Coding Theorems for Delay Sensitive Communication over Burst-Erasure Channels

by

Devin Waine-Tak Lui

A thesis submitted in conformity with the requirements for the degree of Master of Applied Science
Graduate Department of Electrical and Computer Engineering
University of Toronto

Copyright © 2011 by Devin Waine-Tak Lui
Abstract

Coding Theorems for Delay Sensitive Communication over Burst-Erasure Channels

Devin Waine-Tak Lui
Master of Applied Science
Graduate Department of Electrical and Computer Engineering
University of Toronto
2011

In this thesis, we consider error-correction codes for systems which have burst erasure channels, but where the packet delay is constrained. The packet delay itself is the time difference between the arrival of a source packet at the encoder and the reconstruction of that source packet at the decoder. While such a framework was introduced by Martinian (2004) and his co-authors, several problems remain open.

We make three contributions in this thesis. First we develop a rigorous converse proof for the point-to-point case and thus complete the result of Martinian (2004). Our proof technique is also applied to a multicast channel model and new results are obtained. Secondly we study the case when there are multiple parallel links between the encoder and decoder and obtain the capacity in some special cases. Finally we study a setup when there are multiple source streams, each with a different delay constraint, and obtain capacity results.
Acknowledgements

During my two years as a graduate student here, I learnt much from the people who surrounded me at the University of Toronto. I have many of my peers and teachers to thank for teaching me and helping me along the way. Without them, this thesis would not be possible. First and foremost, I would like to thank my supervisor Professor Ashish Khisti for taking me as his first Master student, and for giving me guidance over these two years. His understanding and opportunities that he gave me have been invaluable for my own growth. I also thank Ahmed Badr for the support that he offered, and for the chance to cooperatively work with him.

My sincerest thanks goes to those who have taught me during my stay here at the University of Toronto. Professors Teng Joon Lim, Frank Kschischang, Wei Yu, Ravi Adve, Ben Liang and Peter Marbach have all taught me at one time or another, and the knowledge I gained from their classes has been essential. I also thank Lei Zhou for being an excellent TA and friend. I would like to thank Frank Kshischang and Teng Joon Lim again for their personal support. My thanks also goes to the professors on my committee, including Professor Sean Victor Hum for taking time to help me complete my defence. Finally, I thank Bruno Korst Fagundes for his support and help when I was a TA but also when I was an undergraduate student.

I also give my thanks to my fellow graduate students who saw me through to the end, especially William Chou, Jefferson Lee, Lei Zhang, Patrick Li for their intelligent discussions. For sticking with me through graduate school, I thank Chunpo Pan, Benjamin Smith, Dohyoung Lee, Gaurav Jain, Hui Xiong, Tony Liang and Neeraj Sood.

My thanks go to my friends Jacky Mak, Daniel Hyunh, Hu Hong, Wallace Wee and Gerry Chen for their continued support.

I would also like to thank the staff Diane Silva, Judith Levene and Vladimiro Cirillo for their dedicated work.

I would especially like to thank my family and my parents especially for being supportive of me while I pursued my degree.

The work in this thesis has been graciously funded by the Edward S. Rogers Sr Graduate Scholarship and the Natural Sciences and Engineering Research Council of Canada (NSERC).

Finally, I thank anybody whom I may have left out.
## Contents

1 Introduction .............................................. 1  
   1.1 MDS Block Codes .................................... 2  
   1.2 Low Delay Block Codes .............................. 3  
       1.2.1 Low Delay Block Code Construction .......... 4  
       1.2.2 Low Delay Block Code Decoding .............. 5  
   1.3 Types of Delays .................................... 7  

2 Streaming Burst Erasure Codes ......................... 9  
   2.1 Streaming Codes .................................... 9  
   2.2 System Model for Streaming Codes ................. 10  
   2.3 Streaming Code Construction ...................... 12  
   2.4 Converse Proof using Periodic Erasure Channel Sequences .... 14  
   2.5 Converse Proof using Information Theory .......... 16  
   2.6 Summary .......................................... 20  

3 Multicast System ........................................ 22  
   3.1 Multicast Double Receiver System ................. 22  
   3.2 New Upper Bound — (e) Region .................... 27  
   3.3 Improved Upper Bound for the (f)-region .......... 33  
   3.4 Summary .......................................... 38  

4 Multi-Link Streaming Erasure Codes .................... 40  
   4.1 System Model for two links ....................... 40  
   4.2 Double-Link Streaming Capacity ................... 42  
   4.3 Achievable Codes for Bidirectional Separation Case .... 42  
       4.3.1 Code for \((T \geq B), (d < \frac{T+B}{2})\) ....... 42  

iv
4.3.2 Code for \((T \geq B), (T + B \geq d \geq \frac{T+B}{2})\) ........................................... 44
4.3.3 Code for \((T \geq B), (d > T + B)\) ................................................................. 46
4.3.4 Code for \((T < B), (B > d \geq B - T)\) ......................................................... 47
4.3.5 Code for \((T < B), (d \geq B)\) .............................................................. 47

4.4 Converse Proof ................................................................. 47
4.4.1 \((T \geq B), (d \leq T + B)\) ................................................................. 47
4.4.2 \((T \geq B), (T + B \geq d \geq \frac{T+B}{2})\) ........................................... 49
4.4.3 \((T \geq B), (d > T + B)\) ................................................................. 50
4.4.4 \((T < B), (d < B - T)\) .............................................................. 50
4.4.5 \((T < B), (B > d \geq B - T)\) ......................................................... 51
4.4.6 \((T < B), (d \geq B)\) .............................................................. 53

4.5 System Model for Multiple Links ................................................... 54

4.6 Multi-Link Streaming Capacity ......................................................... 56

4.7 Achievable Codes ................................................................. 57
4.7.1 Code for \((T \geq B), (d \leq \frac{T+B}{2})\) ......................................................... 57
4.7.2 Code for \((T \geq B), (T + B \geq d \geq \frac{T+B}{2})\) ........................................... 58
4.7.3 Code for \((T \geq B), (d > T + B)\) ......................................................... 59
4.7.4 Code for \((T < B), (d < B - T)\) .............................................................. 60
4.7.5 Code for \((T < B), (B > d \geq B - T)\) ......................................................... 60
4.7.6 Code for \((T < B), (d \geq B)\) .............................................................. 62

4.8 Converse Proof ................................................................. 62
4.8.1 \((T \geq B), (d \leq \frac{T+B}{2})\) ................................................................. 62
4.8.2 \((T \geq B), (T + B \geq d \geq \frac{T+B}{2})\) ........................................... 63
4.8.3 \((T \geq B), (d > T + B)\) ................................................................. 64
4.8.4 \((T < B), (d < B - T)\) .............................................................. 64
4.8.5 \((T < B), (B > d \geq B - T)\) ......................................................... 64
4.8.6 \((T < B), (d \geq B)\) .............................................................. 65

4.9 Summary ................................................................. 65

5 Multi-Source Streaming Codes ......................................................... 66
5.1 System Model ................................................................. 66
5.2 \((B, T) (B, B)\) code, \(T \geq 3B\) ................................................................. 67
5.2.1 Converse Proof ................................................................. 68
5.2.2 Achievable Codes ................................................................. 72
Definitions and Notations

Definitions

SCo — Streaming Erasure Code. A single-link, single-source streaming code with the parameters \((B, T)\). (Section 2.3)

DE-SCo — Diversity Embedded Streaming Erasure Code. A single-link, single-source streaming code that supports two users with parameters \((B_1, T_1)\) and \((B_2, T_2)\). (Section 3.1)

MDS Code — Maximum Distance Separable code.

Special Notation

\[
\begin{align*}
s_{[b]}^a &= \begin{cases} 
  s[a], s[a+1], \ldots, s[b-1], s[b], & a \leq b \\
  \emptyset, & \text{otherwise}
\end{cases} \\
W_{[b]}^a &= \begin{cases} 
  W_a, W_{a+1}, \ldots, W_{b-1}, W_b, & a \leq b \\
  \emptyset, & \text{otherwise}
\end{cases}
\end{align*}
\]
Chapter 1

Introduction

This thesis studies the effect of delay sensitive media on the design of communication systems. Imagine that we have a media source streaming content to a viewer over a communication channel. The viewer does not have to wait until the entire file is received, instead the viewer can start playing the media as soon as an initial amount of data is received. Clearly in such applications we are not interested in the time it takes to receive the entire media file, but instead, the delay of each video or audio packet from the source is of relevance. Each packet needs to be received within a certain deadline at the receiver to enable a smooth playback of the media and avoid effects such as buffering or stuttering.

When stringent end-to-end delay constraints are imposed, the underlying channel dynamics need to be carefully considered. We illustrate this for an erasure channel. In such systems, transmitted packets either arrive at the receiver without error, or they do not arrive at all. This type of channel can model the arrival of packets at the higher layers of the protocol stack such as internet protocol (IP) layer or application layer. Fig. 1.1(a) and Fig. 1.1(b) illustrate two channel models that have identical capacities but different delay dynamics. Both channels have 10 erasures in a block-length of 30. In Fig. 1.1(a), the erasures are spread out whereas in Fig. 1.1(b) the erasures are concentrated in a single burst. Fig. 1.1(a) models the scenario of an erasure channel with independent, identically distributed (i.i.d.) erasures while Fig. 1.1(b) models a channel where packet erasures happen in bursts separated by a long guard period.

Clearly both channels are able to communicate 20 information packets in a block length of 30 channel packets. Thus both systems have identical capacities when delay is not taken into account. However special low-delay error control codes exist for the burst-erasure channel with a much lower decoding delay. Such an observation was first made by
Chapter 1. Introduction

Link: 

(a)

Link:

(b)

Figure 1.1: Erasure Channel Sequences for (a) a memoryless erasure channel (b) a burst erasure channel.

Martinian in [12, Chapters 6-11] and more recently in [1], [2], and [3]. In section 1.2, we describe in detail the low-delay block code constructions proposed in [12], as the thesis will build upon these constructions.

Bursty channels have been studied before in classical literature. Gallager considers a bursty binary symmetric channel in [6], where he has bursts of $b$ channel uses where bit flips are possible, but in between each consecutive bursts is a guard period of at least $g$ channel uses where the bits are not flipped. For this scenario, he allows the decoding delay for each source digit to be large enough so that they do not affect the rate. Gallager finds an upper bound on capacity, and shows that capacity can be approached with cyclic or convolutional codes.

Sahai in [16] also considers the effects of an end-to-end delay constraint on a memoryless channel. He shows that the system can be characterized by error exponents, where an error is the event that a source packet is not decoded by its deadline. He then shows how the error exponent is affected by delay constraints, the presence of feedback and the type of channel code used.

1.1 MDS Block Codes

Maximum distance separable (MDS) block code, such as a Reed-Solomon code are most widely used error-correcting codes over erasure channels. For a systematic MDS block code, a codeword consists of $k$ source symbols followed by $n - k$ parity symbols. The codeword can be recovered when $n - k$ or fewer symbols are erased. In general, all the $k$ symbols need to be received at the decoder before the entire codeword can be recovered. This means that any missing source symbol, cannot be recovered until $k$ symbols are recovered. The maximum possible delay occurs when the first $n - k$ symbols
of a codeword are erased, then they are only recovered at the end of the codeword. The maximum delay is then equal to the delay of the first source symbol, which is \( n - 1 \). For example, Figure 1.2, shows an \((n, k) = (9, 5)\) codeword where the first \( n - k = 4 \) symbols are erased.

![Figure 1.2: Example of an \((n, k) = (9, 5)\) codeword with 4 erasures.](image)

**1.2 Low Delay Block Codes**

Instead of MDS block codes, let us consider a new class of codes called low delay block codes. They were introduced by Martinian [12] and have the property that each source symbol can be recovered with a maximum delay of \( T \) time units, no matter the position of the burst erasure. Instead of being defined as an \((n, k)\) block code, they are instead described as \((B, T)\) low delay block codes where \( B \) is the burst erasure length that the code can withstand. Figure 1.3 shows a codeword for a low delay block code and indicates where each erased source symbol can be recovered. With the same rate, this type of code has better delay properties than MDS codes over a burst erasure channel.

![Figure 1.3: Example of a low delay block code with \((B, T) = (4, 5)\)](image)

Martinian in [12] discovered a \((B, T)\) low delay block code with the restriction \( T \geq B \) that has \( n = B + T \) and \( k = T \), giving a rate of

\[
R = \frac{T}{T + B}.
\]
For an example of this code, consider the simple case where $B = 2$ and $T = 3$. To form the block code, we will need $T = 3$ source symbols and $B = 2$ parity symbols. Assume that the source symbols are $\{s_0, s_1, s_2\}$.

Then the block code can be given as

$$\left(s_0, s_1, s_2, s_0 + s_2, s_1 + s_2\right).$$

It is easy to verify that each source symbol $s_0, s_1, s_2$ can be recovered from a burst of 2 erasures, with a maximum delay of 3.

Likewise the $(4, 5)$ low-delay code in Fig. 1.3 is constructed as

$$\left(s_0, s_1, s_2, s_3, s_0 + s_4, s_1 + s_4, s_2 + s_4, s_3 + s_4\right).$$

### 1.2.1 Low Delay Block Code Construction

We describe the general structure of a low delay block code [2]. For a $(B, T)$ code where $T \geq B$, there will be $T$ source symbols and $B$ parity symbols. Let the source symbols be $\{s_0, s_1, \ldots, s_{T-1}\}$. We assign the first $B$ source symbols to the row vector $u$ and the next $T - B$ source symbols to the vector $n$.

$$u = (s_0, s_1, \ldots, s_{B-1})$$

$$n = (s_B, s_{B+1}, \ldots, s_{T-1}) \quad (1.1)$$

The symbols in $u$ are called the urgent symbols and $n$ are called the non-urgent symbols. The reason is that erased urgent symbols are always recovered last and their delay is exactly $T$, while erased non-urgent symbols are recovered before and have delay less than $T$. This is verified in the next section for decoding.

Let $H$ be a full rank $(T - B) \times B$ matrix. The matrix $H$ must also have the property that $(I_{T-B} \ H)$ is a generator matrix for a linear code that can recover from an erasure burst of length $B$, without a delay constraint where $I_{T-B}$ is a $(T - B) \times (T - B)$ identity matrix. Then the block code can be given as

$$\left(u, \ n, \ u + nH\right). \quad (1.2)$$

The generator matrix is then

$$G = \begin{pmatrix} I_B & 0 & I_B \\ 0 & I_{T-B} & H \end{pmatrix}. \quad (1.3)$$

---

1Throughout this thesis the addition is over the underlying finite field of appropriate size.
We describe next how this block-code can recover every symbol from an erasure burst of length $B$ with a delay no larger than $T$.

### 1.2.2 Low Delay Block Code Decoding

A burst of length $B$ erases $B$ consecutive symbols. The decoding strategy is to first recover any missing non-urgent symbols from $n$, then to recover the missing urgent symbols from $u$. Let us consider some cases with different burst erasure locations.

**Case 1**

![Figure 1.4: A burst erasure that covers the urgent and non-urgent source symbols](image)

Figure 1.4 shows the case where both urgent and non-urgent symbols are erased. The last $\Delta$ urgent symbols and the first $B - \Delta$ non-urgent symbols are erased, where $0 \leq \Delta \leq B$. Notice that the first $B - \Delta$ urgent symbols are not erased, so we can subtract them from the first $B - \Delta$ parity symbols. Using these $B - \Delta$ parity symbols and the non-erased non-urgent symbols, erased non-urgent symbols can be recovered. The first non-urgent symbol has the worst-case delay for non-urgent symbols and it is sent at time $B$, but decoded after the $(B - \Delta)$th parity check is received. Its delay will be equal to $[(T - B) + (B - \Delta) - 1]$, or $(T - \Delta - 1)$, which is less than the delay constraint of $T$.

Next, the last $\Delta$ parity symbols are used to recover the missing urgent symbols. As each parity symbol is received, the non-urgent ($nH$) component can be subtracted, and the urgent symbol can be recovered. The urgent symbol that is recovered is the one that was sent $T$ time units ago, so the delay for each urgent symbol is $T$.

**Case 2**

In this next case, Figure 1.5 has the burst erasure covering the non-urgent source symbols and parity symbols. The last $\Delta$ non-urgent source symbols and first $B - \Delta$ parity symbols are erased. Decoding is simple in this case: subtract the urgent symbols $u$ from the parity...
Figure 1.5: A burst erasure that covers the non-urgent source symbols and parity symbols, and then use the received non-urgent symbols in $n$ and the received parity symbols to decode the missing non-urgent symbols. There are enough parity symbols to perform decoding.

The worst-case delay is for the first erased symbol, which is recovered when all the parity symbols are received. The delay is then equal to $B + \Delta - 1$. According to the diagram, $\Delta \leq \min(B, T - B) \leq T - B$, so the worst case delay is less than or equal to $T - 1$.

If we consider the case where the burst erasure is contained within the $n$ block and only erases non-urgent symbols, we can recover the missing symbols in the same way.

Case 3

Figure 1.6: A burst erasure that erases all three types of symbols

When $T - B < B - 1$, it is possible to have the case in Figure 1.6. The erasure burst erases some urgent symbols, all of the non-urgent source symbols and some parity symbols. Once again, the first $T - B$ parity symbols received can be used to recover the non-urgent symbols $n$. Then, the final $\Delta$ parity symbols received can recover the missing urgent symbols. The worst case delay is for the first erased urgent symbol, which is always equal to $T$.

Case 4

This final case deals with the possibility of a wrap-around burst erasure of length $B$. A $(B, T)$ code is not required to recover from a burst of this nature, but Martinian’s low
delay block codes have the ability regardless. Since all the non-urgent source symbols $n$ are received, they can simply be cancelled out from the parity symbols. Then the missing urgent symbols can be read off from the received parity symbols. The delay for the urgent symbols is equal to $T$.

### 1.3 Types of Delays

The system block diagram in Figure 1.8 can be used to illustrate two different types of delays: video processing delays and network delays. The video processing delay is the delay incurred by the decoding of forward error correction (FEC) codes and the video itself. In other words, it is the delay of the higher layers: session, presentation and application layers. Network delay is the delay incurred by the physical properties of the channel and by packet routing. It is the total delay of the physical, data link, network and transport layers.

The type of FEC code and video encoding used affects the video processing delay, but not the network delay. However, in this thesis, we are specifically focused on the decoding delay of the FEC encoder and decoder.

Now that we have established that the FEC code decoding delay is the type of delay we are interested in, we must still define a metric to measure this type of delay. The right metric must be tractable, but also meaningful in the sense that it affects a user’s streaming experience.

One method is to measure the block-delay, which is the time until a single message or part of a file is decoded at the FEC decoder, starting from the first transmission. However, in the case of streaming media, the user is not interested in the time to transfer a large block of data. The media must be able to play continuously, so we must look for another notion of delay.

Instead, we choose to use packet delay as our notion of delay. The packet delay is
the time from the arrival of a source packet at the FEC encoder, to the decoding of that source packet at the FEC decoder on the receiver side. It is independent of the type of source, which simplifies the analysis.

Figure 1.8: High level system diagram
Chapter 2

Streaming Burst Erasure Codes

This thesis builds upon the work done by Emin Martinian in his Ph.D. thesis [12]. In Part II of his thesis, he introduces the concept of packet decoding delay, and uses it to investigate a new class of codes with a delay constraint.

2.1 Streaming Codes

In Chapter 1, we discussed low-delay block codes. A block code maps a block of \( k \) source symbols, and creates a codeword of a fixed length, say \( n \), where \( n \geq k \). The input symbols and output symbols are assumed to have the same alphabet. The rate of the code is defined as \( k/n \), and the encoder takes in blocks of \( k \) source packets and produces blocks of \( n \) channel packets. As we have more channel packets than source packets, the channel packets have to be sent at a faster packet rate than the input stream.

