On the capacity region for deterministic two-way channels and write-unidirectional memories
van Overveld, W.M.C.J.

DOI:
10.6100/IR345254

Published: 01/01/1991

Document Version
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the author's version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Citation for published version (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.
ON THE CAPACITY REGION FOR
DETERMINISTIC TWO-WAY CHANNELS AND
WRITE-UNIDIRECTIONAL MEMORIES

W.M.C.J. VAN OVERVELD
On the Capacity Region for Deterministic Two-Way Channels and Write-Unidirectional Memories

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de Rector Magnificus, prof. ir. M. Tels, voor een commissie aangewezen door het College van Dekanen in het openbaar te verdedigen op vrijdag 25 januari 1991 om 16.00 uur

door

Wilhelmina Maria Cornelia Johanna van Overveld

geboren te Dongen
Dit proefschrift is goedgekeurd door
de promotoren
prof. dr. ir. J. P. M. Schalkwijk
en
prof. dr. J. H. van Lint

Cover image: 'Tekenen', by M. C. Escher (1948)
Summary

In this thesis, two subjects in information theory are treated, to wit Two-Way Channels (TWCs) and Write Unidirectional Memories (WUMs).

A TWC is a model of communication in two directions: two users want to send information to each other at the same time. The channel they use produces a (discrete) output for each user depending on both inputs. The channel is memoryless, but each user may use previous inputs and outputs when selecting the next input symbol. We investigate how efficiently this channel can be used, i.e., we study its capacity region.

In the thesis, several isolated old results are collected in one theory. We use the so-called 'graphical representation', which relates communication over the channel to the partitioning of the unit square. This was known already for one example of a TWC, the Binary Multiplying Channel (BMC), but it turns out that this method can be used for a large class of channels. With this approach it is possible to derive good lower bounds for the capacity region.

A WUM models the use of a rewritable optical disk. Binary information is recorded on such a disk and per writing cycle, symbols can be changed in just one direction: either zeroes can be changed into ones (over the entire disk; we call this a 1-cycle), or ones can change into zeroes (a 0-cycle). We assume that 0- and 1-cycles alternate. Again we are interested in the capacity: we want to record as much information as possible on a given disk.

We distinguish four cases, depending on whether the writer and/or the reader knows the state of the disk before writing. For each of these situations we derive the capacity. For some of these cases, the capacity was known already; in this thesis, we generalize the results to a more general type of WUM, namely the WUM over an alphabet containing \( q \) symbols. We then have \( q \) different cyclically executed writing cycles. For this situation we derive the capacity region as well, for three out of the four cases. For the fourth case (neither writer nor reader knows the old state) we give an inner bound which is believed to equal capacity.

Finally we study connections between TWCs and WUMs. Since the capacity region for the binary WUM in two cases is equal to a fundamental outer bound for the capacity of the BMC, we look for a generalization of this phenomenon over arbitrary alphabets. This gives rise to a generalization of TWCs with \( q \) users, for which we derive some properties.
Preface

This thesis could not have been written without the support of many people. Among others, I would especially like to thank the following persons:

- my advisor prof. dr. J. P. M. Schalkwijk, for introducing me to information theory in general and Two-Way Channels in particular. The many discussions we had about the Binary Multiplying Channel were very valuable for me.

- Frans Willems, for suggesting the subject of Write Unidirectional Memories to me and for giving me every possible assistance with my research in this area.

- professors J. H. van Lint, J. L. Massey and E. C. van der Meulen, for their careful proof-reading of the manuscript. Their remarks have greatly contributed to the final version of this thesis.

- my colleagues in the Information and Communication Theory group at the Eindhoven University of Technology, for providing a perfect atmosphere for the realization of this thesis.

- my family — especially my father — and friends, for showing constant interest in what I was doing during the past five years.
Contents

Summary i
Preface ii
Contents iii

Chapter 1: Introduction 1
1.1. General introduction 1
1.2. Outline of the thesis 3

Chapter 2: Representation of Two-Way Channels 6
2.1. Two-Way Channel model 6
2.2. Historical perspective: previous results 12
2.3. Graphical representation of deterministic channels 19
   2.3.1. The inner bound resolution 19
   2.3.2. The inner bound resolution for the derived channel 22
2.4. Simplification of graphical representations 25
2.5. Partition patterns 30

Chapter 3: Strategy Trees for Two-Way Channels 34
3.1. The rate of a tree 34
3.2. Fixed length trees 42
   3.2.1. Fixed length trees and the capacity region 42
   3.2.2. Fixed length trees ending in rootlike rectangles 43
   3.2.3. From fixed length trees to codes and vice versa 46
3.3. Variable length trees 50
   3.3.1. Variable length trees and the capacity region 50
   3.3.2. Variable length trees ending in rootlike rectangles 53
   3.3.3. From variable length trees to fixed length codes 56
   3.3.4. From variable length trees to variable length codes and vice versa 60
3.4. Bootstrap trees 64
   3.4.1. Definition and statement of result 64
   3.4.2. Proof of the achievability of bootstrap rates 68
3.5. Relations between strategy trees 87
Contents

3.6. Examples of strategy trees 98
   3.6.1. Strategy trees for the Binary Multiplying Channel 98
   3.6.2. Miscellaneous results 110

Chapter 4: The Write Unidirectional Memory 114
4.1. Binary WUM codes 114
   4.1.1. Introduction 114
   4.1.2. Definitions 116
   4.1.3. Previous results 121
4.2. WUM codes over arbitrary alphabets 125
   4.2.1. The model 125
   4.2.2. Statement of results 127
   4.2.3. Proof of achievability for case 1 and 2 128
   4.2.4. Converse for case 1 and 2 133
   4.2.5. Proof for case 3 136
   4.2.6. Proof for case 4: achievability 147
4.3. Bounding the average rate 151
Appendix A 163
Appendix B 167
Appendix C 172

Chapter 5: Some Connections between Write Unidirectional Memories and Two-Way Channels 174
5.1. Introduction: the binary WUM and the BMC 174
5.2. The q-ary WUM and generalizations of the BMC 180
   5.2.1. The q-ary 'two-way channel' 180
   5.2.2. The q-ary BMC 188
   5.2.3. A q-ary ring channel resembling the WUM 192

List of Symbols 200
Subject Index 203
References 205
Samenvatting 209
Curriculum Vitae 210

— iv —
Chapter 1: Introduction

1.1. General introduction

This thesis deals with two unsolved problems in the field of information theory. In this field, different kinds of communication situations are studied and answers are given to questions like

- 'How fast (and yet reliably) can data be transmitted over a given channel and how should that be done?'
- 'How efficiently can two persons transmit messages to each other over one channel at the same time?'
- 'How much information can be stored in a defective computer memory?'

In 1948, C.E. Shannon [SHA48] gave an answer to the first question, using a mathematical model of the channel under consideration. For several restricted cases he showed how to compute the capacity of a channel, which is a formal description of 'how efficiently' the channel can be used. This publication initiated the research in the theory of channel coding.

The second question was considered by the same author in 1961 [SHA61]. It proved to be essentially more difficult than the first one, because of the interaction between the users. Problems in which two or more users try to transmit and/or receive messages simultaneously are treated in so-called multi-user information theory. For an overview of the type of problems considered there, we refer the reader to [MEU77] or [ELG80]. In the case of two users transmitting to
Chapter 1

each other, the channel is called a two-way channel, abbreviated as TWC. The first part of this thesis is devoted to some special classes of TWCs.

The topic launched in the third question is in fact again a subject of study in channel coding theory, but now for a specific type of channel. In the case of the first question, we usually think of the channel as a medium for communication in space (from the location of the transmitter to the location of the receiver); in the same way a memory cell can be used for communication in time, from the storage time to some later instant of data retrieval. In the second part of this thesis we study a channel similar to this, called a Write Unidirectional Memory (WUM).

Finally we study an interesting connection between the two topics (TWCs and WUMs). It appears that a fundamental bound on the capacity region of the binary multiplying channel (BMC, one of the least complex TWCs) is exactly the same as the capacity region of the WUM. This similarity suggests a generalization of the BMC in the same way as the WUM can be generalized. This 'generalized BMC' is investigated in the last part of this thesis.

Although we try to keep this thesis as self-contained as possible, it cannot be avoided that some basic knowledge about mathematics and information theory is assumed. The reader who is not familiar with concepts like entropy, mutual information and channel capacity is referred to one of the many textbooks ([GAL68], [ASH65]) that present an appropriate introduction to this field. All notations (except the standard ones in mathematics, like \( \mathbb{R} \), \( \{0,1\} \), \( Pr\{X = 0\} \), et cetera) will be explained where they are first introduced. Furthermore, a list of symbols can be found in the back of this thesis.
1.2. Outline of the thesis

This thesis is organized as follows. Apart from this introductory chapter, in which we explain the kinds of problems that are studied and mention the main results, there are four more chapters. Chapters 2 and 3 are devoted to TWCs, Chapter 4 is about WUMs; in the final chapter these two subjects are linked together.

In Chapter 2, we start with the definition of the TWC model (Section 2.1) and we mention some previous results (Section 2.2). Apart from Shannon's theorems on TWCs in general, we mainly concentrate on the BMC. The results for the BMC obtained by Schalkwijk and some of his students were found by looking at the channel in a graphical way, in which the construction of codes for the BMC could be seen as the partitioning of a square. In Sections 2.3 through 2.5 we generalize this idea to arbitrary deterministic TWCs. We give a formal definition of the 'graphical representation', we explain how it should be interpreted and we show how it can be used to classify channels.

In Chapter 3, the graphical representation is used to find inner bounds for the capacity region of TWCs. Schalkwijk used the partitioning of the square to bound the capacity region of the BMC, by changing a partitioning corresponding to a code into a Markov chain. Furthermore he reasoned that one of the states (the 'bad' state) of the Markov chain could be left out of consideration. The rate found in this way is the best bound known for the capacity region, although it was never rigidly proved that this rate is indeed achievable.

The techniques used by Schalkwijk are generalized to arbitrary TWCs in Chapter 3 of this thesis. In Sections 3.1 through 3.4, we define different types of
trees in which partitionings of the square can be represented: fixed length and variable length trees, ending either in rectangles or in arbitrarily shaped vertices, and so-called 'bootstrap' trees. For each of these trees, the rate of the tree is defined and we prove that this rate is achievable for the TWC. Since this result includes both Schalkwijk's original Markov chain and the Markov chain without the 'bad' state, this proves that Schalkwijk's scheme indeed yields an inner bound for the BMC capacity. In Section 3.6, we give examples of different TWCs and we show how inner bounds for the capacity can be found using the trees.

Furthermore we prove (Section 3.5) that the achievable rate regions for the different types of trees are all equal and that they are equal to the capacity region. This shows, for instance, that variable length trees do not yield higher rates than fixed length trees, even though the class of fixed length trees is just a small subclass of the class of variable length trees.

Chapter 4 is split into three sections. In Section 4.1, we introduce the binary WUM as studied by Willems/Vinck, Borden, Simonyi and others. We mention previous results concerning the capacity region for the binary WUM in four cases (depending on whether the encoder and/or decoder is informed about the state of the WUM). In Section 4.2, we generalize the concept of the WUM to arbitrarily large alphabets. We define the so-called 'q-ary WUM' and generalize the definitions from Section 4.1. The rest of this section is devoted to the capacity region in the four cases for the q-ary WUM. For three out of four cases, we are able to determine the capacity region; for the fourth case, we derive an inner bound region which we conjecture to be the actual capacity region.

In Section 4.3, finally, we look at the maximum achievable rate sum in each of the four cases for the q-ary WUM. Using the results from Section 4.2, we derive expressions for this rate sum. We prove that the maximum achievable rate
sum for case 4 is equal to the maximum rate sum in the inner bound region found in Section 4.2, which seems to affirm our conjecture.

Chapter 5 deals with possible connections between TWCs and WUMs. In Section 5.1, we point out a similarity between the binary WUM and the BMC: the so-called 'Shannon outer bound' for the BMC equals the WUM capacity for cases 1, 2 and 3. This leads us to investigations how this may be generalized to the situation with arbitrary alphabets. In other words, we are looking for a generalization of the BMC having a 'generalized Shannon outer bound' equal to the \(q\)-ary WUM capacity region for any of the four cases.

In Section 5.2, we define a generalization of the TWC called a \(q\)-ary ring channel. We generalize Shannon's TWC theorems to this channel, which gives us a 'generalized Shannon outer bound'. In Section 5.3, some examples of \(q\)-ary ring channels are examined to see whether their outer bound equals the WUM capacity region. The last one of these examples seems to have this property, although we do not have a proof. The other two channels studied do not have the desired property for \(q > 2\). Nevertheless these ring channels provide promising material for further research.
Chapter 2: Representation of Two-Way Channels

2.1. Two-way channel model

Two-way channels were introduced by Shannon in 1961 [SHA61]. He presented a model of a communication situation in which two users want to send information to each other simultaneously over the same physical channel. In Figure (2.1.1), the model for a two-way channel (TWC) called $\mathcal{K}$ is shown.

![Diagram of a two-way channel](image)

(2.1.1) Figure. A two-way channel.

The two users (or terminals) are called 0 and 1. The (time-discrete) channel $\mathcal{K}$ consists of two input alphabets $\mathcal{X}_0$ and $\mathcal{X}_1$ (for user 0 and 1, respectively), two output alphabets $\mathcal{Y}_0$ and $\mathcal{Y}_1$ and a so-called transition matrix $P^*(y_0, y_1 | x_0, x_1)$, with

$$P^*(y_0, y_1 | x_0, x_1) \in [0, 1] \text{ for } x_0 \in \mathcal{X}_0, x_1 \in \mathcal{X}_1, y_0 \in \mathcal{Y}_0, y_1 \in \mathcal{Y}_1,$$

and

$$\sum_{(y_0, y_1) \in \mathcal{Y}_0 \times \mathcal{Y}_1} P^*(y_0, y_1 | x_0, x_1) = 1 \text{ for } x_0 \in \mathcal{X}_0, x_1 \in \mathcal{X}_1.$$
The channel is used as follows. At each time instant (e.g. each second; this plays no role in the rest of the paper), both users select symbols \( I_0 \) and \( I_1 \) from their respective input alphabets \( X_0 \) and \( X_1 \) and use these as inputs for channel \( K \). Depending on the input pair, the channel produces two output symbols \( Y_0 \) and \( Y_1 \); symbol \( Y_0 \in Y_0 \) can be observed by user 0 and user 1 observes \( Y_1 \in Y_1 \). The pair \((Y_0, Y_1)\) is statistically related to input pair \((I_0, I_1)\) in a way described by \( P^* \):

\[
Pr\{ (Y_0, Y_1) = (y_0, y_1) \mid (I_0, I_1) = (z_0, z_1) \} = P^* (y_0, y_1 \mid z_0, z_1).
\]

(Here and in the rest of this thesis, upper case letters are used for random variables and the corresponding lower case letters denote actual outcomes of the stochastic process that is considered.)

Channel \( K \) is assumed to be discrete, which expresses the fact that both the input and the output alphabets have finite cardinalities. If some alphabet, say \( A \), has \( q \) elements, then without loss of generality we will take \( A = \{0,1,\ldots,q-1\} \).

It follows from the channel model that the channel is memoryless. This means that the probability distribution of the output pair \((Y_0, Y_1)\) at instant \( t \) is determined completely by the input pair at that instant. However, the output pair does depend on all previous input pairs since we allow feedback: each user may choose the next input symbol depending on all previous input and output symbols observed by him. Hence we have the following situation. Let \( I_0^N = (I_0^0, I_0^1, \ldots, I_0^{N-1}) \) and \( I_1^N = (I_1^0, I_1^1, \ldots, I_1^{N-1}) \) be input sequences of length \( N \) and let \( Y_0^N \) and \( Y_1^N \) be the corresponding output sequences. Then, with \( P(z_0^N, z_1^N, y_0^N, y_1^N) := Pr\{ (I_0^N, I_1^N) = (z_0^N, z_1^N), (Y_0^N, Y_1^N) = (y_0^N, y_1^N) \} \) we have

\[
P(z_0^N, z_1^N, y_0^N, y_1^N) = \prod_{n=0}^{N-1} P(z_0^n, z_1^n \mid z_0^n, y_0^n) \cdot P(z_1^n, y_1^n \mid z_1^n, y_1^n) \cdot P^* (y_0^n, y_1^n \mid z_0^n, z_1^n, y_0^n, y_1^n).
\]
This notation should be interpreted in such a way that for \( n = 0 \), \( \mathbb{Z}^n \) and \((x_0, x_1, \ldots, x_{n-1})\) denote the empty sequence.

An interesting example of a discrete memoryless TWC is the so-called binary multiplying channel, or BMC, first studied by Blackwell and described in [SHA61].

(2.1.2) Example. The binary multiplying channel:

\[
x_0 = x_1 = y_0 = y_1 = \{0, 1\}
\]

and

\[
P^*(y_0, y_1 | x_0, x_1) = \begin{cases} 
1 & \text{if } y_0 = y_1 = x_0 x_1 \\
0 & \text{otherwise.}
\end{cases}
\]

The channel in Example (2.1.2) is a deterministic TWC and it is also a T-channel. Both of these concepts are defined below.

(2.1.3) Definition. A TWC with parameters \((X_0, X_1, Y_0, Y_1, P^*)\) is called deterministic if the output pair is completely determined by the input pair; in other words, all entries in the transition matrix \(P^*\) are either 0 or 1. In this case we can define a function \(f_K: X_0 \times X_1 \to Y_0 \times Y_1\), where \(f_K(x_0, x_1)\) is the output pair generated by channel \(K\) if the input pair is \((x_0, x_1)\).

For the BMC example, we have \(f_{\text{BMC}}(x_0, x_1) = (x_0x_1, x_0x_1)\).

(2.1.4) Definition. A TWC with parameters \((X_0, X_1, Y_0, Y_1, P^*)\) is called a T-channel if the transition matrix is such that for all input pairs \((x_0, x_1)\), the output pair has \(Y_0 = Y_1\). Such a channel is also known as a single-output TWC; its output is called \(Y\) (instead of \(Y_0\) or \(Y_1\)).
Consider an arbitrary (discrete, memoryless, but not necessarily deterministic) two-way channel $\mathcal{X}$. The channel can be used for communication in the following way (see Figure (2.1.5)).

Let $\mathcal{M}_0 \subset \mathcal{W}$, $\mathcal{M}_1 \subset \mathcal{W}$, $N \in \mathbb{W}$ and let $\mathcal{M}_i := \{0, 1, \ldots, N-1\}$ for $i \in \{0, 1\}$. The sources generate messages $\mathcal{V}_0$ and $\mathcal{V}_1$, respectively, with $\mathcal{V}_0 \in \mathcal{M}_0$ and $\mathcal{V}_1 \in \mathcal{M}_1$. The sets $\mathcal{M}_0$ and $\mathcal{M}_1$ are called the message sets. $\mathcal{V}_0$ and $\mathcal{V}_1$ are independent and uniformly distributed: for all $w_0 \in \mathcal{M}_0$, $w_1 \in \mathcal{M}_1$,

$$
\Pr\{\mathcal{V}_0 = w_0, \mathcal{V}_1 = w_1\} = \Pr\{\mathcal{V}_0 = w_0\} \cdot \Pr\{\mathcal{V}_1 = w_1\},
$$

$$
\Pr\{\mathcal{V}_0 = w_0\} = 1/\mathcal{W}_0, \quad \Pr\{\mathcal{V}_1 = w_1\} = 1/\mathcal{W}_1.
$$

A message is transmitted to the other terminal by selecting appropriate input symbols for the channel in $N$ consecutive instances of time. The selection is made by the encoders. Let $i \in \{0, 1\}$, $j \in \{0, 1\}$, $i \neq j$. At instant $n$, $n \in \{0, 1, \ldots, N-1\}$, user $i$ can select his input symbol $X_{i,n}$ depending on the message $\mathcal{V}_i$ he is transmitting, but $X_{i,n}$ may also depend on the sequence of $n$ $Y_i$-symbols that user $i$ has observed up to this moment. After all $N$ symbol pairs have been sent over the channel, the decoder of user $i$ estimates user $j$’s message $\mathcal{V}_j$ as $\hat{\mathcal{V}}_j$ by looking at the output sequence $I_{i}^{N}$ and knowing the fact that his own transmitted message is $\mathcal{V}_i$. 

(2.1.5) Figure. Communication situation for a TWC.
Chapter 2

Formally this process can be described by a code, see Definition (2.1.6). The reader familiar with the classical one-way channel (cf. [GAL68], [SHA48]) will recognize definitions (2.1.6) through (2.1.10), since they are straightforward generalizations of some one-way channel coding concepts.

(2.1.6) Definition. An \((W_0, W_1, N)\) code consists of \(2N\) encoding functions \(f_i, 0, f_i, 1, \ldots, f_i, N-1\) \((i=0, 1)\) and two decoding functions \(g_0\) and \(g_1\). For \(n \in \{0, 1, \ldots, N-1\}\), the function

\[
f_i, n : M_i \times Y_i^n \rightarrow X_i
\]

maps the message \(V_i\) and the received \(Y_i\)-sequence \(Y_i^n = (Y_i, 0, Y_i, 1, \ldots, Y_i, n-1)\) into the next channel input symbol \(X_i, n\) for user \(i:\)

\[
X_i, n = f_i, n(V_i, Y_i^n).
\]

The decoding function

\[
g_i : M_i \times Y_i^N \rightarrow M_j
\]

describes the way in which \(V_j\) is estimated by user \(i\) \((i \neq j)\):

\[
\hat{V}_j = g_i(V_i, Y_i^N).
\]

The parameter \(N\) is called the block length (or codeword length) of the code. \(\Box\)

For a fixed code, we can study the probability that either one of the messages will be decoded incorrectly if this code is used. Hence we define the following.

(2.1.7) Definition. The (average) error probabilities \(P_{e, 0}\) and \(P_{e, 1}\) of a code are defined by

\[
P_{e, i} := \sum_{w_i \in M_i} Pr\{V_i = w_i \text{ and } \hat{V}_i \neq w_i\}, \ i \in \{0, 1\}.
\] \(\Box\)
We will look at the rate pair (sometimes called the rate point) of a code, which can be interpreted as the average amount of information (in bits) that is sent over the channel – from terminal 0 to terminal 1 and vice versa – per channel use.

(2.1.8) Definition. For an \((M_0, M_1, N)\) code, the rate pair \((R_0, R_1)\) is defined by
\[
R_i := \frac{\log(M_i)}{N}, \quad i \in \{0, 1\}.
\]
Here, like everywhere else in this thesis, the logarithm is to the base 2.

The next two definitions are important for finding out how efficiently the two-way channel can be used for reliable communication, in terms of high rates and yet small error probability.

(2.1.9) Definition. A rate pair \((R_0, R_1)\) is called achievable if for all \(\epsilon > 0\) and all sufficiently large \(N\), an \((M_0, M_1, N)\) code exists having error probabilities \(P_{e,0}\) and \(P_{e,1}\), such that
\[
\frac{\log(M_i)}{N} \geq R_i - \epsilon \quad \text{and} \quad P_{e,i} < \epsilon \quad \text{for} \quad i \in \{0, 1\}.
\]
The expression 'achievable rate' suggests that a code having this rate exists, which is not necessarily true. 'Approachable' might be a more appropriate word, but the term 'achievable' is more commonly used in the literature.

(2.1.10) Definition. The capacity region of a two-way channel is the set of all achievable rate pairs. If the TWC is called \(\mathcal{X}\), then its capacity region will be denoted by \(C_X\); if no confusion is possible, it will just be denoted by \(C\).

It should be noted that a capacity region is a closed convex set (cf. [SHA61]).
2.2. Historical perspective: previous results

The first results on capacity regions for TWCs were obtained by Shannon [SHA61]. For a discrete memoryless TWC with parameter set \((X_0, X_1, Y_0, Y_1, P^*)\), he derived an upper and lower bound for the capacity region. Before we can state this result, we need to introduce some notation.

(2.2.1) Notation. Consider a TWC as described above and let \(P(z_0, z_1)\) be some probability distribution on \(I_0\) and \(I_1\). Then \(I(I_0; Y_1 | I_1)\) denotes the conditional mutual information [GAL68] between \(I_0\) and \(Y_1\) conditional on \(I_1\):

\[
I(I_0; Y_1 | I_1) = \sum_{x_0, x_1, y_1} P(x_0, x_1, y_1) \cdot \log \left( \frac{P(x_0, x_1, y_1) P(z_1)}{P(y_1) P(x_0, z_1)} \right).
\]

Here the distributions \(P(x_0, x_1, y_1)\), \(P(x_1, y_1)\) and \(P(z_1)\) are found using the given distribution \(P(z_0, z_1)\) and the channel transition matrix \(P^*\). Thus:

\[
P(x_0, x_1, y_1) = P(z_0, x_1) \cdot \sum_{y_0} P^*(y_0, y_1 | x_0, z_1).
\]

Similarly, we denote entropy and conditional entropy as \(H(I_0)\) and \(H(Y_1 | I_0)\).

Hence we have e.g.

\[
I(I_0; Y_1 | I_1) = H(Y_1 | I_1) - H(Y_1 | I_0, I_1),
\]

\[
H(Y_1 | I_1) = H(Y_1, I_1) - H(I_1).
\]

(2.2.2) Definition. The Shannon inner bound region \(G_I\) is defined as

\[
\text{co} \cup \{ (r_0, r_1) \in \mathbb{R}^2 \mid 0 \leq r_0 \leq I(I_0; Y_1 | I_1), 0 \leq r_1 \leq I(Y_1; Y_0 | I_0) \}.
\]

where 'co' denotes taking the the convex hull and the union is taken over all product distributions \(P(x_0, x_1) = P(x_0) P(x_1)\).
(2.2.3) Definition. The Shannon outer bound region \( G_o \) is defined as

\[
\bigcup \left\{ \left( \mathbf{x}_0, \mathbf{x}_1 \right) \in \mathbb{R}^2 \mid 0 \leq \mathbf{x}_0 \leq I(X_0;Y_1|I_1), 0 \leq \mathbf{x}_1 \leq I(X_1;Y_0|I_0) \right\},
\]

where the union is taken over all simultaneous distributions \( P(x_0, x_1) \). No 'co' is necessary here, because \( G_o \) is already convex, as shown in [SHA61].

(2.2.4) Theorem (Shannon '61). For a TWC with parameters \( (X_0, X_1, Y_0, Y_1, P^*) \) and capacity region \( C \), we have

\[ G_i \subseteq C \subseteq G_o. \]

Shannon also proved that the capacity region for a TWC can be expressed in terms of the inner bound regions for its 'derived channels'. The definition of a derived channel is given below. Theorem (2.2.6) states Shannon's result.

(2.2.5) Definition. Let \( \kappa \) be a TWC and let \( N \in \mathbb{N} \). The \( N^{th} \) derived channel of \( \kappa \), denoted as \( \kappa^{(N)} \), is defined such that each use of \( \kappa^{(N)} \) corresponds to \( N \) successive uses of \( \kappa \). To be more precise, input letters \( x_0^N, x_1^N \) for \( \kappa^{(N)} \) are strategies of length \( N \) for \( \kappa \), where the next input \( x_i \) (for \( \kappa \)) may depend on previous input and output symbols \( x_i, y_i \) within the same block of length \( N \); output letters \( y_0^N, y_1^N \) for \( \kappa^{(N)} \) are blocks of outputs for \( \kappa \): \( y_0^N = y_0^N \) and \( y_1^N = y_1^N \).

(2.2.6) Theorem (Shannon '61). Let \( \kappa \) be a TWC with capacity region \( C \). Let, for \( N \in \mathbb{N} \), \( G_i^{(N)} \) denote the Shannon inner bound for the derived channel \( \kappa^{(N)} \). Then

\[ \forall N \in \mathbb{N} \left[ G_i^{(N)} / N \subseteq C \right] \quad \text{and} \quad C = \lim_{N \to \infty} G_i^{(N)} / N, \]

where \( G_i^{(N)} / N := \{ (g_0/N, g_1/N) \mid (g_0, g_1) \in G_i^{(N)} \} \).
Chapter 2

The limit occurring in Theorem (2.2.6) involves a sequence of regions converging to another region. Although the meaning of this limiting process may be clear intuitively, we still give a formal definition of this notion because it will be needed in Chapter 3, where we will prove theorems similar to (2.2.6).

(2.2.7) Definition. The distance between \( p \in \mathbb{R}^2 \) and \( A \subset \mathbb{R}^2 \), \( d(p, A) \), is defined as

\[
d(p, A) = \inf \{ \, d(p, a) \mid a \in A \}\]

where \( d(p, a) \) is the Euclidean distance between the points \( p \) and \( a \).

(2.2.8) Definition. Let \( \{A_N\} \in \mathbb{N} \) be a sequence of regions in \( \mathbb{R}^2 \). Then \( A_\infty \) is the limiting region of \( \{A_N\} \), denoted as \( \lim_{N \to \infty} A_N = A_\infty \), if (i) and (ii) hold.

(i) \( \forall a \in A_\infty \, \forall \varepsilon > 0 \exists N_0 \in \mathbb{N} \forall N > N_0 \left[ d(a, A_N) < \varepsilon \right] \)

(ii) \( \forall a \notin A_\infty \exists \varepsilon > 0 \exists N_0 \in \mathbb{N} \forall N > N_0 \left[ d(a, A_N) > \varepsilon \right] \).

Theorem (2.2.6) theoretically solves the problem of finding the capacity region for a TWC. However, since it involves taking a limit, the actual boundaries of this region are still not computable for most TWCs. Obviously, if \( G_i = G_o \) for some TWC, then its capacity region is determined by Theorem (2.2.4). This equality holds for some TWCs (e.g., TWCs with a 'symmetrical structure', cf. [SHA61]), but Shannon showed that there are also TWCs for which \( G_i \neq G_o \). The least complex example of such a channel is Example (2.1.2), the BMC. By evaluating (2.2.2) and (2.2.3) for this channel, it follows that (cf. [SHA61])

(2.2.9) \( G_i = \{ \, (a \cdot h(\beta), \beta \cdot h(a)) \mid 0 \leq a \leq 1, 0 \leq \beta \leq 1 \} \)

and

(2.2.10) \( G_o = \{ \, (\beta \cdot h\left(\frac{1-a}{\beta}\right), a \cdot h\left(\frac{1-\beta}{a}\right)) \mid 0 \leq a \leq 1, 0 \leq \beta \leq 1, a+\beta \geq 1 \} \).
Here \( h(\cdot) \) denotes the binary entropy function:

\[
h(x) = -x \cdot \log(x) - (1-x) \cdot \log(1-x) \quad \text{for} \quad 0 < x < 1, \quad h(0) = h(1) = 0.
\]

Figure (2.2.11) shows the situation for the BMC. The regions \( G_i \) and \( G_o \) are bounded by the respective convex curves and the axis segments.

(2.2.11) Figure. Shannon bounds for the BMC.

One can check that \( G_o \) is strictly larger than \( G_i \) by looking at the equal rate point \((r_0 = r_1)\) on the border of both regions: For \( G_i \), we find \( r_0 = r_1 = 0.61695 \) and for \( G_o \), we have \( r_0 = r_1 = 0.69424 \).

At the time of Shannon's paper, in 1961, it was not yet known whether the inner bound for the BMC was strictly interior to the capacity region; the best BMC code known at that time (in the symmetrical case) had \( r_0 = r_1 = 0.57143 \). This code was developed by Hagelbarger [SHA61]. However, this is not a code in the sense of Definition (2.1.6), since it has codewords of different lengths (for a description of the Hagelbarger code, see Section 3.3.4 of this thesis).

The problem 'can the capacity region exceed the Shannon inner bound?' was open not only for the BMC; in fact, it was unknown whether this could happen for any TWC at all. This is a fundamental problem: if the capacity is equal to the inner bound, then it is impossible to take advantage of possible
statistical dependency that can be created between the two users by means of previous inputs and outputs. Indeed, the proof of the Shannon inner bound [SHA61] is based on the existence of codes achieving any rate in $\mathcal{G}_i$ that do not 'use' previous output symbols (in terms of Definition (2.1.6): the function value $f_{i,n}(V_i, F_i^n)$ varies with $V_i$ only.)

An attempt to solve the aforementioned problem was first made by Jelinek [JEL64]. He devised a coding scheme for an arbitrary TWC, by decomposing the TWC into two one-way channels. Unfortunately the rate of this coding scheme was still within the Shannon inner bound region. Jelinek (incorrectly) conjectured that, at least for the BMC, it would not be possible to achieve rates outside $\mathcal{G}_i$.

Not until 1979 it was shown that a capacity region can exceed the Shannon inner bound. Dueck [DUE79] constructed an example of a TWC (not the BMC) for which $C \neq \mathcal{G}_i$. In 1982, Schalkwijk [SCH82] showed that the same is true for the BMC by developing a scheme with rate $(\ell_0, \ell_1)$, $\ell_0 = \ell_1 = 0.61914$. Like Hagelbarger's code, this was not a code in the sense of Definition (2.1.6), but Tolhuizen [TOL85] proved that Schalkwijk's scheme can be used to construct a code having the same rate. Schalkwijk improved his scheme in 1983 using a technique called 'bootstrapping' [SCH83a] and obtained an achievable symmetrical rate pair with $\ell_0 = \ell_1 = 0.63056$. Because of the bootstrapping, it was not obvious that this new rate could also be achieved with 'strict-sense' codes. Nevertheless this is true, as shown in [OVE88a] and in Section 3.4 of this thesis. In 1986, Schalkwijk [SCH86] conjectured this rate point to be optimum, but recently, using a more intricate bootstrap scheme [SCH90], he found a slightly higher rate: $0.630555299$ instead of $0.630555256$. For the BMC, this gives a total improvement upon the Shannon inner bound from $0.61695$ to $0.63056$.

The Shannon inner bound could be exceeded not only for specific channels: Han [HAN84] developed a coding scheme for the arbitrary TWC, yielding an
achievable rate region which strictly includes Shannon’s inner bound for a large class of channels. Unfortunately this result cannot be compared to Schalkwijk’s rate point, since computing Han’s achievable rate region for the BMC involves quite complex numerical calculations. In the same paper [HAN84], Han proved that the capacity region is equal to the Shannon inner bound for the Gaussian TWC (which is a TWC having $X_0 = X_1 = Y_0 = Y_1 = \mathbb{R}$ and $Y_0 = aI_0 + bI_1 + N_0$, $Y_1 = cI_0 + dI_1 + N_1$ where the noise terms $N_0$ and $N_1$ are Gaussian).

As for the Shannon outer bound: it took even longer before this was improved. Again, let us consider the symmetrical rate pair $(I_0, I_1)$ with $I_0 = I_1$ for the BMC. In 1986, Zhang, Berger and Schalkwijk [ZHA86] lowered the bound from 0.69424 to 0.64891. Around the same time, the tightest upper bound known up to now was found by Hekstra and Willems [HEK89]: $I_0 = I_1 = 0.64628$. The true capacity region of the BMC is still unknown.

For arbitrary TWCs, the Shannon outer bound can be improved upon by the technique described in [ZHA86]; the outer bound for the BMC is just an example of the general Zhang–Berger–Schalkwijk (ZBS) bound. However, it is very hard to evaluate the ZBS bound in general, because it involves optimization over probability distributions of auxiliary random variables that may take values in large alphabets. The bound in [HEK89], which is known as the dependence balance bound, is also applicable to more channels than just the BMC: it holds for any T–channel. In general it is stronger than the ZBS bound, but unfortunately it has the same disadvantage: the optimization over arbitrary random variables hampers numerical computation of the bound.

Recently, some problems related to Shannon’s classical two–way situation have been studied. Salehi [SAL88] looked at the situation in which the two sources are correlated, but the channel is ‘restricted’. The restriction on the channel is
the fact that the users are not allowed to look at previous outputs before they
decide on their next input symbols. In this situation we have a different kind of
dependency between the users, originating from the message pair rather than from
the fact that the new input pair depends on old output pairs. Salehi derived an
achievable rate region for this case, which is typically somewhere in between $G_i$
and $G_o$ (depending on the kind of correlation); as in Theorem (2.2.6), he was able
to express the capacity region as his inner bound region for the $N^{th}$ derived
channel, for $N \to \infty$.

Tiersma [TIE89] studied an extension of the TWC for three users. In this
situation, each of the three users is sending messages to the other two via the
three-way channel which produces an output triple $(Y_0, Y_1, Y_2)$ depending on the
input triple $(I_0, I_1, I_2)$. Tiersma derived bounds on the capacity region that
generalize the Shannon inner and outer bound for the TWC. As an example, he
looked at the BMC with $Y_0 = Y_1 = Y_2 = I_0 \cdot I_1 \cdot I_2$. For both the inner and outer
bound region, he computed the rate point with highest sum rate, $I_0 + I_1 + I_2$.
He also developed a coding scheme reminiscent of the one in [SCH82], which
achieved a sum rate slightly below his inner bound value.

Finally we should mention the work of numerous students in the Informa-
tion Theory Group of the Electrical Engineering Department of the Eindhoven
University of Technology. Between 1982 and 1988, many of them have worked on
problems related to Schalkwijk's coding scheme for the BMC. Some of them tried
to improve the scheme or to construct different types of strategies for the BMC
([HAZ85], [LEU84], [OVE85], [SME83], [TOL85]); others concentrated on more
complicated channels (e.g. with ternary input and output alphabets), tried to
characterize them and to find good strategies for them ([ENG87], [GAA85],
[HAM87], [HEI87], [JAC86], [LOR88]). Since part of their work is used in this
thesis, we will describe their results in the sections where they are used.
2.3. Graphical representation of deterministic channels

2.3.1. The inner bound resolution

In this section we will introduce the graphical representation of a deterministic TWC. This is a way to depict the probability distribution of input and output symbols in a unit square. This approach was first used by Schalkwijk [SCH82] (and could also be applied to multiple access channels, as shown by Zhang, Berger and Massey [ZHA87]). It appears to be a useful way to represent distributions for deterministic TWCs. Therefore we will restrict ourselves to deterministic TWCs in the rest of this thesis.

From now on, \( X \) will be a discrete memoryless deterministic TWC with alphabets \( X_0, X_1, Y_0 \) and \( Y_1 \). Let the relation between inputs and outputs be given by \((Y_0, Y_1) = f_X(I_0, I_1)\). To avoid trivialities, we will assume that the capacity region \( C \) is not confined to one of the coordinate axes in the \((I_0, I_1)\) plane; this implies that both users can reliably transmit a positive amount of information to each other, however low the rate of the code may be (e.g. by time sharing; we will come back to this in Section 3.3.2). For this case, we have \( H(Y_1 | I_0, I_1) = H(Y_0 | I_0, I_1) = 0 \), since \( Y_0 \) and \( Y_1 \) can be determined from \( I_0 \) and \( I_1 \). Now the Shannon inner bound defined in (2.2.2) can be rewritten as follows.

\[
G_i = \co \{ (I_0, I_1) \mid 0 \leq I_0 \leq H(Y_1 | I_1), 0 \leq I_1 \leq H(Y_0 | I_0), (Y_0, Y_1) = f_X(I_0, I_1), P(x_0, x_1) = P(x_0)P(x_1) \}.
\] (2.3.1.1)

Consider a distribution \( P(x_0, x_1) = P(x_0)P(x_1) \). For user \( i \) (\( i \in \{0, 1\} \)), we represent \( P(x_i) \) by a unit interval which is partitioned into \(|X_i|\) subintervals, one for each possible input symbol, such that the size of each subinterval corresponds to the probability that that input symbol is chosen. As an example, take
$X_0 = \{0,1,2\}$ with $P(0) = 0.2$, $P(1) = 0.5$, $P(2) = 0.3$. The corresponding partitioning of the unit interval for user 0 is depicted in Figure (2.3.1.2).

![Partitioning of a unit interval for user 0.](image)

(2.3.1.2) Figure. Partitioning of a unit interval for user 0.

Since the users are independent, the simultaneous distribution $P(x_0, x_1)$ can be represented in a square being the Cartesian product of the unit intervals for both users. The square is partitioned into subrectangles corresponding to pairs $(x_0, x_1)$.

