (4.8). As a result, the achievable R_1 and R_2 for the original network must satisfy (4.8) as well. (4.7) is a symmetric version of (4.8).

With the additional information at d_1, the decoding function (see (2.7)) at d_1 for the new network becomes

$$\hat{\mathbf{W}}_1 = f_{d_1}([\mathbf{Y}_{\{s_1,s_2,r\}\to\{d_1,d_2\}}, \mathbf{Z}]_1^n). \tag{A.10}$$

For any $t \in [n]$, define

$$U(t) \triangleq (\mathbf{W}_2, [\mathbf{Y}_{\{s_1,s_2,r\}\to\{d_1,d_2\}}, \mathbf{Z}]_1^{t-1}). \tag{A.11}$$

We then have

$$
\begin{aligned}
nR_1 &= H(\mathbf{W}_1|\mathbf{W}_2) \\
&\leq I(\mathbf{W}_1; \hat{\mathbf{W}}_1|\mathbf{W}_2) + n\epsilon_1 && \text{(A.12)} \\
&\leq I(\mathbf{W}_1; [\mathbf{Y}_{\{s_1,s_2,r\}\to\{d_1,d_2\}}, \mathbf{Z}]_1^n|\mathbf{W}_2) + n\epsilon_1 && \text{(A.13)} \\
&= I(\mathbf{W}_1; [\mathbf{Z}]_1^n|\mathbf{W}_2) \\
&\quad + I(\mathbf{W}_1; [\mathbf{Y}_{\{s_1,s_2,r\}\to\{d_1,d_2\}}]_1^n|\mathbf{W}_2, [\mathbf{Z}]_1^n) + n\epsilon_1 && \text{(A.14)} \\
&= \sum_{t=1}^{n} I(\mathbf{W}_1; \mathbf{Y}_{\{s_1,s_2,r\}\to\{d_1,d_2\}}(t) \\
&\qquad |\mathbf{W}_2, [\mathbf{Z}]_1^n, [\mathbf{Y}_{\{s_1,s_2,r\}\to\{d_1,d_2\}}]_1^{t-1}) + n\epsilon_1 && \text{(A.15)} \\
&= n\epsilon_1 + \sum_{t=1}^{n} \big(I(\mathbf{W}_1; \mathbf{Y}_{r\to\{d_1,d_2\}}(t)|U(t), [\mathbf{Z}]_1^n) \\
&\quad + I(\mathbf{W}_1; \mathbf{Y}_{s_1\to\{d_1,d_2\}}(t)|U(t), \mathbf{Y}_{r\to\{d_1,d_2\}}(t), [\mathbf{Z}]_1^n) \\
&\quad + I(\mathbf{W}_1; \mathbf{Y}_{s_2\to\{d_1,d_2\}}(t) \\
&\qquad |U(t), \mathbf{Y}_{\{r,s_1\}\to\{d_1,d_2\}}(t), [\mathbf{Z}]_1^n) \big) && \text{(A.16)} \\
&\leq n\epsilon_1 + \left(\sum_{t=1}^{n} I(\mathbf{W}_1; \mathbf{Y}_{r\to\{d_1,d_2\}}(t)|U(t), [\mathbf{Z}]_1^n) \right) \\
&\quad + nt_{s_1} p_1(d_1, d_2) + 0 && \text{(A.17)} \\
&\leq n\epsilon_1 + nt_{s_1} p_1(d_1, d_2) \\
&\quad + \sum_{t=1}^{n} I(X_r(t); \mathbf{Y}_{r\to\{d_1,d_2\}}(t)|U(t), [\mathbf{Z}]_1^n), && \text{(A.18)}
\end{aligned}
$$

where (A.12) follows from Fano's inequality where ϵ_1 goes to 0 when $\epsilon \to 0$; (A.13) follows from the data processing inequality and (A.10); (A.14), (A.15), and (A.16) follow from the chain rule and the fact that the distribution of $\mathbf{Z}$ is independent of $\mathbf{W}_1$ and $\mathbf{W}_2$; (A.17) follows from the observation that the second term of the summation can be upper bounded by Wald's lemma (similar to (A.7)) and $\mathbf{Y}_{s_2\to\{d_1,d_2\}}(t)$ is independent of $\mathbf{W}_1$ given $\mathbf{Z}$ (similar to (A.4)); and (A.18) follows from the data processing inequality.

To continue, we define the time sharing random variable $Q_t \in \{1, 2, ..., n\}$ with $\mathsf{Prob}(Q_t = i) = \frac{1}{n}$ for all $i \in \{1, 2, ..., n\}$ and Q_t being independent of $[\mathbf{Z}]_1^n$, $\mathbf{W}_1$, and $\mathbf{W}_2$. Since the mutual information is always non-negative, we can rewrite (A.18) as

$$
\begin{aligned}
&(R_1 - t_{s_1} p_1(d_1, d_2) - \epsilon_1)^+ \\
&\leq \sum_{t=1}^{n} \frac{1}{n} I(X_r(t); \mathbf{Y}_{r\to\{d_1,d_2\}}(t) | U(t), [\mathbf{Z}]_1^n) \\
&\leq \sum_{t=1}^{n} \frac{1}{n} H(\mathbf{Y}_{r\to\{d_1,d_2\}}(t) | U(t), [\mathbf{Z}]_1^n) \qquad \text{(A.19)} \\
&= \sum_{q_t=1}^{n} \mathsf{Prob}(Q_t = q_t) \cdot H(\mathbf{Y}_{r\to\{d_1,d_2\}}(q_t) | U(q_t), [\mathbf{Z}]_1^{q_t}, Q_t = q_t) \qquad \text{(A.20)}
\end{aligned}
$$

where (A.19) follows from the definition of the mutual information; (A.20) follows from replacing the time index t by the time sharing random variable Q_t and the distribution of $U(q_t)$ and $\mathbf{Y}_{r\to\{d_1,d_2\}}(q_t)$ does not depend on the future channel realization $[\mathbf{Z}]_{q_t+1}^n$.

We define three binary random variables $\Theta_\sigma \triangleq 1_{\{\sigma(Q_t)=r\}}$, $\Theta_{Z_1} \triangleq 1_{\{Z_{r\to d_1}(Q_t)=1\}}$, and $\Theta_{Z_2} \triangleq 1_{\{Z_{r\to d_2}(Q_t)=1\}}$, which are functions of Q_t and $[\mathbf{Z}]_1^{Q_t}$. Then we can rewrite (A.20) as the following.

$$
\begin{aligned}
&(R_1 - t_{s_1} p_1(d_1, d_2) - \epsilon_1)^+ \\
&\leq \sum_{q_t=1}^{n} \frac{1}{n} H(\mathbf{Y}_{r\to\{d_1,d_2\}}(q_t) | U(q_t), [\mathbf{Z}]_1^{q_t}, Q_t = q_t, \Theta_\sigma, \Theta_{Z_1}, \Theta_{Z_2}) \qquad \text{(A.21)} \\
&= \sum_{q_t=1}^{n} \frac{1}{n} \sum_{\substack{\forall u, [z]_1^{q_t}, \\ \theta_\sigma, \theta_{Z_1}, \theta_{Z_2}}} p_{U(q_t), [\mathbf{Z}]_1^{q_t}, \Theta_\sigma, \Theta_{Z_1}, \Theta_{Z_2}}(u, [z]_1^{q_t}, \theta_\sigma, \theta_{Z_1}, \theta_{Z_2}) \\
&\quad \cdot H(\mathbf{Y}_{r\to\{d_1,d_2\}}(q_t) | U(q_t) = u, [\mathbf{Z}]_1^{q_t} = [z]_1^{q_t}, \\
&\qquad\qquad \Theta_\sigma = \theta_\sigma, \Theta_{Z_1} = \theta_{Z_1}, \Theta_{Z_2} = \theta_{Z_2}) \qquad \text{(A.22)} \\
&= \sum_{q_t=1}^{n} \frac{1}{n} \sum_{\substack{\forall u, [z]_1^{q_t}, \theta_{Z_1}, \theta_{Z_2} \\ \text{s.t. } \max\{\theta_{Z_1}, \theta_{Z_2}\}=1}} p(u, [z]_1^{q_t}, 1, \theta_{Z_1}, \theta_{Z_2}) \\
&\quad \cdot H(X_r(q_t) | U(q_t) = u, [\mathbf{Z}]_1^{q_t} = [z]_1^{q_t}, \\
&\qquad\qquad \Theta_\sigma = 1, \Theta_{Z_1} = \theta_{Z_1}, \Theta_{Z_2} = \theta_{Z_2}) \qquad \text{(A.23)}
\end{aligned}
$$

where (A.21) follows from the fact that Θ's are functions of Q and $[\mathbf{Z}]_1^{Q_t}$; (A.22) follows from the definition of the conditional entropy; and (A.23) follows from the fact that $Y_{r\to\{d_1,d_2\}}(q_t)$ is not erasure only if $\sigma(q_t)=r$ and at least one of $Z_{r\to d_1}$ and $Z_{r\to d_2}$ equals to one and furthermore $Y_{r\to\{d_1,d_2\}}(q_t) = X_r(q_t)$ under such a condition, where we use $p(u,[z]_1^{q_t},1,\theta_{Z_1},\theta_{Z_2})$ as the shorthand of $p_{U(q_t),[\mathbf{Z}]_1^{q_t},\Theta_\sigma,\Theta_{Z_1},\Theta_{Z_2}}(u,[z]_1^{q_t},1,\theta_{Z_1},\theta_{Z_2})$.

We can further simplify (A.23) by the following steps. We first note that conditioning on $U(q_t)=u$, $[\mathbf{Z}]_1^{q_t-1}=[z]_1^{q_t-1}$, and $\Theta_\sigma=1$, the random variable $X_r(q_t)$ is independent of $\mathbf{Z}(q_t)$, Θ_{Z_1}, and Θ_{Z_2}. Notice that $[\mathbf{Z}]_1^{q_t-1}$ is a subset of $U(q_t)$. Therefore, we have

$$\begin{aligned} &H(X_r(q_t)|U(q_t)=u,[\mathbf{Z}]_1^{q_t}=[z]_1^{q_t},\Theta_\sigma=1,\Theta_{Z_1}=\theta_{Z_1},\\ &\qquad\qquad \Theta_{Z_2}=\theta_{Z_2})\\ &= H(X_r(q_t)|U(q_t)=u,\Theta_\sigma=1). \end{aligned} \tag{A.24}$$

Also the joint probability can be rewritten as

$$\begin{aligned} &\sum_{\substack{\forall u,[z]_1^{q_t},\theta_{Z_1},\theta_{Z_2}\\ \text{s.t. } \max\{\theta_{Z_1},\theta_{Z_2}\}=1}} p_{U(q_t),[\mathbf{Z}]_1^{q_t},\Theta_\sigma,\Theta_{Z_1},\Theta_{Z_2}}(u,[z]_1^{q_t},1,\theta_{Z_1},\theta_{Z_2})\\ &= \sum_{\forall u} p_{U(q_t),\Theta_\sigma}(u,1)\cdot \sum_{\substack{\forall z,\theta_{Z_1},\theta_{Z_2}\\ \text{s.t. } \max\{\theta_{Z_1},\theta_{Z_2}\}=1}}\\ &\quad p_{\mathbf{Z}(q_t),\Theta_{Z_1},\Theta_{Z_2}|U(q_t),\Theta_\sigma}(z,\theta_{Z_1},\theta_{Z_2}|u,1) \qquad (A.25)\\ &= \left(\sum_{\forall u} p_{U(q_t),\Theta_\sigma}(u,1)\right)\cdot p_r(d_1,d_2). \qquad (A.26) \end{aligned}$$

where (A.25) follows from the basic probability definition, and (A.26) follows from that the assumption that the channel is memoryless.

(A.24) and (A.26) helps us rewrite (A.23) as

$$\begin{aligned} (A.23) &= t_r\cdot p_r(d_1,d_2)\\ &\cdot \frac{\sum_{q_t=1}^n \frac{1}{n}\sum_{\forall u} p(u,1)\cdot H(X_r(q_t)|u,1)}{t_r} \end{aligned} \tag{A.27}$$

where $p(u,1)$ and $H(X_r(q_t)|u,1)$ are the shorthand for $p_{U(q_t),\Theta_\sigma}(u,1)$ and $H(X_r(q_t)|U(q_t)=u,\Theta_\sigma=1)$, respectively.