For streaming media systems it is more natural to study streaming codes that take a continuous stream of source packets as input and produce another continuous stream of packets as output. The sizes of the source packets and output channel packets are allowed to be different, but the packet rate is the same for the input and output streams of the encoder. In general we will assume that each source packet consists of \( k \) sub-symbols and each channel packet consists of \( n \) sub-symbols over some finite field. The rate of such a streaming code is \( k/n \).

Figure 2.1 shows the difference between an encoder for systematic block codes and streaming codes. In each case, the encoder is adding parity information, so the bit rate of the channel stream is higher than the input stream. However, in Figure 2.1(a) for block codes, the 5 channel packets must be sent in the same time as the 3 source packets, but
in Figure 2.1(b) for streaming codes, the 3 source packets are sent at the same packet rate as the 3 channel packets.

![Diagram of encoder for block codes and streaming codes](image)

(a) Encoder for block codes

(b) Encoder for streaming codes

Figure 2.1: An encoder for (a) block codes (b) streaming codes. The $s[t]$ are source symbols/packets, $p[t]$ are parity symbols and $x[t]$ are channel packets.

We will use streaming codes because their analysis is more tractable and manageable than with block codes. For example, given an integer time index $t$ and a delay constraint $T$, the streaming code decoder must be able to recover the source packet $s[0]$ to $s[t - T]$. However, with block codes, defining which source packets are decoded is unnecessarily complex.

### 2.2 System Model for Streaming Codes

We shall consider a variation of the system model described in [12, Chapter 7]. Imagine that media is being streamed from the sender to the receiver. The sender obtains the media from a source stream, which contains packetized data. The source packets arrive regularly at the sender, and must be sent to the receiver. The receiver itself must be able to obtain each source packet within a certain deadline.

The system can be formally characterized in the following way. Time is discretized into time slots. At the beginning of time slot $t \geq 0$, where $t$ is an integer, a source packet $s[t]$ arrives at the transmitter. Source packets are modelled as being independent of each other and they can be treated as i.i.d. symbols drawn from the same distribution. $H(s)$ is defined to be the entropy of a single source packet, such that

$$H(s[t]) = H(s), \forall t \geq 0.$$
The sender encodes all of the source packets that it has received into a channel packet \( x[t] \) using the encoding function
\[
x[t] = f_t(s[0], \ldots, s[t]), \; t \geq 0.
\] (2.1)

Note that the entropy of the channel packets is the same for all time. We denote this value with the symbol \( H(x) \), such that
\[
H(x[t]) = H(x), \; \forall t \geq 0.
\]

We also define the size of a channel packet in bits with the symbol \( |x| \). We assume that the channel packets are sufficiently large such that
\[
|x| \geq H(x). \quad (2.2)
\]

The channel packet \( x[t] \) is sent over the channel, and the received packet is labelled as \( y[t] \). The packet \( y[t] \) arrives near the end of time slot \( t \), so it arrives in the same time slot that \( x[t] \) was sent.

The channel is a burst erasure channel, which erases at most \( B \) consecutive packets. Only a single burst of erasures occur. The output \( y[t] \) is defined as
\[
y[t] = \begin{cases} \star & t \in [j, j + B - 1] \\ x[t] & \text{otherwise} \end{cases}
\] (2.3)

for some \( j \in \mathbb{Z}^+ \).

Finally, the receiver attempts to decode the original source packets from the received channel packets. The variable \( T \) is the allowable delay, which means that the receiver must decode source packet \( s[t] \) by time \( t + T \). The decoder implements the decoding function \( g_t(\cdot) \), such that for \( t \geq 0 \)
\[
\begin{cases} \hat{s}[t] = g_t(y[0], \ldots, y[t + T]) \\ Pr(\hat{s}[t] \neq s[t]) = 0. \end{cases}
\] (2.4)

The rate of a streaming code is defined to be
\[
R \triangleq \frac{H(s)}{|x|} \leq \frac{H(s)}{H(x)}.
\] (2.5)

Typically, the size of channel packets, \( |x| \), is fixed. For example, Internet protocol (IP) defines a maximum transmission unit (MTU), which is the maximum size of IP packets. Then we can adjust the rate by changing the entropy of the source packets \( H(s) \).
A channel code that meets the requirements of (2.1) and (2.4) when using the channel described by (2.3) is called a $\langle B, T \rangle$ burst erasure code.

The following table lists the main differences between classical information theory framework, and the framework just described.

<table>
<thead>
<tr>
<th>Classical</th>
<th>Streaming</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rate measured in bits per channel use</td>
<td>Rate measured in bits per bits</td>
</tr>
<tr>
<td>Encode a message $W \in {1, \ldots, 2^{nR}}$ into a block of $n$ channel symbols $x$</td>
<td>Encode source packets $s[t]$ into channel packets $x[t]$ for $t \geq 0$</td>
</tr>
<tr>
<td>Recover message $W$ from $n$ received symbols $y$</td>
<td>Recover each source packet $s[t]$ from received packets $y[0], \ldots, y[t+T]$</td>
</tr>
<tr>
<td>Capacity is maximum rate given that $W$ is decoded correctly with probability 1</td>
<td>Capacity is maximum rate given that each $s[t]$ is decoded correctly by time $t + T$ with probability 1</td>
</tr>
<tr>
<td>Capacity found for block length $n \to \infty$</td>
<td>Converse found for time $t \to \infty$</td>
</tr>
</tbody>
</table>

The burst-delay capacity $C$, for the system can be defined as the maximum achievable rate of a $\langle B, T \rangle$ code. The capacity is given in the following theorem from [12].

**Theorem 1.** The capacity of a single-link unicast system which satisfies (2.1), (2.3) and (2.4) is given by

$$C = \begin{cases} \frac{T}{T+B}, & T \geq B \\ 0, & T < B. \end{cases}$$

(2.6)

To prove capacity, we will need two parts to the proof. We need to show that the capacity is achievable, and we also need to show that no $\langle B, T \rangle$ code can have a rate above capacity. We first establish the forward part i.e., achievability.

### 2.3 Streaming Code Construction

Section 1.2 gives a code construction for a $\langle B, T \rangle$ low delay block code for $T \geq B$ which has the desired rate. It is possible to convert a block code to a streaming code using a technique called *diagonal interleaving* [12]. Recall that a streaming code has input source packets and output channel packets, where each packet is made up of multiple sub-symbols. As the capacity is $\frac{T}{T+B}$, each source packet shall contain $T$ sub-symbols and each channel packet can contain $T + B$ sub-symbols, where each sub-symbol has the same finite field.
Each source packet \( s[t] \) can be broken up into \( T \) sub-symbols

\[
s[t] = (s_0[t], s_1[t], \ldots, s_{T-1}[t])
\]  

(2.7)

where each \( s_i[t] \) is a source sub-symbol. Since the code is systematic, each channel packet has the form

\[
x[t] = (s_0[t], s_1[t], \ldots, s_{T-1}[t], p_0[t], \ldots, p_{B-1}[t]).
\]  

(2.8)

If each of the channel packets is placed side by side, then diagonal interleaving means that each of the main diagonals is a codeword of a low delay block code. Figure 2.2 shows one of the main diagonals for \( B = 2, T = 3 \). Each main diagonal can be written as the vector

\[
(s_0[t], s_i[t + i], \ldots, s_{T-1}[t + T - 1], p_0[t + T], \ldots, p_j[t + T + j], \ldots, p_{B-1}[t + T + B - 1]).
\]  

(2.9)

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( s_0[0] )</td>
<td>( s_0[1] )</td>
<td>( s_0[2] )</td>
<td>( s_0[3] )</td>
<td>( s_0[4] )</td>
<td>( s_0[5] )</td>
</tr>
<tr>
<td>( s_1[0] )</td>
<td>( s_1[1] )</td>
<td>( s_1[2] )</td>
<td>( s_1[3] )</td>
<td>( s_1[4] )</td>
<td>( s_1[5] )</td>
</tr>
</tbody>
</table>

Figure 2.2: The main diagonal of a stream of channel packets

Once diagonal interleaving is applied, then each source sub-symbol in the channel packet is protected with a \((B, T)\) low delay block code. Thus, no matter where the \( B \) erasures occur, each source sub-symbol can be recovered with a maximum delay of \( T \).

We shall use SCo to refer to the resulting streaming code.

For an example, consider the \((2, 3)\) low delay block code given earlier in Section 1.2.

\[
\left( s_0, s_1, s_2, s_0 + s_2, s_1 + s_2 \right)
\]

To form the SCo code, the block code must be diagonally interleaved into the channel packets, in order to form the streaming code. For ease of visualization, let each source packet \( s[t] \) be defined as

\[
s[t] = (a_t, b_t, c_t).
\]
Then each parity sub-symbol is constructed as

\[ p_0[t] = a_{t-3} + c_{t-1} \]
\[ p_1[t] = b_{t-3} + c_{t-2}. \]

The streaming code will look like Figure 2.3. Each of the main diagonals is a codeword in the SCo block code.

\[
\begin{array}{cccccccc}
  a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 \\
  b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 \\
  c_0 & c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \\
  a_{-3} + c_{-1} & a_{-2} + c_0 & a_{-1} + c_1 & a_0 + c_2 & a_1 + c_3 & a_2 + c_4 & a_3 + c_5 \\
  b_{-3} + c_{-2} & b_{-2} + c_{-1} & b_{-1} + c_0 & b_0 + c_1 & b_1 + c_2 & b_2 + c_3 & b_3 + c_4 \\
\end{array}
\]

Figure 2.3: The main diagonal of a stream of channel packets

2.4 Converse Proof using Periodic Erasure Channel Sequences

Martinian in [12, Chapter 7] provides the following converse proof.

**Proof.** For \( T \geq B \), consider the erasure channel sequence in Figure 2.4, where the grey squares represent the time slots where packet erasures have occurred, and the white squares represent the time slots when the channel packets are not erased. The channel has a repeating pattern of \( B \) erased channel packets followed by \( T \) non-erased channel packets, which repeats indefinitely.

A \((B, T)\) code can be used to recover all of the source packets, using the following argument. The first \( B \) channel packets are erased, but the next \( T \) packets are received. At this point, the receiver can use the \((B, T)\) code to recover the first \( B \) source packets \( s[0], \ldots, s[B - 1] \). Theoretically, the receiver could use these recovered source packets to reconstruct the missing channel packets \( x[0], \ldots, x[B - 1] \). At this point, we can treat the first burst erasure as having never occurred. Furthermore assuming that the code is systematic, the \( T \) received channel packets \( x[B], \ldots, x[B + T - 1] \) can be used to recover
Chapter 2. Streaming Burst Erasure Codes

The second part of the proof is to find the capacity of the periodic erasure channel sequence from Figure 2.4. When the channel packet is not erased, the receiver receives $H(x)$ bits, but when the channel packet is erased, the receiver gets 0 bits of information. The bit rate (per time unit) of the channel is then equal to:

$$\frac{T}{T+B} \cdot H(x). \quad (2.10)$$

The source stream must have an equivalent or smaller bit rate, which means

$$\frac{H(s)}{R} \leq \frac{T}{T+B} \cdot H(x) \quad (2.11)$$

When $T < B$, we use an erasure channel sequence where each channel packet is erased. At time $T$, we have only seen a burst of $T + 1 \leq B$, so the $(B,T)$ decoder can be used to recover $s[0]$. Then, $x[0]$ can be reconstructed from $s[0]$. At the next time $T + 1$, the only erased channel packets are $x[1]$ to $x[T + 1]$. We are still seeing a burst of $T + 1 \leq B$ and can also recover $s[1]$. Extending this argument, we can recover all of the source packets in the channel sequence where all channel packets are erased. The bit rate of this channel sequence is 0, and thus 0 is the upper bound on the rate of the code.

This converse proof gives an additional interesting result. For $T \geq B$, we only require the $(B,T)$ code to be able to decode from one burst erasure of length $B$, but it so happens that the code also works perfectly in the periodic erasure channel sequence of Figure 2.4, which has multiple erasure bursts of length $B$ which are separated by $T$ time slots.

Although the streaming code is assumed to be systematic, Martinian’s proof can easily be extended to work for non-systematic codes also. When a channel packet is received, a systematic code enables the decoder to recover the corresponding source packets. At this point we have recovered all the source packets in the first period and can proceed with a similar argument for subsequent periods.

![Figure 2.4: The periodic erasure channel sequence used to prove an upper bound on capacity]

The capacity of the periodic erasure channel sequence from Figure 2.4 is:

$$\frac{T}{T+B} \cdot H(x).$$

The source stream must have an equivalent or smaller bit rate, which means

$$\frac{H(s)}{R} \leq \frac{T}{T+B} \cdot H(x)$$

When $T < B$, we use an erasure channel sequence where each channel packet is erased. At time $T$, we have only seen a burst of $T + 1 \leq B$, so the $(B,T)$ decoder can be used to recover $s[0]$. Then, $x[0]$ can be reconstructed from $s[0]$. At the next time $T + 1$, the only erased channel packets are $x[1]$ to $x[T + 1]$. We are still seeing a burst of $T + 1 \leq B$ and can also recover $s[1]$. Extending this argument, we can recover all of the source packets in the channel sequence where all channel packets are erased. The bit rate of this channel sequence is 0, and thus 0 is the upper bound on the rate of the code.

This converse proof gives an additional interesting result. For $T \geq B$, we only require the $(B,T)$ code to be able to decode from one burst erasure of length $B$, but it so happens that the code also works perfectly in the periodic erasure channel sequence of Figure 2.4, which has multiple erasure bursts of length $B$ which are separated by $T$ time slots.

Although the streaming code is assumed to be systematic, Martinian’s proof can easily be extended to work for non-systematic codes also. When a channel packet is received, a systematic code enables the decoder to recover the corresponding source packets. At this point we have recovered all the source packets in the first period and can proceed with a similar argument for subsequent periods.

![Figure 2.4: The periodic erasure channel sequence used to prove an upper bound on capacity]
packet immediately. But when considering non-systematic codes, the decoding can be held off until \( T \) time units later and the result will be the same.

One restriction these proofs have is that the streaming codes are required to be deterministic, meaning that each channel packet \( x[t] \) is a deterministic (non-random) function of the source packets \( s[0], \ldots, s[t] \). In the proof, we implicitly assume that any channel packet \( x[t] \) can be reconstructed if the source packets up to time \( t \) are known.

## 2.5 Converse Proof using Information Theory

While Martinian’s converse proof in Section 2.4 is valid for the system model described, we can introduce a more rigorous converse proof. Martinian’s proof can be modified by rewriting it to use the information theory framework. The idea of using periodic erasure channels sequence to prove the converse is still valid, but it can be written using a different method.

Let us use the following notation:

\[
\begin{align*}
s[a,b] &= \begin{cases} 
  s[a], s[a+1], \ldots, s[b-1], s[b], & a \leq b \\
  \emptyset, & \text{otherwise}
\end{cases} \\
W[a,b] &= \begin{cases} 
  W_a, W_{a+1}, \ldots, W_{b-1}, W_b, & a \leq b \\
  \emptyset, & \text{otherwise}
\end{cases}
\end{align*}
\]

(2.12) (2.13)

To aid us in our proof, let us introduce the terms

\[
\begin{align*}
V_i &= s \left[ \frac{(i+1)(T+B)-1}{i(T+B)} \right] \\
W_i &= x \left[ \frac{(i+1)(T+B)-1}{i(T+B)+B} \right]
\end{align*}
\]

(2.14)

where \( i \geq 0 \). Note that \( V_i \) refers to a group of source packets, whereas \( W_i \) is a group of channel packets in the \( i \)-th period. Figure 2.5 shows the time slots that the packets come from.

**Proof.** We start with the following equations, which are a result of the \((B,T)\) code. If the first \( B \) channel packets are erased, and then the next \( T \) channel packets are received perfectly, the \((B,T)\) code can be used to recover the source packets \( s[0], \ldots, s[B-1] \). Using the conditional entropy notation, this can be written \(^1\) as:

\[
H\left(s^{B-1} | W_B\right) = 0.
\]

(2.15)

\(^1\)If vanishing small error probability with packet size is desired, Fano’s Inequality can be invoked.
Although the next $T$ channel packets $x^{[B+T-1]}_{B}$ are received, we cannot assume that the corresponding source packets $s^{[B+T-1]}_{B}$ are able to be decoded because the code may not be systematic. To recover those source packets, we can use the next group of $T$ non-erased packets in $x^{[2(B+T)-1]}_{(B+T)+B}$. In general, we may not need all of these channel packets, but the proof is simpler if we have it all available. We can then write the relation

\[
H\left(s^{[B+T-1]}_{B}\right) \bigg| x^{[B-1]}_{0}W^{1}_{0}\bigg) = 0. \tag{2.16}
\]

The equations (2.15) and (2.16) can be generalized to

\[
H\left(s^{[i(B+T)+B-1]}_{i(B+T)}\right) \bigg| x^{[i(B+T)-1]}_{0}W_{i}\bigg) = 0 \tag{2.17}
\]

\[
H\left(s^{[(i+1)(B+T)-1]}_{(i+1)(B+T)}\right) x^{[i(B+T)+B-1]}_{0}W^{i+1}_{i}\bigg) = 0 \tag{2.18}
\]

Note that the above expressions still only assume that there was one burst erasure of length $B$. For instance, in (2.17), we assume that the packets $x^{[i(B+T)+B]}_{i(B+T)}$ were erased so they are not used in the expression.