Let channel $K$ be used once with the given input distribution. Since $K$ is deterministic, we can assign to each of the subrectangles the channel output pair $(y_0, y_1) = f_K(x_0, x_1)$. The following example will make this clear.

(2.3.1.3) Example. The ternary TWC $K$ for which $X_0 = X_1 = Y_0 = Y_1 = \{0,1,2\}$ with $f_K$ following from the output matrices below.

<table>
<thead>
<tr>
<th>$I_0$</th>
<th>$I_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 1 2</td>
</tr>
<tr>
<td>0</td>
<td>1 0 0</td>
</tr>
<tr>
<td>1</td>
<td>2 0 2</td>
</tr>
<tr>
<td>2</td>
<td>1 2 2</td>
</tr>
</tbody>
</table>

Assume that the users independently choose their input distribution, which means that the simultaneous input distribution is a product distribution: $P(x_0, x_1) = P(x_0) \cdot P(x_1)$. If this distribution is used on $K$, then the unit square is
partitioned as shown in Figure (2.3.1.4). In a subrectangle corresponding to 
\((z_0, z_1)\), the corresponding output \((y_0, y_1)\) is written.

\[
\begin{array}{ccc}
I_1=0 & I_1=1 & I_1=2 \\
I_0=0 & 10 & 01 & 00 \\
I_0=1 & 20 & 00 & 20 \\
I_0=2 & 10 & 22 & 21 \\
\end{array}
\]

\textbf{(2.3.1.4) Figure.} Partitioning of the unit square for channel \(K\).

Figure (2.3.1.4) summarizes the operation of channel \(K\): the input and 
output alphabets and the relation between inputs and outputs can be obtained 
from it. For this reason we refer to pictures like (2.3.1.4) as a \textit{graphical representation} of the TWC. Since we have a product distribution for the inputs, the area of the \((x_0, x_1, y_0, y_1)\) rectangle in Figure (2.3.1.4) equals

\[
P(x_0, x_1, y_0, y_1) = P(x_0) \cdot P(x_1) \cdot \phi((y_0, y_1) = f_K(x_0, x_1)).
\]

In this notation, \(\phi(I) := 1\) if assertion \(I\) is true and otherwise \(\phi(I) := 0\). Therefore the distribution \(P(x_0, x_1, y_0, y_1)\) can be derived from the areas of all subrectangles in the figure and, consequently, \(H(Y_1 | I_1)\) and \(H(Y_0 | I_0)\) can be computed from the figure. By varying the lengths of the subintervals for the two users (i.e., 
\(P(x_0)\) and \(P(x_1)\)), we find all points in region \(G_i\), see (2.3.1.1). This 
already shows that the graphical representation of a channel is an important tool 
for the computation of achievable rates. In the rest of this chapter and in Chapter 
3, the importance of pictures like these will become more transparent.
Chapter 2

We say that Figure (2.3.1.4) depicts an inner bound resolution. 'Inner bound' because of the independence between the inputs; 'resolution' because if the channel is used with the given input distribution, then an amount of uncertainty is resolved, namely $I(I_0; Y_1 | I_1) (= H(Y_1 | I_1))$ and $I(I_1; Y_0 | I_0) (= H(Y_0 | I_0))$ in the two directions (before transmission, the uncertainty about the other user's input was $H(I_0 | I_1)$ and $H(I_1 | I_0)$, respectively; afterwards these amounts are lowered to $H(I_0 | I_1, Y_1)$ and $H(I_1 | I_0, Y_0)$, respectively. Each difference equals an average mutual information.).

It should be noted that the correspondence between probabilities and areas is possible only because we started with independent $I_0$ and $I_1$. An arbitrary distribution $P(z_0, x_1)$ cannot be drawn by partitioning the unit square into rectangles; therefore, in general, the Shannon outer bound cannot be computed using such a picture. Nevertheless, the kind of dependency that we will encounter in strategies (the dependency arising from previous input and output pairs) can be represented in a square. How this can be done is explained in the next section.

2.3.2. The inner bound resolution for the derived channel

Again we consider Example (2.3.1.3). In Section 2.3.1, we represented the simultaneous distribution of inputs and outputs in a unit square, for a single transmission over the channel with independent users. The distribution of the input $I_i$, $P(x_i)$, could be chosen freely by user $i$. In Figure (2.3.1.4), the distribution choice of user 0 is visible as the positions of the horizontal lines cutting the square; user 1’s choice corresponds to the vertical lines in the picture. We call these lines, or more accurately, the positions of these lines, the thresholds for user 0 and 1. We have called this the graphical representation of an inner bound resolution.

Now we consider an inner bound resolution for $K^{(2)}$ and its graphical
representation. As an illustration, suppose that user 0 sent $I_{00}=2$. The number of strategies of length 2 for user 0 that start with $I_{00}=2$ equals 9: indeed, user 0 can either receive $V_{00}=1$ or $V_{00}=2$; in either case he can send $I_{01}=0$, 1 or 2, depending on $V_{00}$. The probability distribution for these 9 strategies, $P(z_0^2)$, is drawn as the horizontal thresholds in Figure (2.3.2.1) that partition the lower part of the unit square into 9 rows.

![Diagram of horizontal thresholds](image)

(2.3.2.1) Figure. Partitioning of (part of) the square according to $P(z_0^2)$.

If we draw all horizontal and vertical thresholds according to $P(z_0^2)$ and $P(z_1^2)$ in the unit square, then we have the graphical representation for the inner bound resolution for $\mathcal{K}^{(2)}$. In other words, the strategy distribution $P(z_0^2) \cdot P(z_1^2)$ corresponds to a partitioning of the square into subrectangles.

In this way we can represent an inner bound resolution for the derived channel $\mathcal{K}^{(N)}$ for all $N$ and for any channel $\mathcal{K}$. Consider the graphical representation of a transmission over $\mathcal{K}^{(N)}$ for some channel $\mathcal{K}$. Then each point on user 0's unit interval (i.e., each horizontal line in the square) corresponds to a choice of
input letter $\tilde{I}_0^N$ for the TWC $\mathcal{K}(N)$ and every vertical line corresponds to a choice of $\tilde{I}_1^N$. The relation between the points on the two unit intervals and the input strategies $\tilde{I}_0^N$ and $\tilde{I}_1^N$ is described more formally in Definition (2.3.2.2).

(2.3.2.2) Definition. Let $N \in \mathbb{N}$ and let $\tilde{X}_i^N$ be the input alphabet of user $i$ for the TWC $\mathcal{K}(N)$. Two mappings $\sigma_0$ and $\sigma_1$:

$$\sigma_i : [0,1) \rightarrow \tilde{X}_i^N \ (i = 0,1),$$

are strategy functions if the set-valued inverse $I_i$ of $\sigma_i$, defined by

$$I_i(\tilde{x}_i^N) := \{ \theta_i \in [0,1) \mid \sigma_i(\theta_i) = \tilde{x}_i^N \},$$

is a subinterval of $[0,1)$ for all $\tilde{x}_i^N \in \tilde{X}_i^N$. We call $I_i(\tilde{x}_i^N)$ a basic interval for user $i$. A region $I_0(\tilde{x}_0^N) \times I_1(\tilde{x}_1^N)$ is called a basic rectangle. $\square$

Now it is not difficult to see that a pair of strategy functions uniquely defines a product distribution on the pair $(\tilde{I}_0^N, \tilde{I}_1^N)$: we have

$$Pr\{\tilde{I}_i = \tilde{x}_i^N\} = Pr\{\sigma_i(\theta_i) = \tilde{x}_i^N\} = Pr\{\theta_i \in I_i(\tilde{x}_i^N)\} = \text{length} (I_i(\tilde{x}_i^N))$$

if $\theta_i$ is distributed uniformly over the interval $[0,1)$. In the same way the area of the basic rectangle $I_0(\tilde{x}_0^N) \times I_1(\tilde{x}_1^N)$ equals $Pr\{\tilde{I}_0^N = \tilde{x}_0^N \text{ and } \tilde{I}_1^N = \tilde{x}_1^N\}$. Therefore the strategy functions describe a strategy distribution and hence an inner bound resolution for $\mathcal{K}(N)$. Conversely, if a strategy distribution $P(\tilde{x}_0^N, \tilde{x}_1^N) = P(\tilde{x}_0^N)P(\tilde{x}_1^N)$ is given, then this corresponds to an inner bound resolution for $\mathcal{K}(N)$; its graphical representation provides a way to find the corresponding strategy functions.

We can rewrite the expression for $\mathcal{G}_i^{(N)} / N$ in Theorem (2.2.6) using the strategy functions. The distribution $P(\tilde{x}_0^N, \tilde{x}_1^N) = P(\tilde{x}_0^N)P(\tilde{x}_1^N)$ needed in Theorem (2.2.6) is derived from the uniform distribution of $(\theta_0, \theta_1)$ on the unit square, via the relation $P(\theta_0, \theta_1) = P(\theta_0)P(\theta_1)$ and using the strategy functions $\sigma_0$ and $\sigma_1$:

$$Pr\{(\tilde{x}_0^N, \tilde{x}_1^N) = (\tilde{x}_0^N, \tilde{x}_1^N)\} = Pr\{\theta_0 \in I_0(\tilde{x}_0^N)\} \cdot Pr\{\theta_1 \in I_1(\tilde{x}_1^N)\}.$$
Let us now define $H(\tilde{T}_N^1|\theta_1)$. Note that $\theta_1$ ranges over $[0,1)$, so we cannot use the definition for discrete variables. For given $\sigma_0$ and $\sigma_1$, let $I_k, k = 1, 2, \ldots, I$, enumerate all possible basic intervals $I_1(\tilde{T}_N^1)$. Then we define

$$H(\tilde{T}_N^1|\theta_1) := \sum_{k=1}^I Pr(\theta_1 \in I_k) \sum_{\tilde{T}_N^1 = \tilde{y}_1^N} Pr(\tilde{T}_N^1 = \tilde{y}_1^N|\theta_1 \in I_k) \log Pr(\tilde{T}_N^1 = \tilde{y}_1^N|\theta_1 \in I_k).$$

With this definition, we find

$$H(\tilde{T}_N^1|\tilde{T}_0^N) = H(\tilde{T}_N^1|\theta_1),$$

which yields the following theorem.

(2.3.2.3) Theorem. We have

$$G_{\tilde{T}_N^1/N} = \text{co } \{ (R_0/N, R_1/N) \mid R_0 \leq H(\tilde{T}_N^1|\theta_1), R_1 \leq H(\tilde{T}_N^1|\theta_0),$$

$$\tilde{T}_N^0 = \tilde{f}_\theta(N)(\tilde{T}_0^N, \tilde{T}_N^1), \tilde{T}_N^0 = \sigma_0(\theta_0), \tilde{T}_N^1 = \sigma_1(\theta_1),$$

$$\theta_0, \theta_1 \text{ uniformly distributed on } [0,1)]$$

where the union is taken over all strategy functions $\sigma_0, \sigma_1$.

Henceforth, we will call the pair $(H(\tilde{T}_N^1|\theta_1)/N, H(\tilde{T}_0^N|\theta_0)/N)$ occurring in Theorem (2.3.2.3) the rate pair of the strategy distribution corresponding to $\sigma_0, \sigma_1$. In Chapter 3, we will use (2.3.2.3) for the computation of points in $G_{\tilde{T}_N^1/N}$ and other regions that can be found using similar graphical representations.

2.4. Simplification of graphical representations

Pictures like Figure (2.3.1.4) tend to become illegible if we draw all thresholds for a $K(N)$ resolution in a unit square, even for small values of $\tilde{N}$. The aim of
Chapter 2

this section is to simplify the figures. In Section 2.3.2, we considered strategies as input letters for $K(N)$ without looking at the separate transmissions for $K$. Here we will take a slightly different viewpoint: we will look at the $N$ transmissions step-by-step. This will enable us to draw the thresholds for $P(z_i^K)$ in $N$ separate unit squares: one for each transmission. Later on in this section we will simplify this even further by splitting the square into separate regions.

As we have seen in Figure (2.3.2.1), the $K^{(2)}$ resolution for Example (2.3.1.3) is already too complicated to draw in one square. Therefore we will take the BMC (Example (2.1.2)) as our example in this section. Figure (2.4.1) shows a $K^{(2)}$ resolution for the BMC. Each row corresponds to one strategy $z_0^2$ and each column corresponds to one strategy $z_1^2$. As an example, the third row from the bottom corresponds to the strategy where user 0 first sends $I_{00} = 1$; if $V_0 = 1$, then he sends $I_{01} = 1$ and if $V_0 = 0$ then he sends $I_{01} = 0$. The numbers in the rectangles represent the two outputs $Y_0, Y_1$ (remember that the BMC is a T-channel, so $Y_0$ and $Y_1$ denote the first and second output for both users).

![Graphical representation of $K^{(2)}$ resolution for the BMC.](image)

(2.4.1) Figure. Graphical representation of $K^{(2)}$ resolution for the BMC.

Figure (2.4.2) shows how the above figure can be decomposed into two separate squares, one for the first transmission and one for the second transmission. Each
number in a rectangle in the top figure represents the output $Y_0$ corresponding to the strategy pair; the numbers in the bottom figure are the second outputs, $Y_1$.

\[
\begin{array}{cccccc}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 \\
\end{array}
\]

\[
\begin{array}{cccccc}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 \\
\end{array}
\]

(2.4.2) Figure. The $\kappa^{(2)}$ resolution of Figure (2.4.1) with separate pictures for each transmission.

A further simplification can be obtained as follows. If the two users each choose a strategy (hence a pair $(\theta_0, \theta_1)$, corresponding to a row and a column) and send the first input according to this strategy, then user $i$ can determine in which part of the square the pair $(\theta_0, \theta_1)$ lies, using $\theta_i$ and $Y_0$. The common knowledge of both users enables them to determine a part of the square of which they both know that $(\theta_0, \theta_1)$ lies in it. In this way, by looking at all strategy
pairs, we can partition the square into such regions. These regions will be called 'connected'. The definition of this is given below.

(2.4.3) Definition. Consider a graphical representation of a $\mathcal{K}(N)$ transmission, with the output sequences $(I_0^m, I_1^m)$ written in each subrectangle. Let $0 \leq n \leq N$. We call two rectangles in one row 0–connected at level $n$ if the same $I_0^n$ (i.e., the first $n$ symbols of $I_0^m$) is written in both rectangles; two rectangles in one column are 1–connected at level $n$ if they have the same $I_1^n$. Two rectangles are connected at level $n$ if they are either 0– or 1–connected at level $n$. The transitive closure of this relation leads to the definition of connected regions at level $n$. 

This definition implies that two rectangles in a row are 0–connected at level $n$ if and only if user 0 cannot determine in which of the two rectangles the pair $(\theta_0, \theta_1)$ lies after the first $n$ transmissions. To fully appreciate the definition of 0– and 1–connectedness, it is best to illustrate it using a non–T–channel; we take Figure (2.3.1.4). The top row of this figure contains three rectangles. The left one is not 0–connected at level 1 to any other, but the other two are 0–connected at level 1 because they both have $y_0^1 = 0$. There are three connected regions at level 1: the rectangle in the middle, the two right–hand rectangles in the bottom row and the remaining part of the square. For the BMC in Figure (2.4.2), we find two connected regions at level 1: the rectangle in the lower right–hand corner corresponding to $y_0^1 = 1$ and the L–shaped region where $y_0^1 = 0$.

With this definition, we now introduce the concept of 'strategy tree'. A strategy tree is a representation of a $\mathcal{K}(N)$ resolution in a tree with depth $N$, where the $(n+1)^{th}$ transmission over $\mathcal{K}$ is found at level $n$ ($0 \leq n \leq N-1$) in the tree. For the first transmission, we draw the thresholds corresponding to the distribution $P(z_0^N), P(z_1^N)$ and we write the first output pairs $(y_0^0, y_1^0)$ (or just $y_0^0$ for a T–
channel) in all rectangles. The square representing this transmission is the root of the strategy tree. Let the first transmission be such that the square is split into \( q \) connected regions at level 1: \( A_1, A_2, \ldots, A_q \) with \( U(A_i) = (0,1)^2 \). The second transmission will then be drawn in \( q \) separate pictures, one for each \( A_i \), and these appear in the tree at level 1 as successors of the root. Each vertex \( A_i \) is cut into subrectangles, in which the second output pair \( (r_{01}, r_{11}) \) (or just \( r_1 \)) is written. Each \( A_i \) is decomposed into connected regions at level 2 and we proceed with the third transmission. Subdividing the square further with decomposition into connected regions at all levels thus results in a tree where each vertex represents a partitioning of the part of the square corresponding to its predecessor. As an example, the tree corresponding to Figures (2.4.1) and (2.4.2) is drawn below.

\[
\begin{array}{ccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 \\
\end{array}
\]

\[
\begin{array}{ccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 \\
0 & 0 \\
0 & 1 \\
0 & 1 \\
\end{array}
\]

\[
\begin{array}{ccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 \\
0 & 0 \\
0 & 1 \\
0 & 1 \\
\end{array}
\]

(2.4.4) Figure. Strategy tree for a \( \kappa^{(2)} \) resolution for the BMC.
Chapter 2

A $\mathcal{K}^{(N)}$ strategy tree has $N$ levels of vertices, where the root has level 0 and the leaves have level $N-1$. Let the two users apply strategies corresponding to a pair $(\Theta_0, \Theta_1)$ and let $n < N$. By definition of connectedness we see that, after the first $n$ transmissions, the transmitted input sequence $I^n_i$ and output sequence $Y^n_i$ received up to that moment enable user $i$ to determine the vertex at level $n$ corresponding to the connected region at level $n$ in which $(\Theta_0, \Theta_1)$ lies. Thus, in the $(n+1)^{th}$ transmission, the users have to look at the partitioning of this vertex only instead of the whole unit square. This observation will be important in Chapter 3, where we are going to compute the rate of a strategy distribution using its strategy tree. As a consequence of these computations, we will also be able to present further simplifications of strategy trees in Chapter 3.

2.5. Partition patterns

In this section we present another way to represent the fact that two rectangles are 0-connected or 1-connected at level 1, if the input alphabets of the channel have at most three elements. From this representation, called the partition pattern of the channel, we obtain an equivalence relation on ternary TWCs. This enables us to count how many 'essentially different' ternary channels there are; by generalization of this argument, we find the number of nonequivalent TWCs over arbitrary alphabets.

Consider a deterministic ternary TWC with $\mathcal{X}_0 = \mathcal{X}_1 = \mathcal{Y}_0 = \mathcal{Y}_1 = \{0, 1, 2\}$; for a moment, we will take Example (2.3.1.3). In Figure (2.5.1), the inner bound resolution for this channel is shown, with all input probabilities equal to $1/3$. The unit square is divided into nine subsquares, some of which are 0-connected or 1-connected at level 1. If two squares in a row are 0-connected, we draw a dashed vertical line
between them (viewed cyclically: if the first and last one are connected, we draw dashed lines to the left and right ends of the row). Between two 1-connected squares, we draw a dashed horizontal line. The configuration of solid and dashed lines in the figure (i.e., the graphical representation of the inner bound resolution without output symbols) is called the partition pattern of the TWC.

\[
\begin{array}{c|c|c}
10 & 01 & 00 \\
\hline
20 & 00 & 20 \\
\hline
10 & 22 & 21 \\
\end{array}
\]

(2.5.1) Figure. Partition pattern of $\kappa$.

Since we are interested in codes, achievable rates and capacity regions for TWCs, it is clear that the actual input and output symbols of a particular channel are not important: there will be many ternary channels that have the same capacity region as Example (2.3.1.3). For instance, if we permute the output alphabets of one or more users, this will not influence the mutual information corresponding to a $\kappa(N)$ resolution: only the conditional entropy of an output symbol is used, not the output symbol itself. It is also possible to change any code for the original channel to a code having the same rate for the channel with permuted alphabets. Because of this, we have the following theorem, first stated in [JAC86].

(2.5.2) Theorem. All ternary deterministic channels having the same partition pattern have the same capacity region.

\[\square\]
Chapter 2

The importance of this theorem follows if we determine the number of TWCs with all alphabets equal to \( \{0,1,2\} \). This number is equal to \( 9^9 \): for every one of the nine input combinations \( (x_0, x_1) \), we can define the outcome \( (y_0, y_1) \) in nine ways. Hence there are more than \( 3 \cdot 10^8 \) ternary TWCs. If we count all TWCs that have the partition pattern of Figure (2.5.1), we find \( (6^6)/2 = 23,328 \) channels with this pattern. Therefore Theorem (2.5.2) gives a significant reduction of the number of channels to be inspected.

However, if we permute the input alphabets and/or permute the user numbers 0 and 1, the characteristics of a channel do not change either (swapping users reflects every rate pair — and hence the capacity region — in the \( x_0 = x_1 \) axis, but we will consider a capacity region and its reflection the same). Therefore we define the following equivalence relation.

(2.5.3) Definition. Two ternary TWCs \( \mathcal{K}_1 \) and \( \mathcal{K}_2 \) are equivalent \( \mathcal{K}_1 \cong \mathcal{K}_2 \) if the partition pattern of \( \mathcal{K}_1 \) can be transformed into the partition pattern of \( \mathcal{K}_2 \) by permutation of rows, columns and/or user numbers. \( \square \)

The following theorem is now immediate.

(2.5.4) Theorem. TWCs that are equivalent under relation \( \cong \) have the same (or reflected) capacity regions. \( \square \)

Using Pólya theory, Jacobs [JAC86] found that the number of equivalence classes of \( \cong \) equals 322. This was a considerable improvement over an earlier result by Gaal [GAA85], who had defined an equivalence relation with 152,176 classes.

The partition pattern approach has the disadvantage that it requires
alphabet size at most three: with four rectangles in a row, 0-connected but non-adjacent rectangles cannot be drawn using dashed and solid lines. Van der Ham [HAM87] generalized the theory developed by Jacobs and was able to compute the number of equivalence classes of TWCs for arbitrary alphabets. Unfortunately he found that the number of classes increases rapidly with the alphabet size: for alphabet size four he found 2,336,169 different classes! For this reason, channels over larger alphabets have not been studied in great detail yet.

Jacobs found that partition patterns could be used to derive bounds on capacity regions. To this aim she introduced 'dominating part patterns', further investigated by Engels [ENG87]. Since very little is known about this, we shall use partition patterns in this thesis as representations of ternary channels only.
Chapter 3: Strategy Trees for Two-Way Channels

3.1. The rate of a tree

In this chapter we look again at the inner bound resolution for the derived channel $\mathcal{X}(N)$, studied in Section 2.3.2. In Theorem (2.3.2.3), we found that the mutual information appearing in the Shannon inner bound expression could be written as \( \frac{H(T_1|t_0)}{N}, \frac{H(T_0|t_0)}{N} \) for given strategy functions $\sigma_0, \sigma_1$. We will introduce the notation $\mathcal{L}(\sigma_0, \sigma_1)$ for this:

\[
\mathcal{L}(\sigma_0, \sigma_1) := \left( \frac{H(T_1|t_0)}{N}, \frac{H(T_0|t_0)}{N} \right).
\]

The rate pair $\mathcal{L}(\sigma_0, \sigma_1)$ can be computed from the graphical representation of the derived channel by using the areas of subrectangles in the square, in exactly the same way as was shown in Section 2.3.1 for $N = 1$. Here we look at the square as a whole, i.e., the simplifications introduced in Section 2.4 are not reflected in the computations.

However, we also study some related notions in this chapter: more general types of partitionings of the unit square that can be derived from trees. For these partitionings we wish to define rate pairs too, because then we have the analogue of Theorem (2.2.6) stating that these rates are achievable and they cover the whole capacity region if the partitionings are sufficiently complex.

The rate of such a tree, say $\mathcal{L}(T)$ if the tree is called $T$, will be defined as the weighted average of the mutual information in each vertex of the tree. See Definition (3.1.3) below. Theorem (3.1.4) states the fact that for the $\mathcal{X}(N)$ inner
bound resolution as described in Section 2.3.2, it makes no difference whether we compute the rate using (3.1.1), or using Definition (3.1.3).

First of all we will define the rate of a vertex in a tree. To this aim we consider a \(K^N\) resolution with strategy functions \(\sigma_0\) and \(\sigma_1\). If \(A\) is a vertex in the corresponding strategy tree at level \(n\), then we define

\[
H(Y^n_{1n}|\theta_1, I^n_1, (\theta_0, \theta_1) \in A) := - \sum_{y_{1n}, \theta_1^n} P_r\{Y_{1n} = y_{1n}, \theta_1^n = \theta_1, I_1^n = y_1^n | (\theta_0, \theta_1) \in A\} \cdot \log P_r\{Y_{1n} = y_{1n} | \theta_1^n = \theta_1, I_1^n = y_1^n, (\theta_0, \theta_1) \in A\},
\]

and \(H(Y^n_{0n}|\theta_0, I^n_0, (\theta_0, \theta_1) \in A)\) is defined analogously.

(3.1.2) Definition. Let \(N \in \mathbb{N}\). Consider an inner bound resolution for the derived channel \(K^N\) with strategy functions \(\sigma_0\) and \(\sigma_1\) and strategy tree \(T\). Let \(A\) be a part of the unit square that occurs as a vertex at level \(n\) in \(T\). The rate of \(A\), \(\tau(A) = (r_0(A), r_1(A))\), is defined by

\[
\tau(A) := (H(Y^n_{1n}|\theta_1, I^n_1, (\theta_0, \theta_1) \in A), H(Y^n_{0n}|\theta_0, I^n_0, (\theta_0, \theta_1) \in A))
\]

where \((\theta_0, \theta_1)\) is uniformly distributed over the unit square and \((I_{0n}^{n+1}, I_{1n}^{n+1})\) depends on \((\theta_0, \theta_1)\) via the functions \(\sigma_0, \sigma_1\) and \(f_K\).

The weight of \(A\), \(w(A)\), is defined as the area of region \(A\), so

\[
w(A) := P_r\{ (\theta_0, \theta_1) \in A \}.
\]

The rate of \(A\) can be understood as the amount of information that is transmitted with the inputs \((I_{0n}, I_{1n})\) if we only look at region \(A\). Hence we consider the case where after the first \(n\) transmissions, the outputs \(y^n_0\) and \(y^n_1\) are such that both users know that connected region \(A\) is the only part of the square in which \((\theta_0, \theta_1)\) can be.
(3.1.3) Definition. Consider a strategy tree for $\mathcal{K}(N)$ as in Definition (3.1.2). Call this tree $T$ and call its vertices $A_n,i$, where $n \in \{0,1,\ldots,N-1\}$ ranges over the levels and $i \in \{0,1,\ldots,m_n-1\}$ ranges over all nodes at level $n$. Here $m_n$ is the number of nodes at level $n$. The rate of the tree $T$, $\mathcal{R}(T)$, is defined by

$$\mathcal{R}(T) = (\mathcal{R}_0(T), \mathcal{R}_1(T)) := \frac{\sum_n \sum_i w(A_n,i) \cdot r(A_n,i)}{\sum_n \sum_i w(A_n,i)}.$$

Now we show that the rate of a $\mathcal{K}(N)$ resolution can be computed by (3.1.3).

(3.1.4) Theorem. Consider a $\mathcal{K}(N)$ strategy distribution with strategy functions $\sigma_0$ and $\sigma_1$ and strategy tree $T$. The following equality holds.

$$\mathcal{R}(\sigma_0,\sigma_1) = \mathcal{R}(T).$$

Proof: As before, let the vertices of $T$ be labeled as $A_n,i$, where $n$ denotes the level of the vertex and $i \in \{0,1,\ldots,m_n-1\}$ ranges over all vertices in level $n$. Then

$$H(Y_1^n|\theta_1) = H((Y_{10}, Y_{11}, \ldots, Y_{1N-1})|\theta_1)$$

$$= \sum_{n=0}^{N-1} H(Y_{1n}|\theta_1, Y_1^n) \quad \text{[by the chain rule]}
$$

$$= \sum_{n=0}^{N-1} H(Y_{1n}|\theta_1, Y_1^n, A_n),$$

if $A_n$ is a random variable with values in $\{A_n,i \mid 0 \leq i < m_n\}$ and $A_n$ assumes the value of the vertex uniquely determined by $\theta_1$ and $Y_1^n$ (which is the same as the vertex determined by $\theta_0$ and $Y_0^n$). Hence $A_n$ is a deterministic function of $\theta_1$ and $Y_1^n$, which explains the last equality in the above derivation. We find

$$H(Y_{1n}|\theta_1, Y_1^n, A_n) = \sum_i P_r\{(\theta_0, \theta_1) \in A_n,i\} \cdot H(Y_{1n}|\theta_1, Y_1^n, (\theta_0, \theta_1) \in A_n,i)$$

$$= \sum_i w(A_n,i) \cdot r_0(A_n,i).$$
and hence
\[ H(\overline{I}_1^N|\theta_1) = \sum_{n=0}^{N-1} \sum_{i=0}^{m_n-1} w(A_{n,i}) \cdot r_0(A_{n,i}). \]

By definition of the strategy tree \( T \), the connected regions occurring in a fixed level of \( T \) partition the unit square. Therefore
\[
\sum_{n=0}^{N-1} \sum_{i=0}^{m_n-1} w(A_{n,i}) = \sum_{n=0}^{N-1} 1 = N.
\]

Finally we have
\[
H(\overline{I}_1^N|\theta_1)/N = \frac{\sum_n \sum_i w(A_{n,i}) \cdot r_0(A_{n,i})}{\sum_n \sum_i w(A_{n,i})},
\]

which is the first component of the tree rate according to Definition (3.1.3). Clearly the second component is computed analogously.

Definition (3.1.3) can be reduced to (3.1.5) in the case of a T–channel.

(3.1.5) \( r(A) = (H(Y_n|\theta_1, (\theta_0, \theta_1) \in A), H(Y_n|\theta_0, (\theta_0, \theta_1) \in A)). \)

This holds because for a T–channel, all subrectangles within one connected region \( A \) at level \( n \) correspond to the same output sequence, say \( y^n(A) \). This follows from Definition (2.4.3). Therefore \((\theta_0, \theta_1) \in A \) implies \( I_0^n = I_1^n = y^n(A) \), so \( y^n \) is not needed in the condition. Formula (3.1.5) is convenient for computing the rate of a T–channel strategy tree.

As an example, consider the strategy tree in Figure (2.4.4). Let us assume that all rows and columns are equally wide, so that each subrectangle in the square has area \((1/6) \cdot (1/6)\). The rate of the root, \( r(A_0,0) \), is given by
\[
r_0(A_0,0) = \sum_{\bar{x}_1,\bar{x}_2} P\{\theta_1 \in I_1(\bar{x}_2)\} \cdot H(Y_0|\theta_1 \in I_1(\bar{x}_2)).
\]
= 2 \cdot (1/6) \cdot 0 + 4 \cdot (1/6) \cdot h(2/3)

and similarly \( r_1(A_0,0) = (2/3) \cdot h(2/3) \). For the left-hand leaf, \( A_1,0 \), we have

\[
\begin{aligned}
&\quad r_0(A_1,0) = \sum \Pr[\theta_1 \in I_1(\xi_1^2)] \cdot \mathcal{H}(Y_0 | \theta_1 \in I_1(\xi_1^2), (\theta_0, \theta_1) \in A_1,0) \\
&= \frac{1}{20/36} \{(6/36) \cdot 0 + (6/36) \cdot h(1/2) + 2 \cdot (2/36) \cdot 0 + 2 \cdot (2/36) \cdot h(1/2)\} \\
&= 1/2
\end{aligned}
\]

and \( r_1(A_1,0) = 1/2 \). Finally, for the right-hand leaf \( A_1,1 \), we find

\[
\begin{aligned}
&\quad r_0(A_1,1) = r_1(A_1,1) = \frac{1}{16/36} \{(2/36) \cdot 0 + 2 \cdot (4/36) \cdot h(1/2)\} = 1/2.
\end{aligned}
\]

From this example we see that the rate of a vertex at level \( n \) only depends on the areas of subrectangles corresponding to a certain output \( Y_n \) within one column or within one row. This means that if we permute certain rows and/or columns within a vertex, this does not change the lengths or widths of subrectangles and hence \( \mathcal{H}(Y_n | \theta_1, (\theta_0, \theta_1) \in A) \) and \( \mathcal{H}(Y_n | \theta_0, (\theta_0, \theta_1) \in A) \) are not changed by permutation (even if not all rows and columns are equally wide!). It also means that, when computing the \( r_0 \)-component of a rate, two 1-connected rectangles do not have to distinguished: we only need the total area of both rectangles and not the two separate areas. Therefore we can draw two neighboring 1-connected rectangles as one big rectangle by deleting the part of the horizontal threshold that separates them. Of course we can also delete vertical thresholds between 0-connected rectangles (therefore, in the case of a T-channel, the term 'connected regions' has a very practical meaning: if there are no borders between two rectangles, then they are connected).

In other words, if we permute rows and columns in the vertices of Figure (2.4.4) and delete borders between connected rectangles such that Figure (3.1.6)
results, the rate of each vertex in Figure (3.1.6) is still the same as each rate in Figure (2.4.4). The 'permuted' thresholds in Figure (3.1.6) have a useful interpretation. The horizontal thresholds occurring in a vertex $A$ at level $n$ represent the probability distribution of $I_{0n}$ conditional to the previous inputs and outputs of user 0. Similarly the vertical thresholds depict the distribution of $I_{1n}$ given $I_1^n$ and $I_1^0$. This will be explained further below Figure (3.1.7).

![Strategy Trees for TWCs](image)

(3.1.6) Figure. Simplified strategy tree for a $k^{(2)}$ resolution for the BMC.

It should be noted that the above observations arising from computations for the BMC only hold for T–channels. For a channel like Example (2.3.1.3), we are not allowed to permute complete columns in a vertex, since we are interested in $H(Y_{1n}, I_1^n, \theta_0, \theta_1) \in A_{n, i}$ instead of $H(Y_{1n}, I_1^n, \theta_0, \theta_1) \in A_{n, i}$. Hence
we may only permute parts of columns corresponding to the same \( I_{11} \), i.e., the parts of columns lying in 1-connected rectangles. Similarly we may permute rows only within 0-connected rectangles. As an example, the part of the square corresponding to \( I_{00}=2 \) as depicted in Figure (2.3.2.1) may be transformed to Figure (3.1.7) (assuming that the total lengths and widths of the rectangles are the same in both figures). The reader should convince himself that this does not influence the rate. It goes without saying that a picture like Figure (3.1.7) is considerably more comprehensible than Figure (2.3.2.1).

\[
\begin{array}{ccc}
I_{10} = 0 & I_{10} = 1 & I_{10} = 2 \\
I_{01} = 0 & & I_{01} = 0 \\
I_{01} = 1 & & I_{01} = 1 \\
I_{01} = 2 & & I_{01} = 2 \\
I_{00} = 1 & & I_{00} = 2 \\
\end{array}
\]

(3.1.7) Figure. Simplification of Figure (2.3.2.1).

In Figure (3.1.7), the permuted thresholds in each of the connected rectangles again represent distributions \( P(x_{01} \mid x_{00}, y_{00}) \). If we normalize the thresholds to the unit interval (i.e., consider user 0’s interval corresponding to \( I_{00}=2 \) as having size 1), the thresholds for the left-hand rectangle represent \( P(x_{01} \mid I_{00}=2, y_{00}=1) \) and the thresholds for the right-hand pair of rectangles depict \( P(x_{01} \mid I_{00}=2, y_{00}=2) \). Note that there is just one probability distribution for the two 0-connected rectangles on the right, since both of these rectangles correspond to \( I_{00}=2 \) and \( y_{00}=2 \). In the figure, this is reflected in the fact that the horizontal lines run over the entire area of these two rectangles.
This simplification makes it possible to draw strategy trees also for non-T-channels, if $N$ is not too large. Figure (3.1.8) illustrates this: it shows the complete strategy tree for a $\kappa^2$ resolution over the channel of Example (2.3.1.3). The detail shown in Figure (3.1.7) appears in two separate vertices of the tree: the left-hand rectangle occurs in the bottom part of the leaf on the left and the two 0-connected rectangles form the leaf on the right.

(3.1.8) Figure. Strategy tree for the channel of Example (2.3.1.3).

In the remaining part of this chapter, we will draw all strategy trees using the simplifications described in Section 2.4 and in this section.
Chapter 3

3.2. Fixed length trees

3.2.1. Fixed length trees and the capacity region

Here we will introduce a new notation for the inner bound resolution of the derived channel $k(N)$: we will call it an $F(N)$ tree. In this way the analogy between fixed and variable length trees (that will be studied in Section 3.3) will be stressed, since the 'F' in this notation is an abbreviation for 'fixed length' and the trees in Section 3.3 will be denoted as 'V(N) trees' where the 'V' means 'variable length'.

For an arbitrary tree, we define the distance between two vertices on a path from the root to a leaf as usual; we define the depth of a tree as the maximum distance between any vertex and the root. Hence a strategy tree for a $k(N)$ resolution has depth $N-1$.

(3.2.1.1) Definition. Let $N \in \mathbb{N}$. An $F(N)$ tree for $k$ is a strategy tree for an inner bound resolution for $k(N)$. Furthermore we define

$$F(N) := \{ \mathcal{A}(T) \mid T \text{ is an } F(N) \text{ tree for channel } k \},$$

$$F := \bigcup_{N \in \mathbb{N}} F(N).$$

By (3.1.1) and Theorem (3.1.4), we can now state Theorem (2.2.6) as

$$\forall N \in \mathbb{N} \left[ \text{co } F(N) \subset C \right] \quad \text{and} \quad \lim_{N \to \infty} \text{co } F(N) = C.$$  

In Theorem (3.2.1.2) this is formulated differently, namely in such a form that it can be generalized for variable length (and other) trees. For an arbitrary region $A$, let $\overline{A}$ denote the closure of $A$. We can state the following.
Strategy Trees for TWCs

(3.2.1.2) Theorem. We have
\[ \forall N \in \mathbb{N} \left[ \text{co } \mathcal{F}(N) \subset C \right] \quad \text{and} \quad \lim_{N \to \infty} \mathcal{F}(N) = \overline{\mathcal{F}} = C. \]

Proof: see Section 3.5.

The reason why we postpone the proof to Section 3.5 is the following. The proof of Theorem (3.2.1.2) is similar to the proofs of the generalized theorems in the following sections; we can use this theorem and a part of its proof to prove some of the other theorems. It seems more natural to give all of the proofs together in one section because then the relation between them is more evident.

3.2.2. Fixed length trees ending in rootlike rectangles

In this section we will study a special type of fixed length tree, namely one in which all connected regions that are not partitioned any further (that is, the resulting regions of the splitting in the leaves) must be so-called rootlike rectangles, defined below.

(3.2.2.1) Definition. Consider a strategy tree of depth \( N-1 \). For \( 0 \leq n \leq N \), a rootlike rectangle at level \( n \) is a connected region \( A \) at level \( n \) that occurs in the tree as a result of the partitioning of the square after \( n \) transmissions such that the following holds for \( A \).

\begin{enumerate}
  \item[(i)] The region \( A \) is a rectangle; that is, a uniform distribution of \((\theta_0, \theta_1)\) over \( A \) implies independence between \( \theta_0 \) and \( \theta_1 \).
  \item[(ii)] For \( i \in \{0,1\} \): only one output sequence \( x_i^n \) for user \( i \) corresponds to region \( A \). Hence if \((\theta_0, \theta_1) \in A\), \( y_0^n \) and \( y_1^n \) do not depend on the positions (or basic intervals) of \( \theta_0 \) and \( \theta_1 \) within \( A \).
\end{enumerate}
Let us take a more detailed look at rootlike rectangles. Since we are considering 'full' trees only (i.e., trees in which every part of the square is partitioned at every level), all rootlike rectangles at level \( n \) for \( n < N \) are vertices at level \( n \), and all rootlike rectangles at level \( N \) are regions remaining after splitting in the leaves. Note that for a T-channel, condition (i.i) is void. This follows from the discussion in Section 2.4. In that case, any rectangular connected region is a rootlike rectangle. Also note that if vertex \( A \) is a rootlike rectangle at level \( n \), then \( \mathcal{A}(A) = (H(Y_{1n}|\theta_1, (\theta_0, \theta_1) \in \mathcal{A}), H(Y_{0n}|\theta_0, (\theta_0, \theta_1) \in \mathcal{A})) \), because previous outputs add no knowledge about \( Y_{1n} \) and \( Y_{0n} \) if the users know that \( (\theta_0, \theta_1) \in \mathcal{A} \).