We now focus on flow 2. By Fano's inequality, for some $\epsilon_2 > 0$ that goes to 0 as $\epsilon \to 0$, with similar steps as in (A.12)–(A.18), we can also show that

$$
\begin{aligned}
&nR_2 = H(\mathbf{W}_2) \\
\leq & I(\mathbf{W}_2; [\mathbf{Y}_{\{s_1,s_2,r\}\to d_2}, \mathbf{Z}]_1^n) + n\epsilon_2 \\
= & I(\mathbf{W}_2; [\mathbf{Z}]_1^n) + I(\mathbf{W}_2; [\mathbf{Y}_{\{s_1,s_2,r\}\to d_2}]_1^n | [\mathbf{Z}]_1^n) + n\epsilon_2 && \text{(A.28)} \\
= & \sum_{t=1}^{n} I(\mathbf{W}_2; \mathbf{Y}_{\{s_1,s_2,r\}\to d_2}(t) | [\mathbf{Y}_{\{s_1,s_2,r\}\to d_2}]_1^{t-1}, [\mathbf{Z}]_1^n) \\
& + n\epsilon_2 && \text{(A.29)} \\
= & n\epsilon_2 + \sum_{t=1}^{n} \big(I(\mathbf{W}_2; Y_{r\to d_2}(t) | [\mathbf{Y}_{\{s_1,s_2,r\}\to d_2}]_1^{t-1}, [\mathbf{Z}]_1^n) \\
& + I(\mathbf{W}_2; Y_{s_2\to d_2}(t) | [\mathbf{Y}_{\{s_1,s_2\}\to d_2}]_1^{t-1}, [\mathbf{Y}_{r\to d_2}]_1^t, [\mathbf{Z}]_1^n) \\
& + I(\mathbf{W}_2; Y_{s_1\to d_2}(t) | [\mathbf{Y}_{s_1\to d_2}]_1^{t-1}, [\mathbf{Y}_{\{s_2,r\}\to d_2}]_1^t, [\mathbf{Z}]_1^n) \big) && \text{(A.30)} \\
\leq & n\epsilon_2 + \sum_{t=1}^{n} I(\mathbf{W}_2; Y_{r\to d_2}(t) | [\mathbf{Y}_{\{s_1,s_2,r\}\to d_2}]_1^{t-1}, [\mathbf{Z}]_1^n) \\
& + nt_{s_2} p_2(d_2) + 0 && \text{(A.31)}
\end{aligned}
$$

where (A.28), (A.29), and (A.30) follows from the chain rule and the independence between $\mathbf{W}_2$ and $[\mathbf{Z}]_1^n$; and (A.31) follows from similar derivation as in (A.17). We then have

$$
\begin{aligned}
(A.31) = & n\epsilon_2 + nt_{s_2} p_2(d_2) \\
& + \sum_{t=1}^{n} \big(H(Y_{r\to d_2}(t) | [\mathbf{Y}_{\{s_1,s_2,r\}\to d_2}]_1^{t-1}, [\mathbf{Z}]_1^n) \\
& - H(Y_{r\to d_2}(t) | \mathbf{W}_2, [\mathbf{Y}_{\{s_1,s_2,r\}\to d_2}]_1^{t-1}, [\mathbf{Z}]_1^n) \big) && \text{(A.32)} \\
\leq & \, n\epsilon_2 + nt_{s_2} p_2(d_2) \\
& + \sum_{t=1}^{n} \left(H(Y_{r\to d_2}(t) | [\mathbf{Z}]_1^n) - H(Y_{r\to d_2}(t) | U(t), [\mathbf{Z}]_1^n) \right) && \text{(A.33)} \\
= & \, n\epsilon_2 + nt_{s_2} p_2(d_2) + \sum_{t=1}^{n} I(U(t); Y_{r\to d_2}(t) | [\mathbf{Z}]_1^n), && \text{(A.34)}
\end{aligned}
$$

where (A.32) and (A.34) follows from the definition of the mutual information; and (A.33) follows from the fact that *conditioning does not increase the entropy* and $[\mathbf{Y}_{\{s_1,s_2,r\}\to d_2}]_1^{t-1}$ a subset of $U(t)$. Since the mutual information is always non-negative, we now have

$$
\begin{aligned}
&(R_2 - t_{s_2} p_{s_2}(d_2) - \epsilon_2)^+ \\
&\leq \frac{1}{n}\sum_{t=1}^{n} I(U(t); Y_{r\to d_2}(t) | [\sigma, \mathbf{Z}]_1^n) \\
&= \sum_{q_t=1}^{n} \mathsf{Prob}(Q_t = q_t) \cdot I(U(q_t); Y_{r\to d_2}(q_t) | [\mathbf{Z}]_1^{q_t}, Q_t = q_t) \qquad \text{(A.35)} \\
&= \sum_{q_t=1}^{n} \frac{1}{n} \cdot H(Y_{r\to d_2}(q_t) | [\mathbf{Z}]_1^{q_t}, Q_t = q_t) \\
&\quad - \sum_{q_t=1}^{n} \frac{1}{n} \cdot H(Y_{r\to d_2}(q_t) | U(q_t), [\mathbf{Z}]_1^{q_t}, Q_t = q_t), \qquad \text{(A.36)}
\end{aligned}
$$

where (A.35) follows from the definition of the conditional mutual information and the fact that the distribution of $U(q_t)$ and $Y_{r\to d_2}(q_t)$ does not depend on the future channel realization $[\mathbf{Z}]_{q_t+1}^n$; and (A.36) follows from the definition of the mutual information. We first discuss the first summation in (A.36)

$$
\begin{aligned}
&\sum_{q_t=1}^{n} \frac{1}{n} \cdot H(Y_{r\to d_2}(q_t) | [\mathbf{Z}]_1^{q_t}, Q_t = q_t) \\
&= \sum_{q_t=1}^{n} \frac{1}{n} \cdot H(Y_{r\to d_2}(q_t) | [\mathbf{Z}]_1^{q_t}, Q_t = q_t, \Theta_\sigma, \Theta_{Z_2}) \qquad \text{(A.37)} \\
&= \sum_{q_t=1}^{n} \frac{1}{n} \sum_{\substack{\forall [z]_1^{q_t}, \\ \theta_\sigma, \theta_{Z_2}}} p_{[\mathbf{Z}]_1^{q_t}, \Theta_\sigma, \Theta_{Z_2}}([z]_1^{q_t}, \theta_\sigma, \theta_{Z_2}) \\
&\quad \cdot H(Y_{r\to d_2}(q_t) | [\mathbf{Z}]_1^{q_t} = [z]_1^{q_t}, \\
&\qquad \Theta_\sigma = \theta_\sigma, \Theta_{Z_2} = \theta_{Z_2}) \qquad \text{(A.38)} \\
&= \sum_{q_t=1}^{n} \frac{1}{n} \sum_{\forall [z]_1^{q_t}} p_{[\mathbf{Z}]_1^{q_t}, \Theta_\sigma, \Theta_{Z_2}}([z]_1^{q_t}, 1, 1) \\
&\quad \cdot H(X_r(q_t) | [\mathbf{Z}]_1^{q_t} = [z]_1^{q_t}, \\
&\qquad \Theta_\sigma = 1, \Theta_{Z_2} = 1) \qquad \text{(A.39)}
\end{aligned}
$$

where (A.37) follows from the fact that Θ's are functions of Q and $[\mathbf{Z}]_1^{Q_t}$; (A.38) follows from the definition of the conditional entropy; and (A.39) follows from the fact that $Y_{r\to d_2}(q_t)$ is not erasure only if $\sigma(q_t) = r$ and $Z_{r\to d_2}$ equals to one and furthermore $Y_{r\to d_2}(q_t) = X_r(q_t)$ under such a condition.

We can further simplify (A.39) by the following steps. We first note that conditioning on $[\mathbf{Z}]_1^{q_t-1} = [z]_1^{q_t-1}$ and $\Theta_\sigma = 1$, the random variable $X_r(q_t)$ is independent of $\mathbf{Z}(q_t)$ and Θ_{Z_2}. Therefore, we have

$$\begin{aligned} &H(X_r(q_t)|[\mathbf{Z}]_1^{q_t} = [z]_1^{q_t}, \Theta_\sigma = 1, \Theta_{Z_2} = 1) \\ &= H(X_r(q_t)|[\mathbf{Z}]_1^{q_t-1} = [z]_1^{q_t-1}, \Theta_\sigma = 1). \end{aligned} \tag{A.40}$$

Also the joint probability can be rewritten as

$$\begin{aligned} &\sum_{\forall [z]_1^{q_t}} p_{[\mathbf{Z}]_1^{q_t},\Theta_\sigma,\Theta_{Z_2}}([z]_1^{q_t}, 1, 1) \\ &= \sum_{\forall [z]_1^{q_t-1}} p_{[\mathbf{Z}]_1^{q_t-1},\Theta_\sigma}([z]_1^{q_t-1}, 1) \cdot \sum_{\forall z} \\ &\quad p_{\mathbf{Z}(q_t),\Theta_{Z_2}|[\mathbf{Z}]_1^{q_t-1},\Theta_\sigma}(z, 1|[z]_1^{q_t-1}, 1) \qquad \text{(A.41)} \\ &= \left(\sum_{\forall [z]_1^{q_t-1}} p_{[\mathbf{Z}]_1^{q_t-1},\Theta_\sigma}([z]_1^{q_t-1}, 1) \right) \cdot p_r(d_2). \qquad \text{(A.42)} \end{aligned}$$

where (A.41) follows from the basic probability definition, and (A.42) follows from that the assumption that the channel is memoryless.

(A.40) and (A.42) helps us rewrite (A.39) as

$$\begin{aligned} (A.39) &= t_r \cdot p_r(d_2) \\ &\cdot \frac{\sum_{q_t=1}^{n} \frac{1}{n} \sum_{\forall [z]_1^{q_t-1}} p([z]_1^{q_t-1}, 1) \cdot H(X_r(q_t)|[z]_1^{q_t-1}, 1)}{t_r} \end{aligned} \tag{A.43}$$

where $p([z]_1^{q_t-1}, 1)$ and $H(X_r(q_t)|[z]_1^{q_t-1}, 1)$ are the shorthand for $p_{[\mathbf{Z}]_1^{q_t-1},\Theta_\sigma}([z]_1^{q_t-1}, 1)$ and $H(X_r(q_t)|[\mathbf{Z}]_1^{q_t-1} = [z]_1^{q_t-1}, \Theta_\sigma = 1)$, respectively.

Similarly, for the second summation in (A.36),

$$\sum_{q_t=1}^{n} \frac{1}{n} \cdot H(Y_{r\to d_2}(q_t)|U(q_t), [\mathbf{Z}]_1^{q_t}, Q_t = q_t)$$

$$= \sum_{q_t=1}^{n} \frac{1}{n} \cdot H(Y_{r\to d_2}(q_t)|U(q_t), [\mathbf{Z}]_1^{q_t}, Q_t = q_t, \Theta_\sigma, \Theta_{Z_2}) \tag{A.44}$$

$$= \sum_{q_t=1}^{n} \frac{1}{n} \sum_{\substack{\forall u, [z]_1^{q_t}, \\ \theta_\sigma, \theta_{Z_2}}} p_{U(q_t), [\mathbf{Z}]_1^{q_t}, \Theta_\sigma, \Theta_{Z_2}}(u, [z]_1^{q_t}, \theta_\sigma, \theta_{Z_2})$$

$$\cdot H(Y_{r\to d_2}(q_t)|U(q_t) = u, [\mathbf{Z}]_1^{q_t} = [z]_1^{q_t},$$

$$\Theta_\sigma = \theta_\sigma, \Theta_{Z_2} = \theta_{Z_2}) \tag{A.45}$$

$$= \sum_{q_t=1}^{n} \frac{1}{n} \sum_{\forall u, [z]_1^{q_t}} p_{U(q_t), [\mathbf{Z}]_1^{q_t}, \Theta_\sigma, \Theta_{Z_2}}(u, [z]_1^{q_t}, 1, 1)$$

$$\cdot H(X_r(q_t)|U(q_t) = u, [\mathbf{Z}]_1^{q_t} = [z]_1^{q_t},$$

$$\Theta_\sigma = 1, \Theta_{Z_2} = 1) \tag{A.46}$$

where (A.44) follows from the fact that Θ's are functions of Q and $[\mathbf{Z}]_1^{Q_t}$; (A.45) follows from the definition of the conditional entropy; and (A.46) follows from the fact that $Y_{r\to d_2}(q_t)$ is not erasure only if $\sigma(q_t) = r$ and $Z_{r\to d_2}$ equals to one and furthermore $Y_{r\to d_2}(q_t) = X_r(q_t)$ under such a condition.