Next, we will prove the following relation for $n \geq 0$:

\[
H(W^{n}_{0}) \geq H(V^{n-1}_{0}) + H\left(W_{n} \bigg| V^{n-1}_{0}x^{[n(B+T)-1]}_{0}\right). \tag{2.19}
\]

For the base case, substitute $n = 0$ into (2.19). This gives

\[
H(W_{0}) \geq H(V^{-1}_{0}) + H\left(W_{0} \bigg| V^{-1}_{0}x^{[-1]}_{0}\right)
\]

\[
= H(W_{0}) \tag{2.20}
\]

which is true. We assume that (2.19) is true for $n = k$ in the induction step, which gives:

\[
H(W^{k}_{0}) \geq H(V^{k-1}_{0}) + H\left(W_{k} \bigg| V^{k-1}_{0}x^{[k(B+T)-1]}_{0}\right)
\]

\[
\overset{(a)}{=} H(V^{k-1}_{0}) + H\left(s^{[k(B+T)+B-1]}_{k(B+T)}W_{k} \bigg| V^{k-1}_{0}x^{[k(B+T)-1]}_{0}\right)
\]
\[-H\left(s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] V_{0}^{k-1} x \left[ \begin{array}{c} (k+B+T)-1 \\ 0 \end{array} \right] W_k \right)\]

\begin{align*}
&\equiv (b) H(V_0^{k-1}) + H\left(s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] W_k V_{0}^{k-1} x \left[ \begin{array}{c} (k+B+T)-1 \\ 0 \end{array} \right] \right) \\
&\equiv (c) H(V_0^{k-1}) + H\left(s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] V_{0}^{k-1} x \left[ \begin{array}{c} (k+B+T)-1 \\ 0 \end{array} \right] \right) \\
&\quad + H\left(W_k V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)-1 \\ 0 \end{array} \right] \right) \\
&\equiv (d) H(V_0^{k-1}) + H\left(s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] V_{0}^{k-1} \right) \\
&\quad + H\left(W_k V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)-1 \\ 0 \end{array} \right] \right) \\
&\geq (e) H\left(V_0^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] \right) + H\left(W_k V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right)
\end{align*}

(2.21)

Steps (a) uses the joint entropy expansion formula, step (b) uses (2.17) to remove the negative term and step (c) is a joint entropy expansion. Step (d) uses the fact that source packets are independent of each other, so therefore the source packets \(s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] \) must be independent of the past channel packets \(x \left[ \begin{array}{c} (k+B+T)-1 \\ 0 \end{array} \right] \). Step (e) joins the first two terms from (d), and also uses the fact that conditioning reduces entropy in the last term. Next, we add \(H(W_{k+1} W_0^k)\) to both sides,

\begin{align*}
H(W_0^{k+1}) &\geq H(V_0^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] ) + H\left(W_k V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \\
&\quad + H(W_{k+1} W_0^k) \\
&\equiv (a) H(V_0^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] ) \\
&\quad + H\left(W_k V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \\
&\quad + H\left(W_{k+1} V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \\
&\equiv (b) H(V_0^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] ) \\
&\quad + H\left(W_k V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \\
&\quad + H\left(W_{k+1} V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \\
&\equiv (c) H(V_0^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] ) \\
&\quad + H\left(W_k V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \\
&\quad + H\left(W_{k+1} V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \\
&\equiv (d) H(V_0^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] ) \\
&\quad + H\left(W_k V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \\
&\quad + H\left(W_{k+1} V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \\
&\equiv (e) H(V_0^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] ) \\
&\quad + H\left(W_k V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \\
&\quad + H\left(W_{k+1} V_{0}^{k-1} s \left[ \begin{array}{c} (k+B+T)+B-1 \\ k(B+T) \end{array} \right] x \left[ \begin{array}{c} (k+B+T)+B-1 \\ 0 \end{array} \right] \right) \]
Thus, (2.19) is true for \( n = k \geq 0 \), then it is also true for \( n = k + 1 \). Thus, (2.19) is true for \( n \geq 0 \). We take (2.19) and finalize it as

\[
H(W_0^n) \geq H(V_0^{n-1}) + H \left( W_n \left| V_0^{n-1} \cdot x \left[ \begin{array}{c} n(B+T)-1 \\ 0 \end{array} \right] \right. \right)
\]

\[
\geq H(V_0^{n-1}). \tag{2.23}
\]

Next, we expand the groups of channel packets

\[
H(W_0^n) = H \left( x \left[ \begin{array}{c} T+B-1 \\ B \end{array} \right] \cdot x \left[ \begin{array}{c} 2(T+B)-1 \\ (T+B)+B \end{array} \right] \cdot \cdots \cdot x \left[ \begin{array}{c} (n+1)(T+B)-1 \\ n(T+B)+B \end{array} \right] \right)
\]

\[
\leq \sum_{i=0}^{n} \sum_{j=B}^{T+B-1} H(x[i(T + B) + j])
\]

\[
= (n + 1) \cdot T \cdot H(x) \tag{2.24}
\]

and also expand the groups of source packets

\[
H(V_0^{n-1}) = H \left( s \left[ \begin{array}{c} n(T+B)-1 \\ 0 \end{array} \right] \right)
\]

\[
= \sum_{i=0}^{n(T+B)-1} H \left( s[i] \left| s \left[ i-1 \right] \right. \right)
\]

\[
\overset{(a)}{=} \sum_{i=0}^{n(T+B)-1} H(s[i])
\]

\[
= n \cdot (T + B) \cdot H(s) \tag{2.25}
\]
where step (a) is because the source packets are independent. Then we can take (2.23) and write it as

\[
H(W^n_0) \geq H(V^{n-1}_0)
\]

\[
(n + 1) \cdot T \cdot H(x) \geq n \cdot (T + B) \cdot H(s)
\]

\[
\frac{(n + 1)}{n} \cdot \frac{T}{T + B} \geq \frac{H(s)}{H(x)}.
\]

Finally, we conclude that any \((B, T)\) streaming erasure code must satisfy

\[
R \leq \frac{H(s)}{H(x)} \leq \frac{T}{T + B} \quad \text{as } n \rightarrow \infty
\]  

which gives our upper bound of the rate.

Similarly, for \(T < B\), we can construct an erasure channel sequence where all channel packets are erased. We can show that all source packets can still be recovered by 0 channel packets, which will give us the relation

\[
H(s) \leq 0 \cdot H(x)
\]

\[
R \leq \frac{H(s)}{H(x)} \leq 0
\]  

(2.27)

Rate must be non-negative, so \(R = 0\). □

While this alternative method may seem like a redundant version of Martinian’s proof, it is more rigorous and can be used in some situations where Martinian’s proof will not work. One issue, as mentioned before, is that Martinian’s proof expects the code to be deterministic. However, the information theoretic proof still applies when the code is not necessarily deterministic.

Another drawback of Martinian’s converse proof, is that it expects the \((B, T)\) code to be able to perfectly decode each source packet, or else the proof does not work. But if this assumption does not hold, the information theoretic proof can be easily adapted by invoking Fano’s Inequality. Finally in subsequent chapters we will see the need to extend the basic periodic erasure channel argument in several ways. We found information theoretic proof more powerful in these settings.

### 2.6 Summary

In this chapter, we introduced the notion of a streaming code, and showed its differences from a block code, yet at the same time, we showed that it can be constructed from a
block code. We described a streaming setup where the channel is a bursty packet-erasure channel, and showed how to prove its capacity with a forward proof and a converse proof. The forward proof is to simply find a code that can achieve capacity. We showed the converse proof as given by Martinian [12], but we also introduced a new rigorous converse proof method which uses information theoretic notation and will be useful in the next chapter.
Chapter 3

Multicast System

3.1 Multicast Double Receiver System

In this chapter we study a multicast extension of Martinian’s streaming setup. Each receiver is linked to the transmitter by a separate channel, which means each receiver has its own $B$ and $T$ parameter. The problem is to find the best achievable rate for this system. This is the problem considered by [2], [1] and [3]. We let $B_1$ be the burst erasure length of the channel for decoder 1, and $T_1$ be the allowable delay for decoder 1. Similarly, $B_2$ and $T_2$ are the same parameters for decoder 2. Without loss of generality, let us assume that $B_1 \leq B_2$. We will restrict $T_1 \geq B_1$ and $T_2 \geq B_2$, because the capacity will be 0 otherwise.

The system model is the same as Section 2.2, except that we now have two channels and two decoding functions. The encoding function is still

$$x[t] = f_t(s[0], \ldots, s[t]), \quad t \geq 0$$

$$(3.1)$$
Chapter 3. Multicast System

but, the two channels are:

\[
y_1[t] = \begin{cases} 
  \star & t \in [j_1, j_1 + B_1 - 1] \\
  x[t] & \text{otherwise}
\end{cases}
\]

\[
y_2[t] = \begin{cases} 
  \star & t \in [j_2, j_2 + B_2 - 1] \\
  x[t] & \text{otherwise}
\end{cases}
\] (3.2)

for some \( j_1, j_2 \in \mathbb{Z} \). The two decoding functions are:

\[
\hat{s}_1[t] = g_{1t}(y_1[0], \ldots, y_1[t + T_1])
\]

\[
\hat{s}_2[t] = g_{2t}(y_2[0], \ldots, y_2[t + T_2])
\] (3.3)

and must satisfy \( \forall t \geq 0 \)

\[
Pr(\hat{s}_1[t] \neq s[t]) = 0
\]

\[
Pr(\hat{s}_2[t] \neq s[t]) = 0.
\] (3.4)

The rate is still defined as

\[
R \triangleq \frac{H(s)}{|x|} \leq \frac{H(s)}{H(x)}.
\]

The rate and capacity will therefore be a function of four variables: \( B_1, B_2, T_1 \) and \( T_2 \). Recall that a unicast system using a \((B_1, T_1)\) code has a capacity of \( \frac{T_1}{T_1 + B_1} \) while a unicast \((B_2, T_2)\) system has a capacity of \( \frac{T_2}{T_2 + B_2} \). Logically, the doublecast system cannot have a higher capacity than either.

We can start by solving the problem where given \( B_1, T_1 \) and \( B_2 \) and that we want to maintain user 1’s capacity of \( \frac{T_1}{T_1 + B_1} \), what is the lowest value of \( T_2 \) that can be supported. A new type of code known as diversity embedded streaming erasure codes (DE-SCo) is introduced in [2]. The code is best illustrated through an example.

In our example, we have \((B_1, T_1) = (1, 2)\) and \( B_2 = 2 \). We can achieve a unicast rate of \( 2/3 \) for user 1 using the streaming code in Figure 3.2(a). The same rate can be achieved for user 2 if \( T_2 = 4 \), using the streaming code in Figure 3.2(b). Note that the \((2, 4)\) code, we can use two source sub-symbols per packet, instead of four, if diagonal interleaving is applied as shown in the figure. To form a doublecast code, the parity sub-symbols of both codes can be placed together in the same packet as shown in Figure 3.2(c).
(a) SCo Construction for \((B,T) = (1, 2)\)

<table>
<thead>
<tr>
<th></th>
<th>(a_{i-1})</th>
<th>(a_i)</th>
<th>(a_{i+1})</th>
<th>(a_{i+2})</th>
<th>(a_{i+3})</th>
<th>(a_{i+4})</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(b_{i-1})</td>
<td>(b_i)</td>
<td>(b_{i+1})</td>
<td>(b_{i+2})</td>
<td>(b_{i+3})</td>
<td>(b_{i+4})</td>
</tr>
<tr>
<td></td>
<td>(a_i)</td>
<td>(a_{i-1} + b_i)</td>
<td>(a_i + b_{i+1})</td>
<td>(a_{i+1} + b_{i+2})</td>
<td>(a_{i+2} + b_{i+3})</td>
<td></td>
</tr>
</tbody>
</table>

(b) SCo Construction for \((B,T) = (2, 4)\)

<table>
<thead>
<tr>
<th></th>
<th>(a_{i-1})</th>
<th>(a_i)</th>
<th>(a_{i+1})</th>
<th>(a_{i+2})</th>
<th>(a_{i+3})</th>
<th>(a_{i+4})</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(b_{i-1})</td>
<td>(b_i)</td>
<td>(b_{i+1})</td>
<td>(b_{i+2})</td>
<td>(b_{i+3})</td>
<td>(b_{i+4})</td>
</tr>
<tr>
<td></td>
<td>(a_i)</td>
<td>(a_{i-3} + b_{i-2})</td>
<td>(a_{i-2} + b_{i-1})</td>
<td>(a_{i-3} + b_{i-1})</td>
<td>(a_{i-2} + b_{i-1})</td>
<td>(a_{i-1} + b_{i+1})</td>
</tr>
</tbody>
</table>

(c) Doublecast code using code concatenation

Figure 3.2: Unicast streaming codes are in the top two figures. The parity checks of both codes are used to form a \((1, 2), (2, 4)\) doublecast streaming code.
However, the rate of this doublecast code is equal to $1/2$, which is lower than $2/3$. One method of maintaining the rate at $2/3$ for a doublecast code is to simply add the two parity sub-symbols together, so they will still occupy the space of one sub-symbol. This is done in Figure 3.3. The parity checks for user 2 have been delayed by an additional 2 time units before being added to user 1’s parity checks. The reason for this so that user 2’s parity checks can avoid interference from user 1’s parity checks. Because of the shift by two time units, the delay for user 2 is now $T_2 = 6$. However, this delay is still not optimal.

![Figure 3.3: Addition of both parity packets](image)

Finally, we can use the code in Figure 3.4 as a doublecast streaming code. The code is a DE-SCo construction, and in this case, it is able to achieve $T_2 = 5$ with a rate of $2/3$. The technique is to diagonally interleave a $(1, 2)$ code along the main diagonal as usual for user 1. But for user 2, we interleave a $(1, 2)$ code along the other diagonal, and then shift the resulting parity check by 3 time units to the right. Then, we add both of the parity checks together. The DE-SCo construction is optimal, it achieves the lowest value of $T_2$ possible, when the rate of the code is the same as user 1’s unicast capacity.

![Figure 3.4: DE-SCo code construction achieving $(1, 2), (2, 5)$](image)

In general, the DE-SCo construction is able to achieve the lowest $T_2$ value of

$$T_2^* = \alpha T_1 + B_1$$  (3.5)
where $\alpha$ is an integer greater than 1 which satisfies $B_2 = \alpha B_1$. Further information on DE-SCo can be found in [2].

Now that we have shown how user 1’s capacity can be maintained, it would be interesting to see which values of $(B_1, T_1)$, $(B_2, T_2)$ have the same capacity as user 1’s unicast capacity. We want to use the converse proof to find an upper bound on capacity.

First, consider the erasure channel sequence in Figure 3.5(a). If $T_2 > T_1 + B_1$, then we can use this channel sequence to prove an upper bound on capacity. Consider the first period, which has $B_2$ packet erasures followed by $T_2 - B_1$ non-erasures. At time $B_2 + T_2 - B_1 - 1$, we can use the $(B_2, T_2)$ decoder to recover the first $B_2 - B_1$ source packets $s\left[\begin{array}{c} B_2 - B_1 - 1 \\ 0 \end{array}\right]$. The remaining erasures $s\left[\begin{array}{c} B_2 - 1 \\ B_2 - B_1 \end{array}\right]$ can be recovered using the $(B_1, T_1)$ decoder. This gives the following upper bound for rate

$$R \leq \frac{T_2 - B_1}{T_2 - B_1 + B_2}.$$  \hfill (3.6)

The second erasure channel sequence in Figure 3.5(b) is used in the case when $T_2 \leq T_1 + B_1$. Each period has $B_2$ erasures followed by $T_1$ non-erasures. At time $B_2 + T_1 - 1$, the $(B_2, T_2)$ decoder can be used to recover the first $B_2 - T_2 + T_1$ missing source packets $s\left[\begin{array}{c} B_2 - T_2 + T_1 \\ 0 \end{array}\right]$. The remaining source packets $s\left[\begin{array}{c} B_2 - 1 \\ B_2 - T_2 + T_1 \end{array}\right]$ can be recovered using the $(B_1, T_1)$ decoder. The rate is then bounded by

$$R \leq \frac{T_1}{T_1 + B_2}.$$  \hfill (3.7)

![Periodic erasure channel sequences for (a) $T_2 > T_1 + B_1$ (b) $T_2 \leq T_1 + B_1$.](image)

Figure 3.5: Periodic erasure channel sequences for (a) $T_2 > T_1 + B_1$ (b) $T_2 \leq T_1 + B_1$.

Define $C^+$ as

$$C^+ = \begin{cases} \frac{T_2 - B_1}{T_2 - B_1 + B_2}, & T_2 > T_1 + B_1 \\ \frac{T_1}{T_1 + B_2}, & T_2 \leq T_1 + B_1. \end{cases}$$  \hfill (3.8)
The doublecast capacity must be upper bounded by $C^+$ as well as the single user capacities for user 1 and user 2. We can write that the capacity $C$ is upper bounded by $C_U$ where

$$C_U = \min\left\{ C^+, \frac{T_1}{T_1 + B_1}, \frac{T_2}{T_2 + B_2} \right\}. \quad (3.9)$$

The minimum in $C_U$ will depend on the values for $(B_1, T_1)$ and $(B_2, T_2)$. When we draw out the boundaries for where the minimum changes in $C_U$, we end up with the graph in Figure 3.6. In this graph, the values of $B_1$ and $T_1$ are assumed to be fixed, while $B_2$ and $T_2$ can vary. There are six different regions labelled (a) to (f) which are bounded by the lines.

Evaluating $C_U$ for each of the regions (a) to (d) gives

(a) $C^U_a = \frac{T_1}{T_1 + B_1}$

(b) $C^U_b = \frac{T_2 - B_1}{T_2 - B_1 + B_2}$

(c) $C^U_c = \frac{T_1}{T_1 + B_2}$

(d) $C^U_d = \frac{T_2}{T_2 + B_2}$. \quad (3.10)

Each of these bounds is tight in regions (a) to (d). A full treatment is given in [3]. However, for regions (e) and (f), a different technique must be used to provide an upper bound of capacity, and that is the focus of this chapter.

Figure 3.7 is a redrawn version of Figure 3.6, but with $B_1$ and $B_2$ assumed fixed, while $T_1$ and $T_2$ vary. Looking at the capacity equations (3.10), we can see that $C^U_a$ and $C^U_c$ are constant in $T_2$ while $C^U_b$ and $C^U_d$ are constant in $T_1$. This leads to the property that the dotted line drawn in Figure 3.7 is a contour line of constant capacity.

### 3.2 New Upper Bound — (e) Region

In this section we new upper bound for the (e) compared to $C^U$ in (3.9). Recall that the (e)-region is defined by the inequalities

$$T_2 \leq B_2 + B_1$$

$$T_2 \geq T_1 + B_1$$
Figure 3.6: Multicast capacity regions where $B_1$ and $T_1$ are fixed.

Figure 3.7: Multicast capacity regions where $B_1$ and $B_2$ are fixed. This variant of the graph is for the case when $2B_1 < B_2$. 
in addition to the usual inequalities for our problem

\[ T_1 \geq B_1 \]
\[ T_2 \geq B_2 \]
\[ B_2 \geq B_1. \]

We establish the following upper bound on the (e) region:\footnote{We have been able to establish that this bound is the true capacity. However the associated code construction will be reported elsewhere.}

\[ C_e = \frac{T_1}{2T_1 + B_1 + B_2 - T_2}. \]

The key idea is to introduce a technique that involves revealing a subset of source packets to the decoder when they cannot be decoded. By appropriately accounting for the rate of these packets we can get a tighter upper bound compared to \( C_U \).

We illustrate this idea using a periodic erasure channel sequence shown in Fig. 3.9, where each period has \( B_2 \) erasures followed by \( T_1 \) non-erasures. We can assign

\[ a = T_1 + B_2 - T_2 \]
\[ b = B_2 - B_1 \]
\[ c = B_2 \]
\[ d = B_2 + T_1 \]
\[ W_i = x^{\left(\frac{(i+1)d-1}{id+c}\right)} \]
\[ V_i = s^{\left(\frac{id+a-1}{id}\right)} s^{\left(\frac{(i+1)d-1}{id+b}\right)}. \]

The idea behind the converse proof is similar to before, but instead we have two decoding functions to use.

\[ \text{Link:} \quad W_0 \quad W_1 \quad W_2 \quad \ldots \]
\[ B_2 \quad T_1 \quad B_2 \quad T_1 \quad B_2 \quad T_1 \quad \ldots \]

Figure 3.8: The periodic erasure channel sequence used to prove an upper bound on capacity and the locations of the channel packets in \( W_i \)
We use the decoder of receiver 2 to recover $s_{c-1}^{a-1}$ within a delay of $T_2$ using the channel packets $x_{d-1}^{c}$. We then reveal the channel symbols $x_{b-1}^{a}$, hence the technique name “revealing packets”. The decoder of receiver 1 can now be used to recover the next $B_1$ source packets, which are the packets $s_{c-1}^{b}$, using $x_{d-1}^{c}$ again. In general, we may not have a systematic code, so even if $x_{d-1}^{c}$ is received, we may not be able to recover the corresponding source packet $s_{d-1}^{c}$. Instead, $s_{d-1}^{c}$ can be recovered using the second decoder and the first and second sets of $T_1$ channel packets that are not erased, i.e. $x_{d-1}^{c}$ and $x_{2d-1}^{c+d}$.

So far, we have recovered $(T_1 + B_2 - T_2) + B_1 + T_1 = 2T_1 + B_1 + B_2 - T_2$ source packets, using $2T_1$ channel packets. We do not include the source packets $s_{b-1}^{a}$, because it cannot be decoded from the information in the non-erased channel packets. The channel has a period of $B_2 + T_1$ packets, and if we had $n$ periods, then we would be able to recover $n(2T_1 + B_1 + B_2 - T_2)$ source packets using $(n + 1)T_1$ channel packets. Therefore, we can
suppose that the upper bound on the multicast streaming capacity is given by

\[ n \cdot (2T_1 + B_1 + B_2 - T_2) \cdot H(s) \leq (n + 1) \cdot T_1 \cdot H(x) \]

\[ R \leq \frac{H(s)}{H(x)} \leq \frac{n + 1}{n} \cdot \frac{T_1}{2T_1 + B_1 + B_2 - T_2} \]

\[ \lim_{n \to \infty} \frac{T_1}{2T_1 + B_1 + B_2 - T_2} \]

(3.11)

The more formal theoretical proof is given below.