Now consider an \( F(N) \) tree. The root of the tree is a rootlike rectangle at level 0, since it is a square and no previous outputs exist. If we compute the rate of this vertex, \( (H(Y_{10}|\theta_1),H(Y_{00}|\theta_0)) \), then we use the uniform distribution of \( (\theta_0, \theta_1) \) over the square. Therefore \( \theta_0 \) and \( \theta_1 \) are independent and the first input symbols of both users (that follow from the strategy functions) are independently chosen. In the next transmissions, regions that may have arbitrary shapes are subdivided in the tree. The subsequent inputs of the two users will be dependent since they have knowledge about previous inputs and outputs. Only if region \( A \) is again a rootlike rectangle, at level \( n \), say, then (i) and (i.i) imply independence between the new inputs \( X_{0n} \) and \( X_{1n} \) even though the two users share knowledge about previous outputs. Hence in that case the rate is \( (H(Y_{1n}|\theta_1, (\theta_0, \theta_1) \in \mathcal{A}), H(Y_{0n}|\theta_0, (\theta_0, \theta_1) \in \mathcal{A})) \) with independent \( \theta_0 \) and \( \theta_1 \), as if it were the root of the tree. This explains the name 'rootlike rectangle'.

If a rootlike rectangle \( A \) at level \( N \) occurs in the \( F(N) \) tree, we can use \( A \) as the root of a new tree. If the new tree has \( k \) levels, then its root occurs at level \( N \) of the old tree, which changes into an \( F(N+k) \) tree in this way. If we use the \( F(N) \) tree itself for the new tree (\( k = N \)) and repeat the procedure \( t \) times, then \( t \) copies of the \( F(N) \) tree form an \( F(tN) \) tree.
What if the regions remaining after the partitioning in the leaves of the 
\( F(N) \) tree are not rootlike rectangles? Let \( A \) be such a region consisting of several basic rectangles, corresponding to input pairs \( (\bar{z}_0^N, \bar{z}_1^N) \) of the \( F(N) \) tree. Let \( t \in \mathbb{N} \). Now \( A \) may be subdivided using an \( F(t) \) tree with strategy distribution 
\[
P(\bar{z}_0^t, \bar{z}_1^t) = P(\bar{z}_0^t) \cdot P(\bar{z}_1^t),
\]
if the users 'forget' about the dependency in \( A \). This new strategy distribution can be represented in \( A \) by choosing the same set of horizontal thresholds, corresponding to \( P(\bar{z}_0^t) \), for every basic interval \( I_0(\bar{z}_0^N) \) (i.e., every row) in \( A \), and the same set of vertical thresholds \( (P(\bar{z}_1^t)) \) in every column of \( A \). In this way the \( F(t) \) tree is 'appended' below the original tree.

It is easy to see that the rates of the new vertices in the \( F(N+t) \) tree are equal to those in the \( F(t) \) tree itself. If we take the same \( F(N) \) tree \( t \) times we obtain an \( F(tN) \) tree again. Moreover, the \( F(tN) \) tree has the same rate as the \( F(N) \) tree, since the weighted average of the rates in the vertices does not change. In fact, the technique described here is exactly the same as used by Shannon [SHA61] to prove that \( G_i^{(N)} / N \) is included in \( G_i^{(tN)} / (tN) \) for all \( t \).

Now that we know the interpretation and the use of rootlike rectangles, we define the following (compare Definition (3.2.1.1)). The 'c' in the notation can be seen as the root of a tree (the unit square), which denotes the fact that we restrict ourselves to rootlike rectangles.

(3.2.2.2) Definition. Let \( N \in \mathbb{N} \). An \( F_c(N) \) tree is an \( F(N) \) tree such that every connected region in the tree that is not subdivided any further (i.e., every region remaining after the splitting in the leaves) is a rootlike rectangle. Define
\[
F_c(N) := \{ \Delta(T) \mid T \text{ is an } F_c(N) \text{ tree} \},
\]
\[
\mathcal{F}_c := \bigcup_{N \in \mathbb{N}} F_c(N).
\]
The theorem below states the analog of Theorem (3.2.1.2): $\mathcal{F}^0(N)$ converges to the capacity region. As before, we postpone the proof to Section 3.5.

(3.2.2.3) Theorem. We have
\[
\forall N \in \mathbb{N} \left[ \text{co} \mathcal{F}^0(N) \subset C \right] \quad \text{and} \quad \lim_{N \to \infty} \mathcal{F}^0(N) = \mathcal{F}^0 = C.
\]

Proof: see Section 3.5.

As can be seen from Figures (3.1.6) and (3.1.8), most $\mathcal{F}(N)$ trees are not $\mathcal{F}^0(N)$ trees: usually one or more non-rectangular regions will remain in the leaves. If such non-rectangular regions are not allowed, then this poses restrictions on the transmission in the leaves. Therefore it is likely that the rates in the leaves are lower than in the case of an arbitrary $\mathcal{F}(N)$ tree. Because of this restriction, $\mathcal{F}^0(N)$ trees are not very useful. However, if we look at the same restriction for variable length trees (to be studied in Section 3.3.2), it turns out to be quite useful there. In the following section we will consider an even stronger restriction for an $\mathcal{F}(N)$ tree, having even lower rate; still it is an interesting kind of tree since we can easily derive codes from it.

3.2.3. From fixed length trees to codes and vice versa

This section is devoted to a special kind of $\mathcal{F}(N)$ tree, namely one corresponding to a code (in the sense of Definition (2.1.6)). First of all, we will show that any code gives rise to an $\mathcal{F}(N)$ tree. Let $\mathcal{M}_0, \mathcal{M}_1 \in \mathbb{N}$, $N \in \mathbb{N}$ and consider an $(\mathcal{M}_0, \mathcal{M}_1, N)$ code for a deterministic TWC $\mathcal{K}$ defined by function $f_\mathcal{K}$. The message sets are $\mathcal{M}_0$ and $\mathcal{M}_1$ and the encoding functions are called $f_{i,n}$ ($i \in \{0,1\}$, $n \in \{0,1,\ldots,N-1\}$). Using the code, we will construct an $\mathcal{F}(N)$ tree that partitions the unit square. We partition user 0's unit interval into $\mathcal{M}_0$.
Strategy Trees for TWCs

intervals of length $1/N_0$. These intervals are labeled $0, 1, \ldots, N_0 - 1$ such that each interval corresponds to a message $w_0 \in N_0$. In the same way user 1’s unit interval is divided into $N_1$ intervals, indexed with the messages in $N_1$. Hence each of the $N_0 N_1$ rectangles in the square corresponds to a message pair $(w_0, w_1)$.

The strategy functions $\sigma_0$ and $\sigma_1$ will follow from the code. User 0’s subinterval corresponding to message $w_0$ will be a basic interval $I_0(z_{00}^N)$ where the strategy $z_{00}^N$ will follow from the encoding functions $f_{0, n}$ for message $w_0$. Similarly each subinterval of user 1 will be linked — via $w_1$ and the encoding functions $f_{1, n}$ — to a strategy $z_{11}^N$. The independence between the messages of the two users yields a product distribution on $(z_{00}^N, z_{11}^N)$. Let $z_{00}^N$ be the strategy corresponding to $w_0$. The first symbol $x_{00}$ in the strategy $z_{00}^N$ is $f_{00}(w_0)$. In the same way the first letter $x_{10}$ of user 1’s strategy $z_{11}^N$ is determined via $f_{10}$. Hence the output pair $(y_{00}, y_{10})$ for the rectangle corresponding to $(w_0, w_1)$ is

$$(y_{00}, y_{10}) = f_{11}(f_{00}(w_0), f_{10}(w_1)).$$

This explains the partitioning in the root of the tree. The second layer of the tree is partitioned using functions $f_{01}$ and $f_{11}$: if a row corresponds to $w_0$ and user 0’s first output is $y_{00}$, then the input $x_{01}$ for this part of the square is $f_{01}(w_0, y_{00})$.

We can construct a complete $F(N)$ tree like this. Note that the horizontal thresholds in the tree are restricted to coordinates $(i/N_0)$ ($i \in \{0, 1, \ldots, N_0\}$) and the vertical thresholds are multiples of $1/N_1$. We say that the thresholds are on an $(N_0 + 1) \times (N_1 + 1)$ grid. Figure (3.2.3.1) shows an $F(3)$ tree for the BMC that corresponds to a $(3, 3, 3)$ code. We explain the mapping from rows and columns to strategies in this figure by an example. The top row of user 0 corresponds to the following strategy: first send $z_{00} = 0$, such that $y_{00} = 0$ with certainty (root). Then send $z_{01} = 1$ (left-hand vertex at level 1). If $y_{01} = 1$, then send $x_{02} = 1$ (left-hand leaf below this vertex); otherwise send $x_{02} = 0$ (right-hand leaf). This
example also shows how a tree with thresholds on a grid can be interpreted as a code.

\begin{figure}
\centering
\includegraphics[width=0.8\textwidth]{tree.png}
\caption{F(3) tree for the BMC representing a code.}
\end{figure}

(3.2.3.1) Figure. $F(3)$ tree for the BMC representing a code.

It should be remarked that the code represented in Figure (3.2.3.1) has error probability $P_{e,0} = P_{e,1} = 0$, for the appropriate decoding functions: every basic rectangle corresponding to a message pair is a connected region at level 3, and both users can uniquely determine this connected region by looking at their own message and the output sequence of length 3. We also see that the $F(3)$ tree is in fact an $F^0(3)$ tree, since all connected regions at level 3 are squares.

This is true for all codes: any $(M_0, M_1, N)$ code can be represented in an $F(N)$ tree and any zero-error code gives rise to an $F^0(N)$ tree. Conversely, if we have an $F(N)$ tree such that all thresholds are confined to an $(M_0+1) \times (M_1+1)$ grid, then the encoding functions for an $(M_0, M_1, N)$ code can be derived from this tree. Moreover, if every basic rectangle of size $(1/M_0) \times (1/M_1)$ forms a connected region at level $N$ in the tree, then this yields an error-free decoder for the code: each user can determine the connected region at level $N$, and hence the message pair, using his own message and the output sequence. A tree in which such an error-free code can be represented will be called an $F^0(N)$ tree, where the '0'
stands for error probability 0. We also define $T^0(N)$ and $T^0$, as in Definitions (3.2.1.1) and (3.2.2.2). A theorem like (3.2.1.2) and (3.2.2.3) cannot be given for $T^0(N)$ and $T^0$, but in Section 3.5 we present a slightly different theorem about these regions.

(3.2.3.2) Definition. Let $N \in \mathbb{N}$. An $T^0(N)$ tree is an $T^0$ tree on a grid such that all basic rectangles have the same measurements and every basic rectangle is a separate connected region at level $N$. Furthermore

$$T^0(N) := \{ \mathcal{A}(T) \mid T \text{ is an } T^0(N) \text{ tree} \},$$

$$T^0 := \bigcup_{N \in \mathbb{N}} T^0(N).$$

We will show that the rate of a code (Definition (2.1.8)) that is constructed from a tree can be computed via the rate of the tree (Definition (3.1.3)). This is similar to Theorem (3.1.4), where we showed that the rate of a $k(N)$ strategy distribution is equal to the tree rate.

(3.2.3.3) Theorem. Consider an $T^0(N)$ tree $T$ representing an $(N_0, N_1, N)$ code. The rate of the code is equal to $\mathcal{A}(T)$.

Proof: By definition, the code rate is $(\log(N_0)/N, \log(N_1)/N)$. From Theorem (3.1.4) we know that we can compute the tree rate $\mathcal{A}(T)$ either per vertex, or in the unit square as a whole. Here we use the latter method:

$$\mathcal{H}(\tilde{T}^N_1|\theta_1) = \sum_{i = 0, \ldots, N_1-1} \Pr\left\{ \theta_1 \in \left[ \frac{i}{N_1}, \frac{i+1}{N_1} \right] \right\} \cdot \mathcal{H}(\tilde{T}^N_1|\theta_1 \in \left[ \frac{i}{N_1}, \frac{i+1}{N_1} \right]).$$

$$\mathcal{H}(\tilde{T}^N_1|\theta_1) = \sum_{i = 0, \ldots, N_1-1} \frac{1}{N_1} \cdot \mathcal{H}(\tilde{T}^N_1|\theta_1 \in \left[ \frac{i}{N_1}, \frac{i+1}{N_1} \right]).$$

Since the error probability is 0, user 1 is able to determine the basic rectangle by
Chapter 9

the output sequence $X^N$ and by his own message (hence by the column in the square). Hence the $X^N_0$ output sequences $X^N_i$ occurring within one column must be distinct. Since user 0's messages are equally likely, the same holds for the $X^N_1$ sequences within a column. Therefore the entropy of the output sequence in every column is equal to $\log(N_0)$. We find

$$R_0 = H(X^N_0)/(N) = \sum_{i=0}^{N_1-1} \frac{1}{N_1} \log(N_0)/(N) = \log(N_0)/(N),$$

which is the $R_0$ component of the code rate. Because a similar relation holds for the $R_1$ component, this proves the equality of the tree rate and the code rate.

It is interesting to compare this result to Theorem (2.2.6). Shannon has shown that the rate of every $F(N)$ tree is achievable, hence that codes exist having (almost) the same rate as the tree and having arbitrarily small error probability. In this section we have shown the achievability of the rate of a special type of $F(N)$ tree, namely an $F^0(N)$ tree. Not only have we proved, in Theorem (3.2.3.3), the existence of a code with this rate; we have also shown how a such a code can be constructed from the tree. This code even has error probability equal to 0, instead of just tending to 0 with increasing block length. For more examples of zero-error codes and with higher rate, we refer to Section 3.6.

3.3. Variable length trees

3.3.1. Variable length trees and the capacity region

Here we will study one of the generalizations of $F(N)$ trees. If we compute the rate of an $F(N)$ tree, then it may happen that some of the vertices have very
low \(^1\) rates, lower than the (weighted) average rate of the whole tree. Clearly it is in our interest to construct trees with high rates, since these give good lower bounds on the capacity region of the TWC. Therefore it may be advantageous to delete a subtree having a low rate from the \(P(N)\) tree, thereby increasing the rate of the tree in total. This is why we introduce the following concept.

(3.3.1.1) Definition. Let \( N \in \mathbb{N} \). A variable length tree of maximum length \( N \), abbreviated as a \( V(N) \) tree, is a tree that can be obtained from an \( P(N) \) tree by deleting zero or more vertices and the complete subtrees having these vertices as roots in such a way that the \( V(N) \) tree still has depth \( N-1 \).

A variable length tree can be seen as a 'pruned' \( P(N) \) tree. Note that each \( P(N) \) tree is also a \( V(N) \) tree. The definitions for fixed length trees that have been introduced in the previous sections generalize to variable length trees in a natural way. Hence the rate of a \( V(N) \) tree again equals the weighted average of the rates in all vertices.

Rootlike rectangles may also be found in a \( V(N) \) tree; however, as opposed to \( P(N) \) trees, a \( V(N) \) tree may contain a rootlike rectangle at level \( n \), \( n < N \), that is not a vertex. For an example of this, we refer to Figure (3.3.1.2), where the rectangle corresponding to \( r_0 = 1 \) in the root of the \( V(2) \) tree is a rootlike rectangle at level 1 but not a vertex.

\(^1\) We must remark here that the words 'low' and 'high' should not be interpreted too literally, since we are talking about 2-dimensional vectors. A 'high' rate can be seen as having a small Euclidean distance to the border of the actual capacity region, and a 'low' rate is far from the border of \( C \). It may be helpful to think of the sum rate, \( r_0 + r_1 \), as a measure for 'high' or 'low' rates.
Figure (3.3.1.2) illustrates the difference between fixed and variable length trees: it shows an $F(2)$ tree and a $V(2)$ tree for the BMC. All thresholds are in the middle of the intervals, i.e., the next input symbol is 1 with probability 0.5 in vertices $\lambda_{0,0}$ and $\lambda_{1,1}$. We compute the rates of the trees by (3.1.3).

\[
\begin{align*}
\lambda_{0,0} & \rightarrow \lambda_{1,0} \\
1 & \rightarrow 0 \\
0 & \rightarrow 1 
\end{align*}
\]

(Fixed tree) $F(2)$

\[
\begin{align*}
\lambda_{0,0} & \rightarrow 1 \\
0 & \rightarrow \lambda_{1,0} \\
1 & \rightarrow \lambda_{1,1} \\
0 & \rightarrow 0 
\end{align*}
\]

(Variable tree) $V(2)$

\[
\begin{align*}
\lambda_{0,0} & \rightarrow 1, & r(\lambda_{0,0}) & = (1/2, 1/2), \\
\lambda_{0,1} & \rightarrow 3/4, & r(\lambda_{0,1}) & = (2/3, 2/3), \\
\lambda_{1,1} & \rightarrow 1/4, & r(\lambda_{1,1}) & = (1/2, 1/2). \\
\end{align*}
\]

$F(2)$ tree $F(2)$ tree

$V(2)$ tree $V(2)$ tree

Since $9/16 = 0.56250$ and $4/7 \approx 0.57143$, it follows from this example that a $V(N)$ tree can indeed yield a better rate than an $F(N)$ tree. In Section 3.2.1 we found that the set of all rates of $F(N)$ trees tends to the capacity region of the channel. Now we compare this to the set of all rates of $V(N)$ trees.
(3.3.1.3) Definition. For \( N \in \mathbb{N} \), we define
\[
\mathcal{V}(N) := \{ \mathcal{L}(T) \mid T \text{ is a } \mathcal{F}(N) \text{ tree} \},
\]
\[
\mathcal{V} := \bigcup_{N \in \mathbb{N}} \mathcal{V}(N).
\]

The following theorem is a generalization of (3.2.1.2).

(3.3.1.4) Theorem. We have
\[
\forall N \in \mathbb{N} \left[ \mathcal{C} \circ \mathcal{V}(N) \subseteq \mathcal{C} \right] \quad \text{and} \quad \lim_{N \to \infty} \mathcal{V}(N) = \mathcal{V} = \mathcal{C}.
\]

Proof: see Section 3.5.

Theorem (3.3.1.4) is useful for finding inner bounds on \( \mathcal{C} \) that are better than \( \mathcal{F}(N) \), since the class of \( \mathcal{V}(N) \) trees strictly includes the class of \( \mathcal{F}(N) \) trees and therefore \( \mathcal{V}(N) \) generally strictly includes \( \mathcal{F}(N) \). This follows from the examples in Section 3.6, which show that, for comparatively small \( N \), the optimum \( \mathcal{V}(N) \) rate can exceed the optimum \( \mathcal{F}(N) \) rate. On the other hand, finding an outer bound on \( \mathcal{C} \) via outer bounds on \( \mathcal{V}(N) \) for \( N \to \infty \) is much harder. It is unknown how this should be done for \( \mathcal{F}(N) \); yet if this were possible, it would be even more difficult for \( \mathcal{V}(N) \) trees, since these have more 'degrees of freedom' than \( \mathcal{F}(N) \) trees.

3.3.2. Variable length trees ending in rootlike rectangles

In this section we consider the same restriction on trees as in Section 3.2.2, but now for \( \mathcal{V}(N) \) trees. Hence we require again that rootlike rectangles remain as connected regions after the unit square has been subdivided in a \( \mathcal{V}(N) \) tree. Note, however, that these connected regions do not have to occur in the leaves only, as remarked below Definition (3.3.1.1). An example of such a tree is the \( \mathcal{V}(2) \) tree in Figure (3.3.1.2). The following is an obvious generalization of Definition (3.2.2.2).
Chapter 3

(3.3.2.1) Definition. For $N \in \mathbb{N}$, a $\mathcal{V}^D(N)$ tree is a $\mathcal{V}(N)$ tree such that all connected regions not subdivided any further in the tree are rootlike rectangles. Furthermore

$$\mathcal{V}^D(N) := \{ R(T) \mid T \text{ is a } \mathcal{V}^D(N) \text{ tree } \},$$

$$\mathcal{V}^D := \bigcup_{N \in \mathbb{N}} \mathcal{V}^D(N).$$

As we will prove in Section 3.5, the regions $\mathcal{V}^D(N)$ also converge to the capacity region. This is stated in the following theorem.

(3.3.2.2) Theorem. We have

$$\forall N \in \mathbb{N} \left[ \text{co } \mathcal{V}^D(N) \subset C \right] \text{ and } \lim_{N \to \infty} \mathcal{V}^D(N) = \mathcal{V}^D = C.$$  

Proof: see Section 3.5.

It is easy to show that every $\mathcal{V}(N)$ tree can be changed into a $\mathcal{V}^D(N')$ tree for some $N' \geq N$ by adding some extra vertices to the tree. As is explained in the following lemma, the tree can be extended in such a way that the extra vertices have a property that will be needed in Section 3.4.

(3.3.2.3) Lemma. Consider a $\mathcal{V}(N)$ tree for some $N \in \mathbb{N}$. This tree can be extended to a $\mathcal{V}^D(N')$ tree for some integer $N' \geq N$ by adding vertices in which all remaining connected regions are subdivided into rootlike rectangles. This can be done in such a way that in each added vertex at level $n$, user 1 can determine $I_{0n}$ from $I_{1n}$ and vice versa; hence the rate of such a vertex $A$ is

$$(H(I_{0n} | \theta_1, I_{1n}^0, \theta_1, \theta_1) \in A), H(I_{1n} | \theta_0, I_{0n}^1, \theta_0, \theta_1) \in A).$$

Proof: Let $A$ be a connected region in the $\mathcal{V}(N)$ tree that is not partitioned and is not a rootlike rectangle (e.g., the L-shape in vertex $A_{1,0}$ of Figure (3.3.1.2)). $A$ consists of some basic rectangles $I_0^N(z_0^N) \times I_1^N(z_1^N)$ for various $z_0^N$ and $z_1^N$. Assume
there are at least two basic intervals for user 0 that contribute to \( A \) (note that if \( A \) contains just one basic interval for each user, then \( A \) is a rootlike rectangle). Partition the union of all basic intervals for user 0 that correspond to \( A \) into two nonempty sets, say \( I_{0a} \subset [0,1) \) and \( I_{0b} \subset [0,1) \). Since user 0 can transmit information to user 1 at positive rate (by the assumption in Section 2.3.1), two different input letters for user 0, say \( z_0(a) \) and \( z_0(b) \), and an input \( z_1 \) for user 1 exist such that the input pairs \((z_0(a), z_1)\) and \((z_0(b), z_1)\) result in distinct outputs \( y_1 \), say \( y_1(a) \) and \( y_1(b) \) (otherwise user 1 can never distinguish between user 0's inputs). The next transmission in vertex \( A \) is the following: user 0 sends \( z_0(a) \) for \( \theta_0 \in I_{0a} \) and \( z_0(b) \) for \( \theta_0 \in I_{0b} \), user 1 sends \( z_1 \) always. After this transmission, user 1 sees different outputs for the two regions. Now user 0's basic intervals can again be partitioned into two parts for both \( I_{0a} \) and \( I_{0b} \). We repeat this procedure, transmitting the same two input pairs every time, until user 1 sees a different output sequence for each basic interval of user 0. Then the basic rectangles in \( A \) are not 1-connected anymore (Definition (2.4.3)).

After this, we do a similar thing to partition user 1's interval into basic intervals, which is possible since the rate of transmission from user 1 to user 0 is positive too. After a number of transmissions, the basic rectangles in \( A \) are not 0-connected anymore. Hence, after all of these transmissions each basic rectangle is a separate connected region. Clearly there is just one \( Y_0 \) - and one \( Y_1 \) - sequence that corresponds to such a rectangle; in other words, \( A \) has been partitioned into rootlike rectangles. Since we can do this for every region \( A \), a \( P^D(N') \) tree can be constructed. It is obvious that all 'new' vertices in this tree have the property mentioned in the lemma.

\[ \square \]

The technique used in the proof of Lemma (3.3.2.3) will be called slicing. The transmission of symbols to tell the difference between user 0's intervals (i.e.,
the rows in the region) is called row slicing; the partitioning of columns is called column slicing. We will refer to these notions in the sequel.

3.3.3. From variable length trees to fixed length codes

In this section, we present a generalization of Tolhuizen's result [TOL85]. This will show how a code can be constructed from a $V^D(N)$ tree such that the rate of the code approximates the rate of the tree. Let $T$ be a $V^D(N)$ tree. As in Section (3.2.2), the rootlike rectangles that are not subdivided in $T$ can serve as roots of new trees. In this way we can build a $V^D(tN)$ tree out of $t$ copies of $T$ (note that the new tree is indeed a $V^D(tN)$ tree: the connected regions that remain after partitioning are rootlike rectangles again. The same holds for fixed length trees, where $t$ $F^D(N)$ trees form an $F^D(tN)$ tree). Using this, Tolhuizen [TOL85] proved that a $V^D(N)$ tree for the BMC can be used to construct codes having arbitrarily small error probability, such that the code rate approximates the tree rate. Here we generalize this to arbitrary deterministic TWCs. Moreover, we sharpen Tolhuizen's error probability bound because we need this in Section 3.4.

(3.3.3.1) Theorem. Let $N \in \mathbb{N}$ and let $T$ be a $V^D(N)$ tree. Then
\[ \forall \varepsilon > 0 \exists \delta > 0, 0 < \lambda < 1, \exists T_0 \in \mathbb{N} \forall t \geq T_0 \exists C_t \text{ exists with rate } R(C_t) \text{ and} \]
error probabilities $P_{e,i}$ satisfying $H_i(C_t) \geq H_i(T) - \varepsilon$, $P_{e,i} \leq \delta \cdot \lambda^i (i = 0, 1)$. \[ \] Proof: For $t \in \mathbb{N}$, consider the $V^D(tN)$ tree that is constructed by copying the original $V^D(N)$ tree in every rootlike rectangle remaining in the $V^D(N)$ tree and repeating this $t$ times. Let $S_t$ be the set of all rootlike rectangles $[a,b) \times [c,d)$ that remain unpartitioned in the $V^D(tN)$ tree. Then $S_t$ forms a partitioning of the square. Let $\varepsilon > 0$ and $\delta > 0$. Define the set of typical rectangles $r$ as follows:
\[ T(t, \delta) := \{ r \in S_t \mid |n_t(r) - tN| \leq t\delta, |b_t(r) - tB| \leq t\delta, |h_t(r) - tH| \leq t\delta \}. \]
Strategy Trees for TWCs

Here
\[ n_t(r) := n \quad \text{if } r \text{ is a rootlike rectangle at level } n \text{ in the tree}, \]
\[ b_t(r) := -\log(\text{breadth}(r)), \]
\[ h_t(r) := -\log(\text{height}(r)), \]
and \( \bar{N}, \bar{B} \) and \( \bar{H} \) are the means of \( n_1(r'), b_1(r') \) and \( h_1(r') \), respectively, averaged over \( r' \in S_1 \). Notice that in [TOL85], the typical rectangles are defined with \( |\cdots| \leq t^{3/4} \) instead of \( t \delta \). With our definition, we can use the exponential convergence in the weak law of large numbers as it is stated in [ASH65, Lemma 3.6.3, Theorem 3.6.4 and remark in Section 3.8]:

forall \( t \) independently identically distributed random variables \( X_1, X_2, \ldots, X_t \),
\[ \mathbb{P}\{ \sum_{i=1}^{t} X_i - t \cdot X_1 \geq t \delta \} \leq c \cdot \lambda^t \quad \text{for some } c > 0 \text{ and some } 0 < \lambda < 1 \]
independent of \( t \).

Therefore the total area of all \( r \in T(t, \delta) \) tends to 1 exponentially as \( t \to \infty \), since
\[ \mathbb{P}\{|n_t(r) - t\bar{N}| > t \delta \text{ or } |b_t(r) - t\bar{B}| > t \delta \text{ or } |h_t(r) - t\bar{H}| > t \delta\} \]
\[ \leq \mathbb{P}\{|n_t(r) - t\bar{N}| > t \delta\} + \mathbb{P}\{|b_t(r) - t\bar{B}| > t \delta\} + \mathbb{P}\{|h_t(r) - t\bar{H}| > t \delta\}. \]

For \( t \in \mathbb{N} \), we now construct a code \( C_t \). Let \( M_{0t} := \lfloor 2^{t(\bar{B} + 2\delta)} \rfloor \), \( M_{1t} := \lfloor 2^{t(\bar{B} + 2\delta)} \rfloor \).
Let user 0 have \( M_{0t} \) messages and let user 1 have \( M_{1t} \) messages. We regard the message pairs as equally spaced points in a lattice in the unit square. The probability that a lattice point lies in one of the typical rectangles is equal to
\[ \sum_{r \in T(t, \delta)} \frac{\#\text{lattice points in } r}{M_{0t} \cdot M_{1t}} \geq \sum_{r \in T(t, \delta)} \frac{(\text{height}(r) \cdot M_{0t} - 1) \cdot (\text{breadth}(r) \cdot M_{1t} - 1)}{M_{0t} \cdot M_{1t}} \]
for large enough $t$. Since the total area of all $r \in T(t, \delta)$ is at least $1 - (c') \cdot (\lambda')^t$ for some $c'$ and $\lambda'$, the probability that a message point does not lie in a typical rectangle is bounded by $c \cdot \lambda^t$ for some $c > 0$ and $0 < \lambda < 1$ independent of $t$.

The code $C_t$ is defined as follows. Each message point corresponds to a row and column in the square and hence, via the $V(tN)$ tree, to a pair of strategies for the two users. Each user applies the strategy corresponding to his message, but he transmits at most $a_t$ symbols, where $a_t := \lceil t(\bar{N}+\delta) \rceil$. After the $n$th transmission ($n = 1, 2, \ldots, a_t$) each user determines the connected region at level $n$ in which the message point must lie. If the path from the root to the leaf in which the message point lies contains less than $a_t$ vertices, then the users continue with sending dummy letters.

The message point lies in exactly one of the rootlike rectangles $r \in S_t$. If this rectangle $r$ is typical, then after the $a_t$ transmissions the users know that the point lies in $r$, since $n_t(r) \leq a_t$. Now $r$ contains at most $\lceil 2^{-t(\bar{N}-\delta)} \rceil \cdot [3t\delta] + 1$ horizontal lattice lines since $h_t(r) \geq t(\bar{N}-\delta)$. Therefore user 0 can specify his message in $[3t\delta]+1$ transmissions by row slicing (cf. Lemma (3.3.2.3)) and so can user 1 by column slicing. Hence it takes $\ell_t$ transmissions to determine the message pair, with $\ell_t := a_t + 2[3t\delta] + 2$. If less transmissions are needed, dummy letters are added until both users have sent $\ell_t$ input letters. If $r$ is not typical, dummy letters are sent until we have $\ell_t$ inputs.

The code $C_t$ has block length $\ell_t$ and its error probability tends to zero.
exponentially as $t \to \infty$. Its rate is
\[
\left( \frac{\log(N_0 t)}{t} - \frac{1}{t} \right) \to \left( \frac{-\beta + 2\delta}{N + 7\delta}, \frac{-\beta + 2\delta}{N + 7\delta} \right) \text{ if } t \to \infty.
\]

This rate tends to $(H/N, H/N)$ if we let $\delta \downarrow 0$. The only thing left to show is
\[
H/N = I_0(r) \quad \text{and} \quad H/N = I_1(r).
\]

We show this for the $I_0$ part only, since the $I_1$ part is proved analogously. We have
\[
H = \sum_{r \in S_1} \text{area}(r) \cdot (-\log(\text{height}(r))
\]
\[
= \sum_{r = r_0 \times r_1} \text{Pr}\{\theta_0 \in r_0\} \cdot \text{Pr}\{\theta_1 \in r_1\} \cdot (-\log(\text{Pr}\{\theta_0 \in r_0 | \theta_1 \in r_1\})
\]
where $r = r_0 \times r_1$ denotes the projection of the rectangle in the two directions: $r_0$ is user 0's subinterval of $[0,1)$ corresponding to $r$ and $r_1$ is user 1's subinterval corresponding to $r$. We will write $\bar{y}_1(r)$ for the $y_1$ sequence of length $n_1(r)$ corresponding to $r (= r_0 \times r_1)$ in the tree $T$. Now $H$ can be written as
\[
\sum_{r_1} \text{Pr}\{\theta_1 \in r_1\} \cdot \\
\sum_{r_0} \text{Pr}\{\bar{y}_1 = \bar{y}_1(r) | \theta_1 \in r_1\} \cdot (-\log(\text{Pr}\{\bar{y}_1 = \bar{y}_1(r) | \theta_1 \in r_1\})
\]
\[
= \sum_{r_1} \text{Pr}\{\theta_1 \in r_1\} \cdot H(\bar{y}_1 | \theta_1)
\]
\[
= H(\bar{y}_1 | \theta_1)
\]
\[
= \sum_{n} H(Y_{1n} | \theta_1, \bar{y}_1^{\bar{n}}).
\]

It should be noted that not all $Y_{1n}$ exist, in general, because of the variable length of the tree $T$. We solve this problem by adding a suitable number of *'s to the end of a $Y_0$—and $Y_1$ sequence such that all sequences obtain length $N$. The *'s
do not change the entropy (they give contribution 0), since the pair \((\theta_1, Y_1^n)\) (and also \((\theta_0, Y_0^n)\)) determines whether the *'s must be added. Let \(i\) range over all vertices \(A_{n,i}\) at level \(n\) in the tree. As in the proof of Theorem (3.1.4), user 1 can determine the vertex \(A_{n,i}\) given \(\theta_1\) and \(Y_1^n\). Hence \(H(Y_1^n | \theta_1)\) may be expressed as

\[
\sum_n \sum_i \Pr\{(\theta_0, \theta_1) \in A_{n,i}\} \cdot H(Y_{1n} | \theta_1, Y_1^n, (\theta_0, \theta_1) \in A_{n,i})
\]

Furthermore we have

\[
\tilde{N} = \sum_n n_1(r) \cdot \text{area}(r)
\]

\[
= \sum_n \sum_i \phi(A_{n,i} \text{ contains } r) \cdot \text{area}(r)
\]

\[
= \sum_n \sum_i \phi(A_{n,i} \text{ contains } r) \cdot \text{area}(r)
\]

\[
= \sum_n \sum_i \omega(A_{n,i}).
\]

Hence \(\tilde{N}/N = R_0(T)\).

3.3.4. From variable length trees to variable length codes and vice versa

In Section 3.2.3, we constructed a zero-error code from an \(P(N)\) tree. In Section 3.3.3, we constructed a (fixed length) code with vanishing error probability from a \(V(N)\) tree. Here we will construct a 'variable length code' (which is not a code in the sense of Definition (2.1.6)!) with error probability 0, if the thresholds in a \(V(N)\) tree are on an \(\mathbb{N}_0^1 \times \mathbb{N}_1^1\) grid and the unit square is partitioned into \(\mathbb{N}_0^1 \times \mathbb{N}_1^1\) basic rectangles. As in Theorem (3.2.3.3), we will show that the tree rate is equal to the code rate for these codes, if the code rate is defined appropriately.

Consider a \(V(N)\) tree (in fact, a \(P^2(N)\) tree) with thresholds on a grid such that the \(\mathbb{N}_0^1 \times \mathbb{N}_1^1\) basic rectangles remain as connected regions after the partitioning.
An example for the BMC is shown below (compare this to Figure (3.2.3.1)).

(3.3.4.1) Figure. \( V(3) \) tree for the BMC representing a code.

We can use this tree to construct a code in the following way. Let user \( i \) have \( \mathcal{M}_i \) possible messages, \( i \in \{0,1\} \). Each of the \( \mathcal{M}_0 \) basic intervals for user 0 (each row) corresponds to a message of user 0 and each column corresponds to a message of user 1. Suppose the users want to transmit message pair \( (w_0, w_1) \) to each other. The first input symbol that they have to use in the code can be found from the root of the tree: if the row corresponding to message \( w_0 \) has \( I_{00} = x \) in the tree, then the first symbol of user 0's codeword will be \( f_{00}(w_0) = x \). In the same way user 1 can determine his first input, \( f_{10}(w_1) \).

After this transmission they both know in which of the connected regions at level 1 of the tree the pair \( (w_0, w_1) \) lies. If this connected region, say \( A \), consists of just one basic rectangle, then both users can now determine the other user's message. Hence they can stop, or proceed with a new message pair. If \( A \) contains more than one basic rectangle in some row or column, the users choose a second input symbol. They look at the vertex representing \( A \) to find out which input \( I_{01} = f_{01}(w_0, y_{00}) \) must be sent for row \( w_0 \) and which \( I_{11} = f_{11}(w_1, y_{10}) \)
Chapter 3

is needed in column $w_1$.

In this way the users go on until the connected region has shrunk to one basic rectangle. As soon as they have reached this, they stop (notice that both users determine to stop at the same time!). At this point, both users know the message pair corresponding to the basic rectangle, which can only be the transmitted pair $(w_0, w_1)$. Therefore the transmission of messages is error free.

We call this system a variable length code (with error probability equal to zero; we will not consider variable length codes with positive error probability). Obviously it is possible to define encoding and decoding functions (although not in the sense of Definition (2.1.6)) for it. We shall not do this here; a more formal definition of variable length codes can be found in [TIE89].

As we know, a fixed length code (cf. (2.1.6)) has three parameters: $N_0$, $N_1$ and $N$. The parameters $N_0$ and $N_1$ apply to a variable length code too, but a block length cannot be defined for such a code. Instead, we have $N_0N_1$ possibly different codeword lengths: one for each message pair. Therefore we define the following.

(3.3.4.2) Definition. Consider a variable length code constructed from a $F^D(N)$ tree with grid size $(N_0+1) \times (N_1+1)$. For each message pair $(w_0, w_1)$, we define $N(w_0, w_1)$ as the length $n$ of the output sequence $y^m_0$ (or $y^m_1$, as they are equally long) if the transmitted pair is $(w_0, w_1)$. Define the average codeword length $\bar{N}$ by

$$\bar{N} := \sum_{(w_0, w_1)} N(w_0, w_1) / (N_0 \cdot N_1).$$
(3.3.4.3) Definition. For a variable length code with parameters \( w_0, w_1 \) and \( N(w_0, w_1) \) \((0 \leq w_0 \leq N_0 - 1, 0 \leq w_1 \leq N_1 - 1)\) we define the rate pair of the code, \( \mathcal{R} \), as
\[
\mathcal{R} := (R_0, R_1), \quad R_i := \frac{\log(N_i)}{N}, \quad i \in \{0, 1\}.
\]

The following theorem generalizes Theorem (3.2.3.3), where we proved that the rate of a fixed length code is equal to the rate of the corresponding tree.

(3.3.4.4) Theorem. Consider a variable length code with error probability 0 that is constructed from a \( \mathcal{V}^0(N) \) tree \( T \). Then the rate of this code, as defined in Definition (3.3.4.3), is equal to the rate of the \( \mathcal{V}^0(N) \) tree, \( \mathcal{R}(T) \).

Proof. In the proof of Theorem (3.3.3.1), we showed that the rate of the \( \mathcal{V}^0(N) \) tree equals \((\bar{R}/\bar{N}, \bar{B}/\bar{N})\) with \( \bar{R} := -\log(\text{height}(r)) \) averaged over all rootlike rectangles \( r \) remaining after partitioning, \( \bar{B} := -\log(\text{breadth}(r)) \) averaged over \( r \), and \( \bar{N} := \) the average block length as defined in (3.3.4.2). Since all basic rectangles have equal size, \( \text{height}(r) = 1/N_0 \) for all basic rectangles \( r \), so \( \bar{R} = \log(N_0) \). Similarly \( \bar{B} = \log(N_1) \), proving the theorem after division by \( \bar{N} \).

In Section (3.2.3), we defined an \( \mathcal{V}^0(N) \) tree as the kind of tree in which a fixed length code with error probability 0 could be represented. In Definition (3.3.4.5) below we define something similar for variable length codes. For fixed length trees, we could not prove that \( \mathcal{V}^0(N) \) converges to the capacity region. For variable length trees, however, we can show that the regions \( \mathcal{V}^0(N) \) tend to \( \mathcal{C} \).