We can further simplify (A.46) by the following steps. We first note that conditioning on $U(q_t) = u$, $[\mathbf{Z}]_1^{q_t-1} = [z]_1^{q_t-1}$, and $\Theta_\sigma = 1$, the random variable $X_r(q_t)$ is independent of $\mathbf{Z}(q_t)$ and Θ_{Z_2}. Notice that $[\mathbf{Z}]_1^{q_t-1}$ is a subset of $U(q_t)$. Therefore, we have

$$H(X_r(q_t)|U(q_t) = u, [\mathbf{Z}]_1^{q_t} = [z]_1^{q_t}, \Theta_\sigma = 1, \Theta_{Z_2} = 1)$$

$$= H(X_r(q_t)|U(q_t) = u, \Theta_\sigma = 1). \tag{A.47}$$

Also the joint probability can be rewritten as

$$\sum_{\forall u,[z]_1^{q_t}} p_{U(q_t),[\mathbf{Z}]_1^{q_t},\Theta_\sigma,\Theta_{Z_2}}(u,[z]_1^{q_t},1,1)$$

$$= \sum_{\forall u} p_{U(q_t),\Theta_\sigma}(u,1) \cdot \sum_{\forall z}$$

$$p_{\mathbf{Z}(q_t),\Theta_{Z_2}|U(q_t),\Theta_\sigma}(z,1|u,1) \tag{A.48}$$

$$= \left(\sum_{\forall u} p_{U(q_t),\Theta_\sigma}(u,1) \right) \cdot p_r(d_2). \tag{A.49}$$

where (A.48) follows from the basic probability definition, and (A.49) follows from that the assumption that the channel is memoryless.

(A.47) and (A.49) helps us rewrite (A.46) as

$$(A.39) = t_r \cdot p_r(d_2) \cdot \frac{\sum_{q_t=1}^{n} \frac{1}{n} \sum_{\forall u} p(u,1) \cdot H(X_r(q_t)|u,1)}{t_r} \tag{A.50}$$

where $p(u,1)$ and $H(X_r(q_t)|u,1)$ are the shorthand for $p_{U(q_t),\Theta_\sigma}(u,1)$ and $H(X_r(q_t)|U(q_t)=u,\Theta_\sigma=1)$, respectively.

Combining (A.43) and (A.50), we can rewrite (A.36) in the following form.

$$\begin{aligned}
&(R_2 - t_{s_2} p_{s_2}(d_2) - \epsilon_2)^+ \\
&\le t_r \cdot p_r(d_2) \\
&\cdot \left(\frac{\sum_{q_t=1}^{n} \frac{1}{n} \sum_{\forall [z]_1^{q_t-1}} p([z]_1^{q_t-1},1) \cdot H(X_r(q_t)|[z]_1^{q_t-1},1)}{t_r} - \right. \\
&\left. \frac{\sum_{q_t=1}^{n} \frac{1}{n} \sum_{\forall u} p(u,1) \cdot H(X_r(q)|u,1)}{t_r} \right).
\end{aligned} \tag{A.51}$$

Summing up $\frac{(A.27)}{p_r(d_1,d_2)}$ and $\frac{(A.51)}{p_r(d_2)}$, we thus have

$$\frac{(R_1 - t_{s_1} p_1(d_1,d_2) - \epsilon_1)^+}{p_r(d_1,d_2)} + \frac{(R_2 - t_{s_2} p_2(d_2) - \epsilon_2)^+}{p_r(d_2)}$$

$$\le t_r \cdot \frac{\sum_{q_t=1}^{n} \frac{1}{n} \sum_{\forall [z]_1^{q_t-1}} p([z]_1^{q_t-1},1) \cdot H(X_r(q_t)|[z]_1^{q_t-1},1)}{t_r} \tag{A.52}$$

$$\le t_r, \tag{A.53}$$

where (A.53) is based on the following observations. We first note that by definition

$$\begin{aligned} t_r &= \sum_{q_t=1}^{n} \frac{1}{n} \mathsf{Prob}(\sigma(q_t) = r) \\ &= \sum_{q_t=1}^{n} \frac{1}{n} \sum_{\forall [z]_1^{q_t-1}} p([z]_1^{q_t-1}, 1). \end{aligned}$$

Therefore, the fraction term in (A.52) can be viewed as the normalization of the conditional entropy $H(X_r(q_t)|[z]_1^{q_t-1}, 1)$. Since each conditional entropy is no larger than 1 (with the base of the logarithm being q), we thus have (A.53).

(A.53) holds for arbitrary $\epsilon > 0$. Letting $\epsilon \to 0$[1], we thus have the following final inequality.

$$\frac{(R_1 - t_{s_1} p_1(d_1, d_2))^+}{p_r(d_1, d_2)} + \frac{(R_2 - t_{s_2} p_2(d_2))^+}{p_r(d_2)} \leq t_r,$$

which gives us (4.8). (4.7) can be proven by symmetry. The proof of the outer bound is thus complete.

[1]As a result, $\epsilon_1 \to 0$ and $\epsilon_2 \to 0$.

B. DETAILED ACHIEVABILITY ANALYSIS

The feasibility for Policy $\Gamma_{s_1,0}$ and Policy $\Gamma_{s_1,1}$ has been proven in Section 4.3.1. In the following discussion about the rank of spaces, we again rely on the first order, expectation-based analysis and assume the application of the law of large numbers implicitly.

Policy $\Gamma_{s_1,2}$: Similar to the analysis for Policy $\Gamma_{s_1,1}$, assuming $q \geq 2$, the condition that (3.3) being non-empty is equivalent to whether the following rank-based inequality is satisfied.

$$\begin{aligned} &\mathrm{Rank}(S_2) - \mathrm{Rank}(S_2 \cap (S_1 \oplus S_r)) \\ =&\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r) - \mathrm{Rank}(S_1 \oplus S_r) > 0. \end{aligned} \tag{B.1}$$

where (B.1) follows from Lemma 3.1.2.

Similar to the discussion in $\Gamma_{s_1,0}$ and $\Gamma_{s_1,1}$, we will quantify individual ranks at the end of $\Gamma_{s_1,2}$, the policy of interest, and prove that even in the end of $\Gamma_{s_1,2}$, the rank difference in (B.1) is strictly larger than 0. Therefore, throughout the entire duration of $\Gamma_{s_1,2}$, (B.1) is larger than 0 and $\Gamma_{s_1,2}$ is always feasible.

We first focus on $\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r)$. Since $S_1 \oplus S_2 \oplus S_r$ is a subset of the exclusion set in $\Gamma_{s_1,0}$, every time a $\Gamma_{s_1,0}$ packet is received by one of d_1, d_2, and r, $\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r)$ will increase by one. On the other hand, notice that $S_1 \oplus S_2 \oplus S_r$ is a superset of the inclusion set in $\Gamma_{s_1,1}$ and $\Gamma_{s_1,2}$. Hence $\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r)$ remains the same throughout $\Gamma_{s_1,1}$ and $\Gamma_{s_1,2}$. As a result, in the end of policy $\Gamma_{s_1,2}$, we have

$$\mathsf{E}\{\mathrm{Rank}(S_1 \oplus S_2 \oplus S_r)\} = n\omega_{s_1}^0 p_1(d_1, d_2, r). \tag{B.2}$$

We now focus on $\mathrm{Rank}(S_1 \oplus S_r)$. Since $S_1 \oplus S_r$ is a subset of the exclusion sets of $\Gamma_{s_1,0}$, $\Gamma_{s_1,1}$ and $\Gamma_{s_1,2}$, every time a packet of $\Gamma_{s_1,0}$, $\Gamma_{s_1,1}$, or $\Gamma_{s_1,2}$ is received by one of d_1 and r, $\mathrm{Rank}(S_1 \oplus S_r)$ will increase by one. As a result, in the end of policy $\Gamma_{s_1,2}$, we have

$$\mathsf{E}\{\mathrm{Rank}(S_1 \oplus S_r)\} = n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2) p_1(d_1, r). \tag{B.3}$$

Jointly, (B.2), (B.3), and (4.15) imply (B.1) in the end of $\Gamma_{s_1,2}$.

Policy $\Gamma_{s_1,3}$: Similar to the analysis of the previous policies, assuming $q \geq 2$, the condition that (3.4) being non-empty is equivalent to whether the following rank-based inequality is satisfied.

$$\begin{aligned}
&\mathrm{Rank}(S_r) - \mathrm{Rank}(((S_2 \cap S_r) \oplus S_1) \cap S_r) \\
=&\mathrm{Rank}((S_2 \cap S_r) \oplus S_1 \oplus S_r) - \mathrm{Rank}((S_2 \cap S_r) \oplus S_1) && \text{(B.4)} \\
=&\mathrm{Rank}(S_1 \oplus S_r) - (\mathrm{Rank}(S_1) + \mathrm{Rank}(S_2 \cap S_r) \\
&- \mathrm{Rank}(S_1 \cap S_2 \cap S_r)) && \text{(B.5)} \\
=&\mathrm{Rank}(S_1 \oplus S_r) - \mathrm{Rank}(S_1) - (\mathrm{Rank}(S_2) + \mathrm{Rank}(S_r) \\
&- \mathrm{Rank}(S_2 \oplus S_r)) + \mathrm{Rank}(S_1 \cap S_2 \cap S_r) > 0, && \text{(B.6)}
\end{aligned}$$

where (B.4) follows from Lemma 3.1.2; (B.5) follows from simple set operations and from Lemma 3.1.2; and (B.6) follows from Lemma 3.1.2.

Similar to the previous discussion, we will quantify individual ranks at the end of $\Gamma_{s_1,3}$, the policy of interest and prove that even in the end of $\Gamma_{s_1,3}$, the rank difference in (B.6) is strictly larger than 0. Therefore, throughout the entire duration of $\Gamma_{s_1,3}$, (B.6) is larger than 0 and $\Gamma_{s_1,3}$ is always feasible.

By similar analysis,[1] in the end of $\Gamma_{s_1,3}$ we have

$$\mathsf{E}\{\mathrm{Rank}(S_1)\} = n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2 + \omega_{s_1}^3)p_1(d_1), \quad \text{(B.7)}$$

$$\mathsf{E}\{\mathrm{Rank}(S_2)\} = n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^3)p_1(d_2), \quad \text{(B.8)}$$

$$\mathsf{E}\{\mathrm{Rank}(S_r)\} = n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2)p_1(r), \quad \text{(B.9)}$$

$$\mathsf{E}\{\mathrm{Rank}(S_1 \oplus S_r)\} = n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2)p_1(d_1, r), \quad \text{(B.10)}$$

$$\mathsf{E}\{\mathrm{Rank}(S_2 \oplus S_r)\} = n\omega_{s_1}^0 p_1(d_2, r) \quad \text{(B.11)}$$

What remains to be decided is the value of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ at the end of Policy $\Gamma_{s_1,3}$. To proceed, we introduce an auxiliary node a in the following way. Whenever a

[1]The derivation of (B.8) for the case of Policy $\Gamma_{s_1,3}$ uses the following inequality as well.

$$(3.4) \subseteq (S_r \backslash (S_2 \cap S_r)) = (S_r \backslash S_2).$$

vector $\mathbf{v}$ sent by s_1 is received by both d_1 and r, we let the auxiliary node a observe such $\mathbf{v}$ as well. The knowledge space of a, denoted by S_a is thus the linear span of all vectors received by both d_1 and r.