**Proof.** From the \((B_1, T_1)\) decoder, we have for \(i \geq 0\):

\[ H\left(s \begin{bmatrix} id+c-1 \\ id+b \end{bmatrix} | x \begin{bmatrix} id+b-1 \\ 0 \end{bmatrix} W_i \right) = 0 \]

(3.12)

From the \((B_2, T_2)\) decoder, we have for \(i \geq 0\):

\[ H\left(s \begin{bmatrix} id+a-1 \\ id \end{bmatrix} | x \begin{bmatrix} id-1 \\ 0 \end{bmatrix} W_i \right) = 0 \]

(3.13)

\[ H\left(s \begin{bmatrix} (i+1)d-1 \\ id+c \end{bmatrix} | x \begin{bmatrix} id+c-1 \\ 0 \end{bmatrix} W_i^{i+1} \right) = 0. \]

(3.14)

We want to use mathematical induction to prove that for \(n \geq 0\)

\[ H(W_0^n) \geq H(V_0^{n-1}) + H\left(W_n \bigg| V_0^{n-1} x \begin{bmatrix} nd-1 \\ 0 \end{bmatrix} \right). \]

(3.15)

The base case for (3.15) is given by substituting \(n = 0\) into it:

\[ H(W_0) \geq H(V_0^{-1}) + H\left(W_0 \bigg| V_0^{-1} x \begin{bmatrix} -1 \\ 0 \end{bmatrix} \right) \]

\[ = H(W_0) \]

(3.16)

which is obviously true. Let us assume that (3.15) is true for \(n = k\). This gives:

\[ H(W_0^k) \geq H(V_0^{k-1}) + H\left(W_k \bigg| V_0^{k-1} x \begin{bmatrix} kd-1 \\ 0 \end{bmatrix} \right). \]

(3.17)

We can manipulate the expression in two parts. In the first part, we use \(W_k\) to recover the source packets \(s \begin{bmatrix} kd+a-1 \\ kd \end{bmatrix}\) and \(s \begin{bmatrix} kd+c-1 \\ kd+b \end{bmatrix}\).

\[ H(W_0^k) \geq H(V_0^{k-1}) + H\left(W_k \bigg| V_0^{k-1} x \begin{bmatrix} kd-1 \\ 0 \end{bmatrix} \right) \]

\[ = H(V_0^{k-1}) + H\left(s \begin{bmatrix} kd+a-1 \\ kd \end{bmatrix} W_k \bigg| V_0^{k-1} x \begin{bmatrix} kd-1 \\ 0 \end{bmatrix} \right) - H\left(s \begin{bmatrix} kd+a-1 \\ kd \end{bmatrix} \bigg| V_0^{k-1} x \begin{bmatrix} kd-1 \\ 0 \end{bmatrix} \right) W_k \]

\[ \overset{(a)}{=} H(V_0^{k-1}) + H\left(s \begin{bmatrix} kd+a-1 \\ kd \end{bmatrix} W_k \bigg| V_0^{k-1} x \begin{bmatrix} kd-1 \\ 0 \end{bmatrix} \right) \]

\[ = H(V_0^{k-1}) + H\left(s \begin{bmatrix} kd+a-1 \\ kd \end{bmatrix} \bigg| V_0^{k-1} x \begin{bmatrix} kd-1 \\ 0 \end{bmatrix} \right) + H(W_k \bigg| V_0^{k-1} s \begin{bmatrix} kd+a-1 \\ kd \end{bmatrix} x \begin{bmatrix} kd-1 \\ 0 \end{bmatrix}) \]
We use (3.13) to remove the negative term before step (a). Similarly, we remove the negative term before step (c) using (3.12). Steps (b) and (d) use the fact that source packets are independent of each other and of previous channel packets.

In the second part, we add $H(W_{k+1}|W_0^k)$ to both sides of the inequality. Because the channel code is not necessarily systematic, we will use the additional channel packets in $W_{k+1}$ to help decode the source packets $s^{(k+1)d-1}_{kd+c}$.

\begin{align*}
H(W_0^{k+1}) \geq & \ H\left(V_0^{k-1}s^{kd+a-1}_{kd}\right) + H\left(W_kV_0^{k-1}s^{kd+a-1}_{kd}x^{kd-1}_{0}\right) \\
\geq & \ H\left(V_0^{k-1}s^{kd+a-1}_{kd}\right) + H\left(W_kV_0^{k-1}s^{kd+a-1}_{kd}x^{kd+b-1}_{0}\right) \\
= & \ H\left(V_0^{k-1}s^{kd+a-1}_{kd}\right) + H\left(s^{kd+c-1}_{kd+b}W_kV_0^{k-1}s^{kd+a-1}_{kd}x^{kd+b-1}_{0}\right) \\
\geq & \ H\left(V_0^{k-1}s^{kd+a-1}_{kd}\right) + H\left(s^{kd+c-1}_{kd+b}\right) + H\left(W_kV_0^{k-1}s^{kd+a-1}_{kd}x^{kd+b-1}_{0}\right) \\
= & \ H\left(V_0^{k-1}s^{kd+a-1}_{kd}\right) + H\left(s^{kd+c-1}_{kd+b}\right) + H\left(W_kV_0^{k-1}s^{kd+a-1}_{kd}x^{kd+b-1}_{0}\right) \\
\geq & \ H\left(V_0^{k-1}s^{kd+a-1}_{kd}\right) + H\left(s^{kd+c-1}_{kd+b}\right) + H\left(W_kV_0^{k-1}s^{kd+a-1}_{kd}x^{kd+b-1}_{0}\right) \\
= & \ H\left(V_0^{k-1}s^{kd+a-1}_{kd}\right) + H\left(s^{kd+c-1}_{kd+b}\right) + H\left(W_kV_0^{k-1}s^{kd+a-1}_{kd}x^{kd+b-1}_{0}\right) \\
\geq & \ H\left(V_0^{k-1}s^{kd+a-1}_{kd}\right) + H\left(s^{kd+c-1}_{kd+b}\right) + H\left(W_kV_0^{k-1}s^{kd+a-1}_{kd}x^{kd+b-1}_{0}\right) \\
\end{align*}

(3.18)
\[ f \leq H(V_0^{k-1}s^{kd+a-1}x^{(k+1)d-1}) + H(W_0^{k-1}s^{kd+b}x^{kd+c-1}) \]
\[ = H(V_0^k) + H(W_0^{k+1}s^{kd+c-1}) \]
\[ = H(V_0^k) + H(W_k|x^{kd+c-1}) + H(W_{k+1}|V_0^{k-1}s^{(k+1)d-1})W_k \]
\[ \geq H(V_0^k) + H(W_{k+1}|V_0^{k-1}s^{(k+1)d-1}) \] (3.19)

Once again, we remove the negative term before step (e) using (3.14). Steps (f) uses the fact that each source packet is independent of each other.

The working in (3.19) shows that if (3.15) is true for \( n = k \), then it is also true for \( n = k + 1 \). By induction, (3.15) is true for \( n \geq 0 \). Finally,
\[ H(W_0^n) \geq H(V_0^{n-1}) + H(W_n|x^{n-1}) \]
\[ \geq H(V_0^{n-1}). \]

Using the fact that all of the channel packets have the same entropy, and all of the source packets have the same entropy, we can continue to get
\[ H(W_0^n) \geq H(V_0^{n-1}) \]
\[ \frac{n+1}{n} \cdot \frac{T_1}{2T_1 + B_1 + B_2 - T_2} \geq \frac{H(s)}{H(x)}. \] (3.20)

Finally, we get
\[ R \leq \frac{H(s)}{H(x)} \leq \frac{T_1}{2T_1 + B_1 + B_2 - T_2}. \] (as \( n \to \infty \)) (3.21)

Therefore, any \((B_1, T_1), (B_2, T_2)\) code in the (e)-region must satisfy (3.21).

3.3 Improved Upper Bound for the (f)-region

Recall that the (f)-region is described by the following relations:
\[ B_1 \leq T_1 < B_2 \leq T_2 < T_1 + B_1. \]

In this section we will illustrate a new upper bounding technique for this region. We will restrict our discussion to the special case of \( T_2 = B_2 \), although this technique can be generalized. The generalization will be reported elsewhere.

We establish the following upper bound on the multicast capacity.
\[ C_f = \frac{T_1}{2T_1 + B_1}. \] (3.22)
We can prove the upper bound of the rate in the (f)-region using the periodic erasure channel sequence given in Figure 3.11. Each period has $B_2$ erasures followed by $T_1$ non-erasures.

![Figure 3.11: The periodic erasure channel sequence used to prove an upper bound on capacity](image)

It so happens that the $B_2 = T_2$ restriction means that we can prove the converse by only analyzing one period. The reason will be made clear later. But this simplifies the proof and allows us to study the technique of double counting source packets more easily.

![Figure 3.12: The periodic erasure channel sequence used to prove an upper bound on capacity](image)

In Figure 3.12, we have the first period of the erasure channel. The key is to show that the received channel packets $x_{[B_2+T_1-1]}$ alone can recover all of the source packets in the period, but there is enough information in the channel packets to recover some of the source packets twice. The fact that we have two decoders allows some of the source packets to be decoded by mutually exclusive groups of channel packets. The proof uses this to show that the channel packets, when put together, must hold redundant information. As we have seen from the previous converse proofs, the more information that each channel packet must hold, the lower the rate will be.

The source packets that can be recovered by $x_{[B_2+T_1-1]}$ are $s_{T_1-1}$ and $s_{B_2+T_1-1}$, using user 2’s decoder and $s_{B_2-B_1}$ using user 1’s decoder. As Figure 3.12 shows, the first two groups of source packets overlap. The overlap consists of the packets $s_{B_2-B_1}$. The reason why we can use a single period in the proof is because the $B_2 = T_2$ constraint allows us to decode the final group of source packets $s_{B_2+T_1-1}$ using only the packets.
Chapter 3. Multicast System

\[ x \left[ \frac{B_2 + T_1 - 1}{B_2} \right] \] and does not require any future channel packets.

Assuming that what we have just described is possible, then we have \( T_1 \) channel packets recovered \( 2T_1 + B_1 \) source packets. We should be able to write the relation:

\[
(2T_1 + B_1) \cdot H(s) \leq T_1 \cdot H(x)
\]

\[
R \leq \frac{H(s)}{H(x)} \leq \frac{T_1}{2T_1 + B_1}
\]

(3.23)

The formal proof shows that this is indeed possible.

**Proof.** We can split the proof into three major parts.

1. The source packets \( s \left[ \frac{T_1 - B_1 - 1}{0} \right] \) can be recovered from the available channel packets \( x \left[ \frac{B_2 + T_1 - B_1 - 1}{B_2} \right] \) using the \((B_2, B_2)\) decoder, so we can write

\[
H \left( s \left[ \frac{T_1 - B_1 - 1}{0} \right] x \left[ \frac{B_2 + T_1 - B_1 - 1}{B_2} \right] \right) = 0.
\]

(3.24)

Next, we can write

\[
H \left( x \left[ \frac{B_2 + T_1 - B_1 - 1}{B_2} \right] \right) = H \left( s \left[ \frac{T_1 - B_1 - 1}{0} \right] x \left[ \frac{B_2 + T_1 - B_1 - 1}{B_2} \right] \right) - H \left( s \left[ \frac{T_1 - B_1 - 1}{0} \right] x \left[ \frac{B_2 + T_1 - B_1 - 1}{B_2} \right] \right)
\]

\[
\overset{(a)}{=} H \left( s \left[ \frac{T_1 - B_1 - 1}{0} \right] x \left[ \frac{B_2 + T_1 - B_1 - 1}{B_2} \right] \right)
\]

\[
= H \left( s \left[ \frac{T_1 - B_1 - 1}{0} \right] \right) + H \left( x \left[ \frac{B_2 + T_1 - B_1 - 1}{B_2} \right] s \left[ \frac{T_1 - B_1 - 1}{0} \right] \right)
\]

\[
\geq H \left( s \left[ \frac{T_1 - B_1 - 1}{0} \right] \right) + H \left( x \left[ \frac{B_2 + T_1 - B_1 - 1}{B_2} \right] s \left[ \frac{T_1 - B_1 - 1}{0} \right] x \left[ \frac{T_1 - B_1 - 1}{0} \right] \right).
\]

(3.25)

We used (3.24) to remove the negative term before step (a).

2. In this step, we want to prove the following inequality for \( m \geq B_2 + T_1 - B_1 - 1 \):

\[
\sum_{i=B_2}^{m} H(x[i]) \geq H \left( s \left[ \frac{m-B_2}{0} \right] \right) + H \left( s \left[ \frac{m-T_1}{B_2-B_1} \right] \right) + H \left( x \left[ \frac{m}{B_2} \right] s \left[ \frac{m-B_2}{0} \right] s \left[ \frac{m-T_1}{B_2-B_1} \right] x \left[ \frac{m-B_2}{0} \right] \right)
\]

(3.26)

Using the first decoder with a \((B_1, T_1)\) property, we can write the following relation:

\[
H \left( s[i - T_1] x \left[ i - T_1 + B_1 \right] x \left[ i - T_1 - 1 \right] \right) = 0.
\]

(3.27)

Using the \((B_2, B_2)\) decoder, we can write the following relation:

\[
H \left( s[i - B_2] x[i] x \left[ i - B_2 - 1 \right] \right) = 0
\]

(3.28)
which can be used in the following steps

\[ \begin{align*}
H\left(x[i] \mid x\left[i-B_2+1 \atop 0 \right]\right) &= H\left(s[i-B_2]x[i] \mid x\left[i-B_2+1 \atop 0 \right]\right) - H\left(s[i-B_2]x[i] \mid x\left[i-B_2+1 \atop 0 \right]\right) \\
&= H\left(s[i-B_2]x[i] \mid x\left[i-B_2+1 \atop 0 \right]\right) \\
&= H\left(s[i-B_2]\right) + H\left(x[i] \mid s[i-B_2]x\left[i-B_2+1 \atop 0 \right]\right)
\end{align*} \]

(3.29)

The second decoder’s (3.28) was used to remove the negative term before step (a).

Now we can use mathematical induction to prove (3.26). For the base case, we substitute \( m = B_2 + T_1 - B_1 - 1 \)

\[ \sum_{i=B_2}^{B_2+T_1-B_1-1} H(x[i]) \geq H\left(s\left[T_1-B_1-1 \atop 0 \right]\right) + H\left(s\left[B_2-B_1-1 \atop 0 \right]\right) \\
+ H\left(x\left[B_2+T_1-B_1-1 \atop 0 \right]\mid s\left[T_1-B_1-1 \atop 0 \right]\right) \left(s\left[B_2-B_1-1 \atop 0 \right]\right) x\left[T_1-B_1-1 \atop 0 \right]
\]

(3.30)

This is proved by the result of (3.25).

Assume that (3.26) is true for \( m = j \), which gives us

\[ \sum_{i=B_2}^{j} H(x[i]) \geq H\left(s\left[j-B_2 \atop 0 \right]\right) + H\left(s\left[j-T_1 \atop B_2-B_1 \right]\right) + H\left(x\left[j \atop B_2 \right]\mid s\left[j-B_2 \atop 0 \right]\right) \left(s\left[j-T_1 \atop B_2-B_1 \right]\right) x\left[j-B_2 \atop 0 \right]
\]

(3.31)

We add \( H(x[j + 1]) \) to both sides, and use the result of (3.29) to give:

\[ \sum_{i=B_2}^{j+1} H(x[i]) \overset{(a)}{\geq} H\left(s\left[j-B_2 \atop 0 \right]\right) + H\left(s\left[j-T_1 \atop B_2-B_1 \right]\right) + H\left(x\left[j \atop B_2 \right]\mid s\left[j-B_2 \atop 0 \right]\right) \left(s\left[j-T_1 \atop B_2-B_1 \right]\right) x\left[j-B_2 \atop 0 \right] \\
+ H(s[j + 1 - B_2]) + H\left(x[j+1] \mid s[j + 1 - B_2]x\left[j+1-B_2 \atop 0 \right]\right) \\
\geq H\left(s\left[j+1-B_2 \atop 0 \right]\right) + H\left(s\left[j-T_1 \atop B_2-B_1 \right]\right) \\
+ H\left(x\left[j+1 \atop B_2 \right]\mid s\left[j+1-B_2 \atop 0 \right]\right) \left(s\left[j-T_1 \atop B_2-B_1 \right]\right) x\left[j+1-B_2 \atop 0 \right]
\]

(3.32)
Chapter 3. Multicast System

37

\( \subseteq H( s[j+1-B_2] ) + H( s[j-T_1]_{B_2-B-1} ) + H( s[j+1-B_2] ) s[j+1-B_2] x[j+1-B_2] ) + H( s[j+1-B_2] ) s[j-T_1]_{B_2-B-1} ) + H( s[j+1-B_2] ) s[j+1-B_2] x[j+1-B_2] ) + H( x[j+1] ] s[j+1-B_2] s[j+1-T_1]_{B_2-B-1} ) x[j+1-B_2] ) \)

\( \sum_{i=B_2}^{j+1} H(x[i]) \geq H( s[j+1-B_2] ) + H( s[j+1-T_2]_{B_2-B-1} ) + H( x[j+1] ] s[j+1-B_2] s[j+1-T_1]_{B_2-B-1} ) x[j+1-B_2] ) \)

Step (a) is the addition of (3.29) and (3.26), step (b) uses (3.27) to remove the negative term in the previous step, and step (c) uses the fact that the source packets are independent of each other. The result is the form (3.26) for \( m = l + 1 \). By induction, we have proved (3.26) for \( m \geq B_2 + T_1 - B_1 - 1 \).

3. We substitute \( m = B_2 + T_1 - 1 \) into (3.26)

\( \sum_{i=B_2}^{B_2+T_1-1} H(x[i]) \geq H( s[T_1-1]_{B_2-B-1} ) + H( s[B_2-1]_{B_2-B-1} ) + H( x[B_2+T_1-1] s[T_1-1]_{B_2-B-1} s[B_2-1] x[T_1-1]_{B_2-B-1} ) \).