(3.3.4.5) Definition. Let \( N \in \mathbb{N} \). A \( \mathcal{V}^0(N) \) tree (or strategy) is a \( \mathcal{V}^0(N) \) tree (or strategy) on a grid in which a 0—error variable length code is represented such that the largest codeword length occurring in the code is \( N \). Also
Chapter 3

\[ V^0(N) := \{ T \mid T \text{ is a } V^0(N) \text{ tree} \}, \]
\[ V^0 := \bigcup_{N \in \mathbb{N}} V^0(N). \]

(3.3.4.6) Theorem. We have

\[ \forall N \in \mathbb{N} \left( \text{co } V^0(N) \subseteq C \right) \quad \text{and} \quad \lim_{N \to \infty} V^0(N) = V^0 = C. \]

Proof: see Section 3.5.

Because of Theorem (3.3.4.6) it is interesting to try to develop variable length codes. Not only do these provide an easy way to use the channel for actual communication — and even error free — but we also know that variable length codes achieve capacity: if the average block length is large enough, variable length codes exist with rates that are close to the maximum possible rate for the TWC.

Variable length codes have already been mentioned in [SHA61]: the Hagelbarger code is the least complex variable length code for the BMC. It is in fact the code described by the \( V^0(2) \) tree in Figure (3.3.1.2). Each user has two messages and the average block length is 7/4. More complex examples of variable length codes for the BMC are shown in Section 3.6.

3.4. Bootstrap trees

3.4.1. Definition and statement of result

In Section 3.3 we have considered a possible improvement over fixed length trees. We managed to achieve higher rates, in general, by deleting some of the low-rate vertices in an \( F(N) \) tree. (We stress the fact that this improvement only holds for finite \( N \), since we showed that the limiting region for \( N \to \infty \) is the same
for both fixed — and variable length trees.) However, with $V(N)$ trees we are limited in the choice of the vertices to be deleted from the $F(N)$ tree: if we want to delete a low-rate vertex somewhere halfway the tree, we have to delete the complete subtree below it (which might contain vertices with high rates) as well!

In this section we provide a way to get around this problem. Using a technique introduced by Schalkwijk [SCH83a] for the BMC, it is sometimes possible to delete single vertices from a tree without deleting the subtree. We will prove that the average rate of the remaining set of vertices (which is not a tree in the strict sense anymore) is an achievable rate for the TWC again. Schalkwijk called this technique 'bootstrapping', since it is based on the idea of 'pulling oneself up by one's own bootstraps': a given tree with rate $\mathbb{X}(0)$ is used to develop a coding scheme having a higher rate, $\mathbb{X}(1)$, which in turn is used to construct a code with even higher rate, $\mathbb{X}(2)$, et cetera, until we reach the rate of the tree without the deleted vertex. ²

Clearly we cannot delete an arbitrary vertex from a tree and still keep an achievable rate. Definition (3.4.1.1) states necessary requirements for a vertex such that it has the above property.

(3.4.1.1) Definition. Consider a $V(N)$ tree $T$ for some $N \in \mathbb{N}$. The set of possible bootstrap vertices (PBV) for $T$ is the set of vertices in the tree for which (3.4.1.2), the so-called PBV property, holds. Here $A$ denotes a vertex at level $n$.

(3.4.1.2) $r(A) = (H(I_{0n}|\theta_1, I_{1n}^n, (\theta_0, \theta_1) \in A), H(I_{1n}|\theta_0, I_{0n}^n, (\theta_0, \theta_1) \in A)).$

² The term 'bootstrapping' is not uncommon in science: the Oxford English Dictionary [BUR72] quotes several technical papers (the earliest one from 1946), in which this word is used with a similar meaning.
Note that the slicing vertices occurring in Lemma (3.3.2.3) have the PBV property. The PBV property means that the transmission in a vertex $A$ satisfying (3.4.1.2) is such that the a priori uncertainty about output $y_{in}$ for user $i$, knowing that we are in vertex $A$, is equal to the uncertainty about the other user's input. Suppose that we use a code based on the tree, as in Theorem (3.3.3.1). In Section 3.4.2 we will show that the information transmitted in vertex $A$ does not have to be transmitted by sending the inputs as described by $A$; the users can directly transmit the information about their inputs using another code. If this 'other' code has a rate higher than $r(A)$, then the rate of the complete coding scheme will be higher than the rate of the tree.

The new code can be used to repeat the procedure. Now the information about the inputs in vertex $A$ can be transmitted using the new code, which again increases the total rate. This is where idea of 'bootstrapping' is used. If we repeat the procedure infinitely many times, it will turn out that the rate of $A$, $r(A)$, does not occur in the formula for the code rate anymore: $A$ has become 'invisible'. Note that this can be done only for vertices with the PBV property; all other vertices necessarily remain 'visible' in the rate formula. This explains the following definition.

(3.4.1.3) Definition. Consider a $V(n)$ tree $T$ and its PBV set. A type assignment for $T$ is defined as follows. To each vertex in $T$, we assign a type, 'visible' or 'invisible'. For the root and for (other) vertices not in the PBV set, the type must be 'visible'. For vertices in the PBV set (except for the root) we may choose the type at will. The visible distance between visible vertices is defined analogously to the ordinary distance (cf. Section 3.2.1), but now we only count visible vertices on the path between the vertices. The visible depth of $T$ is defined analogously. 

- 66 -
Strategy Trees for TWCs

(3.4.1.4) Definition. Let $N \in \mathbb{N}$. A $B(N)$ tree (bootstrap tree) is defined as a $\mathcal{V}(N_0)$ tree for some $N_0 \geq N$, with a type assignment such that the visible depth is $N-1$. □

An example of this is given in Figure (3.4.1.5).

(3.4.1.5) Figure. $F(5)$, $V(5)$ and $B(4)$ tree.

At the left, an $F(5)$ tree is shown in schematic form. In the middle, the tree has been pruned to a $V(5)$ tree. The three vertices in this tree labeled 'PBV' form the PBV set. The $B(4)$ tree on the right is constructed from the $V(5)$ tree by considering two of the PBV vertices invisible ('inv', the open circles).

The next definition describes how the rate of a $B(N)$ tree should be computed. As said before, invisible vertices do not contribute to the rate.

(3.4.1.6) Definition. Let $T$ be a $B(N)$ tree constructed from a $\mathcal{V}(N_0)$ tree with vertex set $\mathcal{V}$. Partition $\mathcal{V}$ into $\mathcal{V}_{vis}$ and $\mathcal{V}_{inv}$, the sets of visible and invisible vertices, respectively. The rate of the $B(N)$ tree, denoted as $k_{vis}(T)$, is defined as

$$k_{vis}(T) := \frac{\sum_{A \in \mathcal{V}_{vis}} w(A) \cdot \tau(A)}{\sum_{A \in \mathcal{V}_{vis}} w(A)}.$$
The definition and theorem below are generalizations of the ones in Section 3.3. They state that the rate of a bootstrap tree as defined above is indeed achievable and tends to capacity if the visible depth is large enough.

(3.4.1.7) Definition. We define, for $N \in \mathbb{N}$:

$$B(N) := \{ L_{\text{vis}}(T) \mid T \text{ is a } B(N) \text{ tree} \},$$

$$B := \bigcup_{N \in \mathbb{N}} B(N).$$

(3.4.1.8) Theorem. We have

$$\forall N \in \mathbb{N} \left[ \text{co} B(N) \subset C \right] \text{ and } \lim_{N \to \infty} B(N) = \overline{B} = C.$$

Proof: Part of this proof follows the proofs for fixed or variable length strategies, especially where the limit is concerned. For this part we refer to Section 3.5. The fact that every point in $B(N)$ is achievable, however, is proved in an entirely different way. Since this proof is quite involved, we devote Section 3.4.2 to it.

Theorem (3.4.1.8) offers a way to find significantly better inner bounds on the capacity region of certain TWCs. The most famous example of this technique is Schalkwijk's bootstrapping strategy for the BMC. In Section 3.6, this strategy is shown, along with some more bootstrap trees for other TWCs.

3.4.2. Proof of the achievability of bootstrap rates

Here we will prove the achievability of every rate point corresponding to a bootstrap tree as defined in the previous section; hence we will show $B(N) \subset C$ for all $N \in \mathbb{N}$. For the case of the BMC with Schalkwijk's bootstrap tree [SCH83a], the achievability was proved by the author in [OVE88a]. This proof was based on a term paper by Hazewindus [HAZ85], which contained the basic idea for the proof.
(although the paper itself was very concise and unfortunately incorrect). Here we basically use the same argument as in [OVE88a], but in a slightly more general form since we are discussing arbitrary deterministic TWCs now.

Consider a $V(N_0)$ tree with a type assignment and the corresponding $B(N)$ tree, $N \leq N_0$. We use Lemma (3.3.2.3) to change the $V(N_0)$ tree into a $V'(N'_0)$ tree for some $N'_0 \geq N_0$. The slicing vertices that had to be added in this procedure are given type 'invisible'; as shown in the lemma, these vertices satisfy (3.4.1.2).

Because of this, the bootstrap tree corresponding to the $V'(N'_0)$ tree has the same set of visible vertices (and hence, the same rate) as the $B(N)$ tree following from the $V(N_0)$ tree. Let the $V'(N'_0)$ tree be called $T$. Its rate is $l(T) = (l_0(T), l_1(T))$.

We assume that both components of $l(T)$ are positive. The case where one of the two components is zero is trivial and follows by continuity. We will show that $l(T)$ is the first of a sequence of achievable rates, $l(n)$ for $n = 0, 1, 2, \ldots$ with $l(0) = l(T)$. The limit of this sequence will be the rate of the $B(N)$ tree, $l_{\text{vis}}(T)$:

$$
\lim_{n \to \infty} l(n) = l_{\text{vis}}(T) = (l_{\text{vis},0}(T), l_{\text{vis},1}(T))
$$
as computed by Definition (3.4.1.6). We also assume $l_{\text{vis},i}(T) > 0$ for $i = 0, 1$.

In this section we prove that for each $n$, a sequence of block codes exists such that the rate approximates $l(n)$ and both error probabilities decrease exponentially as the block length increases. As in [OVE88a], the main ingredients of the proof are Theorem (3.3.3.1) and the correlated source coding result by Slepian and Wolf [SLE73]. The Slepian–Wolf result is stated in Lemma (3.4.2.1) below, in the form in which we are going to use it.

(3.4.2.1) Lemma. For each vertex $A_n, i$ at some level $n$ in the tree $T$, we have the following. Let $\varepsilon > 0$. Real numbers $c > 0$ and $0 < \lambda < 1$ exist such that, for large enough $N \in \mathbb{N}$, an encoder–decoder pair exists for which the following holds.
Chapter 3

\( I \) symbols \( I_{0n} \) that correspond to \( I \) independent transmissions in vertex \( A_n,i \) can be encoded by user 0 into

\[
\left( H(I_{0n} | \theta_1, I^n_1, (\theta_0, \theta_1) \in A_n,i) + \epsilon_z \right) I
\]

binary symbols such that, if user 1 decodes these using \( \theta_1 \) and \( I^n_1 \), the probability of a decoding error is at most \( c \cdot \lambda^{I} \). Similarly user 1 can encode \( I_{1n} \) symbols in

\[
\left( H(I_{1n} | \theta_0, I^n_0, (\theta_0, \theta_1) \in A_n,i) + \epsilon_z \right) I
\]

binary symbols with a decoding error probability for user 0 of at most \( c \cdot \lambda^{I} \).

**Proof:** This follows from the theorem in [SLE73]. In fact we do not consider the classical case of two correlated information streams that have to be encoded by separate encoders; we only look at the case where one information stream, consisting of i.i.d. symbols \( I_{0n} \), is encoded and where the decoder has side information, namely the i.i.d. sequences \( I^n_1 \) and \( \theta_1 \). This is indeed a special case of the aforementioned theorem; however, we must make two remarks about this.

Firstly, in the proof in [SLE73] it is needed that the decoder’s side information, say \( U \), is described with symbols in a finite alphabet \( \mathcal{U} \). Here we do not have this: \( \theta_1 \) is a real number. However, in Section 2.3.2 we defined \( H(I^n_1 | \theta_1) \) in terms of the basic intervals \( I_k \) of user 1. Therefore the index \( k \) of the interval \( I_k \) containing \( \theta_1 \) bears all information: the position of \( \theta_1 \) within \( I_k \) adds no further knowledge about \( I_{0n} \) or \( I_{1n} \). So, if \( I(\theta_1) \) is the index of the basic interval in which \( \theta_1 \) lies, we could have stated the lemma with \( H(I_{0n} | I(\theta_1), I^n_1, (\theta_0, \theta_1) \in A_n,i) \) instead of with \( H(I_{0n} | \theta_1, I^n_1, (\theta_0, \theta_1) \in A_n,i) \); these represent the same entropy.

Secondly it should be noted that the theorem in [SLE73] is stated with a bound on the error probability that does not depend on the block length \( I \). However, Košlev [KOS77] proved that the theorem also holds with an exponential error bound (also see [CSI81], problem 3.1.5).

For the \( T^D(N_0^* ) \) tree \( T \), we define the set of visible and invisible vertices as
Strategy Trees for TWCs

Let the total weight of all vertices in these sets be $\mathcal{V}_{\text{vis}}$ and $\mathcal{V}_{\text{inv}}$ respectively. If we define $\mathcal{N}$, $\mathcal{R}$ and $\mathcal{B}$ for the $\mathcal{V}(\mathcal{N}_0)$ tree $T$ as in Theorem (3.3.3.1), we find $\mathcal{N} = \mathcal{V}_{\text{vis}} + \mathcal{V}_{\text{inv}}$. We abbreviate $\mathcal{E}_{\text{vis}}(T)$ as $\mathcal{E}_{\text{vis}}$; similarly we define $\mathcal{E}_{\text{inv}}$ as the average rate of the invisible vertices. Thus

\[(3.4.2.2)\]

$$
\mathcal{E}_{\text{vis}} = \left( \frac{\sum_{A \in \mathcal{V}_{\text{vis}}} w(A) \cdot \tau(A)}{\mathcal{V}_{\text{vis}}} \right),
$$

$$
\mathcal{E}_{\text{inv}} = \left( \frac{\sum_{A \in \mathcal{V}_{\text{inv}}} w(A) \cdot \tau(A)}{\mathcal{V}_{\text{inv}}} \right),
$$

$$
\mathcal{E}(T) = \left( \frac{\mathcal{E}_{\text{inv}} \mathcal{E}_{\text{inv}} + \mathcal{E}_{\text{vis}} \mathcal{E}_{\text{vis}}}{\mathcal{E}_{\text{inv}} + \mathcal{E}_{\text{vis}}} \right) = \left( \frac{\mathcal{N}}{\mathcal{R}}, \frac{\mathcal{B}}{\mathcal{N}} \right).
$$

Now we construct a Markov chain from the tree $T$ (cf. [SCH82], [SCH83a]). Each state in the chain will correspond to a vertex in the tree. Call the vertices $A_{n,i}$ and let the states of the Markov chain have the same names. The transition probability of going from a state $A_{n,i}$ to $A_{m,j}$ is given by the following three rules.

(i) If $m = n+1$ and $A_{m,j}$ is a successor of $A_{n,i}$ in the tree, then

$$
Pr\{\text{next state} = A_{m,j} \mid \text{previous state} = A_{n,i}\} := \frac{w(A_{m,j})}{w(A_{n,i})}.
$$

(ii) If $A_{m,j}$ is the root ($m = 0$) and in the partitioning of $A_{n,i}$ at least one rootlike rectangle remains that is not partitioned further in $T$, then

$$
Pr\{\text{next state} = A_{m,j} \mid \text{previous state} = A_{n,i}\} := \frac{w(r)}{w(A_{n,i})},
$$

where $w(r)$ is the weight of all rootlike rectangles remaining in $A_{n,i}$.

(iii) In any other case:

$$
Pr\{\text{next state} = A_{m,j} \mid \text{previous state} = A_{n,i}\} := 0.
$$

As a consequence of the above rules, transitions are possible only from a vertex $A_{n,i}$ to a direct successor $A_{n+1,j}$ or from a vertex $A_{n,i}$ to the root $A_{0,0}$;
latter transition is possible only in the case that a rootlike rectangle remains in $A_{n,1}$. Figure (3.4.2.3) illustrates this. It shows the $\mathcal{P}(3)$ tree for Schalkwijk's bootstrap strategy, plus the Markov chain that is derived from it. In this figure, $a$ is the area of the subrectangle in $A_{0,0}$ and $b$ is the area of the two rectangles in $A_{1,0}$ that correspond to $y = 0$. The transition probabilities are written next to the arrows in the Markov chain.

\begin{center}
\begin{tikzpicture}

\node (A0) at (0,0) {$A_{0,0}$};
\node (A1) at (1,-1) {$A_{1,0}$};
\node (A2) at (2,-2) {$A_{2,0}$};
\node (A3) at (3,-3) {$A_{3,0}$};

\draw[->] (A0) -- (A1);
\draw[->] (A1) -- (A2);
\draw[->] (A2) -- (A3);
\end{tikzpicture}
\end{center}

(3.4.2.3) Figure. $\mathcal{P}(3)$ tree and the corresponding Markov chain.

An infinite walk through the Markov chain corresponding to a tree $T$ can be seen as a partitioning of the square derived from infinitely many copies of $T$. Compare this to the proof of Theorem (3.3.3.1), where we took $t$ copies of a $\mathcal{P}(N)$ tree. Whenever we reach a rootlike rectangle that is not partitioned in the original tree $T$, we return to the root of $T$ and partition this rectangle according to the tree.

It is not difficult to see that the Markov chain is irreducible (cf. [FEL70, Ch. XV, Section 4]), since every state can be reached from every other state.
wit, the root can be reached from every state because each partitioning necessarily ends in rootlike rectangles, and from the root we can reach all other states because of the tree structure. We also assume that the chain is aperiodic. This is not a severe restriction, since we can always change a periodic chain into an aperiodic one without changing the corresponding bootstrap tree. Indeed, if the chain is periodic, then we add a vertex below one of the leaves of the \( V(N_0) \) tree in which we use slicing to separate the remaining rootlike rectangle into two halves. In one of these halves, we return directly to the root and to the other half we append another slicing vertex before returning to the root. Now the chain is aperiodic and still irreducible. The slicing vertices are given type 'invisible' (which is possible, see before), so \( \mathcal{L}_{\text{vis}}(T) \) is not affected by this.

For a state \( A \), define \( q_A \) by

\[
q_A = \frac{w(A)}{\sum_{A_i \in \mathcal{N}} w(A_i)} = \frac{w(A)}{N}.
\]

It is left to the reader to check that the \( q_A \)'s satisfy

\[
\forall_{m,j} \left[ q_{A_{m,j}} = \sum_{A_{n,i}} \Pr\{\text{next state} = A_{m,j} \mid \text{previous state} = A_{n,i}\} \cdot q_{A_{n,i}} \right].
\]

Hence by [FEL70, Ch. XV, Section 7], the \( q_A \)'s form the stationary distribution of the Markov chain. So, if \( A(d) \) is the random variable denoting the state that is reached after \( d \) transitions in the chain, starting from the root, then

\[
\forall A \in \mathcal{N} \left[ \lim_{d \to \infty} \Pr\{A(d) = A\} = q_A \right].
\]

Let \( q_{\text{vis}} \) be the probability of being in a visible vertex and define \( q_{\text{inv}} \) likewise:

\[
q_{\text{inv}} = \frac{1 - q_{\text{vis}}}{N}, \quad q_{\text{vis}} = 1 - q_{\text{inv}} = \frac{1}{N}.
\]

Assume \( q_{\text{inv}} > 0 \); otherwise \( \mathcal{L}_{\text{vis}}(T) \) equals \( \mathcal{L}(T) \), which is achievable by Theo-
rem (3.3.1.4). There are three possibilities, given by (3.4.2.5.a through c).

(3.4.2.5.a) $p_1(p) \cdot p_{\text{inv}}^0 > p_0(p) \cdot p_{\text{inv}}^1$
(3.4.2.5.b) $p_1(p) \cdot p_{\text{inv}}^0 < p_0(p) \cdot p_{\text{inv}}^1$
(3.4.2.5.c) $p_1(p) \cdot p_{\text{inv}}^0 = p_0(p) \cdot p_{\text{inv}}^1$.

In the rest of this proof we assume that (3.4.2.5.a) holds. Case (3.4.2.5.b) can be treated by interchanging 0's and 1's everywhere. The third case asks for some minor changes. We will explicitly point out the differences with (3.4.2.5.a).

Now we define the sequence of rates $p(n)$. Two extra variables are needed:
a term $p^*(n)$ to ensure that the direction of $p(n)$ tends to the direction of $p_{\text{vis}}$ (this can be compared to the dissecting state $d$ in [SCH83b]), and a factor $\eta_n$ which affects the Euclidean norm of $p(n)$ such that $\|p(n)\| \to \|p_{\text{vis}}\|$ for $n \to \infty$.

(3.4.2.6) Definition. We define $p^*_1(n), \eta_n$ and $p(n) = (p_0(n), p_1(n))$ recursively:

a) $p^*_1(0) := 0$, $\eta_0 := 1$, $p(0) := p(p)$ and for $n > 0$:

b) $p^*_1(n) := \frac{p_1(n-1)}{p_0(n-1)} \cdot p_{\text{inv}}^0 - p_{\text{inv}}^1$

c) $\eta_n := p_{\text{inv}}^0 / p_0(n-1)$

d0) $p_0(n) := (p_{\text{inv}} p_{\text{inv}}^0 + p_{\text{vis}} p_{\text{vis}}^0) / (p_{\text{inv}}^0 \eta_n + p_{\text{vis}})$

d1) $p_1(n) := (p_{\text{inv}} p_{\text{inv}}^1 + p^*_1(n)) + p_{\text{vis}} p_{\text{vis}}^1) / (p_{\text{inv}}^0 \eta_n + p_{\text{vis}})$.

A first observation about the term $p^*_1(n)$ is made below.

(3.4.2.7) Lemma. $\forall n > 0 \ [p^*_1(n) > 0]$.

(Note: if (3.4.2.5.b) holds, then $p^*_1(n) > 0$; if c holds, then $p^*_1(n) = 0$)

Proof: By induction: $p^*_1(1) > 0$ by (3.4.2.5.a). If $p^*_1(n) > 0$, then by (3.4.2.6.d):
\[
\frac{R_1(n)}{R_0(n)} = \frac{R_1(0) + q_{\text{inv}} R_1^*(n)}{R_0(0)} > \frac{R_1(0)}{R_0(0)}.
\]

So, by (3.4.2.6.b), \( R_1^*(n+1) > R_1^*(1) > 0 \) for all \( n > 0 \).

Furthermore we derive the following relations that will be needed in the sequel.

By (3.4.2.2), (3.4.2.6) and the definitions of \( q_{\text{inv}} \) and \( q_{\text{vis}} \) we obtain

\[
(3.4.2.8) \quad R_0(n+1) = \frac{\bar{H}}{H(q_{\text{vis}} + q_{\text{inv}} n+1)}, \quad R_1(n+1) = \frac{\bar{H} + \bar{H} q_{\text{inv}} R_1^*(n+1)}{H(q_{\text{vis}} + q_{\text{inv}} n+1)}.
\]

Also, c) and d) of (3.4.2.6) yield

\[
(3.4.2.9) \quad R_0(n) = \frac{R_0(0)}{q_{\text{vis}} + q_{\text{inv}} n+1} \quad \text{and} \quad \frac{R_0(n)}{R_0(n-1)} = \frac{V_{\text{inv}} R_{\text{inv},0} + V_{\text{vis}} R_{\text{vis},0}}{V_{\text{inv}} R_{\text{inv},0} + V_{\text{vis}} R_{\text{vis},0}(n-1)}.
\]

Let \( n \in \mathbb{N} \). We will construct a sequence of codes \( C_k(n) \) such that rate \( R(n) \) is achievable with \( C_k(n) \) if \( k \to \infty \). In the following, the expression \( \{a, b\} \) bit is transmitted \( \} \) is an abbreviation for: \( \} \) simultaneously, an amount of information of \( a \) bit is transmitted in the \( 0 \to 1 \) direction and \( b \) bit is transmitted in the \( 1 \to 0 \) direction \( \} \). Consider assertion \( \delta \), stating that \( R(n) \) is achievable.

\[
\delta_n := \forall e_n > 0 \exists c_n > 0 \exists \lambda_n < 1, 0 < a_n < 1 \exists x_n \forall k \geq x_n \left[ \begin{array}{c}
\text{a block code} \\
C_k(n) \text{ for } K \text{ exists allowing at least } [ (R_0(n)-e_n)k ] \text{, } [ (R_1(n)-e_n)k ] \text{ bit} \\
to be transmitted in } k \text{ transmissions, with both error probabilities at most} \\
c_n \cdot (\lambda_n)^{e_n}, e_n := (k)^a n.
\end{array} \right]
\]

(3.4.2.10) Theorem. \( \delta_n \) holds for all \( n \geq 0 \).

Proof: We use induction on \( n \). For \( n = 0 \), the assertion follows from Theorem (3.3.3.1), where the sequence of codes \( C_k(0) \) is constructed with \( a_0 = 1, e_0 = k \).
Chapter 3

Assume that \( A_n \) holds for some \( n \) and let \( \varepsilon_{n+1} > 0 \). Let \( \delta, \varepsilon_x \) and \( \varepsilon_n \) be positive numbers, to be specified later and depending on \( \varepsilon_{n+1} \) only. Let \( k > 0 \).

For the given \( \delta \), we choose a number \( D \in \mathbb{N} \) such that (3.4.2.11) holds. Here \( I(d) \) denotes the state in the chain reached after \( d \) transitions. Note that \( D \) exists since \( q_A \) is the stationary probability of \( A \) (cf. (3.4.2.4)) and the convergence is uniform.

(3.4.2.11) \( \forall d \geq D \forall A \in \mathcal{X}_{\text{inv}} \left| \Pr\{ I(d) = A \} - q_A \right| \leq \delta/2. \)

Let

(3.4.2.12) \( \varepsilon_m := \{ \varepsilon_x (1 + \delta | \mathcal{X}_{\text{inv}} |) + \delta \cdot \sum_{A \in \mathcal{X}_{\text{inv}}} (r_0(A) + r_1(A))/q_{\text{inv}} \} \)

then \( \varepsilon_m \to 0 \) if \( \varepsilon_x \to 0 \) and \( \delta \to 0 \). For \( t \in \mathbb{N} \), let \( a_t \) and \( \ell_t \) be as in Theorem (3.3.3.1) (where \( \delta \) plays the role of the \( \delta \) in Theorem (3.3.3.1)) and define \( b_t \) by

(3.4.2.13) \( b_t := \lfloor t - (\ell_{t-1} - a_t \cdot D) \cdot b_t + t (\ell_{t-1} - a_t \cdot D) \rfloor. \)

Let \( t \) be the largest integer for which (3.4.2.14) holds.

(3.4.2.14) \( k \geq (a_t - D) \cdot b_t + t (\ell_t - a_t + D). \)

Assume that \( a_t > D \), which means that \( t \) must be large enough and hence \( k \) must be large enough (depending on \( \varepsilon_{n+1} \)). Define \( R_1^{i}(n+1) \) by the following equality.

(3.4.2.15) \( \frac{R_{\text{inv},1}^{i} + \varepsilon_m + R_1^{i}(n+1)}{R_1^{i}(n) - \varepsilon_n} = \frac{R_{\text{inv},0}^{i} + \varepsilon_m}{R_0^{i}(n) - \varepsilon_n}. \)

Since \( R_1^{i}(n+1) \to R_1^{p}(n+1) > 0 \) if \( \varepsilon_m \) and \( \varepsilon_n \) tend to zero, it is possible to choose \( \varepsilon_x, \varepsilon_n \) and \( \delta \) such that \( R_1^{i}(n+1) \geq 0 \) by Lemma (3.4.2.7). (Note: if (3.4.2.5.c) holds, it may happen that \( R_1^{i}(n+1) < 0 \), depending on the sign of \( R_1^{i}(n) - R_0^{i}(n) \). In this case we use \( R_0^{i}(n+1) \) instead of \( R_1^{i}(n+1) \). Then \( R_0^{i}(n+1) \geq 0 \) and the rest of the proof should be read with 0's and 1's interchanged.) Hence, by

-76-
(3.4.2.8), it is possible to choose $\varepsilon_{x}, \varepsilon_{n}$ and $\delta$ such that (3.4.2.16) holds.

\[ \mathbb{B} + 2\delta \geq L_{0}(n+1) - \frac{\varepsilon_{n+1}}{2}, \]

\[ (\mathbb{H} + \delta)(1 - (\theta_{\text{inv}} - \delta)\mathbb{M}_{\text{inv}}) + (\theta_{\text{inv}} + \delta)\left[ \frac{L_{0}(n) - \varepsilon_{n}}{L_{1}(n+1)} \right] + 6\delta \]

\[ \mathbb{B} + 2\delta + (\mathbb{H} + \delta)(\theta_{\text{inv}} - \delta)L_{1}(n+1) \geq L_{1}(n+1) - \frac{\varepsilon_{n+1}}{2}. \]

Hence we can choose a large enough $t$ such that (3.4.2.17) holds.

\[ \frac{t[t(\mathbb{H} + 2\delta)]}{(a_{t+1} + \delta)\cdot b_{t+1} + (t+1)(\ell_{t+1} - a_{t+1} + D)} \geq L_{0}(n+1) - \varepsilon_{n+1}, \]

\[ \frac{t[t(\mathbb{H} + 2\delta)]}{(a_{t+1} + \delta)\cdot b_{t+1} + (t+1)(\ell_{t+1} - a_{t+1} + D)} \geq L_{1}(n+1) - \varepsilon_{n+1}. \]

Now we construct the block code $C_{k}(n+1)$. The block length of this code will be $k$. Only $(a_{t} - D)\cdot b_{t} + t(\ell_{t} - a_{t} + D)$ transmissions are used for the sending of information; any remaining transmissions (cf. (3.4.2.14)) consist of dummy symbols. We show that $\{[(L_{0}(n+1) - \varepsilon_{n+1})k], [(L_{1}(n+1) - \varepsilon_{n+1})k]\}$ bit can be transmitted with $C_{k}(n+1)$. In fact,

(3.4.2.18) \{ $t[t(\mathbb{H} + 2\delta)]$, $t[t(\mathbb{H} + 2\delta)] + (a_{t} - D)[t(\theta_{\text{inv}} - \delta)L_{1}(n+1)]$ \}

bit will be transmitted with it. This is at least as much as was required, since

\[ t[t(\mathbb{H} + 2\delta)] \geq [(L_{0}(n+1) - \varepsilon_{n+1})(a_{t} + \delta)\cdot b_{t+1} + (t+1)(\ell_{t+1} - a_{t+1} + D))]. \]
Chapter 9

\[ \geq \lceil (K_0(n+1) - \epsilon_{n+1}) k \rceil. \]

The first inequality follows from (3.4.2.17) and the second one from the definition of \( t \). The inequality for the \( 1 \to 0 \) direction can be found in the same way.

The amount of information that will be transmitted is split up as

\[ (3.4.2.19) \{ t [ t(\delta + 2\delta) + \delta_j ] \} \quad \text{and} \quad \{ 0, (a_t - \theta) [ t(\theta_{\text{inv}} - \delta) K_1(n+1) ] \} \]

bit. With \( \mathcal{M}_0 = [2^t(\delta + 2\delta)] \) and \( \mathcal{M}_1 = [2^t(\delta + 2\delta)] \) as in Theorem (3.3.3.1), the first part can be considered as the information of \( t \) independent message pairs \((w_0, w_1)\) with \( w_i \in \{0, 1, \ldots, \mathcal{M}_i, t-1\} \) for \( i \in \{0, 1\} \). These \( t \) pairs are denoted by \((w_0, w_1)_0, (w_0, w_1)_1, \ldots, (w_0, w_1)_t-1\). Each of the pairs \((w_0, w_1)_j\) can be transmitted with the block code \( \mathcal{C}_t \) that can be constructed from the tree \( T \) using Theorem (3.3.3.1). Here \( \ell_t \) symbols per pair are needed, and the error probability per pair is at most \( c_r (\lambda_p)^t \) for some \( c_r > 0, 0 < \lambda_p < 1 \).

All pairs are encoded simultaneously in the following way. Also, see Figure (3.4.2.27). First consider user 0. Let pair \((w_0, w_1)_j\) correspond to \((x_0, x_1, \ldots, x_{t-1}) \in (\mathbb{F}_0)^\ell t\), the input sequence that would be sent by user 0 using the code \( \mathcal{C}_t \). User 0 transmits \( \mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_{\ell_t-1} \) where \( \mathcal{P}_i \) contains the information about \((x_{i0}, x_{i1}, \ldots, x_{it-1})\) for \( 0 \leq i < \ell_t \). We call \( \mathcal{P}_i \) the \( i \)th packet. We distinguish between the cases \( i \leq D \), \( D < i < a_t \) and \( i > a_t \).

If \( i \leq D \) or \( i > a_t \), we set \( \mathcal{P}_i := (x_{i0}, x_{i1}, \ldots, x_{i, t-1}) \), so \( \mathcal{P}_i \) contains \( t \) symbols. No further coding takes place. For \( D < i < a_t \) we proceed as follows. The symbols \( x_{i0}, x_{i1}, \ldots, x_{i, t-1} \) correspond to independent transmissions in the tree \( T \). More specifically, \( x_{i,j} \) corresponds to a transmission in a vertex at level \( i \) in the \( j \)th replica of the \( \mathcal{T}_0(t)^c \) tree consisting of \( t \) copies of \( T \). This is illustrated in Figure (3.4.2.20), where the vertices for the \( x_{i,j} \) are shown as open circles.
For each transmitted symbol, user 0 can determine the vertex in $T$ it corresponds to. For $A \in \mathcal{K}_{\text{inv}}$, let $W(A)$ := the number of symbols among these $t$ that are transmitted in vertex $A$, and let $W := \sum_{A \in \mathcal{K}_{\text{inv}}} W(A)$. The packet $P_i$ will contain $b_t$ symbols (cf. (3.4.2.13)). We distinguish two cases. If some $A \in \mathcal{K}_{\text{inv}}$ exists for which $|W(A) - q_A t| > \delta t$, then the $b_t$ symbols of $P_i$ are randomly chosen. Otherwise user 0 first transmits the $t - W$ symbols corresponding to visible vertices. The remaining $W$ symbols are compressed by user 0 such that, for all $A \in \mathcal{K}_{\text{inv}}$, the $W(A)$ symbols transmitted in vertex $A$ are represented by $\lfloor (r_0(A) + \varepsilon_x) \cdot W(A) \rfloor$ binary digits. This is possible by Lemma (3.4.2.1) and since $A \in \mathcal{K}_{\text{inv}}$, so (3.4.1.2) holds. The error probability for the compression is at most

\[ \sum_{A \in \mathcal{K}_{\text{inv}}} c_{\lambda_A} W(A) \leq c_{\lambda_S} W^{t} \]

for some $c_{\lambda_A}$, $\lambda_A$, $c_{\lambda_S}$ and $\lambda_S$. How these binary digits are transmitted will be explained later; their number is (using (3.4.2.2) in the second inequality)

\[ \sum_{A \in \mathcal{K}_{\text{inv}}} \lfloor (r_0(A) + \varepsilon_x) \cdot W(A) \rfloor \]
Chapter 9

\[
\leq \sum_{A \in \mathcal{N}_{inv}} ((r_0(A) + \epsilon_z) \cdot (q_A \delta) t + 1) \\
\leq t \cdot q_{inv}^l_{inv, 0} + t(\epsilon_z (1 + \delta |\mathcal{N}_{inv}|) + \delta \cdot \sum_{A \in \mathcal{N}_{inv}} r_0(A) + |\mathcal{N}_{inv}|
\]

\[
\leq t \cdot q_{inv}^l_{inv, 0} + \epsilon_m + |\mathcal{N}_{inv}|
\]

with \( \epsilon_m \) defined in (3.4.2.12). User 0 adds some dummy binary digits until he has

(3.4.2.22) \[ (q_{inv}^l_{inv, 0} + \epsilon_m) \cdot (q_{inv} + \delta) t \]

of them. This is possible if \( t \) is large enough, depending on \( \epsilon_m \) and \( \delta \).

Similarly, user 1 sends symbols of \( I_1 \). For packet \( P_i \) with \( i \leq 0 \) or \( i > a_i \), he sends \( t \) symbols. If \( 0 < i \leq a_i \), he sends \( b_i \) symbols. If \( |\mathcal{N}(A) - q_{\lambda} \cdot t| > \delta t \) for some \( A \in \mathcal{N}_{inv} \), the \( b_i \) symbols are random; otherwise he sends the \( t - N \) symbols corresponding to visible vertices and compresses the remaining \( N I_1 \) symbols to

\[
\sum_{A \in \mathcal{N}_{inv}} \left[ (r_1(A) + \epsilon_z) \cdot N(A) \right]
\]

binary digits. He adds dummy symbols until their number is

(3.4.2.23) \[ (q_{inv, 1}^l + \epsilon_m) \cdot q_{inv} t + |\mathcal{N}_{inv}| \].

Now recall that some extra information must be sent from user 1 to 0. For each packet \( P_i \) with \( 0 < i \leq a_i \), user 1 has exactly \[ t(q_{inv} - \delta) E_1 (n+1) \] bit of extra information to be transmitted (cf. (3.4.2.19)). In the \( i \)th packet, this is done as follows. If for some \( A \in \mathcal{N}_{inv} \) \[ |\mathcal{N}(A) - q_{\lambda} \cdot t| > \delta t \), no extra information is sent. Otherwise, user 1 takes the \[ t(q_{inv} - \delta) E_1 (n+1) \] bit together with the \[ (q_{inv, 1}^l + \epsilon_m) \cdot q_{inv} t + |\mathcal{N}_{inv}| \] that he still has to transmit by (3.4.2.23). If necessary, he adds some information to this until he has

(3.4.2.24) \[ (q_{inv, 1}^l + \epsilon_m + E_1 (n+1)) \cdot (q_{inv} + \delta) t \]
bit. Notice that the latter number is larger than the sum of the two former if $t$ is large enough. At the same time, user 0 still has to transmit the amount of information given by (3.4.2.22). Hence

\[(3.4.2.25)\]

\[
\{ (\mathcal{L}_{\text{inv}}, 0 + \epsilon_1) \cdot (\vartheta_{\text{inv}} + \delta) t \}, \{ (\mathcal{L}_{\text{inv}}, 1 + \epsilon_0 + K_1(n+1)) \cdot (\vartheta_{\text{inv}} + \delta) t \}
\]

bit must be transmitted in this situation. Now we describe how this is done. Let

\[(3.4.2.26)\]

\[
k' := \left\lceil \frac{(\mathcal{L}_{\text{inv}}, 0 + \epsilon_1)(\vartheta_{\text{inv}} + \delta) t}{\mathcal{L}_0(n) - \epsilon_n} \right\rceil = \left\lceil \frac{(\mathcal{L}_{\text{inv}}, 1 + \epsilon_0 + K_1(n+1)) \cdot (\vartheta_{\text{inv}} + \delta) t}{\mathcal{L}_1(n) - \epsilon_n} \right\rceil.
\]

The equality holds because of (3.4.2.15). By assumption $\lambda_n$, a code $C_{(k')}(n)$ with block length $k'$ exists, such that \{

\[
\{ (\mathcal{L}_0(n) - \epsilon_n, k'), (\mathcal{L}_1(n) - \epsilon_n, k') \}
\]

bit can be transmitted with it, with error probability $\leq c_n(\lambda_n) \epsilon_n$, $\epsilon_n = (k')^a n$. We assume that $k'$ is large enough ($k' \geq b'')$ to make this possible. This yields a bound on $t$ and hence on $k$. By (3.4.2.25) and (3.4.2.26), the users can transmit the

\[
\{ (\mathcal{L}_{\text{inv}}, 0 + \epsilon_1) \cdot (\vartheta_{\text{inv}} + \delta) t \}, \{ (\mathcal{L}_{\text{inv}}, 1 + \epsilon_0 + K_1(n+1)) \cdot (\vartheta_{\text{inv}} + \delta) t \}
\]

bit of (3.4.2.25) with this code.