We first argue that $S_a = S_1 \cap S_r$ in the end of policy $\Gamma_{s_1,2}$. Since a only observes those vectors commonly available at both d_1 and r, the knowledge space of S_a is a subset of $S_1 \cap S_r$. Knowing $S_a \subseteq S_1 \cap S_r$, we can quickly check that S_a is a subset of the exclusion sets in Policies $\Gamma_{s_1,0}$, $\Gamma_{s_1,1}$, and $\Gamma_{s_1,2}$. Therefore, every time node a receives a packet during policies $\Gamma_{s_1,0}$, $\Gamma_{s_1,1}$, and $\Gamma_{s_1,2}$, the rank of S_a will increase by one. Therefore, we have

$$\mathsf{E}\{\mathrm{Rank}(S_a)\} = n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2)p_1(d_1 r) \tag{B.12}$$

in the end of $\Gamma_{s_1,2}$. On the other hand, by similar analysis as before, we have

$$\begin{aligned}
\mathsf{E}\{\mathrm{Rank}(S_1)\} &= n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2)p_1(d_1), \\
\mathsf{E}\{\mathrm{Rank}(S_r)\} &= n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2)p_1(r), \\
\mathsf{E}\{\mathrm{Rank}(S_1 \oplus S_r)\} &= n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2)p_1(d_1, r)
\end{aligned}$$

in the end of policy $\Gamma_{s_1,2}$. By Lemma 3.1.2, we thus have $\mathrm{Rank}(S_a) = \mathrm{Rank}(S_1 \cap S_r)$. As a result, we have proven $S_a = (S_1 \cap S_r)$ in the end of $\Gamma_{s_1,2}$.

By the above analysis, we thus have $(S_1 \cap S_2 \cap S_r) = S_a \cap S_2$. By similarly rank-based analysis, in the end of $\Gamma_{s_1,2}$ we have

$$\mathsf{E}\{\mathrm{Rank}(S_2)\} = n(\omega_{s_1}^0 + \omega_{s_1}^1)p_1(d_2) \tag{B.13}$$

$$\mathsf{E}\{\mathrm{Rank}(S_2 \oplus S_a)\} = n(\omega_{s_1}^0 + \omega_{s_1}^1)p_1(d_2, d_1 r) \tag{B.14}$$

where $p_1(d_2, d_1 r)$ in (B.14) is the probability that at least one of node d_2 and node a receives the packet and (B.14) follows from the observation that $S_2 \oplus S_a$ is a subset of the exclusion sets of $\Gamma_{s_1,0}$, $\Gamma_{s_1,1}$ and is a superset of the inclusion set of $\Gamma_{s_1,2}$. By (B.12), (B.13), and (B.14), we have thus proven that

$$\begin{aligned}
&\mathsf{E}\{\mathrm{Rank}(S_1 \cap S_2 \cap S_r)\} = \mathsf{E}\{\mathrm{Rank}(S_a \cap S_2)\} \\
=&\mathsf{E}\{\mathrm{Rank}(S_2)\} + \mathsf{E}\{\mathrm{Rank}(S_a)\} - \mathsf{E}\{\mathrm{Rank}(S_2 \oplus S_a)\} \\
=&n\left(\omega_{s_1}^0 + \omega_{s_1}^1\right) p_1(d_1 d_2 r) + n\omega_{s_1}^2 p_1(d_1 r)
\end{aligned} \tag{B.15}$$

in the end of $\Gamma_{s_1,2}$.

In the following, we will quantify the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during $\Gamma_{s_1,3}$. To that end, we introduce two more auxiliary nodes b and c. In the beginning of $\Gamma_{s_1,3}$, we let node b (resp. c) be aware of the knowledge space $S_1 \cap S_r$ (resp. $S_2 \cap S_r$). During $\Gamma_{s_1,3}$, whenever a packet is received by d_1 (resp. d_2), we let the auxiliary node b (resp. c) observe such a packet as well. From the construction, it is clear that the following equalities hold in the beginning of $\Gamma_{s_1,3}$.

$$S_b = S_1 \cap S_r \tag{B.16}$$

$$S_c = S_2 \cap S_r. \tag{B.17}$$

We will prove that (B.16) and (B.17) hold even in the end of $\Gamma_{s_1,3}$ as well.

In the following, we will prove that (B.16) holds in the end of $\Gamma_{s_1,3}$. We first note that by our construction, we always have $S_1 \supset S_b \supset (S_1 \cap S_r)$. Knowing that S_b is always a subset of S_1 and S_1 is a subset of the exclusion sets in $\Gamma_{s_1,3}$, we can see that everytime d_1 receives a packet during policy $\Gamma_{s_1,3}$, $\mathrm{Rank}(S_b)$ will increase by one. Moreover, only when d_1 receives a packet during policy $\Gamma_{s_1,3}$ will $\mathrm{Rank}(S_b)$ increase. As a result, the increment of $\mathrm{Rank}(S_b)$ during $\Gamma_{s_1,3}$ equals the number of times d_1 receives a packet during $\Gamma_{s_1,3}$. On the other hand, $\mathrm{Rank}(S_1 \cap S_r) = \mathrm{Rank}(S_1) + \mathrm{Rank}(S_r) - \mathrm{Rank}(S_1 \oplus S_r)$. Since both S_r and $S_1 \oplus S_r$ are supersets of the inclusion set of $\Gamma_{s_1,3}$, both $\mathrm{Rank}(S_r)$ and $\mathrm{Rank}(S_1 \oplus S_r)$ remain identical during $\Gamma_{s_1,3}$. Therefore, the increment of $\mathrm{Rank}(S_1 \cap S_r)$ is identical to the increment of $\mathrm{Rank}(S_1)$ during $\Gamma_{s_1,3}$. As a result, the increment of $\mathrm{Rank}(S_1 \cap S_r)$ during $\Gamma_{s_1,3}$ equals the number of times d_1 receives a packet during $\Gamma_{s_1,3}$. We have thus proven $\mathrm{Rank}(S_b) = \mathrm{Rank}(S_1 \cap S_r)$ in the end of $\Gamma_{s_1,3}$, which implies (B.16). (B.17) can be proven by symmetry.

To quantify the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during $\Gamma_{s_1,3}$, we notice that $\mathrm{Rank}(S_1 \cap S_2 \cap S_r) = \mathrm{Rank}(S_b \cap S_c) = \mathrm{Rank}(S_b) + \mathrm{Rank}(S_c) - \mathrm{Rank}(S_b \oplus S_c)$. As a result, the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during policy $\Gamma_{s_1,3}$ is the summation of the increments of $\mathrm{Rank}(S_b)$ and $\mathrm{Rank}(S_c)$ minus the increment of $\mathrm{Rank}(S_b \oplus S_c)$ during $\Gamma_{s_1,3}$. By our construction, the increments of S_b, S_c, and $S_b \oplus S_c$ during $\Gamma_{s_1,3}$ is simply $n\omega_{s_1}^3 p_1(d_1)$,

$n\omega_{s_1}^3 p_1(d_2)$, and $n\omega_{s_1}^3 p_1(d_1, d_2)$, respectively. As a result, the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during $\Gamma_{s_1,3}$ is simply $n\omega_{s_1}^3 p_1(d_1 d_2)$.

Combining (B.15), we have thus proven that

$$\begin{aligned}&\mathsf{E}\{\mathrm{Rank}(S_1 \cap S_2 \cap S_r)\}\\ =&n\left(\omega_{s_1}^0 + \omega_{s_1}^1\right) p_1(d_1 d_2 r) + n\omega_{s_1}^2 p_1(d_1 r) + n\omega_{s_1}^3 p_1(d_1 d_2)\end{aligned} \tag{B.18}$$

in the end of Policy $\Gamma_{s_1,3}$.

Jointly, (B.7) to (B.11), (B.18), and (4.16) imply (B.6) in the end of $\Gamma_{s_1,3}$.

Policy $\Gamma_{s_1,4}$: Similar to the analysis of the previous policies, the condition that (3.5) being non-empty is equivalent to whether the following rank-based inequality is satisfied in the end of $\Gamma_{s_1,4}$.

$$\begin{aligned}&\mathrm{Rank}(S_2 \cap S_r) - \mathrm{Rank}(S_1 \cap S_2 \cap S_2)\\ =&(\mathrm{Rank}(S_2) + \mathrm{Rank}(S_r) - \mathrm{Rank}(S_2 \oplus S_r))\\ &- \mathrm{Rank}(S_1 \cap S_2 \cap S_r) > 0.\end{aligned} \tag{B.19}$$

Similar to the previous discussion, we will quantify individual ranks at the end of $\Gamma_{s_1,4}$ and prove that (B.19) holds in the end of $\Gamma_{s_1,4}$.

By similar analysis, we have

$$\mathsf{E}\{\mathrm{Rank}(S_2)\} = n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^3)p_1(d_2), \tag{B.20}$$

$$\mathsf{E}\{\mathrm{Rank}(S_r)\} = n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2)p_1(r), \tag{B.21}$$

$$\mathsf{E}\{\mathrm{Rank}(S_2 \oplus S_r)\} = n\omega_{s_1}^0 p_1(d_2, r) \tag{B.22}$$

in the end of $\Gamma_{s_1,4}$. What remains to be decided is the value of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ at the end of Policy $\Gamma_{s_1,4}$. In (B.18), we have already quantified $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ in the end of $\Gamma_{s_1,3}$. In the following, we will quantify the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during $\Gamma_{s_1,4}$. By (3.5), we can see that every time d_1 receives a packet during $\Gamma_{s_1,4}$, $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$

will increase by one. As a result, the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during $\Gamma_{s_1,4}$ is $n\omega_{s_1}^4 p_1(d_1)$. Together with (B.18), we have proven that

$$\begin{aligned}
&\mathsf{E}\{\mathrm{Rank}(S_1 \cap S_2 \cap S_r)\} \\
=&n\left(\omega_{s_1}^0 + \omega_{s_1}^1\right) p_1(d_1 d_2 r) + n\omega_{s_1}^2 p_1(d_1 r) \\
&+ n\omega_{s_1}^3 p_1(d_1 d_2) + n\omega_{s_1}^4 p_1(d_1)
\end{aligned} \tag{B.23}$$

in the end of $\Gamma_{s_1,4}$. Jointly, (B.20) to (B.23) and (4.17) imply that (B.19) holds in the end of $\Gamma_{s_1,4}$.

The feasibility of policy $\Gamma_{s_2,k}$, $k = 0, 1, 2, 3, 4$, can be proven by symmetry.

Policy $\Gamma_{r,1}$: We first notice that the inclusion space and exclusion space of Policy $\Gamma_{r,1}$ are the same as of Policy $\Gamma_{s_1,3}$. Hence to prove the feasibility of Policy $\Gamma_{r,1}$, we need to prove that (B.6) holds in the end of $\Gamma_{r,1}$. By similar analysis, we have

$$\begin{aligned}
\mathsf{E}\{\mathrm{Rank}(S_1)\} &= n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2 + \omega_{s_1}^3 + \omega_{s_1}^4) p_1(d_1) \\
&\quad + n\omega_{r,N}^1 p_r(d_1), && \text{(B.24)} \\
\mathsf{E}\{\mathrm{Rank}(S_2)\} &= n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^3) p_1(d_2) \\
&\quad + n\omega_{r,\mathrm{N}}^1 p_r(d_2), && \text{(B.25)} \\
\mathsf{E}\{\mathrm{Rank}(S_r)\} &= n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2) p_1(r), && \text{(B.26)} \\
\mathsf{E}\{\mathrm{Rank}(S_1 \oplus S_r)\} &= n(\omega_{s_1}^0 + \omega_{s_1}^1 + \omega_{s_1}^2) p_1(d_1, r), && \text{(B.27)} \\
\mathsf{E}\{\mathrm{Rank}(S_2 \oplus S_r)\} &= n\omega_{s_1}^0 p_1(d_2, r). && \text{(B.28)}
\end{aligned}$$

in the end of $\Gamma_{r,1}$.

What remains to be decided is the value of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ at the end of Policy $\Gamma_{r,1}$. In (B.23) we have computed the value of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ in the end of $\Gamma_{s_1,4}$. As a result, we only need to quantify the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during $\Gamma_{r,1}$. By the same analysis as when we quantify the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during $\Gamma_{s_1,3}$, the

increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during $\Gamma_{r,1}$ is $n\omega^1_{r,\mathrm{N}} p_r(d_1 d_2)$. By (B.23), we have shown that

$$\begin{aligned}
&\mathsf{E}\{\mathrm{Rank}(S_1 \cap S_2 \cap S_r)\} \\
=&n\left(\omega^0_{s_1} + \omega^1_{s_1}\right) p_1(d_1 d_2 r) + n\omega^2_{s_1} p_1(d_1 r) \\
&+ n\omega^3_{s_1} p_1(d_1 d_2) + n\omega^4_{s_1} p_1(d_1) + n\omega^1_{r,\mathrm{N}} p_r(d_1 d_2)
\end{aligned} \tag{B.29}$$

in the end of $\Gamma_{r,1}$. Jointly, (B.24) to (B.29) and (4.18) imply that (B.6) holds in the end of $\Gamma_{r,1}$.