We can recover \( s[B_2+T_1-1]_{B_2} \) from \( x[B_2+T_1-1]_{B_2} \) given the previous channel symbols \( x[B_2-1]_{0} \) using decoder 2, so we can write

\( H( s[B_2+T_1-1]_{B_2} x[B_2+T_1-1]_{0} ) = 0. \)

We continue with (3.33) as such

\( \sum_{i=B_2}^{B_2+T_1-1} H(x[i]) \geq H( s[T_1-1]_{B_2-B-1} ) + H( s[B_2-1]_{B_2-B-1} ) + H( x[B_2+T_1-1] s[T_1-1]_{B_2-B-1} s[B_2-1] x[T_1-1]_{B_2-B-1} ) + H( x[B_2+T_1-1] s[T_1-1]_{B_2-B-1} s[B_2-1] x[B_2-1]_{B_2-B-1} ) \)
where step (a) makes use of (3.34). Finally, manipulation of the information theoretic converse proof, and Martinian’s converse proof proved using Martinian’s converse proof. However, the other two regions require careful formulae. For four out of six of these regions, the capacity could be easily found and $B_2$ there are two users. The capacity was revealed to be a piecewise function of the four parameters $B_1$, $T_1$, $B_2$ and $T_2$, and we identified regions which have different capacity formulae. For four out of six of these regions, the capacity could be easily found and proved using Martinian’s converse proof. However, the other two regions require careful manipulation of the information theoretic converse proof, and Martinian’s converse proof

\[
\begin{align*}
-H\left( s \left[ \begin{array}{c}
B_2 + T_1 - 1 \\
B_2 
\end{array} \right] \left| s \right[ \begin{array}{c}
T_1 - 1 \\
0 
\end{array} \right] s \left[ \begin{array}{c}
B_2 - 1 \\
B_2 - B_1 
\end{array} \right] x \left[ \begin{array}{c}
B_2 + T_1 - 1 \\
0 
\end{array} \right] \right) \\
\overset{(a)}{=} H\left( s \left[ \begin{array}{c}
T_1 - 1 \\
0 
\end{array} \right] \right) + H\left( s \left[ \begin{array}{c}
B_2 - 1 \\
B_2 - B_1 
\end{array} \right] \right) \\
+ H\left( s \left[ \begin{array}{c}
B_2 + T_1 - 1 \\
B_2 
\end{array} \right] x \left[ \begin{array}{c}
B_2 + T_1 - 1 \\
B_2 
\end{array} \right] \left| s \right[ \begin{array}{c}
T_1 - 1 \\
0 
\end{array} \right] s \left[ \begin{array}{c}
B_2 - 1 \\
B_2 - B_1 
\end{array} \right] x \left[ \begin{array}{c}
B_2 - 1 \\
0 
\end{array} \right] \right) \\
= H\left( s \left[ \begin{array}{c}
B_2 + T_1 - 1 \\
B_2 
\end{array} \right] \right) + H\left( s \left[ \begin{array}{c}
B_2 - 1 \\
B_2 - B_1 
\end{array} \right] \right) \\
+ H\left( x \left[ \begin{array}{c}
B_2 + T_1 - 1 \\
B_2 
\end{array} \right] \left| s \right[ \begin{array}{c}
T_1 - 1 \\
0 
\end{array} \right] s \left[ \begin{array}{c}
B_2 + T_1 - 1 \\
B_2 - B_1 
\end{array} \right] x \left[ \begin{array}{c}
B_2 - 1 \\
0 
\end{array} \right] \right) \\
\geq H\left( s \left[ \begin{array}{c}
T_1 - 1 \\
0 
\end{array} \right] \right) + H\left( s \left[ \begin{array}{c}
B_2 + T_1 - 1 \\
B_2 - B_1 
\end{array} \right] \right)
\end{align*}
\]

(3.35)

where step (a) makes use of (3.34). Finally,

\[
\sum_{i=B_2}^{B_2 + T_1 - 1} H(x[i]) \geq H\left( s \left[ \begin{array}{c}
T_1 - 1 \\
0 
\end{array} \right] \right) + H\left( s \left[ \begin{array}{c}
B_2 + T_1 - 1 \\
B_2 - B_1 
\end{array} \right] \right)
\]

\[
T_1 \cdot H(x) \geq (2T_1 + B_1) \cdot H(s)
\]

\[
R \leq \frac{H(s)}{H(x)} \leq \frac{T_1}{2T_1 + B_1}
\]

(3.36)

which is the proper upper bound.

In general, if we do not have $B_2 = T_2$, then the proof would require us to consider several periods of the erasure channel, and not only one period. If this is the case, then the proof could require two layers of mathematical induction: the inner layer would be used to double count source packets as in this section, and the outer layer would be to show that this is still the case across several periods as in the previous section.

### 3.4 Summary

This chapter introduced the multicast system, and specifically investigated the case where there are two users. The capacity was revealed to be a piecewise function of the four parameters $B_1$, $T_1$, $B_2$ and $T_2$, and we identified regions which have different capacity formulae. For four out of six of these regions, the capacity could be easily found and proved using Martinian’s converse proof. However, the other two regions require careful manipulation of the information theoretic converse proof, and Martinian’s converse proof
cannot be used to find the tightest upper bound for the capacity. The code constructions are not given in this chapter, but can be found in [3] and future publications.
Chapter 4

Multi-Link Streaming Erasure Codes

4.1 System Model for two links

As with our previous models, time is discretized into time slots. At the beginning of time slot \( t \geq 0 \), where \( t \) is an integer, a source packet \( s[t] \) arrives at the transmitter. The source packets are i.i.d. and we define \( H(s) \) such that

\[
H(s[t]) = H(s), \quad \forall t \geq 0.
\]

The transmitter encodes all of the source packets that it has received into two separate channel packets \( x_1[t] \) and \( x_2[t] \) using the encoding functions

\[
x_i[t] = f_{i,t}(s[0], \ldots, s[t]), \quad i \in \{1, 2\}, \quad t \geq 0.
\]

The channel packet \( x_1[t] \) is sent over Link 1 while \( x_2[t] \) is sent over Link 2. Both links are burst-erasure channels, which may erase one or both of \( x_1[t] \) and \( x_2[t] \). The output of the channel is \( y_1[t] \) on Link 1 and \( y_2[t] \) on Link 2. The channel introduces a burst-eraser of length \( B \) onto each link. However, the position of the burst in Link 1 and the position of the burst in Link 2 are dependent. They are separated by at least \( d \) time units, where \( d \) is a positive integer. The channel can be described as

\[
y_1[t] = \begin{cases} x_1[t] & \text{otherwise} \\ * & t \in [j, j + B - 1] \end{cases}
\]

(4.2)

\[
y_2[t] = \begin{cases} x_2[t] & \text{otherwise} \\ * & t \in [j + (-1)^k \delta, j + (-1)^k \delta + B - 1] \end{cases}
\]

(4.3)

for some integer \( j \geq 0 \), \( k \in \{0, 1\} \) and \( \delta \geq d \).
The channel packets of either link have the same entropy. We define \( H(x) \), such that
\[
H(x_1[t]) = H(x_2[t]) = H(x).
\]
We also define the size of each channel packet in bits to be \( |x| \) and
\[
|x| \geq H(x).
\]
Finally, the receiver attempts to decode the original source packets from the received channel packets. The variable \( T \) is the allowable delay, which means that the receiver must decode source packet \( s[t] \) by time \( t + T \). The decoder implements the decoding function \( g_t(\cdot) \), such that for \( t \geq 0 \)
\[
\begin{align*}
\hat{s}[t] &= g_t(y_1[0], \ldots, y_1[t+T], y_2[0], \ldots, y_2[t+T]) \quad \text{Pr}(\hat{s}[t] \neq s[t]) = 0.
\end{align*}
\] (4.4)

We define a \((B, T, d)\) code in this section as a code that can decode the source packets with delay \( T \) over the erasure channel described above.

For two links, we will define rate as
\[
R \triangleq \frac{H(s)}{2|x|} \leq \frac{H(s)}{2H(x)}.
\] (4.5)

In general, for \( m \) links, the rate is defined as
\[
R \triangleq \frac{H(s)}{m|x|} \leq \frac{H(s)}{mH(x)}.
\] (4.6)

Our motivation for studying such a scenario is to gain insights on how the dynamics of the burst erasures across the two links affect the streaming capacity. An example of where this model is applicable is in an adaptive fast frequency-hopping system. The transmitter uses two parallel links and employs frequency hopping on each link. However,
if one of the frequency slots on any given link is revealed to be jammed, the system adapts so that this slot is not visited by the other link for at least $d$ time-units. We also believe that insights gained from parallel links can be used in more complex networks where the input-output relation can be described via transfer matrices. However we leave such extensions for future work.

As before, we use the notation

$$s[\begin{bmatrix} b \\ a \end{bmatrix}] = \begin{cases} s[a], s[a + 1], \ldots, s[b - 1], s[b], & a \leq b \\ \emptyset, & \text{otherwise} \end{cases}$$ (4.7)$$

$$W[\begin{bmatrix} b \\ a \end{bmatrix}] = \begin{cases} W[a], W[a + 1], \ldots, W[b - 1], W[b], & a \leq b \\ \emptyset, & \text{otherwise} \end{cases}$$ (4.8)

### 4.2 Double-Link Streaming Capacity

The capacities of the system are summarized in the tree diagram in Figure 4.2.

### 4.3 Achievable Codes for Bidirectional Separation Case

For some of the cases listed above, we have found code constructions which achieve capacity. Most of the codes are given in a block code form first. Recall from Section 2.3 that a block code can be converted to a streaming code using diagonal interleaving. We will use the same procedure here to convert a double-link block code into a double-link streaming code.

#### 4.3.1 Code for ($T \geq B$), ($d < \frac{T + B}{2}$)

In this region, the capacity is

$$C = \frac{T}{T + B}.$$ (4.9)

This is the same capacity as the single-link case where $T \geq B$. Thus, we can achieve this rate by applying a $(B, T)$ single-link code separately on each link.
Figure 4.2: Summary of the capacity results for the case of minimum bidirectional separation.
4.3.2 Code for \((T \geq B), (T + B \geq d \geq \frac{T+B}{2})\)

For this region, we want to achieve the maximum rate

\[
C = \frac{2d - B}{2d}.
\]

(4.10)

**Code Construction**

Let \(C_1\) be a single-link SCo block code that encodes \(d - B\) source symbols into codewords of length \(d\). It must also decode with a maximum delay of \(T\) from a burst erasure of length \(B\), including wraparound bursts. Let the source symbols be \(u_i = \{u_i[0], \ldots, u_i[d - B - 1]\}\).

\[
(4.11)
\]

If \(d - B < B\), we choose the codeword of \(C_1\) to be

\[
\begin{pmatrix}
u_i \\ u_iH \\ v_i
\end{pmatrix}
\]

where \(H\) is a full rank \((d - B) \times (2B - d)\) matrix. \(H\) has the property that \((I_{d-B} H)\) is a generator matrix for a linear code which can decode from \(2B - d\) consecutive erasures including wraparound bursts, without a delay constraint. This version of \(C_1\) has a maximum delay of \(B \leq T\).

If \(d - B \geq B\), we let the code \(C_1\) be a \((B, d - B)\) single-link block code.

![Code construction for the case when \(T \geq B\) and \(d \geq (T + B)/2\).](image)

Let \(p_i = \{p_i[0], \ldots, p_i[B - 1]\}\) be the \(B\) parity check symbols in the codeword of \(C_1\) which correspond to the source symbols \(u_i\). Let \(u_1\) and \(u_2\) be defined as in (4.11), and \(p_1\) and \(p_2\) are the corresponding parity check symbols produced by \(C_1\). Let \(v\) be a sequence of \(B\) source symbols, i.e. \(v = \{v[0], \ldots, v[B - 1]\}\). The double-link \((B, T, d)\) block code is given by Figure 4.3. Then the block code is diagonally interleaved to form a streaming code.
Decoding

We only see a total of $B$ erasures at any time. First, we assume that the second link burst starts after the first link burst, and we let $k$ be the offset of the second link burst from the first symbol. By symmetry, the same decoding methods will work when the burst on the second link appears first.

**Case 1:** $k \leq d - B$

![Diagram](Image)

Figure 4.4: Burst erasure location when $k < d - B$ and the Link 1 burst erasure starts first

**Burst positions:** The burst erasure only appears in the bottom link, and the top link has no erasures.

**Decoding:** We can find $v$ from the top link by cancelling out $p_1$ from the $p_1 + v$ symbols. The decoder for $C_1$ can then recover the missing $u_2$ symbols using the $p_2$ in the bottom link.

**Delay:** The code $C_1$ has delay less than $T$, so the missing $u_2$ symbols are recovered before the delay deadline.

**Case 2:** $d - B < k \leq d$

**Burst positions:** The burst erasure appears in the last $d - k$ symbols of the bottom link, and the first $B - (d - k)$ symbols of the top link.

**Decoding:** On the bottom link, we have all of the symbols in $u_2$ and can use it to cancel out $p_2$ and recover the first $B - (d - k)$ symbols in $v$. The $v$ symbols that we cannot recover in this way virtually erase some of the symbols in the top link.
In the top link, we have a virtual wraparound burst of length $B$, which $C_1$ can handle. We use the $C_1$ decoder to recover the missing symbols in $u_1$. Then, we can recover the remaining symbols in $v$ from the virtually erased symbols.

**Delay:** The code $C_1$ has delay less than $T$, so the missing $u_1$ symbols are recovered before the delay deadline.

**Note:** If $d - B < B$, it is possible for the erasure burst to completely erase the $u_1$ region. However, the same decoding technique still applies.

**Example**

For example, we can construct a $(2, 4, 5)$ code. For $C_1$, we choose the $(2, 3) \text{ SCo code}$:

$$
\left( u_0 \ u_1 \ u_2 \ u_0 + u_2 \ u_1 + u_2 \right).
$$

The double-link block code is then

$$
\begin{pmatrix}
 u_0 & u_1 & u_2 & v_0 + u_0 + u_2 & v_1 + u_1 + u_2 \\
 w_0 & w_1 & w_2 & v_0 + w_0 + w_2 & v_1 + w_1 + w_2
\end{pmatrix}
$$

After applying diagonal interleaving to convert from a block code to a streaming code, we end up with the code in Figure 4.6.

**4.3.3 Code for $(T \geq B)$, $(d > T + B)$**

For the case when $d > T + B$, we use the $(B, T, B + T)$ code instead.
4.3.4 Code for \((T < B), (B > d \geq B - T)\)

To achieve capacity, we apply a \((B - d, T)\) single-link code to Link 1, then we let \(x_2[t] = x_1[t]\). The bursts in both links only overlap by \(B - d\) time slots, so if we consider both links together, we have only effectively lost \(B - d\) channel packets, and the single-link code can recover from this.

4.3.5 Code for \((T < B), (d \geq B)\)

The capacity in this region is equal to \(\frac{1}{2}\), and at this rate \(H(s) = H(x)\). Since the erasure bursts in each link are non-overlapping, we can repeat the same source symbol in each link, i.e.

\[x_1[t] = x_2[t] = s[t]\]

4.4 Converse Proof

4.4.1 \((T \geq B), (d \leq \frac{T+B}{2})\)

For this region, the periodic erasure channel sequence in Figure 4.7 is used. By showing that all of the source packets can be recovered using the \((B, T, d)\) code, the rate can be
bounded by
\[ R \leq \frac{T}{T + B} \] (4.12)
which is the same as the single-link streaming code capacity. Instead of proving using
the entropy manipulation as before, we will show how the proof works graphically.

Looking back at Figure 4.7, at time \( B + T - 1 \), the source packets \( s \begin{bmatrix} B - 1 \\ 0 \end{bmatrix} \) have been
recovered using the \((B, T, d)\) code. Theoretically we can use them to reconstruct the
missing channel packets \( x_1 \begin{bmatrix} B - 1 \\ 0 \end{bmatrix} \).

When we advance to time \( B + T + d - 1 \), the channel looks like Figure 4.8. We have a
burst erasure on the second link and another one on the first link, and the beginning of
either burst are both separated by \( T + B - d \) time units. The \((B, T, d)\) code can recover
from a burst erasure on either link if they are at least \( d \) time units apart, so we can
recover from the erasures so far if
\[ T + B - d \geq d \]
\[ \frac{T + B}{2} \geq d. \] (4.13)
If $d \leq \frac{T + B}{2}$, then at time $B + T + d - 1$, we can recover all of the source packets up to time $B + d - 1$. Then these source packets can be used to reconstruct the channel packets in the second link erasure $x_2 \left[ \frac{d + B - 1}{d} \right]$. At this point, we have recovered from the first set of burst erasures. Note that Figure 4.8 was drawn assuming $d < B$, but even if this is not true, the steps are the same.

At time $B + 2T - 1$, all of the source packets in the first period $s \left[ \frac{B + T - 1}{0} \right]$ are recovered using the $(B, T, d)$ code. We can repeat the same steps for the subsequent periods to show that when we have $(n + 1)$ such periods, then we can recover $n$ periods of source symbols. When conducting the information theoretic proof, we will eventually end up with

$$n \cdot (T + B) \cdot H(s) \leq 2(n + 1) \cdot T \cdot H(x)$$

$$\frac{H(s)}{2H(x)} \leq \left( \frac{n + 1}{n} \right) \cdot \frac{T}{T + B}$$

$$R \leq \frac{T}{T + B} \quad \text{(as } n \to \infty)$$

(4.14)

4.4.2 $(T \geq B), \ (T + B \geq d \geq \frac{T + B}{2})$

In this region, we use the periodic erasure channel sequence in Figure 4.9 and the same argument to show that the rate is upper bounded by

$$R = \frac{H(s)}{2H(x)} \leq \frac{2d - B}{2d}.$$  

(4.15)

Consider that at time $2d - 1$, the $(B, T, d)$ code can be used to recover the source packets up to time $2d - T - 1$. At this time, we want to recover at least the source packets $s \left[ \frac{B - 1}{0} \right]$ so then we can reconstruct the channel packets of the first burst erasure.
in the first link. If we have

$$2d - T - 1 \geq B - 1$$

$$d \geq \frac{T + B}{2}$$  \hspace{1cm} (4.16)

then we can recover $s \begin{bmatrix} B-1 \\ 0 \end{bmatrix}$ and therefore $x_1 \begin{bmatrix} B-1 \\ 0 \end{bmatrix}$. Then at time $3d - 1$, we can recover all of the source packets up to time $d + B - 1$ and remove the burst erasure from the second link $x_2 \begin{bmatrix} d+B-1 \\ d \end{bmatrix}$. Continuing this method, we can recover $nd$ source packets using $(n + 1)(2d - B)$ symbols of channel packets, so the rate is upper bounded by $R \leq \frac{2d - B}{2d}$.

4.4.3 \hspace{0.5cm} (T \geq B), (d > T + B)

The periodic erasure channel sequence in Figure 4.10, gives an upper bound of

$$R = \frac{H(s)}{2H(x)} \leq \frac{2T + B}{2T + 2B}. \hspace{1cm} (4.17)$$

![Figure 4.10: Periodic erasure channel sequence for case 3.](image)

4.4.4 \hspace{0.5cm} (T < B), (d < B - T)

In this situation, the burst erasures in both links overlap by $B-d$ time units. If $T < B-d$, then we can recover $s[0]$ by time $T$, even if $x_1 \begin{bmatrix} B-d \\ 0 \end{bmatrix}, x_2 \begin{bmatrix} B-d \\ 0 \end{bmatrix}$ are all erased. This implies that the information content in one source packet is contained in zero channel packets, so

$$H(s) \leq 0 \cdot H(x) = 0$$

$$R = 0 \hspace{1cm} (4.18)$$
4.4.5 \((T < B), (B > d \geq B - T)\)

By showing that all of the source packets from the channel erasure sequence in Figure 4.11 are decodable, the rate can be bounded by

\[
R = \frac{H(s)}{2H(x)} \leq \frac{T}{2(T + B - d)} \quad (4.19)
\]

![Diagram](image)

This converse proof is more tricky than the others. Consider that each source packet at time \(t\) must be recovered by time \(t + T\). If we assume that all channel packets before time \(t\) are received perfectly, then we only need to consider the erasures from time \(t\) to \(t + T\). This covers \(T + 1\) time slots inclusive, so we can focus on a sliding window of \(T + 1\) time slots.

We will show that each source packet \(s[t]\) can be recovered by time \(t + T\) for the first half period where \(0 \leq t < T + B - d\). After each \(s[t]\) is received, the channel packet \(x[t]\) can be reconstructed. The value of \(t\) can be further split into three cases.

\(0 \leq t < B - d\): Our sliding window of \(T + 1\) time slots looks like Figure 4.12(a), where the diagonal shaded squares are channel packets that have been reconstructed. If the erasures in the window are part of a \((B, T, d)\) burst erasure pattern, then \(s[t]\) can be recovered at the end of the window. Figure 4.12(b) shows how a burst erasure of length \(B\) can be laid over each link. In order to decode \(s[t]\), we require that the distance between the start of the first and second burst must be at least \(d\). Let us call this distance \(\Delta\), which is equal to

\[
\Delta = B - (T + 1) + (T - t) \\
= B - t - 1
\]
The minimum possible value of $\Delta$ is

$$\min_{0 \leq t < B-d} \Delta = B - (B - d - 1) - 1 = d.$$ 

Since $\min(\Delta) = d$, then $\Delta \geq d$ so $s[t]$ can indeed be recovered by time $t + T$.