At this point in the proof the 'bootstrapping' idea is used: in the new code $C_k(n+1)$, some of the information is transmitted using an old code $C_{(k')}(n)$. The new code itself can again be used to create another code. This gives rise to an infinite sequence of codes with increasing rates, as will be seen later (at least if $\mathcal{L}_{\text{vis}}$ is higher than $\mathcal{L}(?)$).

In this way all invisible-vertex transmissions, plus the extra information $K_1(n+1)$ for the $i^{th}$ packet are encoded in $k'$ symbols. Therefore all information contained in the packet $\mathcal{P}_i$ is encoded in at most
max \{b_t, t - N + \left[ \frac{(k_{\text{inv}}, 0 + \epsilon_m) (\eta_{\text{inv}} + \delta) t}{L_0(n) - \epsilon_n} \right] \}

symbols. Using \(|N(A) - \eta_A| \leq \delta t\) for \(A \in \mathcal{N}_{\text{inv}}\) and \(N = \sum_{A \in \mathcal{N}_{\text{inv}}} N(A)\), we find

\[ t - N + \frac{(k_{\text{inv}}, 0 + \epsilon_m) (\eta_{\text{inv}} + \delta) t}{L_0(n) - \epsilon_n} + 1 \]

\[ \leq t - (\eta_{\text{inv}} - |\mathcal{N}_{\text{inv}}| \delta) t + \frac{(k_{\text{inv}}, 0 + \epsilon_m) (\eta_{\text{inv}} + \delta) t}{L_0(n) - \epsilon_n} + 1 \leq b_t. \]

Hence each packet \(P_i, 0 < i \leq a_t\), can be lengthened until it contains \(b_t\) symbols.

\(P_0\)

\[ x_0,0, \quad x_0,1, \quad \ldots, \quad x_0,t-1 \quad \text{t symb.} \]

\(P_a\)

\[ x_{a-1,0}, \quad x_{a-1,1}, \quad \ldots, \quad x_{a-1,t-1} \quad \text{t symb.} \]

\(P_l+1\)

\[ x_{l+1,0}, \quad x_{l+1,1}, \quad \ldots, \quad x_{l+1,t-1} \quad \text{t symb.} \]

\(P_{a+t}\)

\[ x_{a+t,0}, \quad x_{a+t,1}, \quad \ldots, \quad x_{a+t,t-1} \quad \text{t symb.} \]

\(P_{a+t+1}\)

\[ x_{a+t+1,0}, \quad x_{a+t+1,1}, \quad \ldots, \quad x_{a+t+1,t-1} \quad \text{t symb.} \]

\(P_{l+t-1}\)

\[ x_{l+t-1,0}, \quad x_{l+t-1,1}, \quad \ldots, \quad x_{l+t-1,t-1} \quad \text{t symb.} \]

(3.4.2.27) Figure. Simultaneous transmission of information in packets.

Summarizing, the situation is as in Figure (3.4.2.27): the users transmit \(a_t - D\) packets containing \(b_t\) symbols each and \((\ell_t - a_t + D)\) packets containing \(t\)
symbols each. The total number of transmissions is \((a_t - D) \cdot b_t + t(\ell_t \cdot a_t + B)\), which is what we had to show.

Decoding is done in the natural way: the decoding functions of \(C_{(k')}(n)\) and \(C_t\) together with the decoding function for the Slepian–Wolf compression scheme enable the two users to find an estimate for each other's message.

Finally we show that the error probabilities \(P_{\varepsilon,0}\) and \(P_{\varepsilon,1}\) (cf. (2.1.7)) are bounded by \(c_{n+1}(\lambda_{n+1})^{e_{n+1}}\) for some \(c_{n+1} > 0\), \(0 < \lambda_{n+1} < 1\), \(a_{n+1} > 0\), \(e_{n+1} = k^q n + 1\) and large enough \(k\). Errors can occur in one of the following ways.

1. An \(i, D < i \leq a_t\), and an invisible vertex \(A\) exist such that in the \(i\)th packet \(|M(A) - q_A^t| > \delta t\), and hence the Slepian–Wolf compression is not applied.

2. An \(i, D < i \leq a_t\), exists such that an error occurs in the Slepian–Wolf compression scheme for the \(i\)th packet.

3. An \(i, D < i \leq a_t\), exists such that the \(\{ (R_{inv,0} + \varepsilon_m, q_{inv} + \delta t) \}, \{ (R_{inv,1} + \varepsilon_m + R_{inv}^{(n+1)}(\ell_t), q_{inv} + \delta t) \}\) bit of the \(i\)th packet are not transmitted correctly with the code \(C_{(k')}(n)\).

4. A message pair \((w_0, w_1)\), \(j \in \{0, 1, \ldots, t-1\}\), exists for which code \(C_t\) is in error (though the symbols in all packets may be transmitted correctly).

The probabilities of these events are called \(P_1, P_2, P_3\) and \(P_4\). We find

\[
P_1 \leq (a_t - D) \cdot \sum_{A \in \mathcal{V}_{inv}} \Pr\{ |M(A) - q_A^t| > \delta t \} \\
\leq (a_t - D) \cdot c_t^{(1)}(\lambda t)^t \{\text{by (3.4.2.11) and the weak law of large numbers}\} \\
\leq c(1)^{(1)}(\lambda(1))^{t}
\]

if \(t\) is large enough, for some \(c_L > 0\), \(0 < \lambda_L < 1\), \(c(1) > 0\), and \(0 < \lambda(1) < 1\).
for some \( c_5, \lambda_5, c(2), \lambda(2) \) and \( t \) large enough. Note that this is the place where we use the exponential error bound in the Slepian–Wolf lemma, (3.4.2.1).

\[
P_3 \leq (a_t - \theta) \cdot c_5(\lambda_5) t^{\gamma} \quad \text{by (3.4.2.21)}
\]

for some \( c_5, \lambda_5, c(2), \lambda(2) \) and \( t \) large.

Now let

\[
c' := c(1) + c(2) + c(3) + c(4) > 0
\]

\[
\lambda' := \max \{ \lambda(1), \lambda(2), \lambda(3), \lambda(4) \} \quad (0 < \lambda' < 1),
\]

Then the total error probability \( P_{e,0} + P_{e,1} \) satisfies

\[
P_{e,0} + P_{e,1} \leq (c')(\lambda')(e'_n) \quad \text{with} \quad e'_n := t(a_n) \leq t.
\]

By definition of \( t \) (see (3.4.2.14)), we have \( t^2 \geq F \cdot k \), where \( F \) depends on \( e_{n+1} \) but not on \( k \). Consequently

\[
e'_n = t(a_n) = t^2(a_n/2) \geq F' \cdot k(a_n/2)
\]

for some \( F' \). With \( a_{n+1} := a_n/2 \) and \( e_{n+1} := k(a_{n+1}) \) we find
\( P_{e,0} + P_{e,1} \leq (c^') (\lambda^')^{\ell n+1} = c_{n+1} (\lambda_{n+1})^{\ell n+1} \)

for some \( c_{n+1} > 0 \) and \( 0 < \lambda_{n+1} < 1 \) depending on \( e_{n+1} \). This concludes the proof of Theorem (3.4.2.10).

Now that we know that every rate \( \ell(n) \) is achievable, we still have to prove that the sequence \( \{\ell(n)\}_{n \in \mathbb{N}} \) converges and that its limit is \( \ell_{vis} \), the visible rate of the bootstrap tree.

(3.4.2.28) Lemma. The sequence \( \{\ell(n)\}_{n \in \mathbb{N}} \) defined in (3.4.2.6) converges. □

Proof: With \( \eta_1 \) as defined in (3.4.2.6.c), first let us assume \( \eta_1 \geq 1 \). It follows by induction that the sequence \( \eta_n \) is nondecreasing and \( \ell_0(n) \) is nonincreasing in \( n \). Indeed, if \( \eta_n \geq \eta_{n-1} \), then by the first equality of (3.4.2.9), \( \ell_0(n) \leq \ell_0(n-1) \) and hence, using (3.4.2.6.c), \( \eta_{n+1} \geq \eta_n \). By the second equality of (3.4.2.9) this implies

\[
1 \geq \frac{\ell_0(n)}{\ell_0(n-1)} = \frac{\ell_{inv,0} + \ell_{vis,0}}{\ell_{inv,0} + \ell_{vis,0}(n-1)}
\]

Hence

\[ \ell_{vis,0} < \ell_0(n-1) \]

for all \( n \). Therefore the sequence \( \ell_0(n) \) is bounded from below by \( \ell_{vis,0} \) and hence it has a limit that we call \( \ell_0 \).

In the case where \( \eta_1 < 1 \), we find analogously that \( \eta_n \) is nonincreasing and \( \ell_0(n) \) is nondecreasing. Now \( \ell_0(n) \) is bounded from above by \( \ell_{vis,0} \) and again the sequence converges to a limit \( \ell_0 \). We define

\[
\mu_n := \frac{\ell_{inv,0} \eta_n}{\ell_{inv,0} + \ell_{vis}}
\]

Since \( \eta_n \ell_1(n-1) = \ell_{inv,1} + \ell_{vis}(n) \) by b and c of (3.4.2.6), we obtain
\[ \forall n > 0 \left[ I_1(n) = \mu_n I_1(n-1) + (1 - \mu_n) \cdot I_\text{vis,1} \right]. \]

In other words, \( I_1(n) \) can be written as a convex combination of \( I_1(n-1) \) and \( I_\text{vis,1} \). This reflects the proof of Theorem (3.4.2.10): in the code with rate \( I(n) \), the information of the visible vertices is transmitted with rate \( I_\text{vis} \) and the information of the invisible vertices is transmitted with a code of rate \( I(n-1) \).

If \( I_1(0) \leq I_\text{vis,1} \), then \( I_1(n) \) will be a nondecreasing sequence bounded from above by \( I_\text{vis,1} \); if \( I_1(0) \geq I_\text{vis,1} \), then the sequence is nonincreasing with lower bound \( I_\text{vis,1} \). In either case \( I_1(n) \) converges to a limit, say \( I_1 \). This concludes the proof of the lemma.

Now we investigate the limit \( (I_0, I_1) \). If we substitute
\[
I_0(n-1) = I_0(n) = I_0, \quad I_1(n-1) = I_1(n) = I_1
\]
in (3.4.2.6), then we find
\[
I_0 = \frac{V_\text{inv} I_\text{inv,0} + V_\text{vis} I_\text{vis,0}}{V_\text{inv} I_\text{inv,0}/I_0 + V_\text{vis}},
\]
\[
I_1 = \frac{V_\text{inv} I_\text{inv,0}/I_0 + V_\text{vis} I_\text{vis,1}}{V_\text{inv} I_\text{inv,0}/I_0 + V_\text{vis}}.
\]

By multiplying both sides of these equalities by \( V_\text{inv} I_\text{inv,0}/I_0 + V_\text{vis} \) we find
\[
I_0 = I_\text{vis,0}, \quad I_1 = I_\text{vis,1}.
\]

This completes the proof of the achievability of the bootstrap rate \( I_\text{vis}(T) \).
3.5. Relations between strategy trees

Here we present the proofs of the theorems in the previous sections. For reference purposes, we once more list the theorems themselves.

(3.2.1.2) Theorem. $\forall N \in \mathbb{N} \ [ \co \mathcal{T}(N) \subset C], \ \lim_{N \to \infty} \mathcal{T}(N) = \bar{\mathcal{T}} = C$.  \hfill $\Box$

(3.2.2.3) Theorem. $\forall N \in \mathbb{N} \ [ \co \mathcal{T}^D(N) \subset C], \ \lim_{N \to \infty} \mathcal{T}^D(N) = \bar{\mathcal{T}^D} = C$.  \hfill $\Box$

(3.3.1.4) Theorem. $\forall N \in \mathbb{N} \ [ \co \mathcal{V}(N) \subset C], \ \lim_{N \to \infty} \mathcal{V}(N) = \bar{\mathcal{V}} = C$.  \hfill $\Box$

(3.3.2.2) Theorem. $\forall N \in \mathbb{N} \ [ \co \mathcal{V}^D(N) \subset C], \ \lim_{N \to \infty} \mathcal{V}^D(N) = \bar{\mathcal{V}^D} = C$.  \hfill $\Box$

(3.3.4.6) Theorem. $\forall N \in \mathbb{N} \ [ \co \mathcal{V}^O(N) \subset C], \ \lim_{N \to \infty} \mathcal{V}^O(N) = \bar{\mathcal{V}^O} = C$.  \hfill $\Box$

(3.4.1.8) Theorem. $\forall N \in \mathbb{N} \ [ \co \mathcal{B}(N) \subset C], \ \lim_{N \to \infty} \mathcal{B}(N) = \bar{\mathcal{B}} = C$.  \hfill $\Box$

In the proofs we will use several inclusions between the regions defined, like $\mathcal{T}(N) \subset \mathcal{V}(N)$ and $\mathcal{T}(N) \subset \mathcal{F}(2N)$. Figure (3.5.1) shows some of the relations that will be used. In this figure, arrows denote inclusions: an arrow pointing from region $A$ to $B$ means $A \subset B$.

(3.5.1) Figure. Relations between the strategy trees.
The inclusions pictured in Figure (3.5.1) are easy to prove. In Section (3.4.2) we have shown that \( B(N) \subseteq C \) for all \( N \). Since every \( V(N) \) tree can be regarded as a \( B(N) \) tree (just assign type 'visible' to all vertices), we have \( V(N) \subseteq B(N) \). In the same way it follows that \( V^D(N) \subseteq V(N), F(N) \subseteq V(N) \), et cetera: if we go one step 'down' in the figure, we add restrictions to the trees considered, so the set of rates is a subset of the one directly above it.

It is a well–known fact that, if a set of rates \( \Lambda \) is achievable for the TWC, then also the region \( \text{co} \, \Lambda \) is achievable. In fact, this follows from time sharing and was already used by Shannon [SHA61] in the definition of the inner bound, (2.2.2). Together with the above inclusions, this proves \( \forall N \in \mathbb{N} \left[ \text{co} \, F(N) \subseteq C \right] \) and similar statements for all other types of trees mentioned in the list of theorems. The second part of each theorem will be proved using the following lemma.

(3.5.2) Lemma. Suppose the following holds for a sequence of regions \( A_N \subseteq \mathbb{R}^2 \) and for all \( \mathbf{a} \in \mathbb{R}^2 \) (here \( d(\mathbf{a}, A_N) \) is defined as in (2.2.7)):

\[
\exists N_0 [ \mathbf{a} \in A_{N_0} ] \Rightarrow \forall \varepsilon > 0 \exists N_1 \forall N > N_1 \left[ d(\mathbf{a}, A_N) < \varepsilon \right].
\]

Then \( \{ A_N \}_{N \in \mathbb{N}} \) converges to a limit \( \Lambda \), with

\[ \Lambda := \bigcup_{N \in \mathbb{N}} A_N. \]

Proof: First we prove that (i) of Definition (2.2.8) holds. Let \( \mathbf{a} \in \Lambda \) and \( \varepsilon > 0 \). By definition of closure, an \( N_0 \in \mathbb{N} \) and \( \mathbf{a}_0 \in A_{N_0} \) exist such that \( d(\mathbf{a}, \mathbf{a}_0) < \varepsilon/2 \).

By the assumption in the lemma, we have

\[
\exists N_1 \forall N > N_1 \left[ d(\mathbf{a}_0, A_N) < \varepsilon/2 \right].
\]

Hence, for such an \( N_1 \) and \( N > N_1 \), we have

\[
d(\mathbf{a}, A_N) = \inf \{ d(\mathbf{a}, \mathbf{a}_N) | \mathbf{a}_N \in A_N \} \\
\leq \inf \{ d(\mathbf{a}, \mathbf{a}_0) + d(\mathbf{a}_0, \mathbf{a}_N) | \mathbf{a}_N \in A_N \}.
\]

- 88 -
which proves (i). Now let $a \notin \bar{A}$, so $a$ is in the complement of $\bar{A}$. Since this set is open, $a$ is an interior point of the complement. Hence an $\varepsilon > 0$ exists for which

\[ \forall \bar{a} \in \bar{A} \ [d(a, \bar{a}) > \varepsilon], \text{ or} \]

\[ \forall \bar{A} \in |N| \ [d(a, \bar{A}) > \varepsilon]. \]

This proves that \((ii)\) of Definition (2.2.8) holds.

Using this lemma, we will now prove

\[ \lim_{N \to \infty} \mathcal{F}(N) = \mathbb{F} \]

and the corresponding statements for the other regions. First we show that the condition in Lemma (3.5.2) holds for the sequences $\mathcal{F}(N)$. Let $N_0 \in \mathbb{N}$, $\varepsilon > 0$ and $\bar{f} \in \mathcal{F}(N_0)$. In Section 3.2.2 we showed $\bar{f} \in \mathcal{F}(tN_0)$ for all $t \in \mathbb{N}$. Now we prove

\[ (3.5.3) \forall k \in \mathbb{N} \forall \bar{f}_N \in \mathcal{F}(N) \ d(\bar{f}_N, \mathcal{F}(N+1)) = 0(1/N), \ N \to \infty. \]

Consider an $\mathcal{F}(N)$ tree with rate $\bar{f}_N$. Its vertices are labeled $A_i$, $i = 0, 1, 2, \ldots$. We append $K$ 'layers' of vertices to the leaves in the tree in which no uncertainty is resolved (e.g. both users send a single input symbol for the whole region). This results in an $\mathcal{F}(N+K)$ tree with rate $\bar{f}_{N+K}$.

\[ \bar{f}_{N+K} = \sum_i w(A_i) \sigma(A_i) \frac{N}{N+K} = \bar{f}_N \left(1 - \frac{1}{N+K}\right). \]

Therefore

\[ d(\bar{f}_N, \bar{f}_{N+K}) = \|\bar{f}_{N+K} - \bar{f}_N\| = \|\bar{f}_N \cdot \frac{1}{N+K}\| = \|\bar{f}_N\| \cdot 0(1/N), \ N \to \infty, \]

which proves (3.5.3) because $\bar{f}_N$ is bounded. It follows that for $N = tN_0 + K$ with $t \geq 1$ and $0 \leq K < N_0$:

\[ d(\bar{f}, \mathcal{F}(N)) = d(\bar{f}, \mathcal{F}(tN_0 + K)) = 0(1/t), \ t \to \infty. \]
Therefore, if $N$ is large enough,

$$d(f, T(N)) < \varepsilon.$$ 

Now we can apply Lemma (3.5.2), yielding

$$\lim_{N \to \infty} T(N) = \tilde{f}.$$ 

We will apply the same reasoning to prove

$$\lim_{N \to \infty} T^\omega(N) = \tilde{T}.$$ 

Let $N_0 \in \mathbb{N}$, $\varepsilon > 0$ and $f \in T(N_0)$. As argued in Section 3.2.2 and 3.3.3, we have $f \in T^\omega(tN_0)$ for all $t \in \mathbb{N}$. By appending $k$ layers of dummy vertices to an $T^\omega(tN_0)$ tree we obtain an $T^\omega(tN_0+k)$ tree with the same rate as computed above. Therefore again

$$d(f, T^\omega(tN_0+k)) = o(1/t), \quad t \to \infty.$$ 

Hence $d(f, T^\omega(N)) < \varepsilon$ if $N$ is sufficiently large. Lemma (3.5.2) yields

$$\lim_{N \to \infty} T^\omega(N) = \tilde{T}.$$ 

For the regions $V(N)$ appearing in Theorem (3.3.1.4) we have to change the argument a little, but we can still apply Lemma (3.5.2). Let $N_0 \in \mathbb{N}$, $\varepsilon > 0$ and $v \in V(N_0)$. Then $v \in V(tN_0)$ for all $t \in \mathbb{N}$, as before. We will prove

$$\forall f \in \mathbb{N} \forall v \in V(N) \quad d(v, V(N_0+k)) = o(1/N), \quad N \to \infty.$$ 

Let $k \in \mathbb{N}$. Consider a $V(N)$ tree with rate $\nu_N$ and vertices $A_0, A_1, \ldots$. Let $A_z$ be a leaf at level $z-1$ of the tree. We change the tree into a $V(N+k)$ tree by adding a path of $k$ dummy vertices (with rate $\nu$) below vertex $A_z$. Call the new vertices $A_0^z, A_1^z, \ldots, A_k^z$, then $w(A_j^z) = w(A_0^z)$ for all $j$ and $w(A_i^z) \leq w(A_0^z)$. Since $A_0^z$ has $N$ predecessors in the tree, the weight of the $V(N)$ tree satisfies
Strategy Trees for TWCs

\[ \sum_{i} w(A_i) \geq \sum_{i} w(A_i) = N \cdot w(A_0) \]

Hence the rate of the \( V(N + l) \) tree equals \( v_{N+l} \).

\[ v_{N+l} = \frac{\sum_{i} w(A_i) \sigma(A_i)}{\sum_{i} w(A_i) + \sum_{j} w(A_j)} = \frac{\sum_{i} w(A_i) \sigma(A_i)}{\sum_{i} w(A_i) + l \cdot w(A_0)} \]

Hence

\[ d(v_{N}, v_{N+l}) = \| v_{N+l} \| \]

Since

\[ \sum_{i} w(A_i) + l \cdot w(A_0) \geq (N+l) \cdot w(A_0) \]

we find

\[ d(v_{N}, v_{N+l}) \leq \| v_{N} \| \cdot \frac{l}{N+l} = O(1/N) \text{, } N \to \infty. \]

This proves (3.5.4). With \( N = tN_0 + k \), \( t \geq 1 \) and \( 0 \leq k < N_0 \), we find

\[ d(v, V(N)) = d(v, V(tN_0 + k)) = O(1/t) \text{, } t \to \infty. \]

Therefore \( d(v, V(N)) < \varepsilon \) if \( N \) is large enough. By Lemma (3.5.2), this proves

\[ \lim_{N \to \infty} V(N) = \bar{V}. \]

The above proof for \( V(N) \) trees can be applied to \( V^2(N) \) and \( V^0(N) \) trees as well, in the same way as the proof for \( F(N) \) and \( F^2(N) \) trees is similar. However, the reader should be aware of the implication of 'taking \( t \) copies of a \( V^0(N) \) tree'
(necessary to show \( \nu \in \mathcal{V}^0(N) \Rightarrow \nu \in \mathcal{V}^0(tN) \)). If the \( \mathcal{V}^0(N) \) tree has thresholds on an \((N_0^t+1) \times (N_1^t+1)\) grid, then the \( \mathcal{V}^0(tN) \) tree has thresholds on an \((N_0^t+1) \times (N_1^t+1)\) grid and it represents a variable length code with error probability zero for \((N_0^t, N_1^t)\) messages. Since the rate of the tree (and the rate of the code) is not affected by this, the rest of the proof can be continued as before. We find

\[
\lim_{N \to \infty} \mathcal{V}^0(N) = \mathcal{V}^0 \quad \text{and} \quad \lim_{N \to \infty} \mathcal{V}^0(N) = \mathcal{V}^0.
\]

Finally we show that Lemma (3.5.2) can be applied to the regions \( B(N) \).

Let \( \varepsilon > 0, \ N_0 \in \mathbb{N}, \ h \in B(N_0) \). Consider a \( B(N_0) \) tree with visible rate \( h \) and let \( T \) be the corresponding \( V(N_0) \) tree with type assignment, for some \( N_0' \). If we take \( t \) copies of the tree \( T \) in the usual way, and we define the types of the new vertices according to the types in \( T \), then it is easy to check that this yields a \( B(tN_0) \) tree with rate \( h \). Therefore \( h \in B(tN_0) \). Next, we show

\[
(3.5.5) \quad \forall K \in \mathbb{N} \forall \mathcal{N} \in B(N) \quad d(h_N, B(N+K)) = O(1/N), \ N \to \infty.
\]

Let \( I \in \mathbb{N}, \ N \in \mathbb{N} \). Let \( h_N \) be the visible rate of a \( B(N) \) tree constructed from a \( V(N') \) tree with type assignment. Let \( A_x \) be a visible vertex at visible distance \( N-1 \) (i.e., the maximum possible visible distance) from the root of the tree. We remove any invisible vertices below \( A_x \), which does not change the visible rate. Consider one of the connected regions remaining after the resolution in \( A_x \). We partition this region into rootlike rectangles by slicing (cf. Lemma (3.3.2.3)), in such a way that at least one rectangle remains, say \( I \), having area at most \( 1/(KN) \). The slicing vertices appear in the tree below \( A_x \) and they are given type 'invisible' (by Lemma (3.3.2.3), these vertices have the PBV property). Finally we add a path of \( I \) visible vertices (thus creating a \( B(N+K) \) tree) in which a dummy resolution in \( I \) is done. Call these vertices \( A_0^I, A_1^I, ..., A_{k-1}^I \), then
Strategy Trees for TWCs

For all $j$, let $\mathcal{T}_j$ be a tree. Now we define the rate of this tree as

$$r(\mathcal{T}_j) = \frac{\sum_{\mathcal{A}_i \in \mathcal{A}_j} w(\mathcal{A}_i) \mathcal{I}(\mathcal{A}_i)}{\sum_{\mathcal{A}_i \in \mathcal{A}_j} w(\mathcal{A}_i) + \sum_{j' \neq j} \sum_{\mathcal{A}_i \in \mathcal{A}_{j'}} w(\mathcal{A}_i)}.$$ 

Since

$$\sum_{j} \sum_{\mathcal{A}_i \in \mathcal{A}_j} w(\mathcal{A}_i) \leq 1/N$$

and

$$\sum_{\mathcal{A}_i \in \mathcal{A}_{j'}} w(\mathcal{A}_i) \geq 1,$$

we find

$$||x_{N+1} - x_N|| = ||x_N|| \cdot (1 + O(\frac{1}{N})), \quad N \to \infty$$

which proves (3.5.5). For all $\varepsilon > 0$ and sufficiently large $N$, we can now prove – as before – that $d(\mathcal{B}, \mathcal{B}(N)) < \varepsilon$. By Lemma (3.5.2) we can conclude

$$\lim_{N \to \infty} B(N) = \mathcal{B}.$$

The last part of the six theorems that we have to prove is

$$\mathcal{F} = \mathcal{F}^0 = \mathcal{V} = \mathcal{V}^0 = \mathcal{V}^0 = \mathcal{B} = \mathcal{C}.$$

First we prove that $\mathcal{F}$, $\mathcal{V}$ and $\mathcal{B}$ are equal to $\mathcal{C}$. From Figure (3.5.1) it follows that $\mathcal{F}(N) \subseteq \mathcal{V}(N) \subseteq \mathcal{B}(N) \subseteq \mathcal{C}$ for all $N$. Hence, for $N \to \infty$, we also find $\mathcal{F} \subseteq \mathcal{V} \subseteq \mathcal{B} \subseteq \mathcal{C}$.

Furthermore we have the following lemma.

(3.5.6) Lemma. For all $N$, we have $co \mathcal{F}(N) \subseteq \mathcal{F}$. \hfill \Box

Proof: Let $N \in \mathbb{N}$, let $\mathcal{T}_0$ and $\mathcal{T}_1$ be $F(N)$ trees and let $\lambda \in (0, 1)$. For $t \in \mathbb{N}$, we will construct an $F(tN+1)$ tree $\mathcal{T}$ such that $\mathcal{F}(\mathcal{T}) = \lambda \mathcal{F}(\mathcal{T}_0) + (1-\lambda) \mathcal{F}(\mathcal{T}_1)$ for $t \to \infty$. See Figure (3.5.7). This will prove the fact that $\lambda \mathcal{F}(\mathcal{T}_0) + (1-\lambda) \mathcal{F}(\mathcal{T}_1) \in \mathcal{F}$.

In the root of the tree $\mathcal{T}$ we perform row slicing. By choosing appropriate input symbols (say $z_0(a)$ and $z_0(b)$ for user 0 and $z_1$ for user 1) we can partition the unit square into two rootlike rectangles, one with area $\lambda$ and the other with
area 1−λ. To the rectangle with area λ we append t copies of the tree $T_0$ in the usual way. We partition the other region according to t copies of $T_1$.

\[
\begin{array}{c|c|c}

& z_1 & \\
\hline
z_0(a) & y_1(a) & \lambda \\
\hline
z_1(b) & y_1(b) & 1-\lambda
\end{array}
\]

\[
\begin{array}{c|c}

\text{tree } T_0 & \lambda \\
\hline
\text{tree } T_1 & 1-\lambda
\end{array}
\]

\[
\begin{array}{c|c}

\text{tree } T_0 & \lambda \\
\hline
\text{tree } T_1 & 1-\lambda
\end{array}
\]

(3.5.7) Figure. Combining two $F(N)$ trees into an $F(tN+1)$ tree.

It is easy to see that $T$ is an $F(tN+1)$ tree with rate

\[
\lambda(T) = \{ 1 \cdot \lambda_{\lambda}, 0 \} + \lambda tN \cdot \lambda(T_0) + (1-\lambda) tN \cdot \lambda(T_1) \}/\{tN + 1\},
\]

which tends to $\lambda \cdot \lambda(T_0) + (1-\lambda) \cdot \lambda(T_1)$ if $t \to \infty$. Thus $\lambda \lambda(T_0) + (1-\lambda) \lambda(T_1) \in \mathcal{F}$ and the lemma is proved.

By Theorem (2.2.6), we know that

\[
\lim_{N \to \infty} co \mathcal{F}(N) = C.
\]

With Lemma (3.5.6) this yields $C \subseteq \mathcal{F}$. We have already proved $\mathcal{F} \subseteq C$, so $\mathcal{F} = C$.

Since we know $\mathcal{F} \subseteq \mathcal{V} \subseteq \mathcal{B} \subseteq C$, it immediately follows that $\mathcal{V} = \mathcal{B} = C$. 

- 94 -
To prove that $\mathcal{V}^0 = \mathcal{C}$, we need to think about the definition of a capacity region. Consider a rate point $R$ lying in $\mathcal{C}$. Then, by Definition (2.1.9) and (2.1.10), a sequence of fixed length codes, say $\{\mathcal{C}(n)\}_{n \in \mathbb{N}}$, can be constructed such that the rate of code $\mathcal{C}(n)$ tends to $R$, the block length tends to infinity and the error probabilities $P_{e,0}$ and $P_{e,1}$ tend to 0 with increasing $n$. Let code $\mathcal{C}(n)$ have parameters $\{M_0(n), M_1(n), N(n)\}$. Hence

$$(3.5.8) \quad \lim_{n \to \infty} \left( \frac{\log(M_0(n))}{N(n)}, \frac{\log(M_1(n))}{N(n)} \right) = R = (R_0, R_1).$$

As we have seen in Section 3.2.3, the code $\mathcal{C}(n)$ can be represented in an $F(N(n))$ tree with thresholds on an $(M_0(n)+1) \times (M_1(n)+1)$ grid. Let $P_{e,0}$ and $P_{e,1}$ be $\varepsilon(n)$ for this code. We will change the $F(N(n))$ tree into a $F^0(N')$ tree for some $N' > N(n)$, to be defined later. Suppose that the two users use the code $\mathcal{C}(n)$. They partition the unit square according to the tree, in $N(n)$ steps, after which they try to determine the basic rectangle in which the message pair $(w_0, w_1)$ lies. In $\varepsilon(n) \cdot M_0(n) \cdot M_1(n)$ cases, on the average, they will not be able to do so. In that case, two or more basic rectangles will be connected; indeed, if there is just one basic rectangle remaining after transmission, then this is the correct one and it is known by both users.

If user 1 does not yet know the message of user 0, then both users are aware of this, and row slicing can be performed to tell user 1 the message $w_0$. At worst, this may take $\lceil \log(M_0(n)) \rceil$ transmissions. The row slicing vertices will appear below some of the leaves of the $F(N(n))$ tree. In the same way column slicing helps user 0 to determine the message $w_1$ sent by user 1. We need a path of at most $\lceil \log(M_1(n)) \rceil$ column slicing vertices in the tree for this. For every basic rectangle that could not be determined in $N(n)$ transmissions, we add dummy vertices until we have $N'$ of them, where
Chapter 3

\[ N' := N(n) + \lfloor \log(N_0(n)) \rfloor + \lfloor \log(N_1(n)) \rfloor. \]

Clearly we have constructed a \( F^0(N') \) tree, since the unit square is partitioned into basic rectangles. The rate of this tree is, by Theorem (3.3.4.4),

\[ \left( \frac{\log(N_0(n))}{N}, \frac{\log(N_1(n))}{N} \right) \]

where \( \bar{N} \) is the average codeword length. This rate lies in \( \mathcal{Y}^0 \). With probability \( 1 - \varepsilon(n) \), the \( N(n) \) transmissions in the \( F(N(n)) \) tree are sufficient to determine the message pair; with probability \( \varepsilon(n) \), \( N' \) transmissions are used. Hence

\[ \bar{N} = (1 - \varepsilon(n)) \cdot N(n) + \varepsilon(n) \cdot (N(n) + \lfloor \log(N_0(n)) \rfloor + \lfloor \log(N_1(n)) \rfloor) \]

\[ = N(n) + \varepsilon(n) \cdot (\lfloor \log(N_0(n)) \rfloor + \lfloor \log(N_1(n)) \rfloor). \]

Because of this we have

\[ \lim_{n \to \infty} \frac{\bar{N}}{N(n)} = \lim_{n \to \infty} \left\{ \frac{N(n)}{N(n)} + \varepsilon(n) \cdot \left( \frac{\lfloor \log(N_0(n)) \rfloor}{N(n)} + \frac{\lfloor \log(N_1(n)) \rfloor}{N(n)} \right) \right\} \]

\[ = 1 + \lim_{n \to \infty} \varepsilon(n) \cdot (L_0 + L_1) = 1 \]

by (3.5.8). Consequently

\[ \lim_{n \to \infty} \left( \frac{\log(N_0(n))}{N}, \frac{\log(N_1(n))}{N} \right) = \lim_{n \to \infty} \left( \frac{\log(N_0(n))}{N(n)}, \frac{\log(N_1(n))}{N(n)} \right). \]

Since the rate on the right equals \( \bar{R} \), this proves that every rate in \( C \) can be approximated with rate points in \( \mathcal{Y}^0 \). Therefore \( C \subset \bar{\mathcal{Y}}^0 \). Together with the inclusions pictured in Figure (3.5.1), we find

\[ C = \bar{\mathcal{Y}}^0 \quad \text{and} \quad C = \mathcal{Y}^0. \]

Finally we show that \( \mathcal{Y}^0 \subset \mathcal{Y}^0 \). This yields, with Figure (3.5.1), \( \mathcal{Y}^0 = \mathcal{Y}^0 = C \)

and therefore the six theorems will be proved. This is done by changing a \( F^0(\bar{N}) \)
tree into an $F^\Omega(N^2)$ tree that has almost the same rate as the $F^\Omega(N)$ tree, if $N^2$ is sufficiently large. A similar technique was applied in [OVE87] to show $\bar{F} \supset \bar{V}$.

Let $N \in \mathbb{N}$ and let $T$ be a $F^\Omega(N)$ tree. Let $t \in \mathbb{N}$ and change the tree into a $F^\Omega(tN)$ tree by adding as many copies of the original tree as possible, without exceeding the maximum depth $tN-1$. This means that the $F^\Omega(N)$ strategy tree is repeated in every rootlike rectangle that remains in the tree, unless it appears at a level with number at least $(t-1)N+1$; in that case there is no room for one more copy below this vertex. The $F^\Omega(tN)$ tree has the same rate as the original tree, $\mathbb{K}(T)$. Now we complete the tree by adding dummy vertices with rate $Q$, until we have an $F^\Omega(tN)$ tree. Call this tree $T_t$. Its rate, $\mathbb{K}(T_t)$, satisfies

$$\mathbb{K}(T_t) = \mathbb{K}(T) \cdot (1 - w_d/(tN)),$$

if $w_d$ is the total weight of all dummy vertices. By the construction method we have $w_d \leq N$, or $w_d/(tN) = 0(1/t)$, $t \to \infty$. Hence we can approximate the rate $\mathbb{K}(T)$ in $\mathbb{F}^\Omega$ arbitrarily closely with rates in $\mathbb{F}^\Omega$, if $t$ is large enough. Thus $\mathbb{F}^\Omega \supset \bar{F}^\Omega$.

We have now proved that the sequences $\mathbb{F}(N)$, $\mathbb{F}^\Omega(N)$, $\mathbb{V}(N)$, $\mathbb{V}^\Omega(N)$, $\mathbb{V}^\Omega(N)$ and $\mathbb{B}(N)$ tend to the capacity region for large $N$. However, there is one more type of strategy tree that we have defined in Section 3.2, for which we could not prove the analogue of these theorems. This is the $F^\Omega(K)$ tree, defined in Definition (3.2.3.2). We have seen that every $F^\Omega(K)$ tree describes a fixed length code with error probability 0, and that every code with error probability 0 can be pictured in an $F^\Omega(N)$ tree. Therefore, if we define the zero-error capacity region as done in Definition (3.5.9), Theorem (3.5.10) is immediate.

(3.5.9) Definition. A rate $(\mathbb{L}_0, \mathbb{L}_1)$ is called zero-error achievable for a TWC $K$ if for all $\varepsilon > 0$ and large enough $N$, an $(\mathbb{L}_0, \mathbb{L}_1, N)$ code exists with error probability
Chapter 3

\[ P_{\epsilon,0} = P_{\epsilon,1} = 0 \] and

\[
\frac{\log(N_0)}{N} > R_0 - \epsilon, \quad \frac{\log(N_1)}{N} > R_1 - \epsilon.
\]

The zero-error capacity region for \( \mathcal{K} \), denoted by \( C^0_\mathcal{K} \) (or just \( C^0 \)), is defined as the set of all zero-error achievable rates for \( \mathcal{K} \).

(3.5.10) Theorem. \( \forall N \in \mathbb{N} \) \( [ \co f(\mathcal{K}) \subset C^0 ] \), \( \lim_{N \to \infty} f(\mathcal{K}) = f^0 = C^0. \)

It should be noted that, just like with ordinary one-way channels, the zero-error capacity region is not necessarily equal to the capacity region as defined in Definition (2.1.10). Unfortunately, very little is known about the zero-error capacity for TWCs. In the case of the BMC, we only have some lower bounds following from constructions (see Section 3.6), but we do not even know whether the Shannon inner bound is zero-error achievable for this channel.

3.6 Examples of strategy trees

3.6.1. Strategy trees for the Binary Multiplying Channel

In this section we take the BMC as our two-way channel \( \mathcal{K} \) and we present rate pairs in the regions \( \mathcal{F}(N), \mathcal{F}^0(N), \mathcal{F}^0(N), \mathcal{V}(N), \) et cetera, for some small values of \( N \). We are interested in the maximum achievable average rate for the two users, \( (R_0 + R_1)/2 \). Therefore we define

\[
f(N) := \sup\{ (R_0 + R_1)/2 \mid (R_0, R_1) \in \mathcal{F}(N) \}.
\]

Similarly we can consider \( f^0(N), f^0(N), v(N), \) et cetera, as the supremum of all average rates in the regions \( \mathcal{F}^0(N), \mathcal{F}^0(N), \mathcal{V}(N), \ldots \).
Although $\mathcal{F}(1)$ is a convex region for the BMC (which has been proved by the author), it is not certain that the maximum average rate for some region is always achieved for a rate point with $I_0 = I_1$. We return to this later. This makes it very difficult to find good lower bounds on the average rate.

Even if we restrict our attention to symmetrical rates $(l,l)$, this does not make the search much easier. Since the BMC is a symmetrical channel (it is a $T$-channel with $f_K(x_0, x_1) = f_K(x_1, x_0)$ for all inputs), the obvious thing to do if we want to find a symmetrical rate is to construct a symmetrical tree. This is a tree that can be derived from some $F(N)$ tree for which the strategy functions $\sigma_0$ and $\sigma_1$ are equal. However, strangely enough, it cannot be proved that for each symmetrical rate in some region (say $A(N)$), a symmetrical $A(N)$ tree exists having this rate. Still, no counterexample has been found yet, and all results for symmetrical rates in this section have been obtained using symmetrical trees.