The discussion of Policy $\Gamma_{r,2}$ follows symmetrically.

Policy $\Gamma_{r,3}$ for $\mathbf{v}^{(1)}$: We will prove that for the first $n\omega^1_{r,\mathrm{C}}$ time slots of Policy $\Gamma_{r,3}$, we can always choose $\mathbf{v}^{(1)}$ according to (3.13). To that end, we first notice that the inclusion space and exclusion space in (3.13) are the same as those of Policy $\Gamma_{s_1,4}$. Hence to prove that (3.13) remains non-empty during the first $n\omega^1_{r,\mathrm{C}}$ time slots of Policy $\Gamma_{r,3}$, we need to prove that (B.19) holds in the end of the first $n\omega^1_{r,\mathrm{C}}$ time slots of Policy $\Gamma_{r,3}$. By similar analysis as used in the previous policies, we have

$$\begin{aligned}
\mathsf{E}\{\mathrm{Rank}(S_2)\} = n(\omega^0_{s_1} + \omega^1_{s_1} + \omega^3_{s_1}) p_1(d_2) \\
+ n\omega^1_{r,\mathrm{N}} p_r(d_2),
\end{aligned} \tag{B.30}$$

$$\mathsf{E}\{\mathrm{Rank}(S_r)\} = n(\omega^0_{s_1} + \omega^1_{s_1} + \omega^2_{s_1}) p_1(r), \tag{B.31}$$

$$\mathsf{E}\{\mathrm{Rank}(S_2 \oplus S_r)\} = n\omega^0_{s_1} p_1(d_2, r). \tag{B.32}$$

in the end of the first $n\omega^1_{r,\mathrm{C}}$ time slots of Policy $\Gamma_{r,3}$. What remains to be decided is the value of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ at the end of the first $n\omega^1_{r,\mathrm{C}}$ time slots of Policy $\Gamma_{r,3}$. In (B.29) we have computed the value of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ in the end of $\Gamma_{r,1}$. As a result, we only need to quantify the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during the first $n\omega^1_{r,\mathrm{C}}$ time slots of Policy $\Gamma_{r,3}$. By the same analysis as when we quantify the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$

during $\Gamma_{s_1,4}$, the increment of $\mathrm{Rank}(S_1 \cap S_2 \cap S_r)$ during the first $n\omega^1_{r,\mathrm{C}}$ time slots of Policy $\Gamma_{r,3}$ is $n\omega^1_{r,\mathrm{C}} p_r(d_1)$. By (B.29), we have shown that

$$\begin{aligned}
&\mathsf{E}\{\mathrm{Rank}(S_1 \cap S_2 \cap S_r)\} \\
=&n\left(\omega^0_{s_1} + \omega^1_{s_1}\right) p_1(d_1 d_2 r) + n\omega^2_{s_1} p_1(d_1 r) \\
&+ n\omega^3_{s_1} p_1(d_1 d_2) + n\omega^4_{s_1} p_1(d_1) + n\omega^1_{r,\mathrm{N}} p_r(d_1 d_2) \\
&+ n\omega^1_{r,\mathrm{C}} p_r(d_1)
\end{aligned} \tag{B.33}$$

in the end of the first $n\omega^1_{r,\mathrm{C}}$ time slots of Policy $\Gamma_{r,3}$. Jointly, (B.30) to (B.33) and (4.19) imply that (B.19) holds in the end of the first $n\omega^1_{r,\mathrm{C}}$ time slots of $\Gamma_{r,3}$.

The discussion of the first $n\omega^2_{r,\mathrm{C}}$ time slots of $\Gamma_{r,3}$ follows symmetrically.

The above analysis completes the achievability proof started in Section 4.3.1.

C. BOUND-MATCHING VERIFICATION

Here we are going to verify the proposed parameter assignment in the proof of *Lemma 4.3.1* satisfies (4.9), (4.10), (4.32) to (4.39). For (4.9),

$$\begin{aligned}\sum_{k=0}^{3}\omega_{s_i}^{k} =& \frac{R_i}{p_i(d_j,r)} \\ &+ R_i\left(\min\left\{\frac{1}{p_i(r)},\frac{1}{p_i(d_j)}\right\}-\frac{1}{p_i(d_j,r)}\right) \\ &+ R_i\left(\frac{1}{p_i(r)}-\frac{1}{p_i(d_j)}\right)^{+} \\ &+ \min\left\{R_i\left(\frac{1}{p_i(d_j)}-\frac{1}{p_i(r)}\right)^{+}, t_{s_i}-\frac{R_i}{p_i(r)}\right\} \\ \leq& \frac{R_i}{p_i(r)} \\ &+ \min\left\{R_i\left(\frac{1}{p_i(d_j)}-\frac{1}{p_i(r)}\right)^{+}, t_{s_i}-\frac{R_i}{p_i(r)}\right\} \\ \leq& t_r.\end{aligned}$$

Hence it satisfies (4.9). For (4.10),

$$\begin{aligned}&\omega_{r,\mathrm{N}}^{i}+\omega_{r,\mathrm{N}}^{j}+\omega_{r,\mathrm{C}}^{i} \\ =&\frac{(R_i-t_{s_i}p_i(d_j))^{+}}{p_r(d_i,d_j)}+\frac{(R_j-t_{s_j}p_{s_j}(d_i))^{+}}{p_r(d_i,d_j)} \\ &+\frac{R_i}{p_r(d_i)}-\frac{(R_i-t_{s_i}p_i(d_j))^{+}}{p_r(d_i,d_j)} \\ =&\frac{R_i}{p_r(d_i)}+\frac{(R_j-t_{s_j}p_{s_j}(d_i))^{+}}{p_r(d_i,d_j)}\leq t_r.\end{aligned}$$

Hence it satisfies (4.10). For (4.32),

$$\omega_{s_i}^{0}p_i(d_j,r)=\frac{R_i}{p_i(d_j,r)}p_i(d_j,r)=R_i.$$

Hence it satisfies (4.32). For (4.33),

$$\begin{aligned}
\text{LHS} =& R_i \left(\min \left\{ \frac{1}{p_i(r)}, \frac{1}{p_i(d_j)} \right\} - \frac{1}{p_i(d_j, r)} \right) p_i(r) \\
=& R_i \left(\min \left\{ 1, \frac{p_i(r)}{p_i(d_j)} \right\} - \frac{p_i(r)}{p_i(d_j, r)} \right) \\
\leq& R_i - R_i \frac{p_i(r)}{p_i(d_j, r)}, \\
\text{RHS} =& R_i \frac{p_i(d_j \overline{r})}{p_i(d_j, r)} = R_i - R_i \frac{p_i(r)}{p_i(d_j, r)}.
\end{aligned}$$

Hence it satisfies (4.33). For (4.34),

$$\begin{aligned}
\text{LHS} =& R_i \left(\min \left\{ \frac{1}{p_i(r)}, \frac{1}{p_i(d_j)} \right\} - \frac{1}{p_i(d_j, r)} \right) p_i(d_j) \\
=& R_i \left(\min \left\{ \frac{p_i(d_j)}{p_i(r)}, 1 \right\} - \frac{p_i(d_j)}{p_i(d_j, r)} \right) \\
\leq& R_i - R_i \frac{p_i(d_j)}{p_i(d_j, r)}, \\
\text{RHS} =& R_i \frac{p_i(r \overline{d_j})}{p_i(d_j, r)} = R_i - R_i \frac{p_i(d_j)}{p_i(d_j, r)}.
\end{aligned}$$

Hence it satisfies (4.34). For (4.35),

$$\begin{aligned}
\text{LHS} =& R_i \left(\frac{1}{p_i(r)} - \frac{1}{p_i(d_j)} \right)^+ p_i(r) = R_i \left(1 - \frac{p_i(r)}{p_i(d_j)} \right)^+, \\
\text{RHS} =& R_i \frac{p_i(d_j \overline{r})}{p_i(d_j, r)} \\
& - R_i \left(\min \left\{ \frac{1}{p_i(r)}, \frac{1}{p_i(d_j)} \right\} - \frac{1}{p_i(d_j, r)} \right) p_i(r) \\
=& R_i \left(1 - \frac{p_i(r)}{p_i(d_j, r)} \right) \\
& - R_i \left(\min \left\{ 1, \frac{p_i(r)}{p_i(d_j)} \right\} - \frac{p_i(r)}{p_i(d_j, r)} \right) \\
\geq& R_i \left(1 - \frac{p_i(r)}{p_i(d_j)} \right)^+.
\end{aligned}$$

Hence it satisfies (4.35). For (4.36),

$$\begin{aligned}
\text{LHS} =& \min\left\{R_i\left(1-\frac{p_i(d_j)}{p_i(r)}\right)^+, t_{s_i}p_i(d_j)-R_i\frac{p_i(d_j)}{p_i(r)}\right\},\\
\text{RHS} =& R_i\frac{p_i(r\overline{d_j})}{p_i(d_j,r)}\\
&-R_i\left(\min\left\{\frac{1}{p_i(r)},\frac{1}{p_i(d_j)}\right\}-\frac{1}{p_i(d_j,r)}\right)p_i(d_j)\\
=& R_i\left(1-\frac{p_i(d_j)}{p_i(d_j,r)}\right)\\
&-R_i\left(\min\left\{\frac{p_i(d_j)}{p_i(r)},1\right\}-\frac{p_i(d_j)}{p_i(d_j,r)}\right)\\
\geq& R_i\left(1-\frac{p_i(d_j)}{p_i(r)}\right)^+.
\end{aligned}$$

Hence it satisfies (4.36). For (4.37),

$$\begin{aligned}
\text{LHS} =& \left(R_i-t_{s_i}p_i(d_j)\right)^+,\\
\text{RHS} =& R_i\frac{p_i(r\overline{d_j})}{p_i(d_j,r)}\\
&-R_i\left(\min\left\{\frac{1}{p_i(r)},\frac{1}{p_i(d_j)}\right\}-\frac{1}{p_i(d_j,r)}\right)p_i(d_j)\\
&-\min\left\{R_i\left(1-\frac{p_i(d_j)}{p_i(r)}\right)^+, t_{s_i}p_i(d_j)-R_i\frac{p_i(d_j)}{p_i(r)}\right\}\\
\geq& R_i\left(1-\frac{p_i(d_j)}{p_i(r)}\right)^+\\
&-\min\left\{R_i\left(1-\frac{p_i(d_j)}{p_i(r)}\right)^+, t_{s_i}p_i(d_j)-R_i\frac{p_i(d_j)}{p_i(r)}\right\}\\
\geq& \left(R_i-t_{s_i}p_i(d_j)\right)^+.
\end{aligned}$$

Hence it satisfies (4.37). For (4.38),

$$
\begin{aligned}
\text{LHS} =& R_i - (R_i - t_{s_i} p_i(d_j))^+ \frac{p_r(d_i)}{p_r(d_i, d_j)}, \\
\text{RHS} =& R_i \frac{p_i(d_j r)}{p_i(d_j, r)} \\
& + (p_i(d_j) + p_i(r)) \\
& \cdot R_i \left(\min \left\{ \frac{1}{p_i(r)}, \frac{1}{p_i(d_j)} \right\} - \frac{1}{p_i(d_j, r)} \right) \\
& + R_i \left(1 - \frac{p_i(r)}{p_i(d_j)} \right)^+ \\
& + \min \left\{ R_i \left(1 - \frac{p_i(d_j)}{p_i(r)} \right)^+, t_{s_i} p_i(d_j) - R_i \frac{p_i(d_j)}{p_i(r)} \right\} \\
& + \frac{(R_i - t_{s_i} p_i(d_j))^+}{p_r(d_i, d_j)} (p_r(d_i, d_j) - p_r(d_i)) \\
=& R_i \min \left\{ \frac{p_i(d_j)}{p_i(r)}, \frac{p_i(r)}{p_i(d_j)} \right\} + R_i \left(1 - \frac{p_i(r)}{p_i(d_j)} \right)^+ \\
& + \min \left\{ R_i \left(1 - \frac{p_i(d_j)}{p_i(r)} \right)^+, t_{s_i} p_i(d_j) - R_i \frac{p_i(d_j)}{p_i(r)} \right\} \\
& + (R_i - t_{s_i} p_i(d_j))^+ (1 - \frac{p_r(d_i)}{p_r(d_i, d_j)}) \\
\geq & R_i - (R_i - t_{s_i} p_i(d_j))^+ \frac{p_r(d_i)}{p_r(d_i, d_j)}.
\end{aligned}
$$

Hence it satisfies (4.38). For (4.39),

$$p_r(d_i)(\omega^i_{r,\mathrm{N}} + \omega^i_{r,\mathrm{C}}) = R_i$$

With the above verification, we conclude that this proposed assignment satisfies (4.9), (4.10), (4.32) to (4.39).