Figure 4.12: Sliding window for $0 \leq t < B - d$: (a) the sliding window in the channel sequence (b) with a burst erasure overlay. The diagonal shaded squares are reconstructed packets.

$B - d \leq t < T$: The sliding window is now in the position given by Figure 4.13(a), with the burst erasure overlay in Figure 4.13(b). If we let $\Delta$ be the distance between the start of both links again, we can see that it is constant and equal to

$$\Delta = B - (B - d) = d.$$ 

This means that $s[t]$ can be recovered at time $t + T$.

Figure 4.13: Sliding window for $B - d \leq t < T$: (a) the sliding window in the channel sequence (b) with a burst erasure overlay.
$T \leq t < T + B - d$: The sliding window is now in the position given by Figure 4.14(a). We can place a burst erasure overlay as in Figure 4.14(b). Once again, we let $\Delta$ be the separation between the bursts, which is equal to

$$\Delta = B - (T + B - d - t)$$

$$= d + t - T.$$

We require $\Delta \geq d$ in order for the $(B, T, d)$ decoder to be usable. Finding the minimum $\Delta$ gives

$$\min_{T \leq t < T + B - d} \Delta = d + T - T$$

$$= d.$$

This means that $s[t]$ can be recovered at time $t + T$.

![Figure 4.14](image)

Figure 4.14: Sliding window for $T \leq t < T + B - d$: (a) the sliding window in the channel sequence (b) with a burst erasure overlay.

Combining these three cases, we can recover $s\left[\frac{T+B-d-1}{0}\right]$, and end up with Figure 4.15. We can continue the same argument to recover each half period, and then adapt either Martinian’s converse proof or the information theoretic converse proof to end up with the capacity given by (4.19).

4.4.6 $(T < B), (d \geq B)$

Any $(B, T, d)$ double-link code can also decode with delay $T$ when only a single burst erasure of length $B$ is in either link. Consider a channel erasure sequence where all of the packets in Link 1 are erased, but none of the packets in Link 2 are erased. By viewing a window of $T + 1 \leq B$ time slots at a time, we can show that $s[0]$ and $x_1[0]$ are recovered
Figure 4.15: Periodic erasure channel sequence for case 5, where source and channel packets in the first half period has been recovered.

by time $T$, $s[1]$ and $x_1[1]$ are recovered by time $T + 1$, and so on. We can decode all the subsequent source packets, so in the limit, the rate is bounded by

$$R = \frac{H(s)}{2H(x)} \leq \frac{1}{2} \quad (4.20)$$

### 4.5 System Model for Multiple Links

We can extend the results of the previous section to multiple links. Consider the multi-link model in Figure 4.16, we have $m$ links from the encoder to the decoder and $m \geq 2$. Each link is a burst erasure channel and is capable of erasing $B$ consecutive channel packets. We assume that there are only two burst erasures, each one on a separate link which are separated by at least $d$ time slots. We are considering a minimum bidirectional separation case for the two erasure bursts.

Our model here is similar to the models before, but with multiple links. Time is discretized into time slots, and at the beginning of time slot $t \geq 0$, a source packet $s[t]$ arrives at the sender. The source packets are i.i.d. and

$$H(s[t]) = H(s), \forall t \geq 0.$$  

The encoder encodes all of the source packets received up to time $t$ into $m$ channel packets using the encoding functions

$$x_i[t] = f_{i,t}(s[0], \ldots, s[t]), \quad i \in \{1, 2, \ldots, m\}, \ t \geq 0. \quad (4.21)$$

The channel packets of each link have the same entropy. We define $H(x)$, such that

$$H(x_i[t]) = H(x), \quad i \in \{1, 2, \ldots, m\}$$
Each channel packet $x_i[t]$ is send over Link $i$. We are concerned with a minimum bidirectional separation case and can describe the channels as following. For some $a, b \in \{1, 2, \ldots, m\}, j \in \mathbb{Z}, k \in \{0, 1\}$, where $a \neq b$, we have

$$
\forall i \in \{1, 2, \ldots, m\}, i \neq a, i \neq b
\begin{align*}
y_i[t] &= x_i[t] \\
y_a[t] &= \begin{cases} 
* & t \in [j, j + B - 1] \\
x_a[t] & \text{otherwise}
\end{cases} \\
y_b[t] &= \begin{cases} 
* & t \in [j + (-1)^k\delta, j + (-1)^k\delta + B - 1] \\
x_b[t] & \text{otherwise}
\end{cases}
\end{align*}
$$

Finally, the receiver has to decode the original source packet from the received channel packets. The variable $T$ is the allowable decoding delay, which means that the receiver must decode source packet $s[t]$ by time $t + T$. The decoder implements the decoding function $g_t(.)$, such that for $t \geq 0$

$$
\begin{align*}
\hat{s}[t] &= g_t(y_1\left[\frac{t+T}{0}\right], y_2\left[\frac{t+T}{0}\right], \ldots, y_m\left[\frac{t+T}{0}\right]) \\
Pr(\hat{s}[t] \neq s[t]) &= 0
\end{align*}
$$

(4.22)
where

\[ y_i[a]^b = \begin{cases} 
  y_i[a], y_i[a+1], \ldots, y_i[b-1], y_i[b], & a \leq b \\
  \emptyset, & \text{otherwise}
\end{cases} \quad (4.23) \]

We shall refer to a \((m, B, T, d)\) code as a channel code which can be used to recover source packets with maximum decoding delay \(T\) from the system described in this section. The rate of a multi-link code with equivalent size channel packets on each link is defined as

\[ R = \frac{H(s)}{mH(x)}. \quad (4.24) \]

### 4.6 Multi-Link Streaming Capacity

The capacity of this system can be summarized in the tree diagram in Figure 4.17.

---

**Figure 4.17:** Summary of the capacity results for the case of minimum bidirectional separation in a multi-link system.
4.7 Achievable Codes

4.7.1 Code for \((T \geq B), \ (d \leq \frac{T+B}{2})\)

Code Construction

The capacity of this region is

\[
C = \frac{T + (\frac{m-2}{m})B}{T + B}.
\] (4.25)

The block code is found first. We will use \(mT + (m-2)B\) source symbols and \(mT + mB\) symbols in total. First, let \(u_i\) be a row vector of length \(T\) and \(v_i\) be a row vector of length \(B\).

\[
u_i = (u_i[0], u_i[1], \ldots, u_i[T-1])
\]

\[
v_i = (v_i[0], v_i[1], \ldots, v_i[B-1])
\]

We find a single-link \((B,T)\) block code \(C_1\). We let \(p_i\) be the parity check symbols when we encode \(u_i\) with \(C_1\), and

\[
p_i = (p_i[0], p_i[1], \ldots, p_i[B-1]).
\]

We will use \(u_1, \ldots, u_m\) and \(v_1, \ldots, v_{m-2}\) as our source symbols. This gives a total of \(mT + (m-2)B\) source symbols. We let \(V\) be a matrix whose rows are the vectors \(v_i\)

\[
V = \begin{pmatrix}
v_1 \\
v_2 \\
\vdots \\
v_{m-2}
\end{pmatrix}
\]

The final ingredient for the code are the row vectors \(w_1, \ldots, w_m\) of length \(m-2\). They are chosen such that any \(m-2\) subset of them can form a full rank matrix. In fact, we can find a \((m, m-2)\) MDS block code, and let each \(w_i^T\) be equal to a column vector of the generator matrix. The block code for each link is constructed as:

\[
\text{Link } i : \left( u_i, \ p_i + w_i V \right)
\]

The code in block code form is given in Figure 4.18.
Chapter 4. Multi-Link Streaming Erasure Codes

Figure 4.18: Code for multi-link when \( T \geq B, \ d \leq \frac{T+B}{2} \)

Decoding

The decoding process is quite simple. For each of the \( m - 2 \) links with no erasures, we can cancel out \( p_i \) from the parity check symbols \( p_i + w_i V \). Then at each time \( t \), from each of the links with no erasures, we have \( m - 2 \) linear combinations of the source symbols \( v_i[t-T] \). We can then decode each \( v_i[t-T] \) and use it to isolate only the \( p_i \) component of the links which do have erasures. Then the \( C_1 \) decoder can recover the missing \( u_i \) source symbols.

4.7.2 Code for \( (T \geq B), \ (T + B \geq d \geq \frac{T+B}{2}) \)

Code Construction

When constructing the code, we will need to make use of a code \( C_1 \), where \( C_1 \) is a single-link block code that encodes \( d - B \) source symbols into \( d \) symbols. \( C_1 \) must also decode with a maximum delay of \( T \) from a burst erasure of length \( B \), which includes wraparound bursts. We use the same \( C_1 \) code in Section 4.3.2. We let \( u_i \) be the source symbols and \( p_i \) be the parity symbols of \( C_1 \) such that

\[
\begin{align*}
    u_i &= (u_i[0], \ldots, u_i[d - B - 1]) \\
    p_i &= (p_i[0], \ldots, p_i[B - 1]) \\
    C_1 &= (u_i, \ p_i) 
\end{align*}
\]

(4.26)

If \( d - B < B \), then we let

\[
p_i = (u_i H, u_i)
\]

(4.27)
where $H$ is a full rank $(d - B) \times (2B - d)$ matrix. If $d - B \geq B$, then we let $C_1$ be a $(B, d - B)$ single-link block code. Next, we let $v_i$ be additional source symbols such that

$$v_i = (v_i[0], ..., v_i[B - 1]).$$

(4.28)

We will use $u_1, \ldots, u_m$ and $v_1, \ldots, v_{m-1}$ as source symbols. This is a total of $md - B$ source symbols. We let $V$ be a matrix with $v_i$ as the rows.

$$V = \begin{pmatrix}
   v_1 \\
   v_2 \\
   \vdots \\
   v_{m-1}
\end{pmatrix}$$

(4.29)

Finally, we have the row vectors $w_1, \ldots, w_m$, each has length $m - 1$. They have the property that any $m - 1$ subset of them can form a full rank matrix. The block code for each link is constructed as

$$\text{Link } i : \left( u_i, \ p_i + w_iV \right)$$

(4.30)

and Figure 4.19 shows the code on all the links.

\begin{center}
\begin{tabular}{|c|c|}
\hline
\multicolumn{2}{|c|}{\text{d - B}} \quad \text{B} \\
\hline
\text{Link 1:} & \text{u}_1 & \text{p}_1 + \text{w}_1V \\
\hline
\text{Link 2:} & \text{u}_2 & \text{p}_2 + \text{w}_2V \\
\hline
\text{Link 3:} & \text{u}_3 & \text{p}_3 + \text{w}_3V \\
\hline
\vdots & \vdots \\
\hline
\text{Link m:} & \text{u}_m & \text{p}_m + \text{w}_mV \\
\hline
\end{tabular}
\end{center}

Figure 4.19: Code for multi-link when $T \geq B$, $T + B > d \geq \frac{T + B}{2}$

\section*{Decoding}

We can use a similar decoding method as the decoding method in Section 4.3.2 on the two links that have burst erasures. The difference is that we will also have to use the $w_iV$ symbols in the other links to help deal with the symbols in $V$.

\subsection*{4.7.3 Code for $(T \geq B)$, $(d > T + B)$}

For the case when $d > T + B$, we use the $(B, T, B + T)$ code instead.
4.7.4 Code for \((T < B), (d < B - T)\)

The capacity here is equal to

\[
C = \frac{m - 2}{m}. \tag{4.31}
\]

The easiest way to encode is to apply a \((m, m - 2)\) MDS code vertically across the links in each time slot. Each source packet \(s[t]\) is split into \(m - 2\) subpackets

\[
s[t] = \{s_1[t], \ldots, s_{m-2}[t]\}.
\]

Then let

\[
\left( \begin{array}{cccc} w_1^T & w_2^T & \cdots & w_m^T \end{array} \right)
\]

be equal to the generator matrix for a \((m, m - 2)\) MDS block code, where each \(w_i\) is a row vector with \(m - 2\) elements. Then we assign

\[
x_i[t] = w_i^T(s_1[t], \ldots, s_{m-2}[t]). \tag{4.32}
\]

To decode, remember that there are at most two erasures at any one time. We can decode by using the \((m, m - 2)\) MDS decoder where the codeword is the vector \((x_1[t], \ldots, x_m[t])\).

4.7.5 Code for \((T < B), (B > d \geq B - T)\)

Code Construction

Let \(C_1\) be a \((B - d, T)\) single-link block code. Also let \(u_i\) be the source symbols and \(p_i\) be the parity symbols of \(C_1\) such that

\[
u_i = (u_i[0], \ldots, u_i[T - 1])
\]

\[
p_i = (p_i[0], \ldots, p_i[B - d - 1])
\]

\[
C_1 = \left( u_i, \quad p_i \right) \tag{4.33}
\]

Next, we let \(v_i\) be additional source symbols such that

\[
v_i = (v_i[0], \ldots, v_i[B - d - 1]). \tag{4.34}
\]
We will use $u_1, \ldots, u_{m-1}$ and $v_1, \ldots, v_{m-2}$ as our source symbols. The total number of source symbols is $(m - 1)T + (m - 2)(d - B)$. We let $V$ be a matrix with $v_i$ as the rows.

$$V = \begin{pmatrix}
v_1 \\
v_2 \\
\vdots \\
v_{m-2}
\end{pmatrix}$$  \hspace{1cm} (4.35)

Finally, we have the row vectors $w_1, \ldots, w_m$, each has length $m - 2$. They have the property that any $m - 2$ subset of them can form a full rank matrix. The block code for each of links 1 to $m - 1$ is constructed as

$$\text{Link } i : \left( u_i, \ p_i + w_iV \right)$$  \hspace{1cm} (4.36)

while the last link is

$$\text{Link } m : \left( \sum_{i=1}^{m-1} u_i, \ \sum_{i=1}^{m-1} p_i + w_mV \right)$$  \hspace{1cm} (4.37)

and Figure 4.20 shows the code on all the links.

![Figure 4.20: Code for multi-link when $T \geq B$, $T + B > d \geq T + \frac{B}{2}$](image)

**Decoding**

From the $m - 2$ links that have no erasures, we can isolate the corresponding $w_iV$ symbols, and use it to decode all of the source symbols $v_1$ to $v_{m-2}$. Notice, that the two bursts overlap by $B - d$ time slots. We can recover the original symbols where the bursts don’t overlap, because the last link is a parity check summation of the other links. We can then use $C_1$ to recover from the symbols in the burst overlap.
4.7.6 Code for \((T < B), \ (d \geq B)\)

**Code Construction**

The capacity here is equal to

\[
C = \frac{m - 1}{m}.
\]  

(4.38)

This time, the code will be given as a streaming code, not a block code. We split each source packets \(s[t]\) into \(m - 1\) subpackets

\[
s[t] = \{s_1[t], \ldots, s_{m-1}[t]\}.
\]  

(4.39)

Then we simply assign

\[
x_i[t] = s_i[t], \ 0 < i < m
\]

\[
x_m[t] = \sum_{i=1}^{m-1} s_i[t].
\]  

(4.40)

**Decoding**

Since the bursts do not overlap at all, we can treat the last link as parity checks, and use it to recover any erased packets.

4.8 Converse Proof

The converse proofs for capacity for this system is extremely similar to the proofs in the previous chapter. In fact, they are just the same periodic erasure channels sequence but with an additional \(m - 2\) links which have no erasures.

4.8.1 \((T \geq B), \ (d \leq \frac{T + B}{2})\)

A \((m,B,T,d)\) code in this region can recover all the source packets from the channel in Figure 4.21 using a similar technique as Section 4.4.1. This means that the rate is upper bounded by the ratio of non-erased packets to erased packets, which is

\[
R \leq \frac{2T + (m - 2)(B + T)}{m(B + T)}
\]

\[= \frac{T + \left(\frac{m-2}{m}\right)B}{B + T}.
\]  

(4.41)
4.8.2 \((T \geq B), (T + B \geq d \geq \frac{T + B}{2})\)

Using the periodic erasure channel sequence in Figure 4.22 with the same method as Section 4.4.2, the rate is upper bounded by

\[
R \leq \frac{d - \left(\frac{1}{m}\right)B}{d}. \tag{4.42}
\]

Figure 4.21: Periodic erasure channel sequence for finding the capacity when \(T \geq B\), \(d \leq \frac{T + B}{2}\).
4.8.3 \((T \geq B), (d > T + B)\)

Similar to Section 4.4.3, the periodic erasure channel sequence in Figure 4.23, gives an upper bound of

\[
R \leq \frac{T + \left(\frac{m-1}{m}\right)B}{T + B}.
\]

\[ (4.43) \]

![Figure 4.23: Periodic erasure channel sequence for finding the capacity when \(T \geq B, d \geq T + B\).](image)

4.8.4 \((T < B), (d < B - T)\)

In this situation, the capacity is equal to

\[
C = \frac{m - 2}{m}.
\]

\[ (4.44) \]

Imagine that we have a channel sequence where Link 1 and Link 2 have erasures for all time, but the other links transmit perfectly. The burst erasures in the two links with erasures overlap by \(B - d\) time units. If \(T < B - d\), then we can recover \(s[0]\) by time \(T\) and then reconstruct \(x_1[0]\) and \(x_2[0]\). We can continue this logic and show that all of the source packets can be recovered.

4.8.5 \((T < B), (B > d \geq B - T)\)

We can show that all of the source packets from the channel erasure sequence in Figure 4.24 are decodable by using the same technique as Section 4.4.5. Thus,

\[
R \leq \frac{(\frac{m-1}{m})T + (\frac{m-2}{m})(B - d)}{T + B - d}.
\]

\[ (4.45) \]
Chapter 4. Multi-Link Streaming Erasure Codes

4.8.6 \((T < B), (d \geq B)\)

Any \((B, T, d)\) double-link code can also decode with delay \(T\) when only a single burst erasure of length \(B\) is in either link. Consider a channel erasure sequence where all of the packets in Link 1 are erased, but none of the packets in Link 2 to Link \(m\) are erased. By viewing a window of \(T + 1 \leq B\) time slots at a time, we can show that \(s[0]\) and \(x_1[0]\) are recovered by time \(T\), \(s[1]\) and \(x_1[1]\) are recovered by time \(T + 1\), and so on. We can decode all the subsequent source packets, so in the limit, the rate is bounded by

\[ R \leq \frac{m - 1}{m} \] (4.46)

4.9 Summary

In this chapter we introduced a multi-link system where each source packet must be received within a certain delay deadline. We start of with two links, and while each link is a bursty packet erasure channel, we maintain a minimum separation of \(d\) between the start of the bursts in either link. When finding the capacity of the system, we could have treated each link independently, which would give us an achievable rate equal to the single-link capacity. However, by treating both links together and considering the minimum separation \(d\), we are able to show that capacity non-decreasing in \(d\).

We then use the same methods on a multi-link system where burst erasures can appear on any two links, but again they have a minimum separation of \(d\). We ended up with similar results as the two-link case.
Chapter 5

Multi-Source Streaming Codes

5.1 System Model

Consider that we have two source streams $s_a[t]$ and $s_b[t]$. As before, each source packet needs to be recovered at the receiver by a certain deadline, but the difference is that each source packet has a different deadline. This means that $s_a[t]$ must be recovered by time $t + T_a$ while $s_b[t]$ is recoverable by time $t + T_b$, where $T_a$ and $T_b$ are the allowable delays for source streams $a$ and $b$.