The tables below list the best lower bounds known for the average rates in the various regions. In some cases these correspond to symmetrical rates (and symmetrical trees). These cases are denoted with '(s)' in the tables.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$f^0(N)$ lower bound</th>
<th>$M(N)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.50000 (s)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0.52832 (s)</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>0.50000 (s)</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>0.51699 (s)</td>
<td>6</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>16</td>
<td>0.58402 (s)</td>
<td>650</td>
</tr>
</tbody>
</table>

(3.6.1.1) Table. Lower bounds on $f^0(N)$.

The results in Table (3.6.1.1) are taken from [OVE85]. In this table, $M(N)$ denotes the number of messages (for both users) of the zero-error code that
corresponds to the $F^{O}(N)$ tree. Hence the listed lower bound for $f^{O}(N)$ equals $\log(N(N))/N$. It should be noted that the result for $N = 3$ stems from the code shown in Figure (3.2.3.1). The bounds for $N = 6,7,\ldots,15$ have been omitted since they are all lower than the bound for $N = 16$; they can be found in [OVE85]. The value 0.58402 for $N = 16$ is the highest rate known to be achievable with any $F^{O}(N)$ tree. Therefore this is the sharpest lower bound for the zero-error capacity region of the BMC in the symmetrical case.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$f^{O}(N)$ lower bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.57549</td>
</tr>
<tr>
<td>3</td>
<td>0.59402</td>
</tr>
</tbody>
</table>

\text{(3.6.1.2) Table. Lower bounds on $f^{O}(N)$.}

Although, for $N \in \{2,3\}$, the values of the bounds for $f^{O}(N)$ are considerably higher than the bounds for $f^{O}(N)$, we do not list values for larger $N$. This is partly because we have already argued in Section 3.2.2 that $F^{O}(N)$ trees do not provide interesting bounds on the capacity region, but it is also true that these bounds are hard to compute for larger values of $N$. Computing the maximum rate of an $F^{O}(N)$ tree for fixed $N$ is a finite search problem since the thresholds are confined to a grid, but for $F^{D}(N)$ trees the thresholds can assume values in [0,1] which makes it more complex to find the optimum rate.

The two trees that achieve the rates in Table (3.6.1.2) are given in Figures (3.6.1.3) and (3.6.1.4). In Figure (3.6.1.3), $t_1$ and $u_1$ denote the first thresholds for the two users: $t_1 = Pr\{X_{00} = 1\}$, $u_1 = Pr\{X_{10} = 1\}$. (An arrow to the left indicates the region where $Y = 0$, and an arrow to the right points to the region with $Y = 1$. This holds in all pictures of trees for the BMC in this section.)
The average rate of 0.57549 corresponds to a choice of $t_1 = u_1 = 0.60300$. The reader should notice that the highest possible rate for any symmetrical $F^2(2)$ tree is less than 0.57549, since for a symmetrical tree we necessarily have rate 0 in the square region on the right.

(3.6.1.4) Figure. $F^2(3)$ tree for the BMC.
Chapter 3

The best $F^D(3)$ tree is shown in Figure (3.6.1.4). In this figure, $F^D(2)$ denotes subdivision of the rectangle according to the $F^D(2)$ tree of Figure (3.6.1.3). The parameters $t_1$ and $u_1$ have the same meaning as before, and $t_2$ and $u_2$ are the thresholds for the left-hand vertex at level 1:

$$t_2 = Pr\{I_{01}=1|I_{00}=1, I_0=0\}, \quad u_2 = Pr\{I_{11}=1|I_{10}=1, I_0=0\}.$$ 

This is in accordance with the notation used in [SME83]; a similar notation will be used in the figures to come. For the tree with rate 0.59402 we find:

$$t_1 = u_1 = 0.66752, \quad t_2 = u_2 = 0.33332.$$ 

<table>
<thead>
<tr>
<th>$N$</th>
<th>$f(N)$ lower bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.61695 (s)</td>
</tr>
<tr>
<td>2</td>
<td>0.61695 (s)</td>
</tr>
<tr>
<td>3</td>
<td>0.61964 (s)</td>
</tr>
<tr>
<td>4</td>
<td>0.62138</td>
</tr>
<tr>
<td>5</td>
<td>0.62333</td>
</tr>
<tr>
<td>6</td>
<td>0.62405</td>
</tr>
<tr>
<td>7</td>
<td>0.62496</td>
</tr>
</tbody>
</table>

(3.6.1.5) Table. Lower bounds on $f(N)$. 

Most entries in Table (3.6.1.5) were found by Smeets (cf. [SME83]). They were found via a computer program that optimizes a function (the average rate) over many parameters (the thresholds in the tree). The bounds on $f(1)$ and $f(2)$ are equal to the symmetrical rate point of the Shannon inner bound. The tree corresponding to the bound for $f(3)$ is also symmetrical, but for $N = 4$, Smeets found an asymmetrical tree achieving a higher rate (0.62136) than any computed symmetrical tree. Close inspection of this tree showed that it could not be the global optimum over all $F(4)$ trees since its division of the rectangle remaining at level 1 of the tree was already suboptimal. Therefore we replaced this part of the tree with the optimum $F(3)$ tree (with rate 0.61964, as found by Smeets). After re-optimization this gave the improvement to 0.62138 listed in the table. The
thresholds for this optimum hardly differ from the ones found by Smeets and thus they are not listed here; they may be found in [SME83].

We are not convinced that this new point is the global optimum, because the tree is still asymmetrical. Since the function to be optimized has 54 parameters and has a large number of local optima, it is extremely hard to find its global optimum. This of course holds a fortiori for the $P(N)$ trees with $N = 5, 6$ and $7$. For $N = 5$, we replaced the $P(4)$ tree (with rate 0.62136) in the subsquare by the new one with rate 0.62138, but this did not improve the $P(5)$ tree significantly (to wit, its rate increased from 0.623326 to 0.623334). For this reason the values in the table for $N \geq 5$ are equal to those found by Smeets.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$v^0(N)$ lower bound</th>
<th>$\mathcal{N}(N)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.57143 (s)</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0.59436 (s)</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>0.59259 (s)</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>0.59273 (s)</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>0.60188 (s)</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>0.60989 (s)</td>
<td>16</td>
</tr>
<tr>
<td>8</td>
<td>0.61079 (s)</td>
<td>17</td>
</tr>
<tr>
<td>9</td>
<td>0.61111 (s)</td>
<td>39</td>
</tr>
<tr>
<td>10</td>
<td>0.61154 (s)</td>
<td>57</td>
</tr>
</tbody>
</table>

(3.6.1.6) Table. Lower bounds on $v^0(N)$.

All rates listed in Table (3.6.1.6) were found by Van der Leur [LEU84] in an attempt to construct variable length codes having rates outside $\mathcal{G}_4$. Although she did not succeed in this, it is still remarkable that for just 17 messages a zero-error code could be found having rate larger than 0.61. In the table, $\mathcal{N}(N)$ is the number of messages of the symmetrical variable length code with maximum block length $N$. For $N = 2$, the entry in the table corresponds to the Hagelbarger code (cf. Section 3.3.4) and for $N = 3$, the optimum code is the one in Figure (3.3.4.1).
Chapter 3

Table. Lower bounds on $v^\square(N)$.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$v^\square(N)$ lower bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.59305 (s)</td>
</tr>
<tr>
<td>3</td>
<td>0.61914 (s)</td>
</tr>
<tr>
<td>4</td>
<td>0.60401 (s)</td>
</tr>
</tbody>
</table>

(3.6.1.7) As with $t^\square(N)$ trees, it is not easy to find good lower bounds for $v^\square(N)$ if $N$ becomes large. This is mainly because at every vertex of a $t^\square(N)$ tree we are in a kind of dilemma: either we subdivide the vertex directly into rectangles (which usually gives a high rate for that vertex, as in the L-shape subdivision in Figure (3.3.1.2)) and stop there, or we cut the region into more intricate shapes, postponing the high rate resolutions to levels further from the root.

Because of this we do not list any bounds on $v^\square(N)$ for $N > 4$. For $N = 2$, our bound is the rate of the tree in Figure (3.6.1.8) with $t_1 = u_1 = 0.62587$. This tree may be seen as a Hagelbarger code for two non-equiprobable messages per user.

(3.6.1.8) Figure. $t^\square(2)$ tree for the BMC.

For $N = 3$, the bound in Table (3.6.1.7) is the rate of Schalkwijk's $t^\square(3)$
tree described in [SCH82]. This tree is equal to the tree of Figure (3.6.1.4) (or Figure (3.6.1.15)), without the partitioning of rootlike rectangles (i.e., without the $F^2(2)$ tree on the right and the slicing vertices at level 2). The optimum values of the thresholds for the $F^3(3)$ tree are (notation as in Figure (3.6.1.4)):

$$t_1 = u_1 = 0.67571 \quad \text{(called } \hat{a} \text{ in [SCH82]),}$$

$$t_2 = u_2 = 0.29770 \quad \text{(} = (\hat{a} + \hat{\gamma} - 1)/\hat{a} \text{ in [SCH82]).}$$

(3.6.1.9) Figure. $F^3(4)$ tree for the BMC

The best $F^3(4)$ tree that we could find is shown in Figure (3.6.1.9). The subrectangle remaining at level 1 is partitioned according to Schalkwijk's optimum $F^3(3)$ tree, and the L-shaped region at level 1 is partitioned as in Figure (3.6.1.8). The optimum values of the parameters are $t_1 = u_1 = 0.60401.$

(3.6.1.10) Table. Lower bounds on $v(N).$

<table>
<thead>
<tr>
<th>$N$</th>
<th>$v(N)$ lower bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.61695 (s)</td>
</tr>
<tr>
<td>3</td>
<td>0.62116 (s)</td>
</tr>
<tr>
<td>4</td>
<td>0.62254</td>
</tr>
<tr>
<td>5</td>
<td>0.62448</td>
</tr>
<tr>
<td>6</td>
<td>0.62453</td>
</tr>
<tr>
<td>7</td>
<td>0.62557</td>
</tr>
</tbody>
</table>
As can be seen from Table (3.6.1.10), the best $V(2)$ tree does not yield a rate beyond the Shannon inner bound. The optimum $V(3)$ tree is depicted in Figure (3.6.1.11), with $t_i = u_i$ for $i = 1, 2, 3$, and 

$$t_1 = 0.67414, \; t_2 = 0.95439, \; t_3 = 0.34667.$$ 

(3.6.1.11) Figure. $V(3)$ tree for the BMC.

To find an optimum (or at least a 'good') $V(4)$ tree, we took the asymmetrical $F(4)$ tree as found by Smeets, achieving rate 0.62136, and deleted from it the $F(3)$ tree for the subsquare at level 1. We obtained a tree with rate 0.62233. For the tree with this structure we tried to find the optimum threshold values by computer, but it turned out that the thresholds in Smeets' $F(4)$ tree are locally optimum also for the $V(4)$ tree. However, a slight increase in rate did occur by deleting the vertex at level 3 that corresponds to output sequence $I = (0, 1, 0)$. Once more we optimized the corresponding rate function over the thresholds and
found rate 0.62254. The threshold values are listed in Table (3.6.1.12) below and the tree itself is shown in Figure (3.6.1.13).

<table>
<thead>
<tr>
<th>i</th>
<th>( t_i )</th>
<th>( u_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.68476</td>
<td>0.68897</td>
</tr>
<tr>
<td>2</td>
<td>0.74067</td>
<td>0.76191</td>
</tr>
<tr>
<td>3</td>
<td>0.61594</td>
<td>0.60101</td>
</tr>
<tr>
<td>4</td>
<td>0.94928</td>
<td>0.94585</td>
</tr>
<tr>
<td>5</td>
<td>0.35479</td>
<td>0.35354</td>
</tr>
<tr>
<td>6</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>7</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>8</td>
<td>0.91071</td>
<td>0.85421</td>
</tr>
<tr>
<td>9</td>
<td>0.59413</td>
<td>0.62187</td>
</tr>
<tr>
<td>10</td>
<td>0.58915</td>
<td>0.61627</td>
</tr>
<tr>
<td>11</td>
<td>0.62670</td>
<td>0.56490</td>
</tr>
<tr>
<td>12</td>
<td>1.00000</td>
<td>0.63821</td>
</tr>
<tr>
<td>13</td>
<td>0.32548</td>
<td>0.10000</td>
</tr>
<tr>
<td>14</td>
<td>0.35842</td>
<td>0.10000</td>
</tr>
<tr>
<td>15</td>
<td>1.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>16</td>
<td>0.93740</td>
<td>0.92554</td>
</tr>
<tr>
<td>17</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>18</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>19</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>20</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>21</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>22</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>23</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
</tbody>
</table>

(3.6.1.12) Table. Threshold values for the tree in Figure (3.6.1.13).

For \( N \geq 5 \), we can do the same to change the 'optimum' \( F(N) \) tree into a good \( v(N) \) tree. Because of the numerous possibilities to delete subtrees from such a tree and because of the many thresholds, however, it turns out to be practically impossible to find an optimum \( v(N) \) tree in this way. Therefore we just took Smeets' \( F(N) \) tree and deleted the \( F(N-1) \) tree in the lower right-hand subsquare. Table (3.6.1.10) shows the rates of the \( v(N) \) trees thus constructed for \( N \geq 5 \).
Figure. $f(4)$ tree for the BMC.
For $N = 2$, the best bootstrap tree that could be found is Schalkwijk's tree [SCH83a], shown in Figure (3.6.1.15). Both the vertices at level 1 and 2 have the PBV property, but only the middle vertex (at level 1) has type ‘invisible’. The threshold values are: $t_1 = u_1 = 0.69070$, $t_2 = u_2 = 0.32059$.

For $N = 3$, nothing better could be found than the tree of Figure (3.6.1.15) with the same tree appended to it in the subsquare remaining after the first transmission. Clearly this yields the same rate as the $B(2)$ tree, namely 0.63056. Although Schalkwijk [SCH90] recently found a $B(4)$ tree having a rate exceeding the optimum $B(2)$ rate in the eighth decimal, it still remains an open problem whether or not a tree exists having a rate larger than 0.63056.
3.6.2. Miscellaneous results

Here we mention some results for TWCs other than the BMC. Trees have been constructed for ternary channels by Van der Heijden [HEI87] and Lormans [LOR88]. All results in this section can be found in these two references. From the 322 classes of ternary TWCs found by Jacobs [JAC86], eleven channels have been studied (not including the BMC, which can also be seen as as ternary TWC). For five of these channels, trees could be constructed that achieved rates outside the Shannon inner bound. Figure (3.6.2.1) lists the partition patterns of these TWCs.

\begin{figure}[h]
\begin{center}
\includegraphics[width=\textwidth]{partition_patterns.png}
\end{center}
\end{figure}

\textbf{(3.6.2.1) Figure.} Partition patterns of the investigated channels.

In Table (3.6.2.2) and (3.6.2.3) we list some achievable rates for each of the five channels. The channels are referred to by the names shown in Figure (3.6.2.1). As before, we only look at the average rate, \((\frac{1}{2} + \frac{1}{2})/2. We define \(f(N), v(N), \) et cetera, as in the previous section. In the tables below we give the sharpest lower bounds known for these rate points. Hence e.g. in the column labeled \(f(1)\) we list the highest average rate of all \(F(1)\) trees that have been
considered. The meaning of column \( b(n) \) is explained in the text below the tables. We also list the Shannon outer bound \( G_o \), denoting the maximum average rate of any point in \( G_o \). An \( '(s)' \) indicates that the tree is symmetrical (This applies to \( S_{3,3}, S_{5,5}(d) \) and \( S_{10,10}(b) \) only, since they are the only channels equivalent to symmetrical channels.).

<table>
<thead>
<tr>
<th>channel</th>
<th>( f(1) )</th>
<th>( f(2) )</th>
<th>( f(3) )</th>
<th>( G_o )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S_{3,3} )</td>
<td>0.83483 (s)</td>
<td>0.87576 (s)</td>
<td>0.87523</td>
<td>1.00000</td>
</tr>
<tr>
<td>( S_{5,5}(d) )</td>
<td>0.68207 (s)</td>
<td>0.73176 (s)</td>
<td>0.72430</td>
<td>0.84956</td>
</tr>
<tr>
<td>( S_{10,10}(b) )</td>
<td>0.91830 (s)</td>
<td>0.93865 (s)</td>
<td>0.93606</td>
<td>1.00000</td>
</tr>
<tr>
<td>( S_{3,7} )</td>
<td>1.02818</td>
<td>1.02818</td>
<td>1.02818</td>
<td>1.09477</td>
</tr>
<tr>
<td>( S_{4,6}(a) )</td>
<td>1.04936</td>
<td>1.04936</td>
<td>1.04936</td>
<td>1.09477</td>
</tr>
</tbody>
</table>

(3.6.2.2) Table. Fixed length tree rates for five channels.

<table>
<thead>
<tr>
<th>channel</th>
<th>( v(2) )</th>
<th>( b(2) )</th>
<th>( b(n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S_{3,3} )</td>
<td>0.88495 (s)</td>
<td>0.88547 (s)</td>
<td>0.73881</td>
</tr>
<tr>
<td>( S_{5,5}(d) )</td>
<td></td>
<td></td>
<td>0.94719 (s)</td>
</tr>
<tr>
<td>( S_{10,10}(b) )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( S_{3,7} )</td>
<td>1.03323</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( S_{4,6}(a) )</td>
<td>1.05411</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(3.6.2.3) Table. Variable length and bootstrap tree rates for five channels.

The reason why Table (3.6.2.3) contains only some isolated rates whereas Table (3.6.2.2) is 'full' is the following. Clearly the rate of an \( F(N) \) tree can be calculated by computer, and optimization is done by varying the threshold values. It is far less obvious how good \( V(N) \) or \( B(N) \) trees should be found, since there are many possibilities to prune an \( F(N) \) tree, as mentioned already for the BMC. The
trees achieving the rates of Table (3.6.2.3) could only be found by looking at the best $F(2)$ or $F(3)$ tree for the given TWC and adjusting this tree a little.

The column name '$b(m)$' should be interpreted as follows. Trees achieving rates in this column have been constructed as Markov chains, in the same way as a $B(N)$ tree can be viewed as a Markov chain. For a $B(N)$ tree we have seen that every vertex corresponds to a state in the chain, where the only possible transitions are: either one step down in the tree, or directly back to the root.

$$
\begin{array}{c}
0 & 1 & 2 \\
1 & 0 & 1 \\
\end{array}
\begin{array}{c}
1 \uparrow \downarrow 0,1,2 \\
1 \downarrow 0 \\
1 \uparrow \downarrow 0 \\
\end{array}
\begin{array}{c}
1 \\
2 \\
\end{array}
$$

(3.6.2.4) Figure. A Markov chain for $S_{10,10}(b)$.

The Markov chains considered here behave differently: here it is possible to return to a vertex higher in the tree that is not equal to the root. For an example of this, see Figure (3.6.2.4), which shows one of the trees from [LOR88] in a schematic form. Such a tree cannot be described by a finite bootstrap tree as
defined in Section 3.4.1. The only way to show that the rate of this Markov chain is achievable is by looking at a $B(N)$ tree for all $N \in \mathbb{N}$, where the $B(N)$ tree shows all sequences of $N$ transitions in the Markov chain starting from the root. The rate of this tree (which is achievable by Theorem (3.4.1.8)) tends to the rate of the Markov chain if $N \to \infty$. For this reason we refer to the Markov chain tree as a $B(\infty)$ tree.

With the trees given in this section we end the chapter on two-way channels. It should be understood that the trees presented here are nothing more than examples; by no means do we claim that the rates given in the tables are optimum. Also, many of the results from Section 3.6 are not new.

The new results are the theorems listed in the beginning of Section 3.5. These show that every TWC tree or code that has been constructed in the past years (i.e., the trees described in this section, among others) belongs to one of the types defined throughout this chapter. The theorems link all of these trees together and show that they can indeed be used to find arbitrarily good lower bounds on the capacity region.
Chapter 4: The Write Unidirectional Memory

4.1. Binary WUM codes

4.1.1. Introduction

Chapter 4 is devoted to a model of a communication situation that is called a 'Write Unidirectional Memory' or WUM, as it is abbreviated. In this section we describe the historical background of this model, which can also be found in [WIL89].

The concept of the WUM is relatively new. It dates back from December 1985, when a question was raised by researchers at Philips Research Laboratories, concerning a magneto-optical recording system. (The reader interested in more technical details of this system is referred to [THO87].) In such a system, binary symbols are stored on an optical disc in the following way. Depending on the symbol ('0' or '1') that has to be written on a specific spot of the disk, a magnetic field with the appropriate orientation is generated via an electromagnet. When heating up the spot on the disc by a laser beam, the material of the disc in this spot is magnetized in the direction of the magnetic field.

The direction of magnetization can be retrieved later when a laser beam is pointed at the spot: the polarization of the reflected light can be used to determine the direction of magnetization in this spot.

Because of inductivity, the polarity of the current through the coil of the electromagnet cannot be reversed too often in a short period of time. Hence fast
switching between writing '0's and '1's is not possible. For this reason it was proposed to keep the direction of the magnetization constant as long as possible (i.e., for the time it takes to record the whole disc) but to write symbols only in specific positions, to be chosen by the writer. In this way we only need to switch the laser on and off in different spots of the disc, but the current does not need to be reversed.

Assume that the disc initially contains only '0's. We set the direction of the current in such a way that '1's can be written. Then information is written on the disc by changing some of the '0's into '1's and by leaving the remaining positions unchanged. For obvious reasons we will call this a '1-cycle'. After writing, the information can be retrieved by a person who receives the disc, if he knows that the disc contained only '0's before writing.

The next time that we want to store information on the disc, we reverse the polarity of the current, such that we can only write '0's. A 0-cycle is performed, in which some of the '1's that were on the disc are changed into '0's. It should be noted that the spots containing '0' after the 1-cycle cannot be changed in the 0-cycle. The receiver of the disc must be able to determine what was written during this cycle, even if he does not know the previous state of the disc. Clearly this is a nontrivial task and some way of coding is needed to ensure transmission with a small (or even zero) probability of error.

The above implies that not every spot of the disc can be used to store a

---

3 This describes the situation of 1985. In the mean time (1989) this problem does not exist anymore: solutions have been found that allow fast switching between '0's and '1's. Among other things, different media have been proposed in which the information is stored by amorphous/crystalline phase changes, that can be created by heat only. Here inductivity does not play a role since this system does not use an electromagnet.
complete bit of information. In other words, the rate of a code (formally defined in Section 4.1.2) will be less than 1 bit/spot. The obvious question was: how high can the rate of a code really be?

Before this question can be answered, we must make more specific assumptions about the situation we are interested in. For instance, we can study the case where the encoder looks at the previous state of the disc before he chooses the spots in which to write a new symbol, but we can also consider the case where he is not allowed to look at the old state of the disc. Similarly we must distinguish between the cases in which the decoder is either informed or uninformed about the old state of the disc. Furthermore we can restrict ourselves to the case where the code has to be the same for every cycle, or to the case where we allow just two codes: one for every 0-cycle and one for every 1-cycle. For each of these cases, models have been developed that are described in the next section.

4.1.2. Definitions

The definitions in this section reflect some information-theoretical aspects of the optical recording system. We present the definitions in such a setting that they can easily be transferred to more general situations, like the one with arbitrarily large alphabets as described in Section 4.2.

(4.1.2.1) Definition. Let $N \in \mathbb{N}$. A (binary) Write Unidirectional Memory or WUM of block length $N$ is a sequence of $N$ cells that can assume values in $\{0,1\}$. The state of the WUM at a time $k \in \mathbb{N}$ is denoted by $s_k$ or $(s_k(0),s_k(1),...,s_k(N-1))$, where $s_k(n)$ is the value of the $n$th cell of the WUM at this time.

(4.1.2.2) Definition. Let $N \in \mathbb{N}$ and $c \in \{0,1\}$. A $c$-filter (cf. [SIM89]) is a vector in $\{c,0\}^N$. Here $c$ is called the type of the filter and $0$ is called a hole. A $c$-filter $\text{x}$
The *Write Unidirectional Memory* operates on a state $s_k$ by changing it into a state $s_{k+1}$ (denoted as $s_{k+1} = z \circ s_k$) according to the following rules.

$$s_{k+1}(n) = s_k(n) \text{ if } x(n) = \square \text{ and }$$

$$s_{k+1}(n) = c \quad \text{if } x(n) = c, \text{ for all } n \in \{0,1,\ldots,N-1\}.$$  

We will also use the notation $s_{k+1}(n) = x(n) \circ s_k(n)$ for the operation of a filter on a single cell of the WUM.

Now we describe the way in which a WUM can be used for communication. We must distinguish between four different cases, depending on whether or not the encoder or decoder knows the state of the WUM before writing, since this knowledge will influence the efficiency with which the WUM can be used. Hence we have four cases, which will be named in the same way as in [SIM89].

**Definition.** We define the following four cases for WUM communication. Here the 'old state' stands for the state of the WUM before writing takes place.

Case 1: both encoder and decoder know the old state of the WUM  
Case 2: only the encoder knows the old state of the WUM  
Case 3: only the decoder knows the old state of the WUM  
Case 4: neither encoder nor decoder knows the old state of the WUM.

Figure (4.1.2.4) depicts the communication situation for all four cases. The dashed lines represent the side information about the old state of the WUM, which is present in some of the cases only. Let $k \in \mathbb{N}$ and suppose that the WUM has been used $k-1$ times (or cycles), starting from state $s_1 = (0,0,\ldots,0)$, such that the state before the $k^{th}$ use is $s_k$. The source generates a message $v_k$ from a set $\mathcal{V}_k := \{0,1,\ldots,\mathcal{U}_k-1\}$ for some $\mathcal{U}_k \in \mathbb{N}$. It is assumed that all messages in the set are equally likely.
The encoder chooses a filter $I_k$ that will operate on the state $S_k$ to form a new state, $S_{k+1}$. This filter can be described by two components: its type, say $c_k$ (which can be seen as the direction of the current through the electromagnet, in terms of the optical disc) and the set of positions $n$ where $I_k(n) = c_k$ (which can be seen as the spots that have to be heated by the laser beam for writing). We denote this set as $X_k$. In general, the choice of both $c_k$ and $X_k$ may depend on $k$, $V_k$, and (in cases 1 and 2) on $S_k$. Because of this, the encoder’s choice can be written as either $(c_k, X_k) = f_k(V_k, S_k)$ or $(c_k, X_k) = f_k(V_k)$, depending on the case. Here $f_k$ is a so-called encoding function with either

(4.1.2.5.a) $f_k : \mathcal{M}_k \times \{0,1\}^N \to \{0,1\} \times \mathcal{P}([0,1, \ldots, N-1])$ or

(4.1.2.5.b) $f_k : \mathcal{M}_k \to \{0,1\} \times \mathcal{P}([0,1, \ldots, N-1])$,

where $\mathcal{P}([0,1, \ldots, N-1])$ denotes the power set of $\{0,1, \ldots, N-1\}$.

The decoder takes $\hat{V}_k := g_k(S_{k+1}, S_k)$ (in cases 1 and 3) or $\hat{V}_k := g_k(S_{k+1})$ (in cases 2 and 4) as an estimate for message $V_k$. If $\hat{V}_k = \text{''?''}$, then a good estimate for the message cannot be found. We call $g_k$ a decoding function with either

(4.1.2.6.a) $g_k : \{0,1\}^N \times \{0,1\}^N \to \mathcal{M}_k \cup \{?\}$ or

(4.1.2.6.b) $g_k : \{0,1\}^N \to \mathcal{M}_k \cup \{?\}$.

In the most commonly studied situation (cf. [WIL86], [BOR88], [KUZ88], [SIM89]), $c_k$ does not depend on $V_k$ and $S_k$, but only on $k$. This is indeed a sensible choice, because it allows concatenation to create codes for arbitrarily
large block length. In that case, $c_k \equiv k \mod 2$ is taken. From now on we will assume that this relation between $c_k$ and $k$ holds. Then we can also define the encoding function $f_k$ in (4.1.2.5) as

\begin{align}
(4.1.2.7.a) & f_k: \mathcal{U}_k \times \{0, 1\}^N \rightarrow \{c_k, \gamma\}^N \\
(4.1.2.7.b) & f_k: \mathcal{U}_k \rightarrow \{c_k, \gamma\}^N,
\end{align}

such that the image of the function is the $c_k$-filter $\mathcal{A}_k$ with $c_k := 0$ if $k$ is even and $c_k := 1$ if $k$ is odd. In the sequel we will use (4.1.2.7) rather than (4.1.2.5).

The above model describes one of the most general situations. If the WUM is used an infinite number of times, we theoretically need infinitely many numbers $I_1, I_2, \ldots$ for the number of messages that can be written per cycle, together with an infinite sequence of encoding and decoding functions: $f_1, g_1, f_2, g_2, \text{etcetera}$. For practical reasons, however, we will assume that a code has only a finite number of $I_k$'s, $f_k$'s and $g_k$'s. In most papers on WUMs, the number of messages is assumed to be fixed: $I_k = I$ for all $k$. Some exceptions to this rule are [OVE88b], [OVE88c] and a concluding remark in [COH88].

If the number of messages is fixed, then it is clear (cf. [SIM89]) that only finitely many different encoding and decoding functions exist, since the number of states of the WUM is finite. This is the motive for the assumption in [WIL89] that a period $T$ exists, such that $f_k = f_k \mod T$ and $g_k = g_k \mod T$. Two extensively studied cases are $T = 2$ and $T = 1$. If $T = 2$, the code is called alternating and if $T = 1$, the code is called symmetric. (Note that in the symmetric case, $f_0$ and $f_1$ cannot really be the same since the ranges of the two functions differ. A more correct way to express the symmetry is that the description of $f_1$ can be obtained from the description of $f_0$ by reversing the roles of '0' and '1'.)
Chapter 4

We will look at the alternating case, but with two numbers of messages, $M_0$ and $M_1$. It turns out that this type of WUM code is suitable for generalization to larger alphabets. Therefore we arrive at the following set of definitions.

\[ (4.1.2.8) \text{ Definition.} \] Let $N \in \mathbb{N}$, $M_c \in \mathbb{N}$ and $K_c := \{0,1, \ldots, M_c-1\}$ for $c \in \{0,1\}$. A Case 1–WUM code with parameters $(N,M_0,M_1)$ consists of encoding functions $f_c$ and decoding functions $g_c$, $c \in \{0,1\}$:

\[
\begin{align*}
f_c &: K_c \times \{0,1\}^N \rightarrow \{c,\Box\}^N, \\
g_c &: \{0,1\}^N \times \{0,1\}^N \rightarrow K_c \cup \{?\}.
\end{align*}
\]

\[ (4.1.2.9) \text{ Definition.} \] Let $N \in \mathbb{N}$, $M_c \in \mathbb{N}$ for $c \in \{0,1\}$. A Case 2–WUM code with parameters $(N,M_0,M_1)$ consists of encoding functions $f_c$ and decoding functions $g_c$, $c \in \{0,1\}$:

\[
\begin{align*}
f_c &: K_c \times \{0,1\}^N \rightarrow \{c,\Box\}^N, \\
g_c &: \{0,1\}^N \times \{0,1\}^N \rightarrow K_c \cup \{?\}.
\end{align*}
\]

\[ (4.1.2.10) \text{ Definition.} \] Let $N \in \mathbb{N}$, $M_c \in \mathbb{N}$ for $c \in \{0,1\}$. A Case 3–WUM code with parameters $(N,M_0,M_1)$ consists of encoding functions $f_c$ and decoding functions $g_c$, $c \in \{0,1\}$:

\[
\begin{align*}
f_c &: K_c \rightarrow \{c,\Box\}^N, \\
g_c &: \{0,1\}^N \times \{0,1\}^N \rightarrow K_c \cup \{?\}.
\end{align*}
\]

\[ (4.1.2.11) \text{ Definition.} \] Let $N \in \mathbb{N}$, $M_c \in \mathbb{N}$ for $c \in \{0,1\}$. A Case 4–WUM code with parameters $(N,M_0,M_1)$ consists of encoding functions $f_c$ and decoding functions $g_c$, $c \in \{0,1\}$:

\[
\begin{align*}
f_c &: K_c \rightarrow \{c,\Box\}^N, \\
g_c &: \{0,1\}^N \rightarrow K_c \cup \{?\}.
\end{align*}
\]
The Write Unidirectional Memory

(4.1.2.12) Definition. Consider a WUM code $C$ for any of the four cases defined above. The (average) error probability of $C$, $P_{\text{err}}(C)$, is defined by

$$P_{\text{err}}(C) := \lim_{K \to \infty} \frac{\sum_{k=1}^{K} \Pr\{\hat{V}_k \neq V_k | \text{code } C\}}{K}. \quad \square$$

(4.1.2.13) Definition. Consider a WUM code for any of the four cases, with parameters $(\mathcal{N}, \mathcal{N}_0, \mathcal{N}_1)$. The rate pair, $\mathcal{R} = (\mathcal{R}_0, \mathcal{R}_1)$, of the code is defined by

$$\mathcal{R}_c := \frac{\log(\mathcal{N}_c)}{\mathcal{N}}, \quad c \in \{0, 1\}. \quad \square$$

(4.1.2.14) Definition. Let $i \in \{1, 2, 3, 4\}$. A pair $(\mathcal{R}_0, \mathcal{R}_1)$ is achievable for case $i$ if, for all $\varepsilon > 0$ and all large enough $\mathcal{N}$, a case $i$--WUM code $C$ with block length $\mathcal{N}$ and rate $(\mathcal{R}_0, \mathcal{R}_1)$ exists such that $\mathcal{R}_0 > \mathcal{R}_0 - \varepsilon$, $\mathcal{R}_1 > \mathcal{R}_1 - \varepsilon$ and $P_{\text{err}}(C) < \varepsilon$. \quad \square$

(4.1.2.15) Definition. Let $i \in \{1, 2, 3, 4\}$. The capacity region for case $i$, denoted by $C_i$, is defined as the set of pairs in $\mathbb{R}^2$ that are achievable for case $i$. \quad \square$

In Section 4.1.3, we summarize some of the results obtained for WUM codes as defined above.

4.1.3. Previous results

Since 1985, many constructions for WUM codes have been found achieving 'high' rates (see the footnote in Section 3.3.1), for all of the models described in Section 4.1.2. The codes were constructed using combinatorial theory. An information-theoretical approach (i.e., random coding arguments and the concept of typicality) yielded inner bounds to the capacity regions. Conversely, upper bounds to the capacity could be derived using standard information-theoretical
techniques. In most cases it could be shown that the bounds yield the actual capacity region.

The first code construction was proposed by the Philips researchers themselves and can be found in e.g. [BOR88]. This code, say \( C \), is a case 4 WUM code with parameters \((N, m_0, m_1) = (1, 1, 2)\) and rate \((0, 1)\). It is defined by

\[
\begin{align*}
    f_0(0) &:= (0), & g_0((0)) &:= 0, & g_0((1)) &:= 7, \\
    f_1(0) &:= (n), & f_1(1) &:= (1), & g_1((0)) &:= 0, & g_1((1)) &:= 1.
\end{align*}
\]

It is easy to check that this code has \( P_{\text{err}}(C) = 0 \). Indeed, every 0-cycle is used to 'clean' the WUM, such that after a 0-cycle the state is always \((0)\). In the 1-cycles, 1 bit of information can be stored in the WUM by either changing the state to \((1)\) or leaving it unchanged.

Obviously we can form a code with parameters \((N, 1, 2^N)\) by concatenating \( N \) copies of \( C \). This WUM code is a zero-error code, which means that the decoder can always retrieve the message sent, no matter what the message or the previous state is. This proves that \((0, 1)\) is an achievable rate for case 4. Since this code can also be used if the encoder or the decoder knows the state of the WUM before writing, we have \((0, 1) \in C_i \) for all \( i \in \{1, 2, 3, 4\} \).

As in Section (3.6.1), we define the 'average rate' of a code, \( R_{\text{av}} \), by

\[
    R_{\text{av}} := \left( R_0 + R_1 \right) / 2.
\]

We also define, for \( i \in \{1, 2, 3, 4\} \),

\[
    (4.1.3.1) \quad C_i := \max \{ R_{\text{av}} \mid (R_0, R_1) \in C_i \}
\]

as the maximum average rate achievable by a WUM code for each of the cases.

The above construction shows \( C_i \geq 0.5 \) for all \( i \). However, the achievable rate can be much higher, as is shown in Table (4.1.3.2). This table summarizes
what is known about $C_i$ for all $i$. The column labeled 'constr' gives the highest average rate of any WUM code that has been constructed. It should be noted that the constructed codes are zero-error codes in every case. Further explanation about the codes and the bounds is given below the table.

<table>
<thead>
<tr>
<th>$i$</th>
<th>$C_i$</th>
<th>constr</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.69424</td>
<td>0.69424</td>
</tr>
<tr>
<td>2</td>
<td>0.69424</td>
<td>0.56371</td>
</tr>
<tr>
<td>3</td>
<td>0.69424</td>
<td>0.52832</td>
</tr>
<tr>
<td>4</td>
<td>0.54589</td>
<td>0.50000</td>
</tr>
</tbody>
</table>

(4.1.3.2) Table. Highest achievable rates for binary case $i$–WUM codes.

The code construction for case 1 is the following (cf. [BOR88]). Let $a < 0.5$ and assume that the state of the WUM before a 0–cycle contains $a \cdot N$ 0's. Both encoder and decoder know the positions of these 0's, which are 'useless' to them since they cannot be changed. Hence they only consider the $(1-a) \cdot N$ positions of the 1's in which information can be stored. Since the encoder knows the effect of each 0–filter on the state, he can make sure that the writing is done in such a way that after the 0–cycle, the state contains $a \cdot N$ 1's. The number of different words that can be written in this way equals $\left\lfloor \frac{(1-a) \cdot N}{a \cdot N} \right\rfloor$.

Clearly the same number of words can be written in each 1–cycle, by leaving $a \cdot N$ of the $(1-a) \cdot N$ 0’s unchanged and changing the rest into 1’s. By letting $N \rightarrow \infty$, it follows that

$$\left( (1-a)h\left(\frac{a}{1-a}\right), (1-a)h\left(\frac{a}{1-a}\right) \right) \in C_1.$$  

Thus, by optimization over $a$, we find that $R_{av} = 0.69424$ is achievable for case 1.

The fact that average rates higher than 0.69424 are not achievable was proved by Borden [BOR88] and Simonyi [SIM89], but for zero-error codes only.
Chapter 4

Cohen [COH88] gave a proof for the $\varepsilon$—error case. We do not give the proofs here, since generalizations of them are presented in the next sections.

For case 2, it was proved in [BOR88] that rate $0.69424$ is achievable with $0$—error codes. A random coding argument is needed for this, which is generalized in Section 4.2.3 to codes over larger alphabets. It is clear that higher rates are not achievable, since these are not even achievable in case 1, where the decoder knows the previous state as well. Many code constructions have been presented for this case (which is known as the 'classical' situation because of its origin, the optical recording system). First, Willems and Vinck [WIL86] improved upon the trivial rate $0.5$ code with a code achieving rate $\log(6)/5 \approx 0.51699$. Their result was generalized by Simonyi [SIM89], who found a code with rate $\log(58)/11 \approx 0.53254$. After this, many ad hoc constructions have been found by several people. At the moment, the highest average rate known is $0.56371$ [KOS90].

As in case 2, it is obvious that average rates higher than $0.69424$ cannot be achieved in case 3. In [COH88] it was conjectured that the rate could not exceed $\log(3)/3 \approx 0.52832$. In the same paper, a construction is given of a $0$—error code achieving this average rate. In Section 4.2.5 of this thesis we prove that average rate $0.69424$ is case 3—achievable, as a corollary of a more general theorem.

For case 4, no constructions have been found that improve upon the trivial code with rate $(0,1)$. Many authors have noticed that a WUM in case 4 resembles a $Z$—channel under special circumstances (cf. [COH88], [WIL89]). Computing the capacity of this $Z$—channel gives rate $0.54589$. A rigorous proof that this rate is indeed achievable, however, has not yet been published, as far as known by the author. In Section 4.2.6, we will present the proof of the achievability in a more general situation. For symmetrical codes, the converse is proved in [WIL89]. The more general case of alternating codes is treated in Section 4.3.
4.2 WUM codes over arbitrary alphabets

4.2.1. The model

At the 1988 IEEE Information Theory Symposium, the author gave a talk entitled 'Generalized WUM codes' [OVE88c]. The generalization referred to was the one to different numbers of messages in the two cycles. However, this title inspired Kuznetsov [KUZ88b] to contrive a different kind of generalization, namely to an arbitrary alphabet. Instead of writing binary symbols, 0 and 1, we can use an alphabet of size \( q \), say \( \{0, 1, \ldots, q-1\} \).