D. THE GROWTH RATE OF DEFICIT

In this appendix, we analyze the growth rate of the deficit in NM 2. Before proving Lemma 6.2.7, we need several lemmas. Let $p_k(t) \stackrel{\triangle}{=} -q_k(t) + \mu_{in,k}(t-1)$ and $p_k(t)$ grows sublinearly by the properties in Section 5.1.1. We notice that $D_k(t)$ is the running maximum of $p_k(t)$ because by (6.14)

$$\begin{aligned} D_k(t) &= D_k(t-1) + (\mu_{out,k}(t-1) - Q_k(t-1))^+ \\ &= D_k(t-1) + \max\{0, \mu_{out,k}(t-1) - Q_k(t-1)\} \\ &= \max\{D_k(t-1), -q_k(t-1) + \mu_{out,k}(t-1)\} \\ &= \max\{D_k(t-1), -q_k(t) + \mu_{in,k}(t-1)\}. \end{aligned} \tag{D.1}$$

Let $\overline{p_k}(t) \stackrel{\triangle}{=} -\overline{q_k}(t) + \overline{\mu_{in,k}}(t-1)$ and $p'_k(t) \stackrel{\triangle}{=} p_k(t) - \overline{p_k}(t)$. Note that $\overline{p_k}(t)$ is stable by Lemma 5.1.1 and $p'_k(t)$ grows sublinearly by Lemma 5.1.2 and Lemma 5.1.3. Let $D'_k(t)$ be the running maximum of the $p'_k(t)$ and $\overline{D_k}(t)$ be the running maximum of $\overline{p_k}(t)$. That is,

$$D'_k(t) = \max_{1 \le \tau \le t} p'_k(\tau),$$

$$\overline{D_k}(t) = \max_{1 \le \tau \le t} \overline{p_k}(\tau).$$

Let $T'_k(b) \stackrel{\triangle}{=} \min\{t \ge 0 : p'_k(t) \ge b\}$ be the hitting time of $p'_k(t)$.

Claim D.0.1 *There exist $C > 0$ for all arrival vector realizations such that for all t with* $\mathsf{Prob}(T'_k(b) \le t | \{\mathbf{a}(\tau)\}_{\tau=0}^t) \neq 0$,

$$\mathsf{Prob}(p'_k(t) \ge b | T'_k(b) \le t, \{\mathbf{a}(\tau)\}_{\tau=0}^t) > C. \tag{D.2}$$

Proof Let $\Delta\mu_{in,k}(t) \stackrel{\triangle}{=} \mu_{in,k}(t) - \overline{\mu_{in,k}}(t)$, $\Delta\mu_{out,k}(t) \stackrel{\triangle}{=} \mu_{out,k}(t) - \overline{\mu_{out,k}}(t)$, and $\Delta\mu_k(t) \stackrel{\triangle}{=} \Delta\mu_{in,k}(t) - \Delta\mu_{out,k}(t)$. By definition

$$\begin{aligned} q_k(t) &= \sum_{\tau=0}^{t-1}(\mu_{in,k}(\tau) - \mu_{out,k}(\tau)), \\ \overline{q_k}(t) &= \sum_{\tau=0}^{t-1}(\overline{\mu_{in,k}}(\tau) - \overline{\mu_{out,k}}(\tau)), \\ q_k(t) - \overline{q_k}(t) &= \sum_{\tau=0}^{t-1}\Delta\mu_k(\tau). \end{aligned}$$

Then

$$\begin{aligned} p_k(t) &= -\sum_{\tau=0}^{t-2}(\mu_{in,k}(\tau) - \mu_{out,k}(\tau)) + \mu_{out,k}(t-1), \\ \overline{p_k}(t) &= -\sum_{\tau=0}^{t-2}(\overline{\mu_{in,k}}(\tau) - \overline{\mu_{out,k}}(\tau)) + \overline{\mu_{out,k}}(t-1), \\ p'_k(t) &= \sum_{\tau=0}^{t-2}\Delta\mu_k(\tau) + \Delta\mu_{out,k}(t-1). \end{aligned}$$

We first notice that conditioning on $T'_k(b) \le t$ and the arrival vectors $\{\mathbf{a}(\tau)\}_{\tau=0}^{t}$, $p'_k(t) \ge b$ implies

$$\begin{aligned} & p'_k(t) - p'_k(T'_k(b)) \ge 0 \\ \Rightarrow & \left(\sum_{\tau=0}^{t-2}\Delta\mu_k(\tau) + \Delta\mu_{out,k}(t-1)\right) - \left(\sum_{\tau=0}^{T'_k(b)-2}\Delta\mu_k(\tau) + \Delta\mu_{out,k}(T'_k(b)-1)\right) \\ = & \sum_{\tau=T'_k(b)}^{t-2}\Delta\mu_k(\tau) + (\Delta\mu_{out,k}(t-1) + \Delta\mu_{in,k}(T'_k(b)-1) - 2\Delta\mu_{out,k}(T'_k(b)-1)) \ge 0. \end{aligned}$$

Thus

$$\begin{aligned} & \mathsf{Prob}(p'_k(t) \ge b | T'_k(b) \le t, \{\mathbf{a}(\tau)\}_{\tau=0}^{t}) \\ = & \mathsf{Prob}\Bigg(\sum_{\tau=T'_k(b)}^{t-2}\Delta\mu_k(\tau) + (\Delta\mu_{out,k}(t-1) + \Delta\mu_{in,k}(T'_k(b)-1) \\ & -2\Delta\mu_{out,k}(T'_k(b)-1)) \ge 0 | T'_k(b) \le t, \{\mathbf{a}(\tau)\}_{\tau=0}^{t}\Bigg) \end{aligned} \tag{D.3}$$

We then use the fact that conditioning on the arrival vectors, all the random variables in (D.3) (in front of the conditioning part) are zero-mean, bounded, independent with each other, and with finitely many different distributions.[1] Thus (D.3) can be seen as

$$\mathsf{Prob}(\sum_{l=1}^{L} X_l \geq 0) \tag{D.4}$$

where L is an arbitrary finite number and $\{X_l\}_{l=1}^{L}$ are zero-mean, bounded, and independent random variables with finitely many different distributions. Thus to prove this claim, it is equivalent to show that for any L, there exists $C > 0$ such that $\mathsf{Prob}(\sum_{l=1}^{L} X_l \geq 0) > C$.

We first assume that X_l are not only independently distributed but actually i.i.d. By the central limit theorem, there exists a l_0 such that when $L > l_0$, the probability of interest is $> 1/4$. Choose $C = \min(\min\{\mathsf{Prob}(\sum_{l=1}^{L} X_i \geq 0) : t \leq t_0\}, 1/4)$. Note that $\min\{\mathsf{Prob}(\sum_{l=1}^{L} X_i \geq 0) : t \leq t_0\} \neq 0$ because of the zero-mean assumption and hence $C > 0$.

Now, we consider the case that X_i are not identically distributed but with finitely many different distributions. Let K be the number of different distributions and L_k be the number of random variables with distribution k. We group those X_l, $l = 1, 2, ..., L$ with the same distribution and denote them by $\{X_{k,l_k}\}_{l_k=1}^{L_k}$ for $k = 1, 2, ..., K$. We can write

$$\begin{aligned}
&\mathsf{Prob}(\sum_{l=1}^{L} X_l) \\
=&\mathsf{Prob}(\sum_{l_1=1}^{L_1} X_{1,l_1} + \sum_{l_2=1}^{L_2} X_{2,l_2} + ... + \sum_{l_K=1}^{l_K} X_{K,L_K} \geq 0) \\
\geq&\mathsf{Prob}(\sum_{l_k=1}^{L_k} X_{k,l_k} \geq 0, \forall k) \\
=&\prod_{k=1}^{K} \mathsf{Prob}(\sum_{l_k=1}^{L_k} X_{k,l_k} \geq 0).
\end{aligned} \tag{D.5}$$

We have shown that for each k, there exists C_k such that $\mathsf{Prob}(\sum_{l_k=1}^{L_k} X_{k,l_k} \geq 0) > C_k$. Hence the product in (D.5) is larger than $C \stackrel{\triangle}{=} \prod_{k=1}^{K} C_k$. Notice that the arrival vectors only influence the distribution of $\{L_1, L_2, ..., L_K\}$ and C_k is valid for all possible L_k. Hence this

[1] This is because there only exist finitely many choices of scheduling decision.

C is valid for all possible arrival vector realization. This completes the proof of this claim. ■

Lemma D.0.1 *$D'_k(t)$ grows sublinearly.*

Proof Notice that by Claim D.0.1, there exists C for all arrival vector realizations such that

$$\begin{aligned}&\mathsf{Prob}(p'_k(t) \geq b | T'_k(b) \leq t, \{\mathbf{a}(\tau)\}_{\tau=0}^t)\\ =&\frac{\mathsf{Prob}(p'_k(t) \geq b, T'_k(b) \leq t | \{\mathbf{a}(\tau)\}_{\tau=0}^t)}{\mathsf{Prob}(T'_k(b) \leq t | \{\mathbf{a}(\tau)\}_{\tau=0}^t)}\\ =&\frac{\mathsf{Prob}(p'_k(t) \geq b | \{\mathbf{a}(\tau)\}_{\tau=0}^t)}{\mathsf{Prob}(T'_k(b) \leq t | \{\mathbf{a}(\tau)\}_{\tau=0}^t)} > C \end{aligned} \tag{D.6}$$

Meanwhile, since $D'_k(t)$ is the running maximum of $p'_k(t)$,

$$\mathsf{Prob}(D'_k(t) \geq b | \{\mathbf{a}(\tau)\}_{\tau=0}^t) = \mathsf{Prob}(T'_k(b) \leq t | \{\mathbf{a}(\tau)\}_{\tau=0}^t) < \frac{1}{C}\mathsf{Prob}(p'_k(t) \geq b | \{\mathbf{a}(\tau)\}_{\tau=0}^t) \tag{D.7}$$

Taking the expectation on both side over all possible arrival vectors

$$\mathsf{Prob}(D'_k(t) \geq b) < \frac{1}{C}\mathsf{Prob}(p'_k(t) \geq b) \tag{D.8}$$

Substituting b by ϵt in the above equation and using the fact that $p'_k(t)$ grows sublinearly, $D'_k(t)$ grows sublinearly. The proof is complete. ■

Claim D.0.2 *The following two inequalities are true for all possible realizations.*
(i)$\overline{D_k}(t+1)^2 - \overline{D_k}(t)^2 \leq \max\{\overline{p_k}(t+1)^2 - \overline{p_k}(t)^2, 0\} + \Delta^2$*, where Δ is the supremum over all possible $|\overline{\mu_{out,k}}(t) - \overline{\mu_{in,k}}(t-1)|$ (which is also the supremum over all possible $|\overline{p_k}(t+1) - \overline{p_k}(t)|$).*
(ii)$\max\{\overline{p_k}(t+1)^2 - \overline{p_k}(t)^2, 0\} + \Delta^2 \leq 2|\overline{p_k}(t)|\Delta + 2\Delta^2$.

Proof We first prove (i). There are three possible cases.
Case 1: $\overline{D_k}(t) \geq \overline{p_k}(t+1)$. Since $\overline{D_k}(t)$ is the running maximum of $\overline{p_k}(t)$, $\overline{D_k}(t+1) = \overline{D_k}(t)$ in this case. Thus the left hand side of (i) is zero and the inequality holds.

Case 2: $\overline{D_k}(t) < \overline{p_k}(t+1)$ and $\overline{p_k}(t) \geq 0$. As a result of the definitions of $\overline{D_k}(t)$ and Δ, this condition also implies $\overline{D_k}(t+1) = \overline{p_k}(t+1)$ and

$$\overline{D_k}(t) - \Delta \leq \overline{p_k}(t) \leq \overline{D_k}(t) < \overline{p_k}(t+1).$$

Hence $\overline{D_k}(t+1)^2 - \overline{D_k}(t)^2 \leq \overline{p_k}(t+1)^2 - \overline{p_k}(t)^2 \leq \max\{\overline{p_k}(t+1)^2 - \overline{p_k}(t)^2, 0\} + \Delta^2$.
Case 3: $\overline{D_k}(t) < \overline{p_k}(t+1)$ and $\overline{p_k}(t) < 0$. Since $\overline{D_k}(t)$ is always no less than zero and $-\Delta \leq \overline{p_k}(t) < 0$ implies $\Delta^2 - \overline{p_k}(t)^2 \geq 0$, thus $\overline{D_k}(t+1)^2 - \overline{D_k}(t)^2 \leq \overline{p_k}(t+1)^2 \leq \max\{\overline{p_k}(t+1)^2 - \overline{p_k}(t)^2, 0\} + \Delta^2$.