As before, we divide time into time slots and at the beginning of each time slot, two source packets arrive at the encoder. The source packets are $s_a[t]$ and $s_b[t]$, and each source packet is independent of all other source packets. In one source stream, the source packets have the same entropy for all time, but $s_a[t]$ and $s_b[t]$ may have different entropies.

\[
H(s_a[t]) = H(s_a), \quad \forall t \geq 0 \\
H(s_b[t]) = H(s_b), \quad \forall t \geq 0
\]
The encoder takes all of the source packets that it has received into a channel packet $x[t]$ using the function
\[
x[t] = f_t\left(s_a\left[\frac{t}{0}\right], s_b\left[\frac{t}{0}\right]\right), \ t \geq 0.
\]
\[
H(x[t]) = H(x).
\]

Each channel packet is then sent over a burst erasure channel. The burst erasure channel erases $B$ consecutive packets, and the output $y[t]$ is defined as
\[
y[t] = \begin{cases} 
\star, & t \in [j, j + B - 1] \\
x[t], & \text{otherwise}
\end{cases}
\]
for some integer $j$. Finally, we decode the source packets $s_a[t]$ and $s_b[t]$. Source stream $a$ has the decoding delay $T_a$ and source stream $b$ has the decoding delay $T_b$.

\[
\hat{s}_a[t] = g_{a,t}\left(y\left[t + T_a\right]_0\right) \\
Pr(\hat{s}_a[t] \neq s_a[t]) = 0.
\] (5.1)

\[
\hat{s}_b[t] = g_{b,t}\left(y\left[t + T_b\right]_0\right) \\
Pr(\hat{s}_b[t] \neq s_b[t]) = 0.
\] (5.2)

A $(B, T_a), (B, T_b)$ code in this chapter will be defined as a code that satisfies the described equations. The rate of the code will be characterized by an ordered pair which has the rate for source $a$ and the rate for source $b$.

\[
(R_a, R_b) = \left(\frac{H(s_a)}{H(x)}, \frac{H(s_b)}{H(x)}\right)
\] (5.3)

For this chapter, we will give a region by region analysis which includes both the converse proof and the achievable codes. We will only consider the case when $T_a = T$ and $T_b = B$, so the parameters of the code will be $(B, T), (B, B)$.

### 5.2 $(B, T) (B, B)$ code, $T \geq 3B$

In this case, we have $T_a = T$ and $T_b = B$, where $T \geq 3B$. The capacity in this region is given by the graph in Figure 5.2, and by the formula
\[
0 \leq C_a \leq \frac{T}{T + B}
\]
\[
C_b = \begin{cases} 
\frac{1-C_a}{2}, & C_a < \frac{T-B}{T+B} \\
\frac{T}{T+B} - C_a, & C_a \geq \frac{T-B}{T+B}.
\end{cases}
\] (5.4)
Note that there is a trade off between the capacity for source $a$ and the capacity for source $b$. The converse proof that is given in the next section applies to all $T > 2B$, but we only have a code construction for $T \geq 3B$. Therefore, the capacity in $2B < T < 3B$ is still open.

![Figure 5.2: Capacity for a $(B,T), (B,B)$ code where $T \geq 3B$.](image)

### 5.2.1 Converse Proof

To find capacity, we have to derive a relationship on the maximum possible value of both $R_a$ and $R_b$. As with our previous chapters, we can construct periodic burst erasure channel sequences to help us. We will construct two different periodic erasure channel sequences, which will give us different upper bounds on both rates.

Our first periodic erasure channel sequence is given in Figure 5.3.

![Figure 5.3: The periodic erasure channel sequence for one of the inequality relations between $R_a$ and $R_b$.](image)

Consider that at time $T + B - 1$, we can use the $(B,T) (B,B)$ decoder to recover the source packets $s_a \left[ \begin{array}{c} B-1 \\ 0 \end{array} \right]$ and $s_b \left[ \begin{array}{c} T-1 \\ 0 \end{array} \right]$. For source stream $a$, we have a delay of $T$, so we can only recover up to the source packet from $T$ time slots ago. Similarly for source stream $b$, the delay is $B$ so we can recover up to the source packet $s_b [T-1]$ which is $B$ time slots ago. We can theoretically use the source packets up to time $B - 1$ to recover all of the missing channel packets $x \left[ \begin{array}{c} B-1 \\ 0 \end{array} \right]$. Then we can treat the first erasure
as having never happened, and can continue decoding each period similarly. If we had
\( n \) such periods of channel packets, we would be able to recover \( (n - 1) \) periods of source
packets, so we can write the relation

\[
\frac{n \cdot T \cdot H(x)}{H(x)} + \frac{H(s_b)}{H(x)} \leq \frac{n - 1}{n} \cdot \frac{T}{T + B}
\]

\[
R_a + R_b \leq \frac{T}{T + B} \quad (\text{as } n \to \infty).
\]

\[ (5.5) \]

Figure 5.4: The erasure channel sequence for the second inequality relation between \( R_a \) and \( R_b \)

The second erasure channel sequence is given in Figure 5.4, and actually contains no erasures. As before, we want to find out how many source packets can be decoded by some number of channel packets. A naive approach would be to say that for a long enough channel sequence, we can recover all of the source packets from stream \( a \) and stream \( b \). Then the rate can be bounded by \( R_a + R_b \leq 1 \). However, if we double count the source packets of stream \( b \), using the same method as Section 3.3, we can find a tighter upper bound.

We start by writing relations about the decoding ability of the code. Consider that the \((B, T) (B, B)\) code allows us to decode \( s_a[i - T] \) by time \( i \) if there is a burst erasure from time \( i - T \) to \( i - T + B - 1 \). We can write this decoding ability as

\[
H\left(s_a[i - T]|x[i-1:i-T+B]x[i-T-1]\right) = 0, \ i \geq T.
\]

\[ (5.6) \]

We can also decode \( s_b[i - B] \) by time \( i \) with the burst erasure occurring from time \( i - B \) to \( i - 1 \).

\[
H\left(s_b[i - B]|x[i]x[i-B-1]\right) = 0, \ i \geq B.
\]

\[ (5.7) \]

Finally, we can decode \( s_b[i] \) by time \( i + B \) when the burst erasure occurs from time \( i + 1 \) to \( i + B \). Incidentally, this means that we can recover \( s_b[i] \) at time \( i \) given all of the channel packets from time \( i \) and before.

\[
H\left(s_b[i]|x[i]\right) = 0, \ i \geq 0.
\]

\[ (5.8) \]
We want to prove the following relationship for \( n \geq T \)

\[
\sum_{i=0}^{n-1} H(x[i]) \geq H\left(s_a\left[\frac{n-T-1}{0}\right]\right) + H\left(b\left[\frac{n-B-1}{T-B}\right]\right) + H\left(s_b\left[\frac{n-1}{T}\right]\right) + H\left(x\left[\frac{n-1}{0}\right]s_a\left[\frac{n-T-1}{0}\right]s_b\left[\frac{n-B-1}{T-B}\right]s_b\left[\frac{n-1}{T}\right]\right). \tag{5.9}
\]

We can use mathematical induction. When we substitute \( n = T \), we have

\[
\sum_{i=0}^{T-1} H(x[i]) \geq H\left(s_a\left[\frac{-1}{0}\right]\right) + H\left(b\left[\frac{T-B-1}{T-B}\right]\right) + H\left(s_b\left[\frac{T-1}{T}\right]\right) + H\left(x\left[\frac{T-1}{0}\right]\right) s_a\left[\frac{-1}{0}\right] s_b\left[\frac{T-B-1}{T-B}\right] s_b\left[\frac{T-1}{T}\right] \tag{5.10}
\]

which is true and establishes the base case. Before conducting the induction step, we first take \( H(x[i]) \) where \( i \geq B \):

\[
H(x[i]) \geq H\left(x[i]\left[\frac{i-B-1}{0}\right]\right) = H\left(s_b[i-B]x[i]\left[\frac{i-B-1}{0}\right]\right) - H\left(s_b[i-B]\left[\frac{i-B-1}{0}\right]\right) \tag{a}
\]

\[
= H\left(s_b[i-B]x[i]\left[\frac{i-B-1}{0}\right]\right) \tag{b}
\]

where (a) uses (5.7) and (b) uses the fact that \( s_b[i-B] \) is independent of all source packets and all channel packets with time index less than \( i - B \).

Now let us assume that (5.9) is true for \( n = k \) where \( k \geq T \). When we combine this with (5.11), we have

\[
\sum_{i=0}^{k} H(x[i]) = \sum_{i=0}^{k-1} H(x[i]) + H(x[k]) \geq H\left(s_a\left[\frac{k-T-1}{0}\right]\right) + H\left(s_b\left[\frac{k-B-1}{T-B}\right]\right) + H\left(s_b\left[\frac{k-1}{T}\right]\right) + H\left(x\left[\frac{k-1}{0}\right]s_a\left[\frac{k-T-1}{0}\right]s_b\left[\frac{k-B-1}{T-B}\right]s_b\left[\frac{k-1}{T}\right]\right) + H\left(s_b[k-B]\right) + H\left(x[k]\left[\frac{k-B-1}{0}\right]\right) \geq H\left(s_a\left[\frac{k-T-1}{0}\right]\right) + H\left(s_b\left[\frac{k-B}{T-B}\right]\right) + H\left(s_b\left[\frac{k-1}{T}\right]\right) + H\left(x\left[\frac{k}{0}\right]s_a\left[\frac{k-T-1}{0}\right]s_b\left[\frac{k-B}{T-B}\right]s_b\left[\frac{k-1}{T}\right]\right)
\]
Chapter 5. Multi-Source Streaming Codes

where step (a) uses (5.6), (c) uses (5.8) and (b) and (d) uses the fact that source packets

We have proved by induction that (5.9) must be true for all

\[ (n - T) \geq \frac{n}{n - T} \geq \frac{2H(s_b)}{H(x)} + \frac{2H(s_b)}{H(x)} . \]

Finally, this gives us

\[ R_a + 2R_b \leq \frac{n}{n - T} \xrightarrow{n \to \infty} 1. \]
Now we have the two inequalities

\[ R_a + R_b \leq \frac{T}{T + B}, \quad (5.15) \]
\[ R_a + 2R_b \leq 1. \quad (5.16) \]

From the inequalities, we can find a bound on \( R_b \) as a function of \( R_a \geq 0 \)

\[ R_b \leq \min \left( \frac{T}{T + B} - R_a, \frac{1 - R_a}{2} \right). \quad (5.17) \]

Since \( R_b \) cannot be negative, we restrict \( R_a \leq \frac{T}{T + B} \). Then, the allowable \((R_a, R_b)\) values are shown in the shaded area of Figure 5.5 and the capacity is the boundary of this shaded region.

![Figure 5.5: Allowable rates for \((B, T), (B, B), T \geq 3B\) code.](image)

### 5.2.2 Achievable Codes

To show that our upper bound is achievable, we only need to find three different codes. Looking at Figure 5.2, the three rates that we need to achieve are

\[ (R_a, R_b) = \left( \frac{T}{T + B}, 0 \right) \]
\[ (R_a, R_b) = \left( 0, \frac{1}{2} \right) \]
\[ (R_a, R_b) = \left( \frac{T - B}{T + B}, \frac{B}{T + B} \right). \]
Once we have codes with these three rates, we can achieve any other capacity bound by splitting the bandwidth and using different codes in each part of the bandwidth. The first rate pair is achievable by applying a \((B, T)\) single-link streaming code for source \(a\), while the second rate pair is achievable by using a \((B, B)\) code on source \(b\). We only need to find a code for the last rate pair.

**Code Construction for** \(T \geq 3B\)

For this code, we require that \(T \geq 3B\). We will first construct the code in block form, and then use diagonal interleaving to convert it to a streaming code. From the rate formula, we will have \(T - B\) source symbols for \(a\), \(B\) source symbols for \(b\) and \(T + B\) symbols in total. We let our source symbols be \(\{a[0], \ldots, a[T - B - 1]\}\) and \(\{b[0], \ldots, b[B - 1]\}\). We then introduce the row vectors \(a_1, a_2, a_3\) and \(b_1\) such that

\[
\begin{align*}
    a_1 &= (a[0], \ldots, a[B - 1]) \\
    a_2 &= (a[B], \ldots, a[T - 2B - 1]) \\
    a_3 &= (a[T - 2B], \ldots, a[T - B - 1]) \\
    b_1 &= (b[0], \ldots, b[B - 1]).
\end{align*}
\]

(5.18)

The vectors \(a_1, a_3\) and \(b_1\) have \(B\) symbols each, while \(a_2\) has \(T - 3B\) symbols. We let \(F\) be a \((T - 3B) \times B\) full-rank matrix. The code construction is

\[
\begin{pmatrix}
    a_1 & a_2 & a_3 & b_1 & a_1 + a_2 F + a_3 + b_1
\end{pmatrix}
\]

(5.19)

We can think of \(a_1\) and \(b_1\) as the urgent symbols and \(a_2\) and \(a_3\) as the non-urgent symbols, similar to the SCo construction in Section 1.2.1. The last \(B\) symbols are the parity check symbols. The block code can then be turned into a streaming code using diagonal interleaving.

**Decoding**

The decoding is relatively straightforward. Any erasure burst of length \(B\) will erase at most \(B\) symbols in the block code, but there are enough symbols left to reconstruct the erased source symbols.

The maximum delay of each symbol in \(a_1\) is \(T\) because the corresponding parity symbols are \(T\) time slots later. The delay of each symbol in \(a_2a_3\) is less than \(T\), so the
source symbols in $a$ meet their decoding delay. The symbols in $b_1$ have a maximum delay of $B$, so they also meet their decoding delay.

To prove that the code can recover from burst erasures of length $B$, we only need to consider six essential cases.

Case 1:

![Diagram](image)

Figure 5.6: A burst erasure for decoding in case 1.

In Figure 5.6, the erasure burst erases $\Delta$ symbols in $a_1$ and $B - \Delta$ symbols in $a_2$, where $0 \leq \Delta \leq B$. When the first $B - \Delta$ parity symbols are received, we can subtract the contributions from $a_1$, $a_3$ and $b_1$ symbols from them. Notice that we can subtract the $a_1$ symbols because the first $B - \Delta$ symbols in $a_1$ are not erased. So we are left with the first $B - \Delta$ symbols of $a_2F$, which we combine with the received $a_2$ symbols to decode the missing $a_2$ symbols.

Then as each subsequent parity symbol is received, we cancel out $a_2F$ and $b_1$ and recover each missing symbol in $a_1$ with a delay of $T$.

Case 2:

![Diagram](image)

Figure 5.7: A burst erasure for decoding in case 2.

Here in Figure 5.7, $B$ symbols of $a_2$ are erased. This case can occur if $T - 3B \geq B$. To decode, we wait until the entire codeword is received, then we cancel $a_1$ and $b_1$ from the parity symbols. We combine the parity symbols with the received $a_2$ symbols to decode the missing $a_2$ symbols.
Case 3:

\[ \Delta \quad B - \Delta \]

**Figure 5.8**: A burst erasure for decoding in case 3.

In this case, shown in Figure 5.8, we have \( \Delta \) symbols of \( a_2 \) and \( B - \Delta \) symbols in \( a_3 \) erased, where \( 0 \leq \Delta \leq B \). None of the erased symbols are urgent, so we can wait until the entire codeword is received, then we have enough parity symbols to decode.

Case 4:

\[ \Delta \quad 3B - T - \Delta \]

**Figure 5.9**: A burst erasure for decoding in case 4.

The burst erasure in Figure 5.9 is possible if \( T - 3B < B \), then the erasure burst erases symbols from \( a_1 \), \( a_2 \) and \( a_3 \). When the first \( B - \Delta \) parity symbols are received, then we can recover the missing \( a_2 \) and \( a_3 \) symbols. Then when the last \( \Delta \) parity symbols are received, we can recover each missing symbol from \( a_1 \) with delay \( T \).

Case 5:

\[ \Delta \quad B - \Delta \]

**Figure 5.10**: A burst erasure for decoding in case 5.

**Figure 5.10** shows the erasure position for case 5, where \( \Delta \) symbols of \( a_3 \) and \( B - \Delta \) symbols of \( b_1 \) are erased. Then as each of the first \( B - \Delta \) parity symbols are received,
we subtract the contributions from $a_1$, $a_2$ and the non-erased $a_3$ symbols. This allows us to recover each missing $b_1$ symbol with delay $B$. Next, when the rest of the parity symbols are received, the missing $a_3$ symbols can be recovered.

**Case 6:**

![Figure 5.11: A burst erasure for decoding in case 6.](image)

Finally 5.11 gives the case when $\Delta$ symbols from $b_1$ are erased. They can be recovered from the received parity checks after subtracting the contributions from $a_1$, $a_2$ and $a_3$.

### 5.3 $(B, T)$ $(B, B)$ code, $B \leq T \leq 2B$

We have $T_a = T$ and $T_b = B$ where $B \leq T \leq 2B$. The capacity in this region can be written as

$$0 \leq C_a \leq \frac{T}{T + B}$$

$$C_b = \begin{cases} \frac{1}{2} - \frac{3B - T}{2B} & C_a, \quad C_a < \frac{B}{T + B} \\ \frac{T}{T + B} - C_a, & C_a \geq \frac{B}{T + B}. \end{cases}$$

(5.20)

The graph in Figure 5.12 shows the trade off between the capacity for source $a$ and the capacity for source $b$.

#### 5.3.1 Converse Proof

We construct two periodic erasure sequences and each one gives a different inequality involving $R_a$ and $R_b$. Our first periodic erasure channel sequence in Figure 5.13 is the same as the previous case. The first inequality is also the same:

$$R_a + R_b \leq \frac{T}{T + B}. \quad (5.21)$$
Figure 5.12: Capacity for a \((B,T), (B,B)\) code where \(B \leq T \leq 2B\).

Figure 5.13: The periodic erasure channel sequence for one of the inequality relations between \(R_a\) and \(R_b\)

In the previous section, for \(T \geq 3B\), the second inequality was (5.16). While that relation still applies here, we can find an even tighter bound. The second erasure sequence is given in Figure 5.14. It consists of a pattern of \(B\) erasures followed by \(B\) non-erasures. We want to show that we can recover all of the source packets of stream \(b\), but only a subset of the source packets of stream \(a\), by revealing channel packets in the proof.

Figure 5.14: The periodic erasure channel sequence for one of the inequality relations between \(R_a\) and \(R_b\)

Consider that at time \(2B - 1\), we have received the channel packets \(x\left[\begin{array}{c} 2B-1 \\ B \end{array}\right]\). Source stream \(a\) has the \((B,T)\) property, and since \(T \leq 2B\), we can recover the first \(2B - T\) packets of source \(a\). Source \(b\) has the \((B,B)\) property, so we can also recover the first \(B\) packets of \(b\). So far, we have recovered source symbols \(s_a\left[\begin{array}{c} 2B-T-1 \\ 0 \end{array}\right]\) and \(s_b\left[\begin{array}{c} B-1 \\ 0 \end{array}\right]\). We can then reconstruct the channel packets \(x\left[\begin{array}{c} 2B-T-1 \\ 0 \end{array}\right]\) and reveal the channel packets \(x\left[\begin{array}{c} B-1 \\ 2B-T \end{array}\right]\).
Then at time $4B - 1$, the receiver receives the channel packets $x^{\left[4B-1\right]}_{3B}$. Using all of the channel packets available, which are $x^{\left[2B-1\right]}_{0}x^{\left[4B-1\right]}_{3B}$, we can additionally recover $s_{a}^{\left[2B-1\right]}_{B}$ and $s_{b}^{\left[2B-1\right]}_{B}$. In total, we have available $2B$ channel packets and have recovered $3B - T$ source $a$ packets and $2B$ source $b$ packets.

Continuing the same argument for more periods, we can recover $(n - 1)$ periods of $3B - T$ source $a$ packets and $2B$ source $b$ packets if we have available $n$ periods of $B$ channel packets. Then we can write the following relation

$$
(n - 1) \cdot [(3B - T) \cdot H(s_{a}) + 2B \cdot H(s_{b})] \leq n \cdot 2B \cdot H(x) \\
\frac{3B - T}{B} \cdot \frac{H(s_{a})}{H(x)} + 2 \cdot \frac{H(s_{b})}{H(x)} \leq \frac{n}{n - 1} \\
\left(\frac{3B - T}{B}\right) R_{a} + 2R_{b} \leq 1 \text{ (as } n \to \infty). \quad (5.22)
$$

Now we have the two inequalities

$$
R_{a} + R_{b} \leq \frac{T}{T + B} \\
\left(\frac{3B - T}{B}\right) R_{a} + 2R_{b} \leq 1
$$

We can express $R_{b}$ as being bounded by a function of $R_{a}$

$$
R_{b} \leq \min \left(\frac{T}{T + B} - R_{a}, \frac{1}{2} - \left(\frac{3B - T}{2B}\right) R_{a}\right) \\
0 \leq R_{a} \leq \frac{T}{T + B}. \quad (5.23)
$$

The valid rates are shown in the shaded region of Figure 5.15.