In this situation, the state of a WUM with block length \( N \) can be any vector in \( \{0, 1, \ldots, q-1\}^N \). Again, writing is done in cycles, but now we have \( q \) different types of cycles. As before, in a \( c \)-cycle \( (0 \leq c < q) \), only \( c \)'s can be written. Therefore we define the following. Let \( q \in \mathbb{N} \), \( q \geq 2 \) and \( \mathcal{A} := \{0, 1, \ldots, q-1\} \) throughout this chapter.

(4.2.1.1) Definition. Let \( N \in \mathbb{N} \). A \((q\text{-ary})\) WUM with block length \( N \) consists of \( N \) cells assuming values in \( \mathcal{A} \). The state \( s_k \) of the WUM before the \( k \)\textsuperscript{th} use is a vector in \( \mathcal{A}^N \), where \( s_k(n) \) is the value of the \( n \)\textsuperscript{th} cell \( (0 \leq n < N) \). A state \( s_k \) is changed into a state \( s_{k+1} \) by a \( c \)-filter \( (c \in \mathcal{A}) \) as described in Definition (4.1.2.2).

For a binary WUM, we usually omit the word 'binary'. We do the same in the \( q \)-ary case, since we will always use a fixed \( q \), so no confusion is possible. Similarly, the \( q \)-ary WUM codes defined later will just be called WUM codes.

Again we consider four cases of communication with a WUM. In the \( k \)\textsuperscript{th} transmission (starting with state \( s_1 := (0,0,\ldots,0) \)), the encoder who wants to send message \( V_k \) chooses a \( c_k \)-filter for some \( c_k \in \mathcal{A} \), depending on \( V_k \) and possibly on the state \( s_k \). The decoder who observes \( s_{k+1} \) and possibly \( s_k \) makes an
estimate \( \hat{V}_k \) for the message sent. For reasons explained before, we will restrict ourselves to the case where the type \( c_k \) of the filter chosen by the encoder depends on \( k \) only.

The most straightforward generalization of the situation considered in the previous section seems to be \( c_k = k \mod q \). This is indeed the choice we will make, but we stress that other choices are conceivable. For instance, we could start with a 0-cycle, then have a 1-cycle, 2-cycle, up to a \((q-1)\)-cycle and then go 'down' with a \((q-2)\)-cycle followed by a \((q-3)\)-cycle, et cetera, and finally a 0-cycle. After this we could go 'up' again. It appears that this and other choices do not allow the computation of capacity regions, whereas for the choice \( c_k = k \mod q \), we are able to give quite elegant generalizations of both the results and the proofs mentioned in Section 4.1.3.

For the same reason as above, we restrict ourselves to codes for \( q \) different numbers of messages with \( q \) different encoding and decoding functions; one for each type of cycle. This gives the most natural and most promising generalization of Definitions (4.1.2.8) through (4.1.2.15). Hence we have the following definition, which is a compressed version of (4.1.2.8) through (4.1.2.11) for alphabet \( A \).

**(4.2.1.2) Definition.** Let \( N \in \mathbb{N} \) and \( N_c \in \mathbb{N} \) for \( c \in A \). Define \( \mathcal{M}_c := \{0,1,\ldots,N_c-1\} \).

For \( i \in \{1,2,3,4\} \), a case \( i \)-WUM code with parameters \((N,N_0,N_1,\ldots,N_{q-1})\) consists of \( q \) encoding functions \( f_c \) and \( q \) decoding functions \( g_c \) (\( c \in A \)). Here

\[
\begin{align*}
    f_c &: \mathcal{M}_c \times A^N \rightarrow \{c,\square\}^N & & \text{for } i = 1 \text{ and } i = 2, \text{ and} \\
    f_c &: \mathcal{M}_c \rightarrow \{c,\square\}^N & & \text{for } i = 3 \text{ and } i = 4; \\
    g_c &: A^N \times A^N \rightarrow \mathcal{M}_c \cup \{?\} & & \text{for } i = 1 \text{ and } i = 3, \text{ and} \\
    g_c &: A^N \rightarrow \mathcal{M}_c \cup \{?\} & & \text{for } i = 2 \text{ and } i = 4.
\end{align*}
\]

The error probability for \( q \)-ary WUM codes is defined as in Definition
We can also define the rate, achievability and capacity region in this case, where the only difference with Definitions (4.1.2.13), (4.1.2.14) and (4.1.2.15) is the fact that a rate tuple now has $q$ components instead of 2. This is summarized in the following definition.

(4.2.1.3) Definition. Let $i \in \{1,2,3,4\}$. A tuple $(\ell_0, \ell_1, \ldots, \ell_{q-1}) \in \mathbb{R}_q$ is called achievable for case $i$ if, for all $\varepsilon > 0$ and large enough $N$, a case $i$–WUM code $C$ with parameters $(N, \ell_0, \ell_1, \ldots, \ell_{q-1})$ exists for which $P_{\text{err}}(C) < \varepsilon$ and $\bar{R}_c > \bar{R}_c - \varepsilon$ for all $c \in A$, with $\bar{R}_c := \log(\ell_c) / N$. $\bar{R}$ is called the rate of this code. The capacity region for case $i$, $C_i$, is the set of case $i$–achievable $q$–tuples. $\square$

4.2.2. Statement of results

Here we state the main results of this chapter: we have found the capacity region for cases 1, 2 and 3. For case 4, a region of achievable rate tuples is given which we conjecture to be the capacity region. In the rest of this chapter, superscripts and subscripts like in $a_i^{c+1}$ (see Definition (4.2.2.1) below) should be considered mod $q$. This does not include subscripts (usually called $k$) that denote the number of the cycle, as in $S_{k+1}$.

(4.2.2.1) Definition. We define

$$\mathcal{Z}_1 := \{(\ell_0, \ell_1, \ldots, \ell_{q-1}) \in [0,1]^q \mid \forall c \in A \quad \bar{R}_c = \sum_{i \in A \setminus \{c\}} a_i^c \cdot h\left(\frac{a_i^{c+1}}{a_i^c}\right),$$

for some $a_i^c \geq 0$ such that $\sum_{i \in A} a_i^c = 1$ for all $c$ and $a_i^c \geq a_i^{c+1}$ for all $i \neq c$.\]

In this formula, $0 \cdot h(0/0)$ should be understood as 0. $\square$

(4.2.2.2) Theorem. We have $C_1 = C_2 = \text{co} \mathcal{Z}_1$ ("co" denotes the convex hull). $\square$
Chapter 4

(4.2.2.3) Definition. We define
\[ \mathcal{Z}_3 := \{ (k_0, k_1, \ldots, k_{q-1}) \in [0,1]^q \mid \forall c \in \mathcal{A} \quad \mathcal{P}_c = h(p_c) \{ 1 - \frac{p_c \cdot P}{(1-p_c)(1-P)} \}, \]
for some \( p_c \in [0,1] \) and \( P := \prod_{c \in \mathcal{A}} (1-p_c) \}. \]

(4.2.2.4) Theorem. We have \( C_3 = \co \mathcal{Z}_3 \).

(4.2.2.5) Definition. We define
\[ \mathcal{Z}_4 := \{ (k_0, k_1, \ldots, k_{q-1}) \in [0,1]^q \mid \forall c \in \mathcal{A} \quad \mathcal{P}_c = h(p_c) \cdot \frac{p_c}{1-P} \cdot h(P), \]
for some \( p_c \in [0,1] \) and \( P := \prod_{c \in \mathcal{A}} (1-p_c) \}. \]

(4.2.2.6) Theorem. We have \( C_4 \subseteq \co \mathcal{Z}_4 \).

(4.2.2.7) Conjecture. \( C_4 = \co \mathcal{Z}_4 \).

The proofs of Theorems (4.2.2.2), (4.2.2.4) and (4.2.2.6) are given in the next sections.

4.2.3. Proof of achievability for case 1 and 2

In this section we prove the 'positive half' of Theorem (4.2.2.2), namely \( \co \mathcal{Z}_1 \subseteq C_1 \) and \( \co \mathcal{Z}_1 \subseteq C_2 \). First we show that all \( q \)-tuples in \( \mathcal{Z}_1 \) are achievable for case 2; by Definition (4.1.2.3), this proves that these tuples are also achievable for case 1. Our proof is a generalization of Borden's random coding proof [BOR88] for the achievability of rate 0.69424 for binary case 2–WUM codes.

Let \( \epsilon > 0 \); let, for \( i \in \mathcal{A} \) and \( c \in \mathcal{A} \), \( a^c_i \in (0,1) \) be such that the \( a^c_i \) satisfy the requirements mentioned in Definition (4.2.2.1). The case \( a^c_i = 0 \) or \( a^c_i = 1 \).
The Write Unidirectional Memory

follows by continuity. Let \( N \in \mathbb{N} \). For simplicity, we will consider all numbers \( a_i^c \cdot N \) to be integers, which does not matter if \( N \) is large enough. For \( c \in A \), let

\[
R_c := \sum_{i \in A \setminus \{c\}} a_i^c \cdot \left[ \frac{a_i^{c+1}}{a_i^c} \right], \quad M_c := \lfloor 2^N \cdot (R_c - e) \rfloor \quad \text{and} \quad \mathcal{M}_c := \{0, 1, \ldots, N_c - 1\}.
\]

Clearly \( \mathcal{M}_c \in \mathcal{H}_1 \). For \( c \in A \), consider the set \( T_c \subset \mathcal{A}^N \) defined as

\[
T_c := \{ g \in \mathcal{A}^N \mid \forall i \in A \mathcal{N}(i; g) = a_i^c \cdot N \}
\]

where \( \mathcal{N}(i; g) \) denotes the number of \( i \)'s in \( g \). The cardinality of this set is a multinomial coefficient:

\[
|T_c| = \frac{N!}{\prod_i (a_i^{c+1} \cdot N)!}
\]

We define the vector \( \mathbf{g}^* \) as \( (a_0^0 \cdot N)^1 (a_1^0 \cdot N)^2 (a_2^0 \cdot N)^3 \ldots (q-1)^{a_{q-1}^0 \cdot N} \); that is, a sequence of \( a_0^0 \cdot N \) 0's followed by \( a_1^1 \cdot N \) 1's, etcetera. We have \( \mathbf{g}^* \in \mathcal{T}_q \).

Furthermore we introduce the following concept.

(4.2.3.1) Definition. Let \( g \in \mathcal{A}^N \) and \( c \in A \). The \( c \)-pattern of \( g \) is defined as the set of positions where \( g \) has ' \( c \)' , so

\[
(c \text{-pattern of } g) = \{ n \in \{0, 1, \ldots, N-1\} \mid s(n) = c \}.
\]

We partition the set \( T_c \) into equivalence classes in such a way that a class contains all words that have the same \( c \)-pattern. Now we want to assign a label from \( \mathcal{M}_c \) to every word in \( T_c \) in such a way that words with the same \( c \)-pattern obtain the same label. We do this by randomly assigning labels from \( \mathcal{M}_c \) to each equivalence class; all labels of classes are chosen independently and the labels are equiprobable. Now each word in \( T_c \) gets the label of the equivalence class it belongs to. In this way, the words in all \( q \) sets \( T_c \ (c \in A) \) are labeled.

- 129 -
Chapter 4

Given the labeling, we define the following case 2–WUM code $C$ with parameters $(N, m_0, m_1, \ldots, m_{q-1})$. Let $c \in A$ and $i \in M_c$. Encoding function $f_c$ maps message $w_k = i$ and state $s_k$ into a $c$–filter $s_k \in \{c, \emptyset\}^N$ such that the new state $s_{k+1} = (s_k \circ c) s_k$ is a word in $T_c$ having label $i$. If such a $c$–filter does not exist, we take the $c$–filter having $c$'s in the positions of the $c$–pattern of $s^*$. The decoding function $g_c$ maps the state $s_{k+1}$ to its label in $T_c$ if $s_{k+1} \in T_c$; if $s_{k+1} \notin T_c$ then $g_c(s_{k+1}) := '?'$.

After the first $q - 1$ uses of this WUM code, the state $s_q$ of the WUM is an element of $T_{q-1}$. This is seen as follows. The initial state is $s_1 = 0^N$. If $q = 2$, $s_1$ can be changed into every word in $T_1$ in the first cycle (a 1–cycle). For $q > 2$, it is impossible to change $s_1$ into a word in $T_1$ in the first transmission, since words in $T_1$ contain all $q$ symbols at least once, for large enough $N$. Hence $s_2$ consists of 0's and 1's only and the 1–pattern of $s_2$ equals the 1–pattern of $s^*$. In the next cycle it is again impossible to change the state into a word in $T_2$, if $q > 3$, so both the 1–pattern and the 2–pattern of $s_3$ and $s^*$ are equal. Continuing in this way, we find

$$s_{q-1} = (a_0^0 \cdot N)_1 (a_1^0 \cdot N)_2 (a_2^0 \cdot N) \cdots (q-2) (a_{q-2}^0 \cdot N) (a_{q-1}^0 \cdot N).$$

In the $(q-1)^{th}$ transmission, this can be changed to a word in $T_{q-1}$ (possibly $s^*$).

Now we look at all uses of the WUM beyond the first $q - 1$. We will show that at least one of the WUM codes corresponding to the random labeling has the property that $w_k = \hat{w}_k$ for all $k \geq q$. A code having this property will be called eventually error–free. Although the word 'eventually' suggests that errors may occur in any finite set of initial transmissions, we will only consider the case where errors are confined to the first $q$ transmissions.

An eventually error–free code $C$ has $P_{err}(C) = 0$, since
\[ P_{\text{err}}(\mathcal{C}) = \lim_{K \to \infty} \left( \sum_{k=1}^{K} \frac{Pr(\hat{v}_k \neq v_k)}{K} \right) \]

\[ = \lim_{K \to \infty} \left( \sum_{k=1}^{q-1} \frac{Pr(\hat{v}_k \neq v_k)}{K} + \sum_{k=q}^{K} \frac{Pr(\hat{v}_k \neq v_k)}{K} \right) \]

\[ \leq \lim_{K \to \infty} \frac{(q-1) + 0}{K} = 0. \]

We will compute the probability that the random labeling does not correspond to an eventually error-free WUM code \( \mathcal{C} \). Suppose \( \mathcal{C} \) is not eventually error-free and consider a sequence of transmissions in which the decoder makes an error for some \( k \geq q \). Let \( k \geq q \) be smallest index such that \( w_k \neq \hat{w}_k \) and let \( c \in A \) satisfy \( k \equiv c \mod q \). Since \( k \) is the first time (after the initial \( q-1 \) transmissions) that an error occurs, we have \( s_k \in T_{c-1} \). This follows from the code definition. Now \( w_k \neq \hat{w}_k \) implies that the encoder could not find a \( c \)-filter changing \( s_k \) into a word in \( T_c \) having the right label. Hence the labeling is such that for some \( s \in T_{c-1} \) and some \( i \in \mathcal{M}_c \), no word \( s' \in T_c \) with label \( i \) can be found such that \( s \) can be updated to \( s' \) in a \( c \)-cycle.

For fixed \( c \) and \( s \in T_{c-1} \), the number of words \( s' \) in \( T_c \) to which \( s \) can be updated by a \( c \)-filter equals \( A_c \), where

\[ A_c := \prod_{i \neq c} \left[ \frac{a_i^c}{a_i^{c+1}} \right]^N. \]

Since all of these words \( s' \) have different \( c \)-patterns, their labels were chosen independently during the code construction. Hence the probability that none of the words \( s' \) has label \( i \) equals \( P_c \), where

\[ P_c := \left[ \frac{N_c^{-1}}{N_c} \right] A_c, \]

which is independent of \( s \) and \( i \). By the union bound, the total probability that
the randomly constructed code is not eventually error-free can be bounded by

\[ Pr\{ C \text{ is not eventually error-free} \} \leq \sum_{c \in A} \sum_{\delta \in \mathcal{T}} \sum_{i \in \mathcal{N}_c} P_c \]

\[ = \sum_{c \in A} |\mathcal{T}| \cdot \mathcal{N}_c \cdot \left[ \frac{1}{\mathcal{N}_c} \right] \]

As in [BOR88], we have

\[ |\mathcal{T}| \cdot \mathcal{N}_c \cdot \left[ \frac{1}{\mathcal{N}_c} \right] = |\mathcal{T}| \cdot \mathcal{N}_c \cdot 2 \left[ \frac{\mathcal{N}_c}{\mathcal{N}_c} \right] \]

\[ \leq |\mathcal{T}| \cdot \mathcal{N}_c \cdot 2 \left[ \frac{\mathcal{N}_c}{\mathcal{N}_c} \right]. \]

This is true because \( \ln(1-z) \leq -z \) for \( z \geq 0 \), so \( \log(1-z) \leq -z/\ln(2) \). We have

\[ \left[ \frac{\mathcal{N}_c}{\mathcal{N}_c} \right] = \frac{N!}{\Pi(a_i^c N)!} \]

Applying Stirling's formula, we find that this is

\[ \left[ \frac{\mathcal{N}_c}{\mathcal{N}_c} \right] = \frac{N!}{\Pi(a_i^c N)!} \]

\[ = \left[ \frac{2^N(R_c - \varepsilon)}{[2^{N(R_c - \varepsilon)}]} \right]. \]

Applying Stirling's formula, we find that this is

\[ \left[ \frac{2^N(R_c - \varepsilon)}{[2^{N(R_c - \varepsilon)}]} \right] = \left[ \frac{2^N(R_c - \varepsilon)}{[2^{N(R_c - \varepsilon)}]} \right]. \]

Because

\[ N = 2^{(N \cdot R_c - o(N))}, \quad N \to \omega, \]

the expression can be simplified to

\[ 2 \left[ -2N\varepsilon - o(N)/\ln(2) + o(N \cdot \log(N)) \right], \quad N \to \omega. \]

Therefore \( Pr\{ C \text{ is not eventually error-free} \} \) can be bounded as

\[ Pr\{ C \text{ is not eventually error-free} \} \]
\[ \leq q \cdot 2^{-N\epsilon - o(N)/\ln(2) + 0(N \cdot \log(N))}, \ N \to \infty. \]

Consequently, with probability tending to 1, the random construction yields an eventually error-free case 2-WUM code if \( N \to \infty \). This code has \( P_{\text{err}}(C) = 0 \) and rate approximating \( R \) if the block length is large enough. This proves the case 2- (and case 1-) achievability of all points in \( \mathcal{Z}_1 \).

By concatenating (or time sharing) codes with different rate points in \( \mathcal{Z}_1 \), we find that \( \mathcal{Z}_1 \) is included in the capacity regions \( C_1 \) and \( C_2 \).

4.2.4. Converse for case 1 and 2

Now we show that for a case 1-WUM code with error probability tending to zero, the rate tuple approximates a point in \( \co \mathcal{Z}_1 \). Together with the proof in Section 4.2.3, this proves Theorem (4.2.2.2). Our proof generalizes Willems' proof [WIL89] for the binary equal rate case.

Let \( \epsilon > 0, N \in \mathbb{N} \). Consider a case 1-WUM code \( C \) with block length \( N \), error probability \( P_{\text{err}}(C) < \epsilon \) and rate \( (R_0, R_1, \ldots, R_{q-1}) = (\log(N_0)/N, \ldots, \log(N_{q-1})/N) \). Consider a long sequence of transmissions. Let \( c \in A \) and \( k = c \mod q \). Then

\[ \log(N_c) = H(V_k) = \left\{ \begin{array}{l} \text{since } V_k \text{ and } S_k \text{ are independent} \\ H(V_k \mid S_k) \\ = I(\tilde{V}_k; V_k \mid S_k) + H(V_k \mid \tilde{V}_k, S_k). \end{array} \right. \]

So

\[ (4.2.4.1) \ \log(N_c) = I(\tilde{V}_k; V_k \mid S_k) + H(V_k \mid \tilde{V}_k, S_k). \]

For the first term on the right-hand side of (4.2.4.1), we have

\[ I(\tilde{V}_k; V_k \mid S_k) \leq H(\tilde{V}_k \mid S_k) \]
\[ \leq H(\tilde{V}_k, S_{k+1} \mid S_k) \]
\[ = H(S_{k+1} \mid S_k) \ \{ \text{since } \tilde{V}_k = g_c(S_{k+1}, S_k) \}. \]
\[ N - 1 \leq \sum_{n=0}^{N-1} H(S_{k+1}(n) | S_k(n)) \]

For all \( n, k \) and \( i \), we define

\[ I_k(n) := H(S_{k+1}(n) | S_k(n)) \quad \text{and} \quad \beta^k_i(n) := \Pr\{S_k(n) = i\}. \]

Then, since \( S_{k+1}(n) = I_k(n) \circ S_k(n) \),

\[
\Pr\{S_{k+1}(n) = j, S_k(n) = i\} = \begin{cases} 
\beta^{k+1}_i(n) & \text{if } i \neq c, \ i = j, \\
\beta^k_i(n) & \text{if } i = j = c, \\
\beta^k_i(n) - \beta^{k+1}_i(n) & \text{if } i \neq c, \ j = c, \\
0 & \text{otherwise.}
\end{cases}
\]

Therefore

\[ I_k(n) = H(S_{k+1}(n) | S_k(n)) = \sum_{i \neq k} \beta^k_i(n) \cdot h\left(\frac{(\beta^{k+1}_i(n))}{\beta^k_i(n)}\right). \]

The numbers \( \beta^k_i(n) \) can be computed for large \( k \) if we know the asymptotic distribution of \( S_k \). We can describe this distribution by a Markov chain with \( q \cdot q^N \) states, labeled \((c, s)\) for \( c \in A \) and \( 0 \leq s \leq q^N - 1 \). For fixed \( c \), the states \((c, s)\) correspond to all possible values of \( S_k \) (i.e., all elements of \( A^n \)) if \( k \equiv c \mod q \). The transition probabilities are given by the distribution of \( I_k \) conditional to \( S_k \): the probability of going from a state \((c, s)\) to a state \((c+1, s')\) is equal to the probability that a \( c \)-filter \( I_k = f_c(\Psi_k, S_k) \) was chosen transforming \( S_k \), the WUM state corresponding to \((c, s)\), into \( S_{k+1} \) (corresponding to \((c+1, s')\)). Clearly this Markov chain models the statistical behavior of \( S_k \) because of the periodicity of
the cycles in the WUM code.

It follows from the definition of the code that the chain is irreducible (every state can be reached from every other state) if we consider only those states that can be reached at all. Hence by [FEL70, Ch. XV, Section 9] a unique stationary distribution over the states exists, which can be computed from the transition probabilities given by the code. If we consider the \( n \)th cell of the state \( S_k \) only, the following definition is valid.

\[
\alpha_i^c(n) := \lim_{t \to \infty} \Pr\{S_{tq+c}^{i} = i\} = \lim_{t \to \infty} \beta_i^{tq+c}(n), \text{ for } c \in \mathcal{A}, i \in \mathcal{A}.
\]

The numbers \( \alpha_i^c(n) \) for fixed \( n \) satisfy the requirements mentioned in Definition (4.2.2.1). This is true because for any finite \( k \) (by the definition of \( \beta_i^k(n) \) and because of the way a \( c \)-filter operates)

\[
\sum_i \beta_i^k(n) = 1 \text{ and } \beta_i^k(n) \geq \beta_i^{k+1}(n) \text{ if } i \neq k.
\]

Hence we find the same relations for the limits \( \alpha_i^c(n) \). Now, if \( k \) is sufficiently large and \( k \equiv c \mod q \), (4.2.4.2) yields

\[
I_k(n) \leq \Sigma_{i \neq c} \alpha_i^c(n) \cdot \frac{\alpha_i^{c+1}(n)}{\alpha_i^c(n)} + \varepsilon.
\]

Since this upper bound only depends on \( c \) and not on \( k \), we can define

(4.2.4.3) \( I_c(n) := \Sigma_{i \neq c} \alpha_i^c(n) \cdot \frac{\alpha_i^{c+1}(n)}{\alpha_i^c(n)} \)

which yields

(4.2.4.4) \( I(\tilde{V}_k; V_k | S_k) \leq \sum_{n=0}^{N-1} I_c(n) + N \cdot \varepsilon. \)

Now consider the second term on the right-hand side of (4.2.4.1). By Definition (4.1.2.12), for all \( I \in \mathbb{N} \), a \( k > I \) with \( k \equiv c \mod q \) exists such that
Chapter 4

For such a $k$, we have by Fano's theorem (cf. [GAL68, Theorem 4.3.1]):

$$H(V_k | \hat{V}_k, \xi_k) \leq H(V_k | \hat{V}_k) \leq h(\varepsilon) + q\varepsilon \cdot \log(N_c) =: \varphi(\varepsilon)$$

with $\varphi(\varepsilon)/N \to 0$ if $\varepsilon \to 0$, for constant $N_c$. Hence, combining this with (4.2.4.1) and (4.2.4.4) we find: for all $c \in A$ a $k \in \mathbb{N}$ exists with $k \equiv c \mod q$ such that

$$N-1 \sum_{n=0}^{N-1} R_c(n)/N \leq H(V_k | \hat{V}_k) + \varepsilon + \varphi(\varepsilon)/N$$

for arbitrary $\delta > 0$, if $\varepsilon$ is small enough.

For the rate of the code, $R = (R_0, R_1, \ldots, R_{q-1})$, this yields:

$$N-1 \sum_{n=0}^{N-1} R(n)/N \leq H(V_k | \hat{V}_k) + \varepsilon + \varphi(\varepsilon)/N$$

where '<' denotes that the inequality '<' holds componentwise, and $R(n) = (R_0(n), R_1(n), \ldots, R_{q-1}(n))$ is an element of $Z_1$, by (4.2.4.3). Therefore

$$N-1 \sum_{n=0}^{N-1} R(n)/N \in co Z_1.$$ 

Hence $R$ approximates $co Z_1$ arbitrarily closely if $\varepsilon \downarrow 0$. This concludes the converse part of the proof of Theorem (4.2.2.2).

4.2.5. Proof for case 3

The proof of Theorem (4.2.2.4) is split into an achievability part and a converse part, like the proof of Theorem (4.2.2.2). We start with the proof of the the case 3--achievability of all rates in $co Z_3$. This proof is based on an unpublished result by Willems for case 4--achievability in the symmetrical case for $q = 2$, in which block Markov encoding (described in [COV81]) is used. We present a simplified version of this.
Let $0 < \varepsilon < 1$. For $c \in A$, we fix numbers $p_c \in (0, 1)$. Define

$$P := \prod_{c \in A} (1-p_c) \quad \text{and} \quad \varepsilon_c := h(p_c)\{1 - \frac{p_c \cdot P}{(1-p_c)(1-P)}\}.$$  

Let $N \in \mathbb{N}$ be large. As before, we assume that $p_c \cdot N$ is an integer for all $c$. We will consider an ensemble of possible case 3-WUM codes having a rate tuple, $\vec{R}$, that is approximately equal to the desired rate tuple: $\vec{R}_c \approx \varepsilon_c$ for all $c$. We show that the average error probability over this ensemble tends to 0 with increasing block length. This will imply the existence of a 'good' code achieving rate $\vec{R}$.

In order to be able to compute the probability of error, we construct each code in the ensemble in such a way that the distribution of the states (in a Markov chain, as in Section 4.2.4) is more or less 'uniform'. We will explain later what we mean by this. To this aim, all codewords in the ensemble will be partitioned into blocks of length $N$ that have a fixed number of $c$'s and $\omega$'s. The writing will be such that at each time instant, every $N$-block of this form is equally likely to be written.

For large enough $N$ and $c \in A$, let $N_0(c)$ and $N_1(c)$ be integers such that

$$N_0(c), N_1(c) = \left[ \frac{N}{p_c \cdot N} \right],$$

$$2 \left( h(p_c) \cdot \frac{p_c \cdot P}{(1-p_c)(1-P)} + \varepsilon/4 \right) \leq N_0(c) \leq 2 \left( h(p_c) \cdot \frac{p_c \cdot P}{(1-p_c)(1-P)} + \varepsilon \right),$$

$$2 \left( \frac{R_c - \varepsilon/2}{2} \right) \leq N_1(c) \leq 2 \left( R_c - \varepsilon/3 \right).$$

It can be seen as follows that this is possible. Firstly,
Chapter 4

\[
\left[ \frac{N}{P\cdot N} \right] = 2 \cdot \frac{N(h(p_c) + o(1))}{N \rightarrow \infty}, \text{ and } h(p_c) = h(p_c) - \frac{p_c \cdot P}{(1-p_c)(1-P)} + \mathcal{O}.
\]

Furthermore \( \left[ \frac{N}{P\cdot N} \right] \) has a divisor between \( d \) and \( N \cdot d \) for all \( d \in \{1, 2, \ldots, \left[ \frac{N}{P\cdot N} \right] \} \) (consider an ordered list of all divisors of \( \left[ \frac{N}{P\cdot N} \right] \); two consecutive divisors in the list differ by a factor of at most \( N \), since the expansion of \( \left[ \frac{N}{P\cdot N} \right] \) consists of divisors that are less than or equal to \( N \)). Hence \( M_1(c) \) has a divisor that differs at most by a factor \( N \) from \( 2^{N/\left[ \frac{N}{P\cdot N} \right] - \frac{1}{2}} \). For large \( N \), this number lies in the interval specified by the above inequalities. The quotient \( \left[ \frac{N}{P\cdot N} \right] / M_1 =: M_0(c) \) satisfies the other inequalities if \( N \) is large enough.

For fixed \( c \), let \( S^c \) denote the set of all \( c \)-filters of length \( N \) that contain \( p_c \cdot N \) times \( 'c' \) and \( (1-p_c) \cdot N \) times \( 'o' \). Since \( |S^c| = \left[ \frac{N}{P\cdot N} \right] \), the set \( S^c \) can be partitioned into \( M_0(c) \) subsets each containing \( M_1(c) \) words. We call these subsets \( C_1^c \), \( 0 \leq i \leq M_0(c) - 1 \). The words in \( C_i^c \) are ordered:

\[
C_i^c = \{ \sigma(c, i, 0), \sigma(c, i, 1), \ldots, \sigma(c, i, M_1^c - 1) \}.
\]

Every case \( 3 \)-WUM code in our ensemble will correspond to some choice of the ordered partitionings of \( S^0, S^1, \ldots, S^{c-1} \). Consider a fixed choice of partitionings, that can be denoted as \( \{ C_i^c \}, c \in \mathcal{A}, 0 \leq i \leq M_0(c) - 1 \). The WUM code, say \( \mathcal{C} \), that corresponds to this choice is the following.

Let \( T := \lceil \log(N) \rceil \). The code \( \mathcal{C} \) will have parameters \( (\bar{N}, \bar{N}_0, \ldots, \bar{N}_{q-1}) \) with

\[
\bar{N} := T \cdot N + \sum_{i \in \mathcal{A}} \lceil \log(M_0^i) \rceil,
\]

\[
\bar{N}_c := M_0(c) \cdot (M_1(c))^T \text{ for all } c.
\]

The rate of the code, \( \bar{R} = (\bar{N}_0, \bar{N}_1, \ldots, \bar{N}_{q-1}) \), has
The Write Unidirectional Memory

\[ \tilde{I}_c = \frac{\log(N_0^{(c)}) + T \cdot \log(N_1^{(c)})}{\sum_{i \in A} [\log(N_0^{(i)})] + TN} \geq \frac{0(N) + [\log(N)] \cdot N(R - \epsilon/2)}{0(N) + [\log(N)] \cdot N} \]

Hence, if \( N \) is large enough:

\[ \tilde{I}_c \geq I_c - \epsilon. \]

Let \( c \in A \) and \( k \equiv c \mod q \). The encoding function \( f_c \) is described as follows. Suppose the encoder wants to send message \( w_k \in \{0,1, \ldots, \tilde{I}_c - 1\} \). This message is written as

\[ \begin{aligned}
\text{(4.2.5.3)} & \quad w_k = w_{k,0} + w_{k,1} N_0^{(c)} + w_{k,2} N_1^{(c)} + \ldots + w_{k,T} N_0^{(c)} N_1^{(c)} T^{-1} \\
& \text{with } 0 \leq w_{k,0} < N_0^{(c)} \text{ and } 0 \leq w_{k,t} < N_1^{(c)} \text{ for } t = 1, 2, \ldots, T.
\end{aligned} \]

The number \( w_{k,0} \) is encoded using a trivial time sharing code: the information is written in \( [\log(N_0^{(c)})] \) prescribed positions where \( S_k \) had 'c-1' (hence the decoder can uniquely determine the positions in which 'c's are written), and the remaining \( \sum_{i \neq c} [\log(N_0^{(i)})] \) positions of the first block of the codeword are filled with 'c's.

Now the set \( C_{w_{k,0}}^{(c)} \) is used to encode the numbers \( w_{k,t} \) for \( t \geq 1 \): \( w_{k,t} \) will be encoded by a block of length \( N \), namely \( x(c, w_{k,0}, w_{k,t}) \in C_{w_{k,0}}^{(c)} \). We have \( T \) of these blocks. Let the state \( S_k \) be partitioned according to these blocks:

\[ S_k = (S_{k,0}, S_{k,1}, \ldots, S_{k,T}), \]

where \( S_{k,0} \) has length \( \sum_{i \in A} [\log(N_0^{(i)})] \) and \( S_{k,t} \) for \( t \geq 1 \) has length \( N \).

The decoding functions \( g_c \) are the same for all codes in the ensemble. Let \( \delta > 0; \delta \) will be specified later, and it will only depend on \( p_0, p_1, \ldots, p_{q-1} \) and \( \epsilon \). For the decoder, \( S_{k+1,0} \) and \( S_{k,0} \) uniquely determine the index \( w_{k,0} \) of the set \( C_{w_{k,0}}^{(c)} \) with which the remaining (and largest) part of the message was encoded. Each block \( S_{k+1,t} \) for \( t \geq 1 \) is decoded as follows. The decoder takes \( \hat{w}_{k,t} := \) the
unique \( w \in \{0,1,\ldots,2^c-1\} \) such that the triple \((z(c,w_k,0,w),s_k,t,s_{k+1},t)\) is jointly \(\delta\)–typical (see Definition (4.2.5.4) below). If such a \( w \) does not exist, then he decodes \( \hat{w}_k := g'_c(s_k,s_{k+1}) := \text{'?'}. \) If not \( \hat{w}_k = \text{'?'}, \) then \( \hat{w}_k \) is computed via the \( \hat{w}_k, t \) in a way similar to (4.2.5.3).

**Definition.** Let \( \delta > 0 \). A state \( s = s_k, t \), for \( k = c \mod q \), is \( \delta \)–typical (or just typical, if no confusion about \( \delta \) is possible) if

\[
\forall i \in A \left[ |N(i; s) - N \cdot a^c_i| \leq N\delta \right],
\]

where the \( a^c_i \) satisfy (4.2.5.5):

\[
\begin{align*}
\sum_{i \in A} a^c_i &= 1, \\
 a^{c+1}_i &= a^c_i (1 - p_c) \quad \text{for } i \neq c, \\
 a^{c+1}_c &= a^c_c + p_c (1 - a^c_c).
\end{align*}
\]

(The complete solution of (4.2.5.5) is given in Appendix A of this chapter; for now, it is only important to know that the \( a^c_i \) can be expressed in the \( p_c \) in a unique way.)

For a word \( s \in S^c \) and states \( s_k, t \) and \( s_{k+1}, t \), we say that \((z,s_k,t,s_{k+1},t)\) is jointly \( (\delta-)\)typical (or just \( j.t. \)) if both \( s_k, t \) and \( s_{k+1}, t \) are \( \delta \)–typical and \( s_{k+1}, t = z \circ s_k, t \) holds.