We then prove (ii). Let $\Delta\overline{p_k}(t) \stackrel{\triangle}{=} \overline{p_k}(t) - \overline{p_k}(t-1)$. Then

$$\begin{aligned}
&\max\{\overline{p_k}(t+1)^2 - \overline{p_k}(t)^2, 0\} + \Delta^2 \\
=&\max\{(\overline{p_k}(t) + \Delta\overline{p_k}(t+1))^2 - \overline{p_k}(t)^2, 0\} + \Delta^2 \\
=&\max\{2\overline{p_k}(t)\Delta\overline{p_k}(t+1))^2 + \Delta\overline{p_k}(t+1)^2, 0\} + \Delta^2 \\
\leq&2|\overline{p_k}(t)|\Delta + 2\Delta^2.
\end{aligned}$$

■

Lemma D.0.2 *$\overline{D_k}(t)$ grows sublinearly.*

Proof Following from Claim D.0.2 and taking the expectation on both sides over all possible arrival vectors,

$$\mathsf{E}\{\overline{D_k}(t+1)^2\} - \mathsf{E}\{\overline{D_k(t)^2}\} \leq 2\mathsf{E}\{|\overline{p_k}(t)|\}\Delta + 2\Delta^2.$$

Iteratively summing up from $\tau = 0$ to $\tau = t-1$,

$$\begin{aligned}
\mathsf{E}\{\overline{D_k}(t)^2\} &\leq 2\Delta \sum_{\tau=0}^{t-1} \mathsf{E}\{|\overline{p_k}(t)|\} + 2\Delta^2 t \\
\Rightarrow \frac{1}{t}\mathsf{E}\{\overline{D_k}(t)^2\} &\leq 2\Delta \frac{1}{t} \sum_{\tau=0}^{t-1} \mathsf{E}\{|\overline{p_k}(t)|\} + 2\Delta^2.
\end{aligned}$$

The fact that $p_k(t)$ is stable implies $\frac{1}{t}\sum_{\tau=0}^{t-1} \mathsf{E}\{|\overline{p_k}(t)|\}$ is bounded for all t, say by U. For any $\epsilon > 0$, $\delta > 0$, we then apply the Markov inequality,

$$\mathsf{Prob}(\overline{D_k}(t) > \epsilon t) \leq \frac{1}{\epsilon^2 t^2}\mathsf{E}\{\overline{D_k}(t)^2\} \leq \frac{1}{\epsilon^2 t}\left(2\Delta U + 2\Delta^2\right).$$

Let t_0 be the smallest t such that $\frac{1}{\epsilon^2 t}\left(2\Delta U + 2\Delta^2\right) < \delta$. Then $\mathsf{Prob}(\overline{D_k}(t) > \epsilon t) < \delta$ for all $t > t_0$. ■

Proof of Lemma 6.2.7 Notice that $D_k(t)$ is the running maximum of $p_k(t)$ and $p_k(t) = p'_k(t) + \overline{p_k}(t)$. Hence $D_k(t) \leq D'_k(t) + \overline{D_k}(t)$. From Lemma D.0.1 and Lemma D.0.2, it implies $D_k(t)$ also grows sublinearly.

E. THE GROWTH RATE OF FICTITIOUS PACKETS

In this appendix, we prove Proposition 6.2.2. For each n, there are two possible reasons that SA n generates f.p. at time t. The first one is the null activity (N.A.) occurs at SA n at time t. That is, there exists one queue $k \in \mathfrak{I}_n$ such that $Q_k(t) < \mu_{out,k}(t)$. By the property of $D_k(t)$, it is equivalent to say $D_k(t) > D_k(t-1)$. Thus $\eta_{fp,n}(t)$ generated from the first reason can be bounded by

$$\sum_{\tau=1}^{t} \sum_{k \in \mathfrak{I}_n} I\{D_k(\tau) > D_k(\tau - 1)\}.$$

The second one is that SA n takes a f.p. from one of the queue $k \in \mathfrak{I}_n$. To analyze the second one, we define a binary random variable $I_{k_1,k_2}(t_1, t_2) = 1$ if there is a f.p. entering queue k_2 at time t_2 which is generated from the f.p. entering queue k_1 at time t_1, and $I_{k_1,k_2}(t_1, t_2) = 0$ otherwise. Intuitively speaking, $I_{k_1,k_2}(t_1, t_2)$ depends on the topology of NM 2 and the route that the f.p. passes through, and hence it depends on the network status $B(\tau)$ and the scheduling decisions $x^*(\tau)$ for $t_1 \leq \tau \leq t_2$. Thus the f.p. generated from the second reason can be bounded by

$$\sum_{t_2=1}^{t} \sum_{k \in \mathfrak{I}_n} \sum_{k' \neq k} \sum_{t_1=1}^{t} I_{k',k}(t_1, t_2) I\{D_{k'}(t_1) > D_{k'}(t_1 - 1)\}.$$

Combining the numbers of f.p. generated from both reasons, we have

$$\begin{aligned}
\eta_{fp,n}(t) \leq & \sum_{\tau=1}^{t} \sum_{k \in \mathfrak{I}_n} I\{D_k(\tau) > D_k(\tau - 1)\} \\
& + \sum_{t_2=1}^{t} \sum_{k \in \mathfrak{I}_n} \sum_{k' \neq k} \sum_{t_1=1}^{t} I_{k',k}(t_1, t_2) I\{D_{k'}(t_1) > D_{k'}(t_1 - 1)\} \\
= & \sum_{k \in \mathfrak{I}_n} \sum_{\tau=1}^{t} I\{D_k(\tau) > D_k(\tau - 1)\} \\
& + \sum_{k \in \mathfrak{I}_n} \sum_{k' \neq k} \sum_{t_2=1}^{t} \sum_{t_1=1}^{t} I_{k',k}(t_1, t_2) I\{D_{k'}(t_1) > D_{k'}(t_1 - 1)\}. \qquad \text{(E.1)}
\end{aligned}$$

Notice that the first summation in (E.1) grows sublinearly by Proposition 6.2.1. Hence it remains to show that for each $k \neq k'$, $\sum_{t_2=1}^{t}\sum_{t_1=1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\}$ also grows sublinearly.

Given any $\epsilon > 0$ and $\delta > 0$, by Markov inequality,

$$
\begin{aligned}
&\mathsf{Prob}\left(\sum_{t_2=1}^{t}\sum_{t_1=1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\} > \epsilon t\right)\\
\leq&\frac{1}{\epsilon t}\mathsf{E}\left\{\sum_{t_2=1}^{t}\sum_{t_1=1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\}\right\}\\
=&\frac{1}{\epsilon t}\sum_{t_1=1}^{t}\mathsf{E}\left\{\sum_{t_2=t_1+1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\}\right\}\\
=&\frac{1}{\epsilon t}\sum_{t_1=1}^{t}\mathsf{E}\left\{\mathsf{E}\left\{\sum_{t_2=t_1+1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\}|I\{D'_k(t_1) > D'_k(t_1-1)\}\right\}\right\}\\
=&\frac{1}{\epsilon t}\sum_{t_1=1}^{t}(\mathsf{Prob}(I\{D'_k(t_1) > D'_k(t_1-1)\} = 1)\\
&\cdot\mathsf{E}\left\{\sum_{t_2=t_1+1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\}|I\{D'_k(t_1) > D'_k(t_1-1)\} = 1\right\}\\
&+\mathsf{Prob}(I\{D'_k(t_1) > D'_k(t_1-1)\} = 0)\\
&\cdot\mathsf{E}\left\{\sum_{t_2=t_1+1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\}|I\{D'_k(t_1) > D'_k(t_1-1) = 0\right\}\Bigg)\\
=&\frac{1}{\epsilon t}\sum_{t_1=1}^{t}\mathsf{Prob}(I\{D'_k(t_1) > D'_k(t_1-1)\} = 1)\\
&\cdot\mathsf{E}\left\{\sum_{t_2=t_1+1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\}|I\{D'_k(t_1) > D'_k(t_1-1)\} = 1\right\}
\end{aligned}
\tag{E.2}
$$

$\mathsf{E}\left\{\sum_{t_2=t_1+1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\}|I\{D'_k(t_1) > D'_k(t_1-1)\} = 1\right\}$ is the expected number of f.p. received by queue k up to t, which come from the N.A. occurred at

queue k' at t_1. Notice that NM 1 is acyclic and so is NM 2. Hence this amount is always bounded, say c. Following (E.2), we have

$$\begin{aligned}
&\mathsf{Prob}\left(\sum_{t_2=1}^{t}\sum_{t_1=1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\} > \epsilon t\right)\\
\leq&\frac{c}{\epsilon t}\sum_{t_1=1}^{t}\mathsf{Prob}(I\{D'_k(t_1) > D'_k(t_1-1)\} = 1)\\
=&\frac{c}{\epsilon t}\sum_{t_1=1}^{t}\mathsf{E}\{I\{D'_k(t_1) > D'_k(t_1-1)\}\}\\
=&\frac{c}{\epsilon t}\mathsf{E}\left\{\sum_{t_1=1}^{t} I\{D'_k(t_1) > D'_k(t_1-1)\}\right\}. \qquad \text{(E.3)}
\end{aligned}$$

By Proposition 6.2.1, $n_k(t) = \sum_{t_1=1}^{t} I\{D'_k(t_1) > D'_k(t_1-1)\}$ grows sublinearly. Hence there exists $\epsilon' > 0$, $\delta' > 0$, and t_0 with $(\epsilon' + \delta') < \frac{\delta\epsilon}{c}$ such that $\mathsf{Prob}(n_k(t) > \epsilon' t) < \delta'$ for all $t > t_0$. Furthermore, by definition, $n_k(t) \leq t$ and hence

$$\begin{aligned}
&\mathsf{E}\{n_k(t)\} = \mathsf{Prob}(n_k(t) > \epsilon' t)\mathsf{E}\{n_k(t)|n_k(t) > \epsilon' t\} + \mathsf{Prob}(n_k(t) \leq \epsilon' t)\mathsf{E}\{n_k(t)|n_k(t) \leq \epsilon' t\}\\
\leq&\delta' t + \epsilon' t = (\delta' + \epsilon')t,\ \forall t > t_0.
\end{aligned}$$

Substituting $\mathsf{E}\{n_k(t)\}$ back to (E.3) and yields,

$$\begin{aligned}
&\mathsf{Prob}\left(\sum_{t_2=1}^{t}\sum_{t_1=1}^{t} I_{k',k}(t_1,t_2)I\{D'_k(t_1) > D'_k(t_1-1)\} > \epsilon t\right)\\
\leq&\frac{c}{\epsilon t}\cdot(\epsilon' + \delta')t\\
=&\frac{c(\epsilon' + \delta')}{\epsilon}\\
<&\frac{c}{\epsilon}\cdot\frac{\delta\epsilon}{c} = \delta,\ \forall t > t_0.
\end{aligned}$$

This completes the proof.

F. THE THROUGHPUT REGION OF 5-TYPE LNC SCHEME

In this appendix, we prove Proposition 6.3.1 to Proposition 6.3.3. For Proposition 6.3.3, following the continuity of the linear equations describing the stability region of SPN_ϵ, we have

$$\Lambda_{\text{LNC}} = \lim_{\epsilon \to 0} \Lambda_\epsilon = \Lambda_0.$$

To prove Proposition 6.3.1 and Proposition 6.3.2, it is enough to prove the following two statements: (1) For any $\epsilon > 0$, the effective throughput of any scheme in SPN_ϵ is less than the throughput of another scheme in 5-type LNC 2-user broadcast PEC; and (2) The throughput of any scheme in 5-type LNC 2-user broadcast PEC is less than the effective throughput of another scheme in SPN_0.