### 5.3.2 Achievable Codes

To show that the upper bound of the rate can be achieved, we need to focus on three points of the graph: the $R_{a}$-intercept, the $R_{b}$-intercept and the intersection point. This gives us the three rate pairs:

$$
(R_{a}, R_{b}) = \left(\frac{T}{T + B}, 0\right) \\
(R_{a}, R_{b}) = \left(0, \frac{1}{2}\right) \\
(R_{a}, R_{b}) = \left(\frac{B}{T + B}, \frac{T - B}{T + B}\right).
$$
Once we have these codes, then any rate pair meeting capacity can be achieved using a combination of the codes. The first rate pair is achievable using a \((B, T)\) single-link streaming code with source \(a\) and the second rate pair is achievable using a \((B, B)\) single-link streaming code with source \(b\). For the final rate pair, we have the following code.

**Code Construction**

The rate tells us that we can have \(B\) source \(a\) symbols, \(T - B\) source \(b\) symbols and \(T + B\) symbols in total in the block code. Let us the source \(a\) symbols be \(\{a[0], \ldots, a[B - 1]\}\) and the source \(b\) symbols be \(\{b[0], \ldots, b[T - B - 1]\}\). We assign \(a_1\), \(a_2\) and \(b_1\) to be the row vectors

\[
\begin{align*}
  a_1 &= (a[0], \ldots, a[2B - T - 1]) \\
  a_2 &= (a[2B - T], \ldots, a[B - 1]) \\
  b_1 &= (b[0], \ldots, b[T - B - 1]).
\end{align*}
\]

Let \(F\) be a full rank matrix with dimension \((T - B) \times (2B - T)\). Then the block code is:

\[
\begin{pmatrix}
  a_1_{2B-T} & a_2_{T-B} & b_1_{T-B} & a_1 + b_1F_{2B-T} & a_2 + b_1_{T-B}
\end{pmatrix}
\]

To aid with the decoding explanation, let us call the first set of parity symbols \(p_1\) and the second set as \(p_2\) such that

\[
\begin{align*}
  p_1 &= a_1 + b_1F \\
  p_2 &= a_2 + b_2
\end{align*}
\]
We diagonally interleave the block code to form a streaming code.

**Decoding**

A burst erasure of length $B$ erases up to $B$ symbols, but we added $B$ parity check symbols. Notice that each symbol $a[i]$ only appears twice, once by itself in the start of the code, and once in a parity check symbol at the end of the code. Each source symbol $a[i]$ is $T$ time slots away from its corresponding parity check symbol, so a burst of length $B$ can never affect both of them. The symbols $b[i]$ have a similar property, although they appear more than twice, each $b[i]$ does appear on its own in the middle of the code, and has a corresponding parity check symbol $B$ time slots later. So the delay of each $a[i]$ symbol is at most $T$ and the delay of each $b[i]$ symbol is at most $B$. The block code meets the delay constraints if we can show that the decoder can recover all the source symbols from any burst erasure position. We consider four main cases.

**Case 1**

![Diagram](image)

Figure 5.16: A burst erasure for decoding in case 1.

Figure 5.16 shows case 1. Symbols in $a_1$, $a_2$ and $b_1$ are missing. Note that the first $2B - T - \Delta$ symbols in $a_1$ are received perfectly. When the first $2B - T - \Delta$ parity symbols of $p_1$ are received, we can subtract the contribution from $a_1$ and recover the missing $b_1$ symbols. Then when the remainder of the $p_1$ parity symbols are received, the decoder can recover the missing $a_1$ symbols. Next, when the parity symbols $p_2$ are received, they can be used to recover $a_2$.

**Case 2**

The case shown in Figure 5.17 can occur if $2T - 2B \geq B$. To decode the $B - \Delta$ symbols missing in $b_1$, the decoder uses all of the parity symbols in $p_1$ and the first $T - B - \Delta$ symbols of $p_2$. Then the remaining $\Delta$ symbols of $p_2$ are used to recover $a_2$. 
Figure 5.17: A burst erasure for decoding in case 2.

Case 3

Figure 5.18: A burst erasure for decoding in case 3.

Figure 5.18 shows the burst erasure position for case 3. We decode the $T - B$ missing symbols in $b_1$ using the $\Delta$ parity symbols in $p_1$ that have not been erased, and the first $T - B - \Delta$ symbols in $p_2$. Then we use the final $\Delta$ symbols of $p_2$ to recover the missing $\Delta$ symbols of $a_2$.

Case 4

Figure 5.19: A burst erasure for decoding in case 4.

Case 4 is shown in Figure 5.19, where the final $\Delta$ symbols of $b_1$ are erased. The decoder can recover these source symbols using the parity symbols of $p_2$ which are not erased.
5.4 \((B, T) (B, B)\) code, \(T < B\)

In this region, we have \(T_a = T, T_b = B\) but \(T < B\). The capacity is

\[
C_a = 0 \\
C_b = \frac{1}{2}.
\]  
(5.26)

This is not surprising, because we know that a \((B, T)\) single-link code where \(T < B\) has a capacity of 0.

5.4.1 Converse Proof

To prove the converse, we can construct two erasure channel sequences. The first channel sequence is given in Figure 5.20. Each period consists of \(B\) erased symbols followed by \(B\) non-erased symbols. Using this channel sequence, we can show that the decoder can recover all of the source \(a\) packets and source \(b\) packets. This gives a bound of

\[
R_a + R_b \leq \frac{1}{2}.
\]  
(5.27)

Figure 5.20: The periodic erasure channel sequence for one of the inequality relations between \(R_a\) and \(R_b\)

The next erasure channel is in Figure 5.21 and has erasures in every time slot. None of the source \(b\) packets can be recovered, but all of the source \(a\) packets can be recovered. However, there are no channel packets received by the decoder, so this gives the relation

\[
H(s_a) \leq 0 \cdot H(x) \\
R_a \leq 0
\]  
(5.28)

Figure 5.21: The periodic erasure channel sequence for the second inequality relations between \(R_a\) and \(R_b\)
The rate $R_a$ cannot be negative, so we have $R_a = 0$ for any code in this region. Then the first inequality becomes

$$R_b \leq \frac{1}{2}$$

and we have the capacity of both source streams $a$ and $b$.

### 5.4.2 Achievable Codes

The code that achieves capacity is a $(B, B)$ single-link code applied to the source stream $b$. This gives a rate of $\frac{1}{2}$ for $b$, and a rate of 0 for $a$.

### 5.5 Summary

We introduce the multi-source system, where one receiver must receive separate source streams, but each source stream has a different delay constraint. Once again, we investigated the capacity of this system, but in this system, each source stream has a separate rate. The capacity of one source stream depends on the rate of the other source streams.

In this chapter, we investigated the system with two sources, where one source has a delay of $T_a = T$ and the other source has a delay of $T_b = B$. The capacity, given the parameters $B$ and $T$ cannot be expressed as a single scalar, but can instead be expressed in the form of a graph. The graph is piecewise linear, and the converse proof is to find two inequalities involving $R_a$ and $R_b$. All points outside the graph are not achievable. We provided code constructions to achieve capacity, but the code construction where $2B < T < 3B$ is currently unknown.
Chapter 6

Conclusion

In this thesis, we looked at the work conducted by Martinian in his own Ph.D. thesis [12] on delay sensitive streaming codes and extended it further. First, we improved the converse proofs, which upper bound the capacity of a system, by finding a more rigorous method that uses information theoretic notation. Second, we investigated the effects of delay on multicast, multi-link and multisource systems. We investigated the capacity of these systems and we found code constructions and converse proofs.

The advantages of the information theoretic converse proof over the original proof are that it is applicable to both deterministic and non-deterministic codes, it does not require the decoder to have zero probability of error and it can make use of some special techniques, namely, packet revealing and double counting packets. Martinian’s method for converse proofs was to construct a periodic erasure channel sequence, and then show that the system must be able to decode all source packets for that channel sequence. The capacity of the system must then be bounded by the capacity of the channel sequence. The reason why the special techniques don’t work for Martinian’s proof is that his proof requires that all source packets for a given channel sequence are recoverable. This restriction is incompatible with packet revealing and double counting, because packet revealing allows the decoder to decode a subset, and not all, of the source packets, while double counting allows some source packets to be recovered more than once.

In the multicast system, we investigated the capacity of the system for two users. While Martinian’s proof can be used to find an upper bound on the capacity, the information theoretic proof allows for tighter bounds to be found.

The multi-link system reveals some interesting results. When considering two-links only, where a burst erasure can appear on each link, we can naively treat each link
separately and apply an SCo code on each link. However, we manage to establish that when both links are considered together, and the coupling between the burst erasures in each link is exploited, we can achieve a higher rate.

Finally, the multisource system is investigated. Unlike the other systems considered so far, the system is characterized by multiple rates, one for each source. The consequence is that the capacity is no longer a single scalar, but is a trade-off relation between the rates. For a double source system, as the rate for one source increases, the rate of the other source decreases.

Perhaps the main contribution of this thesis is to introduce a systematic approach for developing upper bounds on the capacity of delay constrained communication over burst erasure channels. This completes the heuristic argument in earlier works and also enables several extensions. On the other hand we believe that the analysis of our coding schemes is still somewhat ad-hoc and a more systematic approach could be developed. One observation is that the underlying code constructions are convolutional codes and perhaps some properties of these codes could be used to simplify our proofs.

The multicast and multi-link systems can be generalized into a system that has multiple links and multiple users, but each user receives a subset of the total set of links. If each user has its own delay deadline, then the problem would be to find the capacity of the system. Conversely, each link could have a fixed rate, then the problem would be to find the minimum delay that each user can achieve.
Appendix A

Generalized Packet Revealing Technique

A general upper bound on the capacity of any streaming erasure code can be determined by the following lemma. This lemma only makes use of the packet revealing technique, and not the double counting technique, so it does not give the tightest bound for capacity in all cases.

First, we let $\phi = \{\phi[0], \ldots, \phi[L-1]\}$ be a finite channel sequence of length $L$, where each $\phi[i]$ can be an erasure or a non-erasure. We let $C$ be the streaming code.

For any given channel sequence $\phi$ and streaming code $C$, let us define the sets $V_0(\phi, C)$ and $W_0(\phi)$. We let $\theta(\phi, n_0)$ be a channel sequence that contains $(n_0 + 1)$ periods of $\phi$. 

Figure A.1: A possible sequence for $\phi$

Figure A.2: A possible sequence for $\theta$. It is made up of $(n_0 + 1)$ periods of $\phi$. 

For any given channel sequence $\phi$ and streaming code $C$, let us define the sets $V_0(\phi, C)$ and $W_0(\phi)$. We let $\theta(\phi, n_0)$ be a channel sequence that contains $(n_0 + 1)$ periods of $\phi$. 

86
where \( n_0 = \lceil \frac{T_{\text{max}}}{L} \rceil \) and \( T_{\text{max}} \) is the maximum allowable delay of the code \( C \). Then, we construct \( V_0(\phi, C) \) so that it is the set of decodable source packets from the first period of \( \theta(\phi, n_0) \), and we let \( W_0(\phi) \) be the set of non-erased channel packets in \( \phi \).

We can find the set \( V_0(\phi, C) \) systematically with the following technique: for each time \( t \), we try to decode the packet \( s[t] \) first, and then we reveal the channel packet \( x[t] \). The pseudo-code for this procedure is given by:

```plaintext
begin
  let \( \theta_0 = \theta(\phi, n_0) \)
  for \( t = 0 \rightarrow L - 1 \)
    if \( \theta_t[t] \) in the sequence \( \theta_t[0:T_i \rightarrow 0] \) is decodable using \( C \)
      include \( s[t] \) in \( V_0(\phi, C) \)
      break
    let \( \theta_{t+1} \) equal \( \theta_t \) but change \( \theta_{t+1}[t] \) into a non-erased packet
  end for
end
```

Then we can construct the following lemma.

**Lemma 1.** Given a streaming code \( C \), we can upper bound the capacity of this code with

\[
C \leq \frac{|W_0(\phi)|}{|V_0(\phi, C)|}
\]

for any finite channel sequence \( \phi \).

**Proof.** We are given a channel sequence \( \phi \) and a code \( C \). For readability, we will let \( V_0 = V_0(\phi, C) \) and \( W_0 = W_0(\phi) \). Then, we define the additional sets \( V_i \) and \( W_i \) for \( i > 0 \). Imagine we have a channel sequence with \( iL \) non-erasures followed by \( (n_0 + 1) \) periods of \( \phi \), then \( V_i \) is the subset of source packets \( s[0:(i+1)L-1] \) that can be recovered using code \( C \). This channel resembles the channel \( \theta(\phi) \), so if \( s[t] \) is in \( V_0 \), then \( s[t+iL] \) should also be in \( V_i \). \( W_i \) is the set of channel packets from the \( i \)th period that are not erased.

\[
V_i = \{ s[j] : s[j-L] \in V_{i-1}, j \in \mathbb{Z}^+ \},
W_i = \{ x[j] : x[j-L] \in W_{i-1}, j \in \mathbb{Z}^+ \}, \quad i > 0
\]

(A.1)

We want to prove using mathematical induction that for \( n \geq -1 \)

\[
H(W_0^{n+n_0}) \geq H(V_0^n) + H\left( W_{n+1}^{n+n_0} \mid V_0^n \left( x^{(n+1)L-1} \right) \right).
\]

(A.2)
When we let $n = -1$, we have
\[ H(W_0^{-n_0-1}) \geq H(V_0^{-1}) + H\left(W_0^{-n_0-1} \left| V_0^{-1} x \left[ \begin{array}{c} -1 \\ 0 \end{array} \right] \right) \right) \]
\[ \geq H(W_0^{-n_0-1}) \quad (A.3) \]
which gives the base case.

For the induction step, assume that (A.2) is true for $n = k$. That gives
\[ H(W_k^{k+n_0}) \geq H(V_k^{k}) + H\left(W_k^{k+n_0} \left| V_k^{k} x \left[ \begin{array}{c} (k+1)L-1 \\ 0 \end{array} \right] \right) \right) \]. \quad (A.4) \]

We now add $H(W_{k+n_0+1}^{k+n_0} | W_0^{k+n_0})$ to both sides:
\[ H(W_0^{k+n_0+1}) \geq H(V_0^{k}) + H\left(W_0^{k+n_0+1} \left| V_0^{k} x \left[ \begin{array}{c} (k+1)L-1 \\ 0 \end{array} \right] \right) \right) \]
\[ + H(W_{k+n_0+1}^{k+n_0} | W_0^{k+n_0}) \]
\[ \geq H(V_0^{k}) + H\left(W_0^{k+n_0+1} \left| V_0^{k} x \left[ \begin{array}{c} (k+1)L-1 \\ 0 \end{array} \right] \right) \right) \]. \quad (A.5) \]

At this point, we want to show that we can recover the source packets from $V_{k+1}$. The technique is to start at $t = (k + 1)L$ and try to recover $s[(k + 1)L]$ using any capabilities of the code. Then whether or not we can recover the source packet, we reveal the channel packet $x[(k + 1)L]$. We repeat this technique for subsequent $t$ up to $t = (k + 2)L - 1$. To do this formally, we need another layer of mathematical induction.

We can generalize (A.5) into
\[ H(W_0^{k+n_0+1}) \geq H(V_0^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t\}) \]
\[ + H\left(W_0^{k+n_0+1} \left| V_0^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t\}, x\left[ \begin{array}{c} t \\ 0 \end{array} \right] \right) \right) \]. \quad (A.6) \]

where $[(k + 1)L - 1] \leq t \leq [(k + 2)L - 1]$.

Assuming that (A.6) is true, then we try to recover $s[t + 1]$ using the second term of the R.H.S. If $s[t + 1]$ is recoverable by the channel symbols $x\left[ \begin{array}{c} t \\ 0 \end{array} \right]$ and $W_{k+1}^{k+n_0+1}$, then it
Appendix A. Generalized Packet Revealing Technique

must be in the set $V_{i+1}$ and we can write

$$H(W_{0}^{k+n_0+1}) \geq H(V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t\})$$

$$+ H\left(s[t+1]W_{k+1}^{k+n_0+1}V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t\}, x\left[t\atop 0\right]\right)$$

$$- H\left(s[t+1]W_{k+1}^{k+n_0+1}V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t\}, x\left[t\atop 0\right]\right)$$

$\quad (a)H(V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t\})$

$$+ H\left(s[t+1]W_{k+1}^{k+n_0+1}V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t\}, x\left[t\atop 0\right]\right)$$

$$= H(V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t\})$$

$$+ H\left(s[t+1]V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t\}, x\left[t\atop 0\right]\right)$$

$$+ H\left(W_{k+1}^{k+n_0+1}V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t + 1\}, x\left[t\atop 0\right]\right)$$

$$= H(V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t + 1\})$$

$$+ H\left(W_{k+1}^{k+n_0+1}V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t + 1\}, x\left[t\atop 0\right]\right). \quad (A.7)$$

In step (a), we set the negative term from the previous step to 0, because we can decode $s[t + 1]$ using code C. If $s[t + 1]$ is not decodable, then we can move straight from (A.6) to the final form in (A.7).

Next, we reveal the channel packet $x[t + 1]$.

$$H(W_{0}^{k+n_0+1}) \geq H(V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t + 1\})$$

$$+ H\left(W_{k+1}^{k+n_0+1}V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t + 1\}, x\left[t+1\atop 0\right]\right). \quad (A.8)$$

This is the same formula as (A.6), except that we have $(t + 1)$ instead of $t$, so (A.6) must be true for all $t \geq [(k + 1)L - 1]$. Note that

$$\{s[j] : s[j] \in V_{k+1}, j \leq (k + 2)L - 1\} = V_{k+1}$$

so when we substitute $t = (k + 2)L - 1$ into (A.6), we have

$$H(W_{0}^{k+n_0+1}) \geq H(V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t + 1\})$$

$$+ H\left(W_{k+1}^{k+n_0+1}V_{0}^{k}, \{s[j] : s[j] \in V_{k+1}, j \leq t + 1\}, x\left[t+1\atop 0\right]\right)$$

$$= H(V_{0}^{k+1}) + H\left(W_{k+1}^{k+n_0+1}V_{0}^{k+1}x\left[(k+2)l-1\atop 0\right]\right) \quad (A.9)$$

which completes our initial induction step and proves (A.2) for all $n \geq -1$.
We finish with

\[
H(W_0^{n+n_0}) \geq H(V_0^n) + H(W_{n+1}^{n+n_0} | V_0^n, x^{(n+1)L-1})
\]

\[
H(W_0^{n+n_0}) \geq H(V_0^n)
\]

\[
|W_0^{n+n_0}| \cdot H(x) \geq |V_0^n| \cdot H(s)
\]

\[
(n + n_0 + 1) \cdot |W_0| \cdot H(x) \geq (n + 1) \cdot |V_0| \cdot H(s).
\]  \hspace{1cm} (A.10)

We rearrange this for

\[
R = \frac{H(s)}{|x|} \leq \frac{H(s)}{H(x)} \leq \left(\frac{n + n_0 + 1}{n + 1}\right) \cdot \frac{|W_0|}{|V_0|} \xrightarrow{n \to \infty} \frac{|W_0|}{|V_0|}.
\]  \hspace{1cm} (A.11)

\[\square\]
Bibliography