To prove the first statement, for any $\epsilon > 0$, we begin with one arbitrary scheme in SPN_ϵ and propose another scheme in LNC problem as follows, named Scheme A. For any time t, whenever we schedule SA_i, if TYPE_i is non-empty, then we also choose the coding vector in TYPE_i and send it out, and we send zero packet if TYPE_i is empty for $i \in \{9, 18, 31, 63, 95\}$. It remains to show that TYPE_i is always nonempty whenever SA_i is scheduled.

We show this by claiming that for any time t and $i \in \{1, 2, 3, 4\}$, $Q_{i,\epsilon}(t) \leq Q_{i,\text{LNC}}(t)$ under Scheme A. We prove this claim iteratively. The statement is trivially true at $t = 0$. Suppose $Q_{i,\epsilon}(t) \leq Q_{i,\text{LNC}}(t)$ at time t. For time $t+1$, we first discuss $i = 1$. Since $a_1(t) = A_1(t)$ for all t, the number of flow-1 packets injected into $Q_{1,\epsilon}$ is the same as the increment in $Q_{1,\text{LNC}}(t) = \text{Rank}(\Omega(t)) - \text{Rank}(A_7(t))$. Suppose the scheduler choose to activate SA_{18} at time t. Since $Q_{i,\epsilon}(t) \leq Q_{i,\text{LNC}}(t)$, we can choose TYPE_{18} at time t as well. And hence $Q_{1,\text{LNC}}(t)$ decreases by one if $\mathbf{Z}_s(t) \in \{(1,1),(1,0),(0,1)\}$. By the random service distribution described before this proposition, $Q_{1,\text{LNC}}(t)$ also decreases by one if $\mathbf{Z}_s(t) \in \{(1,1),(1,0),(0,1)\}$. And hence $Q_{1,\epsilon}(t+1) \leq Q_{1,\text{LNC}}(t+1)$. We now

discuss $i = 3$. $Q_{3,\epsilon}(t+1)$ can increase by one only when SA_{18} is scheduled at time t and $\mathbf{Z}_s(t) = (0,1)$, while $Q_{3,\text{LNC}}(t+1)$ can increase by one only when TYPE_{18} is scheduled at time t and $\mathbf{Z}_s(t) = (0,1)$. Thus the increments are the same. The discussion for scheduling SA_{63} is the same as scheduling SA_{18} in $Q_{1,\epsilon}$. Suppose SA_{31} is scheduled at time t. Since $Q_{i,\epsilon}(t) \leq Q_{i,\text{LNC}}(t)$, we can choose TYPE_{31} at time t as well. $Q_{3,\epsilon}(t+1)$ decrease by one when $\mathbf{Z}_s(t) \in \{(1,0),(1,1)\}$ and by ϵ when $\mathbf{Z}_s(t) = (0,1)$, while $Q_{3,\text{LNC}}(t+1)$ decrease by one when $\mathbf{Z}_s(t) \in \{(1,0),(1,1)\}$. And hence $Q_{3,\epsilon}(t+1) \leq Q_{3,\text{LNC}}(t+1)$. For $i = 2,4$, they are symmetric to $i = 1,3$. And hence we complete the proof of the claim and the first statement.

To prove the second statement, we begin with one arbitrary scheme in LNC problem and propose another scheme in SPN_0 as follows, named Scheme b. For any time t and $i \in \{9,18,31,63,95\}$, whenever we choose TYPE_i, we also schedule SA_i if there is no underflow, and don't schedule any SA if there is underflow for SA_i. It remains to show that there is always no underflow when TYPE_i is scheduled. We show this by claiming that for any time t and $i \in \{1,2,3,4\}$, $Q_{i,\epsilon}(t) = Q_{i,\text{LNC}}(t)$ under Scheme B. The proof of the claim is almost identical to the one for the first claim above. The only difference is when discussing $Q_{3,\text{LNC}}(t+1)$ and choosing TYPE_{31}. $Q_{3,\text{LNC}}(t+1)$ decrease by one when $\mathbf{Z}_s(t) \in \{(1,0),(1,1)\}$, while $Q_{3,\mathbf{0}}(t+1)$ decrease by one when $\mathbf{Z}_s(t) \in \{(1,0),(1,1)\}$. And hence we complete the proof the second claim and the second statement. The proof of the proposition is thus completed.

LIST OF REFERENCES

LIST OF REFERENCES

[1] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *Information Theory, IEEE Transactions on*, vol. 46, no. 4, pp. 1204–1216, 2000.

[2] S.-Y. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, pp. 371–381, Feb 2003.

[3] L. Georgiadis and L. Tassiulas, "Broadcast erasure channel with feedback — capacity and algorithms," in *Proc. 5th Workshop on Network Coding, Theory, & Applications (NetCod)*, (Lausanne, Switzerland), pp. 54–61, June 2009.

[4] C.-C. Wang, "On the capacity of 1-to-K broadcast packet erasure channels with channel output feedback," *IEEE Trans. Inf. Theory*, vol. 58, pp. 931–956, Feb 2012.

[5] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: Practical wireless network," in *Proc. ACM Special Interest Group on Data Commun. (SIGCOMM)*, 2006.

[6] P. Chaporkar and A. Proutiere, "Adaptive network coding and scheduling for maximizing throughput in wireless networks," in *Proc. Ann. ACM Int. Conf. on Mobile Computing & Networking (MobiCom)*, (Montreal, QC, Canada), pp. 135–146, Sept 2007.

[7] T. Cui, L. Chen, and T. Ho, "Energy efficient opportunistic network coding for wireless networks," in *Proc. 27th IEEE Conf. Computer Communications (INFOCOM)*, (Phoenix, AZ), Apr 2008.

[8] W.-C. Kuo and C.-C. Wang, "On the capacity of 2-user 1-hop relay erasure networks—the union of feedback, scheduling, opportunistic routing, and network coding," in *Proc. IEEE Int. Symp. Inform. Theory*, (Saint Petersburg, Russia), Aug 2011.

[9] B. Rankov and A. Wittneben, "Achievable rate regions for the two-way relay channel," in *Proc. IEEE Int. Symp. Inform. Theory*, (Seattle, WA), 2006.

[10] K. Salamatian and R. Khalili, "An information theory for erasure channels," in *Proc. 43rd Ann. Allerton Conf. on Comm.*, (Monticello, IL), Sept 2005.

[11] C. Suh and K. Ramachandran, "Exact-repair MDS code construction using interference alignment," *IEEE Trans. Inf. Theory*, vol. 57, pp. 1425–1442, Mar 2011.

[12] J. Yoo, T. Liu, and F. Xue, "Gaussian broadcast channels with receiver message side information," in *Proc. IEEE Int. Symp. Inform. Theory*, (Seoul, Korea), June 2009.

[13] S. Rayanchu, S. Sen, J. Wu, S. Banerjee, and S. Sengupta, "Loss-aware network coding for unicast wireless sessions: design, implementation, and performance evaluation," in *Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, (Annapolis, MD, USA), pp. 85–96, 2008.

[14] S. Sengupta, S. Rayanchu, and S. Banerjee, "An analysis of wireless network coding for unicast sessions: The case for coding-aware routing," in *Proc. 26th IEEE Conf. Computer Communications (INFOCOM)*, (Anchorage, Alaska, USA), pp. 1028–1036, May 2007.

[15] H. Seferoglu, A. Markopoulou, and K. K. Ramakrishnan, "I^2NC: Intra- and inter-session network coding for unicast flows in wireless networks," in *Proc. 30th IEEE Conf. Computer Communications (INFOCOM)*, (Shanghai, China), pp. 1035–1043, April 2011.

[16] L. Ozarow and S. Leung-Yan-Cheong, "An achievable region and outer bound for the Gaussian broadcast channel with feedback," *IEEE Trans. Inf. Theory*, vol. 30, July 1984.

[17] A. Eryilmaz, D. Lun, and B. Swapna, "Control of multi-hop communication networks for inter-session network coding," *IEEE Trans. Inf. Theory*, vol. 57, pp. 1092–1110, Feb. 2011.

[18] C.-C. Wang, "On the capacity of wireless 1-hop intersession network coding — a broadcast packet erasure channel approach," *IEEE Trans. on Information Theory*, vol. 58, pp. 957–988, Feb 2012.

[19] J. Korner and K. Marton, "General broadcast channels with degraded message sets," *IEEE Trans. Inf. Theory*, vol. 23, pp. 60–64, Jan 1997.

[20] G. Kramer and S. Shamai, "Capacity for classes of broadcast channels with receiver side information," in *Proc. IEEE Inform. Theory Workshop*, (Lake Tahoe, CA), pp. 313–318, Sept 2007.

[21] B. Smith and B. Hassibi, "Wireless erasure networks with feedback," in *Proc. IEEE Int. Symp. Inform. Theory*, pp. 339–343, July 2008.

[22] C.-C. Wang and N. Shroff, "Beyond the butterfly—a graph-theoretic characterization of the feasibility of network coding with two simple unicast sessions," in *Proc. IEEE Int. Symp. Inform. Theory*, (Nice, France), pp. 121–125, June 2007.

[23] F. Xue and X. Yang, "Network coding and packet-erasure broadcast channel," in *Proc. 5th IEEE Ann. Communications Society Conf. on Sensor, Mesh and Ad Hoc Communications and Networks Workshops (SECON)*, (San Francisco, CA), July 2008.

[24] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *Proc. ACM Special Interest Group on Data Commun. (SIGCOMM)*, (Kyoto, Japan), Aug 2007.

[25] D. Koutsonikolas, C.-C. Wang, and Y. Hu, "CCACK: Efficient network coding based opportunistic routing through cumulative coded acknowledgments," in *Proc. 29th IEEE Conf. Computer Communications (INFOCOM)*, (San Diego, CA), pp. 1–9, Mar 2010.

[26] X. Lin, N. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 24, pp. 1452–1463, Aug 2006.

[27] L. Georgiadis, M. J. Neely, and L. Tassiulas, *Resource allocation and cross-layer control in wireless networks*. Now Pub, 2006.

[28] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *Automatic Control, IEEE Transactions on*, vol. 37, no. 12, pp. 1936–1948, 1992.

[29] Y. E. Sagduyu and A. Ephremides, "On broadcast stability region in random access through network coding," in *44th Allerton Annual Conference on Communication, Control and Computing*, p. 38, 2006.

[30] V. Anantharam and S. Verdtu, "Bits through queues," *IEEE Trans. Inf. Theory*, vol. 42, pp. 4–18, 1996.

[31] T. Lutz, G. Kramer, and C. Hausl, "Capacity for half-duplex line networks with two sources," in *Proc. IEEE Int'l Symp. Inform. Theory*, (Austin, Texas, USA), June 2010.

[32] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[33] A. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros, "Capacity of wireless erasure networks," *IEEE Trans. Inf. Theory*, vol. 52, pp. 789–804, Mar. 2006.

[34] T. Ho and H. Viswanathan, "Dynamic algorithms for multicast with intra-session network coding," *Information Theory, IEEE Transactions on*, vol. 55, no. 2, pp. 797–815, 2009.

[35] L. Amini, N. Jain, A. Sehgal, J. Silber, and O. Verscheure, "Adaptive control of extreme-scale stream processing systems," in *Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on*, pp. 71–71, IEEE, 2006.

[36] M. Zaharia, A. Konwinski, A. D. Joseph, R. Katz, and I. Stoica, "Improving mapreduce performance in heterogeneous environments," in *Proceedings of the 8th USENIX conference on Operating systems design and implementation*, pp. 29–42, 2008.

[37] J. Cao, S. A. Jarvis, S. Saini, and G. R. Nudd, "Gridflow: Workflow management for grid computing," in *Cluster Computing and the Grid, 2003. Proceedings. CCGrid 2003. 3rd IEEE/ACM International Symposium on*, pp. 198–205, IEEE, 2003.

[38] L. Jiang and J. Walrand, "Stable and utility-maximizing scheduling for stochastic processing networks," in *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*, pp. 1111–1119, IEEE, 2009.

[39] L. Huang and M. J. Neely, "Utility optimal scheduling in processing networks," *Performance Evaluation*, vol. 68, no. 11, pp. 1002–1021, 2011.

[40] C.-C. Wang and D. J. Love, "Linear network coding capacity region of 2-receiver mimo broadcast packet erasure channels with feedback," in *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*, pp. 2062–2066, IEEE, 2012.

[41] A. El Gamal, "The feedback capacity of degraded broadcast channels," *IEEE Trans. Inf. Theory*, vol. 25, pp. 379–381, March 1978.