Now that we have defined the code ensemble, we will prove that the average error probability over the ensemble tends to zero if the block length \( N \) tends to infinity. In this proof we use that a state \( s_k \) is \( \delta \)–typical with probability tending to 1, which will be proved in Appendix A. Let \( \bar{p}_{\text{err}} \) be the error probability over the ensemble of codes.
\[ P_{\text{err}} = \sum_{A \in \mathcal{C}} Pr\{C = A\} \cdot P_{\text{err}}(A). \]

Here \( C \) is the ensemble of codes and \( C \) is a random variable denoting the choice of code, where \( Pr\{C = A\} = \frac{1}{|C|} \) for all codes \( A \) in the ensemble \( C \). We write \( \mathcal{C} = \{ \mathcal{C}(c) \} \), which denotes the partitioning of the sets \( S^c \) corresponding to \( C \). By Definition (4.1.2.12) and changing the order of summation, we obtain

\[
(4.2.5.6) \quad P_{\text{err}} = \lim_{K \to \infty} \frac{1}{K} \sum_{k=1}^{K} \sum_{A \in \mathcal{C}} Pr\{\hat{V}_k \neq V_k \ \text{and} \ C = A\}.
\]

Let \( k \equiv c \mod q \). Since the first block, \( S_{k+1,0} \), can always be decoded correctly, we find

\[
\sum_{A \in \mathcal{C}} Pr\{\hat{V}_k \neq V_k \ \text{and} \ C = A\}
\]

\[
\leq \sum_{A \in \mathcal{C}} \sum_{t=1}^{T} Pr\{\hat{V}_k, t \neq V_k, t \ \text{and} \ C = A\}
\]

\[
= T \sum_{A^* \in \mathcal{C}^*} Pr\{\hat{V}_{k,1} \neq V_{k,1} \ \text{and} \ \mathcal{C}(c) = A^*\}.
\]

In (*), we have assumed (without loss of generality) that \( w_{k,0} = 0 \). The set \( \mathcal{C}^* \) contains all ordered sets \( \mathcal{C}(c) = \{ z(c,0,0), z(c,0,1), \ldots, z(c,0,\mathcal{M}^{(c)}(c)-1) \} \) where the \( z(c,0,i) \) are distinct elements of \( S^c \). Now we may also assume \( w_{k,1} = 0 \). Let \( \mathcal{C}^* \setminus \{ z \} \) denote the set of all possible \( \mathcal{C}(c) \) that have \( z(c,0,0) = z \), and let \( \mathcal{C}' \) denote \( \mathcal{C}(c) \setminus \{ z(c,0,0) \} \). Then

\[
\sum_{A^* \in \mathcal{C}^*} Pr\{\hat{V}_{k,1} \neq V_{k,1} \ \text{and} \ \mathcal{C}(c) = A^*\}
\]

\[
= \sum_{z \in S^c, A' \in \mathcal{C}^* \setminus \{z\}, z, z'} Pr\{C = A', z(c,0,0) = z, S_{k,1} = z, S_{k+1,1} = z', \hat{V}_{k,1} \neq 0\}.\]
Let \( z, \xi \) and \( \xi' \) be such that \( \xi' = z \circ \xi \). By Definition (4.2.5.4), if \( \xi \) and \( \xi' \) are typical, then also \((z, \xi, \xi')\) is j.t. Therefore \( \hat{v}_{k,1} \neq 0 \) can only occur either if \( \xi \) or \( \xi' \) is not typical, or if a different codeword \( y \in C' \) exists for which \((y, \xi, \xi')\) is jointly typical. Hence we can bound the above sum as

\[
(4.2.5.7) \quad \sum_{\xi: \text{not typical}} \Pr\{s_{k,1} = \xi\} + \sum_{\xi': \text{not typical}} \Pr\{s_{k+1,1} = \xi'\} + \\
+ \sum_{\xi \in S, \xi' \in C^* \setminus \{\xi\}, \xi \text{ and } \xi' \text{ typical}} \Pr\{C' = \xi', \xi(c, 0, 0) = \xi, s_{k,1} = \xi, s_{k+1,1} = \xi'\} \cdot \phi(\exists y \in A', [(y, \xi, \xi') \text{ j.t.}])
\]

We write (4.2.5.7) as \( P_0 + P_1 + P_2 \). In Appendix A it is shown that

\[
\Pr\{s_{k,1} \text{ not typical}\} = \frac{1}{6^3} 0(1/N) \quad \text{and} \quad \Pr\{s_{k+1,1} \text{ not typical}\} = \frac{1}{6^3} 0(1/N), \quad N \to \infty
\]

if \( k \) is large enough, say \( k \geq k_N \) for some \( k_N \) dependent of \( p_0, p_1, \ldots, p_{q-1}, \delta \) and \( N \). Since \( T = \lceil \log(N) \rceil \), we have for large enough \( N \):

\[
T \cdot P_0 = T \cdot \sum_{\xi: \text{not typical}} \Pr\{s_{k,1} = \xi\} \leq \varepsilon/3
\]

and similarly \( T \cdot P_1 \leq \varepsilon/3 \). We can bound \( P_2 \) by

\[
(4.2.5.8) \quad P_2 \leq \sum_{\xi \in S, \xi \text{ and } \xi' \text{ typical}} \Pr\{z(c, 0, 0) = \xi, s_{k,1} = \xi, s_{k+1,1} = \xi'\} \cdot \\
\cdot \sum_{\xi' \in C^* \setminus \{\xi\}} \Pr\{C' = \xi'\} \cdot \sum_{\xi \in S} \phi(\exists y \in A' \text{ and } (y, \xi, \xi') \text{ j.t.}).
\]

Consider fixed typical \( \xi \) and \( \xi' \). Every \( y \) that is j.t. with \((\xi, \xi')\) is contained in
ordered sets $A^*$ (apart from $g$ and $y$, there are $\binom{c}{1} - 2$ other words in $c^0$ that can be chosen from $\mathcal{S}^c$). The total number of ordered sets $A^*$ is

$$\binom{\binom{0}{c}}{\binom{1}{c} - 2} \cdot (\binom{1}{c} - 1) !.
$$

Therefore, for any $z, \emptyset$ and $A'$,

$$\sum_{A' \in \binom{c^a}{\emptyset}} \sum_{y \in \mathcal{S}^c} \Pr\{G' = A'\} \cdot \sum_{y \in \mathcal{S}^c} \phi(y \in A' \text{ and } (y, \emptyset, g') \text{ j.t.})$$

$$= \left[ \binom{\binom{0}{c}}{\binom{1}{c} - 2} \right] \cdot \left[ \binom{\binom{0}{c}}{\binom{1}{c} - 1} \right] \cdot \sum_{y \in \mathcal{S}^c} \phi((y, \emptyset, g') \text{ j.t.})$$

$$\leq \frac{1}{\binom{0}{c}} \cdot \sum_{y \in \mathcal{S}^c} \phi((y, \emptyset, g') \text{ j.t.}).$$

Substituting this in (4.2.5.8) yields

$$(4.2.5.9) \quad P_2 \leq \frac{1}{\binom{0}{c}} \sum_{\emptyset, \emptyset' : \text{ typical}} \Pr\{\mathcal{S}_{k,1} = \emptyset, \mathcal{S}_{k+1,1} = \emptyset'\} \sum_{y \in \mathcal{S}^c} \phi((y, \emptyset, g') \text{ j.t.}).$$

For every $(\emptyset, \emptyset') \text{ j.t. with some } y$, there are (for some constant $c_1$)

$$\binom{\mathcal{N}(c: \emptyset)}{p_c \mathcal{N}(c: \emptyset') + \mathcal{N}(c: \emptyset)} \leq 2 \left[ a_c^c \cdot \frac{P_c - a_c^c + a_c^c}{a_c^c} + c_1 \delta \right] + O(\log(\mathcal{N}))$$

words $y \text{ j.t. with this pair: in the } \mathcal{N}(c: \emptyset') - \mathcal{N}(c: \emptyset) \text{ positions where } s'(n) = c \text{ and } s(n) \neq c, \text{ } y(n) \text{ must be } c \text{ and the remaining } (p_c \mathcal{N} - \mathcal{N}(c: \emptyset') + \mathcal{N}(c: \emptyset)) \text{ positions of the } c \text{'s in } y \text{ can be chosen among the } c \text{'-positions in } \emptyset. \text{ Hence for all}$$
Chapter 4

typical \(g\) and \(g'\), we have

\[
\sum_{y \in \mathcal{D}^e} \phi((y, g, g') j \cdot t) \leq 2 N \left[ a_c^c \cdot h \left( \frac{p_c - a_c^{c+1} + a_c^c}{a_c^c} \right) + c_1 \delta \right] + o(\log(N))
\]

or

\[
P_2 \leq \frac{1}{\mathcal{N}(c)} \cdot 2 N \left[ a_c^c \cdot h \left( \frac{p_c - a_c^{c+1} + a_c^c}{a_c^c} \right) + c_1 \delta \right] + o(\log(N))
\]

With (4.2.5.2), this is less than

\[
N \left[ a_c^c \cdot h \left( \frac{p_c - a_c^{c+1} + a_c^c}{a_c^c} \right) - h(p_c) \cdot \frac{p_c \cdot P}{(1-p_c)(1-P)} + c_1 \delta - \varepsilon/4 \right] + o(\log(N))
\]

From Appendix A, we find \(a_c^{c+1} = \frac{p_c}{1-P}\) and \(a_c^c = \frac{p_c \cdot P}{(1-p_c)(1-P)}\). This yields

\[
a_c^c \cdot h \left( \frac{p_c - a_c^{c+1} + a_c^c}{a_c^c} \right) = h(p_c) \cdot \frac{p_c \cdot P}{(1-p_c)(1-P)}
\]

Hence with \(\delta \leq \varepsilon/(8c_4)\) and \(\delta \leq 1/2\) we have

\[
P_2 \leq 2^{-N\varepsilon/8} + o(\log(N)) < \varepsilon/3T
\]

if \(N\) is large enough.

Now we combine the inequalities for \(P_0\), \(P_1\) and \(P_2\) with (4.2.5.6): for sufficiently large block length \(N\),

\[
\bar{p}_{\text{err}} = \lim_{K \to \infty} \sum_{k = K^e, K} \frac{1}{K - K^e + 1} \cdot T(P_0 + P_1 + P_2)
\]

\[
\leq T(\varepsilon/(3T) + \varepsilon/(3T) + \varepsilon/(3T)) = \varepsilon.
\]

Finally we can conclude that at least one case 3–WUM code \(c\) in the ensemble

\[ -144 - \]
exists for which \( P_{\text{err}}(C) < \epsilon \) and for which the rate tuple \((\tilde{R}_0, \tilde{R}_1, \ldots, \tilde{R}_{q-1})\) satisfies \( \tilde{R}_c > R_c - \epsilon \). This proves the achievability of the rate point \( R \in \mathcal{Z}_3 \). By time sharing it is obvious that every convex combination of points in \( \mathcal{Z}_3 \) is also achievable for case 3. This completes the proof of the statement \( C_3 \supset \text{co } \mathcal{Z}_3 \).

Now we prove the converse, namely that points outside \( \text{co } \mathcal{Z}_3 \) are not achievable. Let \( C \) be a case 3-WUM code with parameters \((N, W_0, W_1, \ldots, W_{q-1})\) and error probability \( P_{\text{err}}(C) < \epsilon \). Let \( c \in A \) and \( k \equiv c \mod q \). We have, exactly as in case 1 and 2 (compare the derivation of (4.2.4.1) and following inequalities):

\[
\log(W_c) = I(\hat{W}_k; V_k | S_k) + H(V_k | \hat{W}_k, S_k),
\]

with

\[
I(\hat{W}_k; V_k | S_k) \leq \sum_{n=0}^{N-1} H(S_{k+1}(n) | S_k(n)).
\]

As in Section 4.2.4, let

\[
R_k(n) := H(S_{k+1}(n) | S_k(n)).
\]

Then

\[
R_k(n) = \sum_{i \notin c} Pr\{S_{k}(n) = i\} \cdot H(S_{k+1}(n) | S_k(n) = i)
\]

\[
= \sum_{i \notin c} Pr\{S_{k}(n) = i\} \cdot H(X_k(n) | S_k(n) = i)
\]

\[
= Pr\{S_{k}(n) \neq c\} \cdot H(I_k(n)).
\]

For a fixed coordinate \( n \), we define \( p_c(n) := Pr\{I_k(n) = c\} \). Note that the numbers \( p_c(n) \) only depend on \( k \mod q (= c) \) because of the code definition. As before, the distribution of the states is described by a Markov chain. Although this Markov chain is different from the one described in Appendix A (where every codeword of fixed composition has the same probability of being written, which is clearly not true for any code), we do find the same stationary distribution if we look at one cell only, since for a fixed cell we have equal probability of writing a
'c' in both models. Hence, as proved in Appendix A, we find for \( k \rightarrow \infty \):

\[
\Pr\{S_k(n) = i\} \rightarrow a_i^c(n),
\]

where the \( a_i^c(n) \) form the solution of (4.2.5.5), if (4.2.5.5) is read with \( a_i^c(n) \) instead of \( a_i^c \) and \( p_c(n) \) instead of \( p_c \). The solution is given in (A.3). However, the notation in Appendix A is slightly different: \( p_i \) should be read as \( p_i(n) \), \( a_i^c \) as \( a_i^c(n) \), \( p_i^i \) as \( 1 - p_i(n) \) and \( P \) as \( P(n) := \Pi_{i}(1 - p_i(n)) \).

For large enough \( k \), we find

\[
R_k(n) \leq (1 - a_i^c(n)) \cdot h(p_c(n)) + \varepsilon/2
\]

\[
= (1 - \frac{p_c(n)P(n)}{(1 - P(n))(1 - p_c(n))}) \cdot h(p_c(n)) + \varepsilon/2.
\]

If we define, for all \( c \),

\[
R_c(n) := (1 - \frac{p_c(n)P(n)}{(1 - P(n))(1 - p_c(n))}) \cdot h(p_c(n)),
\]

then it follows that

\[
I(\tilde{V}_k; \tilde{V}_k|S_k) \leq \sum_{n=0}^{N-1} R_c(n) + N \cdot \varepsilon/2.
\]

As in case 1, the Fano inequality yields: for all \( c \), a \( k \) with \( k \equiv c \mod q \) exists such that

\[
\mathbb{H}(\tilde{V}_k|\tilde{V}_k, S_k) \leq \varphi(\varepsilon).
\]

This shows

\[
R_c \leq \sum_{n=0}^{N-1} R_c(n)/N + \delta
\]

for arbitrarily small \( \delta \), if \( \varepsilon \) is small enough. As in Section 4.2.4, let

\[
\mathbb{H}(n) := (R_0(n), R_1(n), \ldots, R_{q-1}(n)).
\]

By Definition (4.2.2.3), \( \mathbb{H}(n) \in \mathcal{H}_3 \). Now the code rate, \( \mathbb{R} \), can be bounded as
\[ \sum_{n=0}^{N-1} k(n) + (\delta, \delta, \ldots, \delta) \in \mathbb{Z}_3 \]

if \( \epsilon \downarrow 0 \). This concludes the converse part of the proof of Theorem (4.2.2.4).

### 4.2.6. Proof for case 4: achievability

We will prove the case 4–achievability for points in \( \mathbb{R}_4 \) in almost exactly the same way as we proved the achievability for case 3, in Section 4.2.5. Therefore we will skip some of the details and only point out the differences between the two cases.

Let \( 0 < \epsilon < 1 \). For \( c \in A \), let \( p_c \in (0, 1) \). We define

\[
\begin{align*}
P := \prod_{c \in A} (1 - p_c) \quad \text{and} \quad P_c := h(p_c) - \frac{p_c h(P)}{1 - P}.
\end{align*}
\]

Let \( N \in \mathbb{N} \) be large and assume that \( p_c N \) is an integer for all \( c \). Let \( N_0(c) \) and \( N_1(c) \) be such that (compare (4.2.5.2))

\[
\begin{align*}
N_0(c), N_1(c) &= \left[ \frac{N}{p_c N} \right], \\
2^{-N \left[ h(P) \cdot p_c / (1 - P) + \epsilon / 4 \right]} &\leq N_0(c) \leq 2^{-N \left[ h(P) \cdot p_c / (1 - P) + \epsilon \right]}, \\
2^{-N \left[ k - \epsilon / 2 \right]} &\leq N_1(c) \leq 2^{-N \left[ k - \epsilon / 3 \right]}.
\end{align*}
\]

As in case 3, such numbers can be found. We will construct an ensemble of case 4–codes by looking at partitionings of \( S_0, S_1, \ldots, S^{q-1} \), where \( S^c \) denotes the set of all words in \( \{c, \overline{c}\}^N \) that contain \( p_c N \) times \( c \) and \( (1 - p_c) N \) times \( \overline{c} \).

Every \( S^c \) can be partitioned into \( N_0(c) \) subsets containing \( N_1(c) \) words each. We call these subsets \( C_i(c), 0 \leq i \leq N_0(c) - 1 \), and we write

\[
C_i(c) = \{ Z(c, i, 0), Z(c, i, 1), \ldots, Z(c, i, N_1(c) - 1) \}.
\]
Consider a fixed choice of partitionings, denoted as \( \{ e_i^{(c)} \} \), \( c \in \mathcal{A}, 0 \leq i \leq H_0^{(c)} - 1 \). The case 4-WUM code, say \( \mathcal{C} \), that corresponds to this choice is the following.

Let \( T := \lceil \log(N) \rceil \). The code \( \mathcal{C} \) has parameters \( \tilde{H}, \tilde{h}_0, \ldots, \tilde{h}_{q-1} \) that are equal to the parameters for the case 3-WUM codes in Section 4.2.5. With these parameters, we have for the rate tuple \( \tilde{r} \):

\[
\tilde{r}_c \geq \tilde{r}_c - \varepsilon, \quad \text{for all} \quad c \in \mathcal{A}
\]

if \( N \) is large enough.

The encoding function \( f_c \) for \( c \in \mathcal{A} \) is the same as for the case 3-codes (in both cases the encoder cannot use the previous state, \( s_{k} \), so we can take the same functions in case 3 and 4). The decoder, however, is not allowed to use the old state \( s_{k} \) in his estimate for \( \hat{w}_k \). Let \( \delta > 0 \). If the decoder receives \( s_{k+1} \), where

\[
s_{k+1} = (s_{k+1,0}, s_{k+1,1}, \ldots, s_{k+1,T}),
\]

then \( s_{k+1,0} \) determines the index \( w_{k,0} \) of the set \( e_{w_{k,0}}^{(c)} \) that was used to encode the rest of the message. A block \( s_{k+1,t} \) for \( t \geq 1 \) is decoded by taking \( \hat{w}_k, t := \) the unique \( w \in \{0,1,\ldots,H_1^{(c)} - 1\} \) such that \( (z(w), s_{k+1,0}, w), s_{k+1, t} \) is jointly \( \delta \)-typical (see Definition (4.2.6.3)). If such a \( w \) does not exist, then \( \hat{w}_k := '?' \).

(4.2.6.3) Definition. Let \( \delta > 0 \). For a word \( x \in \mathcal{S}^c \) and a state \( z = s_{k+1,t} \), we say that \( z \) and \( z \) are jointly \( \delta \)-typical (j.t.) if \( z \) is \( \delta \)-typical (see Definition (4.2.5)) and \( s(n) = c \) for all positions \( n \) for which \( x(n) = c \).

As in Section 4.2.5, we will now bound the average error probability over the code ensemble,

\[
\bar{p}_{\text{err}} = \sum_{A \in \mathcal{C}} \Pr\{ \mathcal{C} = A \} \cdot p_{\text{err}}(A).
\]

The first part of the derivation in Section 4.2.5 is valid also in this case. Therefore
\[
\sum_{\mathcal{A} \in \mathcal{C}} \Pr\{\hat{V}_k \neq V_k \text{ and } \mathcal{G} = \mathcal{A}\} \\
\leq T \cdot \sum_{\mathcal{A}^* \in \mathcal{C}^*} \Pr\{\hat{V}_{k,1} \neq V_{k,1} \text{ and } \mathcal{G}_0(\mathcal{C}) = \mathcal{A}^*\} \\
= T \cdot \sum_{\mathcal{G} \in \mathcal{D}^C, \mathcal{A}^* \in \mathcal{C}^* \setminus \{\mathcal{G}\}, \mathcal{G}^*} \Pr\{\mathcal{C} = \mathcal{A}^*, \mathcal{G} = \mathcal{G}^*, \mathcal{S}_{k,1}^* = \mathcal{G}, \mathcal{S}_{k+1,1}^* = \mathcal{G}^*, \hat{V}_{k,1} \neq 0\}.
\]

Now \(\hat{V}_{k,1} \neq 0\) can only occur either if \(\mathcal{G}^* (= \mathcal{S}_{k+1,1}^*\) is not typical, or if a different filter \(\mathcal{G} \in \mathcal{C}^*\) exists for which \(\mathcal{G} \text{ and } \mathcal{G}^* \) are jointly typical. Since \(\mathcal{G}\) is the \(c\)-filter operating on \(\mathcal{G}\) to form \(\mathcal{G}^*\), it follows with Definition (4.2.6.3) that \((\mathcal{G}, \mathcal{G}^*)\) is j.t. if \(\mathcal{G}^*\) is typical. Hence we can bound the above sum as (compare (4.2.5.7)):

\[
T \cdot \sum_{\mathcal{G}^* \text{ not typical}} \Pr\{\mathcal{S}_{k+1,1} = \mathcal{G}^*\} + \\
+ T \cdot \sum_{\mathcal{G} \in \mathcal{D}^C, \mathcal{A}^* \in \mathcal{C}^* \setminus \{\mathcal{G}\}, \mathcal{G}^* \text{ typical}, \mathcal{G}} \cdot \phi(\exists \mathcal{G} \in \mathcal{A}^* [(y, \mathcal{G}^*) \text{ j.t.}])
\]

We write this as \(T \cdot P_1 + T \cdot P_2\). As in the previous case, we have (see Appendix A)

\[
\Pr\{\mathcal{S}_{k+1,1} \text{ not typical}\} \leq \frac{1}{\delta^3} \cdot 0(1/N), \quad N \to \infty
\]

if \(k \geq K_N\) for some \(K_N\) dependent of \(p_0, p_1, \ldots, p_{q-1}, \delta\) and \(N\). Since \(T = \lceil \log(N) \rceil\), we have for sufficiently large \(N\):

\[
T \cdot P_1 = T \cdot \sum_{\mathcal{G}^* \text{ not typical}} \Pr\{\mathcal{S}_{k+1,1} = \mathcal{G}^*\} \leq \varepsilon/2.
\]

Also, \(P_2\) can be bounded by
(4.2.6.4) \( P_2 \leq \sum_{\bar{x} \in \mathcal{S}^c, \bar{x}, \bar{x}' : \text{typical}} \Pr\{C'(\bar{x}) = \bar{x}'\} \cdot \sum_{\bar{y} \in \mathcal{S}^c} \phi(\bar{y} \in \mathcal{A}'\text{ and } (\bar{y}, \bar{x}') \text{ j.t.})\).

Consider a typical \( \bar{x}' \). As before, every \( \bar{y} \) that is j.t. with \( \bar{x}' \) is contained in
\[
\left[ \begin{array}{c}
\binom{N(c)}{0} \\
\binom{N(c)}{1}
\end{array} \right] \cdot \binom{N(c)-1}{1}!
\]
ordered sets \( \mathcal{A}' \). The total number of ordered sets \( \mathcal{A}' \) is
\[
\left[ \begin{array}{c}
\binom{N(c)}{0} \\
\binom{N(c)}{1}
\end{array} \right] \cdot \binom{N(c)-1}{1}!.
\]
As in case 3, this yields for any \( \bar{x} \) and \( \bar{x}' \):
\[
\sum_{\mathcal{A}' \in \mathcal{C}^c \setminus \{\bar{x}\}} \Pr\{C'(\mathcal{A}') = \bar{x}'\} \cdot \sum_{\bar{y} \in \mathcal{S}^c} \phi(\bar{y} \in \mathcal{A}'\text{ and } (\bar{y}, \bar{x}') \text{ j.t.})
\leq \frac{1}{\binom{N(c)}{0}} \cdot \sum_{\bar{y} \in \mathcal{S}^c} \phi((\bar{y}, \bar{x}') \text{ j.t.}).
\]
Substituting this in (4.2.6.4), we find
(4.2.6.5) \( P_2 \leq \frac{1}{\binom{N(c)}{0}} \cdot \sum_{\bar{x}' : \text{typical}} \Pr\{\mathcal{S}_{k+1,1} = \bar{x}'\} \cdot \sum_{\bar{y} \in \mathcal{S}^c} \phi((\bar{y}, \bar{x}') \text{ j.t.}).\)

For every typical \( \bar{x}' \), there are (for some constant \( c_1 \))
\[
\left[ \begin{array}{c}
\binom{N(c; \bar{x}')}{0} \\
\binom{N(c; \bar{x}')}{1}
\end{array} \right] \cdot \frac{N}{p_c N^2} \left[ a_c^{c+1} \cdot h(p_c/a_c^{c+1}) + c_1 \delta \right] + o(\log(N))
\]
c-filters \( \bar{y} \) that are j.t. with it, since \( \bar{y} \) must have its \( p_c N \) c's in the positions
where \( \bar{x}' \) has c's. Hence
The Write Unidirectional Memory

\[ p_2 \leq \frac{1}{N(c)} \cdot 2^{N \left[ a_c^{c+1} \cdot h(p_c/a_c^{c+1}) + c_1 \delta \right] + 0(\log(N))} \]

With (4.2.6.2), this is less than

\[ N \cdot 2^{N \left[ a_c^{c+1} \cdot h(p_c/a_c^{c+1}) - h(P) \cdot p_c/(1-P) + c_1 \delta - \epsilon/4 \right] + 0(\log(N))} \]

As in case 3, \( a_c^{c+1} = p_c/(1-P) \). Hence by taking \( \delta \leq \epsilon/(8c_1) \), \( \delta \leq 1/2 \) we have

\[ p_2 \leq 2^{-N \varepsilon/8} + 0(\log(N)) < \epsilon/2 \]

if \( N \) is large enough.

Now the inequalities for \( P_1 \) and \( P_2 \) yield with (4.2.5.6): for large enough \( N \),

\[ P_{err} = \lim_{K \to \infty} \sum_{k=K}^{K+1} \frac{1}{T(P_1 + P_2)} \leq T(\varepsilon/(2T) + \varepsilon/(2T)) = \varepsilon. \]

We conclude the proof of Theorem (4.2.2.6) in the same way as in case 3: at least one case 4–code exists in the ensemble such that the rate \( \hat{R} \) satisfies \( \hat{R}_c > R_c - \varepsilon \) and \( P_{err}(C) < \varepsilon \). By time sharing this proves the achievability of all points in \( co \mathcal{R}_4 \).

4.3 Bounding the average rate

In this section we will study the largest possible value of the average rate of a WUM code. This will be defined analogously to the definition for the binary case (compare (4.1.3.1)).
(4.3.1) Definition. Let $i \in \{1,2,3,4\}$. The average rate $R_{av}$ of a case $i$–WUM code with rate $(k_0,k_1,\ldots,k_{q-1})$ is defined by

$$R_{av} := \sum_{c \in \mathcal{A}} \frac{k_c}{q}.$$ 

The average capacity for case $i$, $C_{i,av}$, is defined by

$$C_{i,av} := \max \{ R_{av} \mid (k_0,k_1,\ldots,k_{q-1}) \in \mathcal{C}_i \}.$$ 

First we will prove the following quite obvious theorem.

(4.3.2) Theorem. $C_{i,av} \geq (q-1)/q$ for all $i$. 

Proof: We will construct a case 4–WUM code with rate $(0,1,1,\ldots,1)$ and error probability 0. This code has average rate $(q-1)/q$, which shows the achievability of this average rate for case 4. Clearly rate $(0,1,1,\ldots,1)$ is achievable for the other cases as well, which proves the theorem.

The code will have parameters $(N,1,2^N,2^N,\ldots,2^N)$. In the 0–cycle, the only 0–filter that can be written is $f_0(0) := (0,0,\ldots,0)$. In other words, we 'clean' the WUM in such a way that we can use it in the following transmissions. In the 1–cycle, the $2^N$ different 1–filters that can be written correspond to every possible word in the set $\{1,0\}^N$. The decoder will be able to see the difference between them, since before writing the state of the WUM contained 0's only.

After writing, there will be only 0's and 1's in the state. Hence if we allow every word from the set $\{2,0\}^N$ to be written in the subsequent 2–cycle, then the decoder will again be able to determine which of these words was written by looking at the positions of the 2's. In this way we can go on until the $(q-1)$–cycle, but then we have to clean the WUM again in the 0–cycle.
If we want to know the exact value of $C_i,_{av}$ then, for $i = 1, 2$ or 3, we can just use Theorems (4.2.2.2) and (4.2.2.4) and compute $C_i,_{av}$ from the regions $Z_i$ and $Z_3$. For case 1 and 2, this means that we have to maximize a function of $q^2$ parameters under some restrictions. Since this involves long and uninteresting computations, the details are given in Appendix B. The result is stated below, in Theorem (4.3.5). First we introduce a polynomial that is needed in the theorem.

**4.3.3 Definition.** Define the polynomial $f: \mathbb{C} \to \mathbb{C}$ by

$$f(z) := z^q(z - 2) + 1.$$  

Let $\zeta_0, \zeta_1, \ldots, \zeta_q$ denote the zeros of the polynomial $f$ and let $\zeta_0$ be the largest real zero (it follows from the lemma below that at least one real zero exists). □

**4.3.4 Lemma.** With $\zeta_i$ as defined above, the following three propositions hold.

(i) All $\zeta_i$ are distinct,

(ii) $\zeta_0 > 1.5$,

(iii) for all $i > 0$: $|\zeta_i| < \zeta_0$.

Proof: (i) For a multiple zero $\zeta$ we would have

$$f(\zeta) = 0 \text{ and } \frac{df(z)}{dz} \bigg|_{z=\zeta} = 0,$$

so

$$\zeta^q(\zeta - 2) + 1 = 0 \text{ and } (q+1)\zeta^q - 2q \cdot \zeta^{q-1} = 0.$$  

The second equation yields $\zeta = 0$ (which is not a zero of $f$) or $\zeta = \frac{2q}{q+1}$. The latter possibility gives, after substitution in the first equation,

$$2 \cdot (2q)^q = (q+1)^q + 1$$

which is impossible because the left-hand side is $0 \pmod{q}$ and the right-hand side is $1 \pmod{q}$. Therefore no multiple zeroes exist.
We have $f(z) > 0$ for all $z \in \mathbb{R}$ with $z \geq 2$; we also have
\[ f(1.5) = - (1.5)^q/2 + 1 \leq - (1.5)^{q^2}/2 + 1 = - 1/8 < 0. \]

Since $f$ is continuous, this implies that the largest real zero lies in $(1.5, 2)$.

We use Rouché's theorem (cf. [SAF76]) to prove $|\zeta_i| \leq 1$ for all $i > 0$. Since $\zeta_0 > 1.5$, this will prove the theorem. Define $g(z) := -2z^q$ for $z \in \mathbb{C}$.

Then, for $\epsilon > 0$ and $|z| = 1 + \epsilon$:
\[ |g(z)| = 2(1 + \epsilon)^q = 2 + 2q\epsilon + O(\epsilon^2) \] and
\[ |f(z) - g(z)| = |z^{q+1} + 1| \leq (1 + \epsilon)^{q+1} + 1 = 2 + (q+1)\epsilon + O(\epsilon^2). \]

Hence, if $\epsilon$ is small enough: $|f(z) - g(z)| \leq |g(z)|$ on the circle with radius $1 + \epsilon$.

Now, if we let $\epsilon \downarrow 0$, Rouché's theorem implies that $f$ and $g$ have equally many zeros (counting multiplicities) inside or on the circle of unity. Obviously, $g$ has $q$ zeros for which $|z| \leq 1$; hence the same holds for $f$. Since $f$ has $q+1$ zeros in total and $|\zeta_0| > 1$, we find $|\zeta_i| \leq 1$ if $i > 0$.

\[ (4.3.5) \text{Theorem.} \quad \text{The average capacities} \ c_{1, \text{av}} \text{ and} \ c_{2, \text{av}} \text{ of the WUM equal} \]
\[ c_{1, \text{av}} = c_{2, \text{av}} = \log(\zeta_0). \]

\[ \text{Proof:} \quad \text{As mentioned before, the proof can be given by maximizing the function} \]
\[ R(a_0^0, a_0^1, \ldots, a_{q-1}^{q-1}), \]
where
\[ R(a_0^0, a_0^1, \ldots, a_{q-1}^{q-1}) := \sum_{c} \sum_{i \neq c} a_i^c \cdot h(\frac{a_i^{c+1}}{a_i^c}), \]
under the constraints $\sum_{i} a_i^c = 1$ for all $c$ and $a_i^c \geq a_i^{c+1}$ for all $i \neq c$. This is done in Appendix B. Here we will present a different proof of the theorem.

First we prove that average rate $\log(\zeta_0)$ is achievable with case 2–codes, if the block length tends to infinity. After that we show that $\log(\zeta_0)$ is an upper bound for the rate of any eventually error–free case 2–code (defined in Section 4.2.3). Since we already know that $\epsilon$–error case 1–codes cannot achieve higher
rates than eventually error–free case 2–codes (which follows from Section 4.2.3 and 4.2.4), this shows that \( \log(\zeta_0) \) is the maximum achievable average rate for both case 1 and case 2.

Let \( N \in \mathbb{N} \). Let \( \mathbf{a} := (a_0, a_1, \ldots, a_{q-1}) \in \mathbb{R}^q \) such that \( a_c \cdot N \in \mathbb{N} \) for all \( c \in \mathcal{A} \), \( 0 \leq a_0 \leq a_1 \leq \ldots \leq a_{q-1} \) and \( \sum_c a_c = 1 \). Consider the following restriction on the use of the WUM. In a \( c \)-cycle, the writing must be such that only words from a set \( \mathcal{T}_c \) can be the new state \( \mathcal{X}_{k+1} \), where \( \mathcal{T}_c \subseteq \mathcal{X}^N \) is the set of words that contain \( a_{i-c-1}^N \) 's for all \( i \in \mathcal{A} \). Hence writing in a \( c \)-cycle means that a word \( \mathcal{X}_k \in \mathcal{T}_{c-1} \) is changed into a word \( \mathcal{X}_{k+1} \in \mathcal{T}_c \). It can be readily checked that this is possible only by changing \( (a_{i-c} - a_{i-c-1})^N \) of the i's into c's (the a_i's are nondecreasing) for all \( i \neq c \). Note that the encoder can take care of this: by choosing an appropriate case 2–code, he can make sure that the restriction is met.

We now adapt the notation of [KUZ83], [KUZ88a] and consider the WUM with sets \( \mathcal{T}_0, \ldots, \mathcal{T}_{q-1} \) as a channel with generalized defects. The words in \( \mathcal{T}_{c-1} \) that can be on the disk before writing in a \( c \)-cycle are the states of the channel, such that each state \( s \) allows some set \( \mathcal{Y}_s \subseteq \mathcal{T}_c \subseteq \mathcal{X}^N \) to be written. Kuznetsov [KUZ83] showed that the capacity \( \mathcal{C}(\mathbf{a}) \) of this channel with generalized defects is

\[
\mathcal{C}(\mathbf{a}) = \lim_{N \to \infty} \min_{s} \frac{\log |\mathcal{Y}_s|}{N}.
\]

To compute the numbers \( |\mathcal{Y}_s| \), we must count the number of ways in which some fixed word \( \mathcal{X}_s \) (representing the state of the channel) in set \( \mathcal{T}_{c-1} \) can be updated to a word in \( \mathcal{T}_c \). Call this number \( N_{\mathbf{a}, c} \). Then

\[
N_{\mathbf{a}, c} = \prod_{i \in \mathcal{A} \setminus \{c\}} \left[ \frac{a_{i-c}^N}{a_{i-c-1}^N} \right]_{i=0}^{q-2} = \prod_{i \in \mathcal{A} \setminus \{c\}} \left[ \frac{a_{i+1}^N}{a_i^N} \right]_{i=0}^{q-2},
\]

which is dependent on neither \( \mathbf{a} \) nor \( c \). Therefore we find that all sets \( \mathcal{Y}_s \) contain the same number of elements,
Chapter 4

\[ |\mathcal{Y}_3| = \frac{q^2}{\prod_{i=0}^{q-2} \left( a_{i+1}^N \right)^{\frac{1}{a_i^N}}} \]

Consequently,

\[ C(a) = \lim_{N \to \infty} \log \left( \frac{q^2}{\prod_{i=0}^{q-2} \left( a_{i+1}^N \right)^{\frac{1}{a_i^N}}} \right) = \sum_{i=0}^{q-2} a_{i+1}^N h\left( \frac{a_i}{a_{i+1}} \right). \]

Therefore, for all vectors \( a \) satisfying the conditions mentioned in the beginning of the proof, rate \( C(a) \) is case 2–achievable if the block length is large enough. Maximizing over \( a \) in a straightforward manner, we find

\[ \max \{ C(a) \mid 0 \leq a_0 \leq \ldots \leq a_{q-1}, \sum a_i = 1 \} = \log(\zeta_0). \]

(This maximization occurs in a concealed form in the proof given in Appendix B. Notice, though, that the maximization in the appendix involves \( q^2 \) parameters, whereas here we have only \( q \).) Hence the average rate in \( q \) consecutive cycles can achieve the value \( \log(\zeta_0) \), which concludes the first half of the proof.

Now we prove the converse. This part generalizes Simonyi's proof [SIM89] of the converse for the binary case 2–WUM with \( \mathcal{C}_0 = \mathcal{C}_1 \). Let \( C \) be an eventually error–free case 2–code with parameters \( N \) and \( N_c \) for \( c \in \mathcal{A} \). We consider \( q \cdot t \) consecutive writing cycles, and we look at the 'fate' of one cell during these cycles. For \( q = 3 \), all possible fates of length at most 3 are shown in Table (4.3.6). As mentioned before, each cell contains '0' before the first writing.

Let \( \mathcal{N}_{t,i} \) denote the number of fates (sequences) of length \( t \) for which the last \((t^{th})\) content of the cell equals \( i \) and let \( S_t \) be the total number of fates of length \( t \). So

\[ S_t = \sum_{i \in \mathcal{A}} \mathcal{N}_{t,i}. \]

We find the following recurrences for the \( \mathcal{N}_{t,i} \):
The Write Unidirectional Memory

<table>
<thead>
<tr>
<th>cycletype</th>
<th>type of cycle</th>
<th>possible fates of cell</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>02</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>022</td>
</tr>
<tr>
<td></td>
<td></td>
<td>020</td>
</tr>
<tr>
<td></td>
<td></td>
<td>111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>110</td>
</tr>
<tr>
<td></td>
<td></td>
<td>122</td>
</tr>
<tr>
<td></td>
<td></td>
<td>120</td>
</tr>
</tbody>
</table>

(4.3.6) Table. Example of 'fate' of a cell.

\[ N_{t+1,i} = N_{t,i} \quad \text{if } t+1 \neq i \text{ mod } q \quad (\text{the last cycle is not an } \iota-\text{cycle}), \]
\[ N_{t+1,i} = S_t \quad \text{if } t+1 = i \text{ mod } q. \]

Summation and repeated application of the first identity leads to

\[ S_{t+1} = 2^t S_t - S_{t-q} \quad \text{with } S_0 = 1, S_1 = 2, S_t = 0 \quad \text{for } t < 0. \]

It should be noted that for \( q = 2 \), this is equivalent to the Fibonacci relation.

By [LOV79, pp. 158–160] and (i) of Lemma (4.3.4), we have the following formula for \( S_t \). With \( \zeta_0, \zeta_1, \ldots, \zeta_q \) as in Definition (4.3.3), we find

\[
S_t = \sum_{i=0}^{q} \zeta_i^{t-q} \prod_{j \neq i} (\zeta_i - \zeta_j)
= \zeta_0^t + (\text{other terms } \zeta_{i_1} \zeta_{i_2} \cdots \zeta_{i_t} \text{ with } (i_1, \ldots, i_t) \in \{0, \ldots, q\}^t). 
\]

Because of Lemma (4.3.4), (iii), it can be seen that the following limit exists:

\[
\lim_{t \to \infty} \frac{\log(S_t)}{t} = \log(\zeta_0).
\]
Chapter 4

We know that each cell can have \( S_q \) different fates of length \( t \); for the sequence of \( q \cdot t \) consecutive writings of words of length \( N \), the number of possibilities is \((S_q t)^N\). Since code \( C \) is eventually error-free, decoding errors may occur in the first \( q-1 \) transmissions only. Hence the decoder must be able to determine which sequence of \( (q \cdot t - q) \) messages out of the \((N_0)^{t-1} \cdot (N_1)^{t-1} \cdots (N_{q-1})^{t-1}\) possible sequences was transmitted during the \((q \cdot t - q)\) most recent cycles. Therefore we must have

\[
(S_q t)^N \geq (N_0)^{t-1} \cdot (N_1)^{t-1} \cdots (N_{q-1})^{t-1},
\]

or

\[
\frac{\sum_c \log(N_c^t)}{N} \leq \frac{\log(S_q t)}{t-1}.
\]

This is true for all \( t \in \mathbb{N} \), so we have

\[
\mathbb{R}_{av} \leq \lim_{t \to \infty} \frac{\log(S_q t)}{qt} = \frac{t}{t-1} \log(\zeta_0),
\]

proving the converse part of the theorem.

Now that we have found an expression for the average capacity in cases 1 and 2, we can compute \( \mathbb{R}_{1, av} \) for several values of \( q \). For \( q \leq 10 \), these numbers can be found in Table (4.3.8). This table also lists values of \( \mathbb{R}_{3, av} \), which have been found by maximizing the function \( \mathbb{R}(p_0, p_1, \ldots, p_{q-1}) \), where

\[
\mathbb{R}(p_0, p_1, \ldots, p_{q-1}) := \sum_c h(p_c) \left\{ 1 - \frac{p_c^p}{(1-p_c)(1-P)} \right\}, \quad P := \Pi(1-p_c).
\]

Unfortunately, no analytical expression could be obtained for the maximum of this function, even though we have just \( q \) parameters (instead of \( q^2 \) as in case 1) and no restrictions here. Therefore the maximization was done numerically. For all values of \( q \leq 10 \), it turns out that the point \( p \) where \( \mathbb{R}(p) \) assumes its maxi-
The Write Unidirectional Memory

mum (as found by computer) satisfies \( p_0 = p_1 = \ldots = p_{q-1} \) and hence also all \( \xi_c \) are equal for the maximum average rate. This was to be expected, since we also saw this in cases 1 and 2: if the average rate is maximum, the rate is symmetrical. This leads us to the conjecture that the same will be true in case 4.

Since the true capacity region for case 4 is unknown, we need a different approach to find \( c_{4,av} \). Theorem (4.3.7) is a justification for Conjecture (4.2.2.7).

(4.3.7) Theorem. We have

\[
C_{4,av} = \max \left\{ \left( \xi \xi_c \right)/q \mid (\xi_0,\xi_1,\ldots,\xi_{q-1}) \in co \mathcal{Z}_4 \right\}.
\]

Proof: It follows from Theorem (4.2.2.6) that

\[
C_{4,av} \geq \max \left\{ \left( \xi \xi_c \right)/q \mid (\xi_0,\xi_1,\ldots,\xi_{q-1}) \in co \mathcal{Z}_4 \right\}.
\]

Therefore it remains to prove that for every case 4–achievable rate, the average rate is in the set \( \left\{ \left( \xi \xi_c \right)/q \mid (\xi_0,\xi_1,\ldots,\xi_{q-1}) \in co \mathcal{Z}_4 \right\} \). The main part of this proof is a generalization of the case 4–converse for binary symmetrical codes given in [WIL89] (also found by Wyner and Ozarow, [WYN88]).

Let \( \varepsilon > 0 \) and let \( \mathcal{C} \) be a case 4–code with parameters \( N \) and \( M_c \) for \( c \in \mathcal{A} \). Assume \( P_{err}(\mathcal{C}) \leq \varepsilon \). Let \( c \in \mathcal{A}, k \equiv c \mod q \). We have

\[
H(\hat{V}_k) = H(\hat{V}_k, S_{k+1}) + I(S_{k+1}; \hat{V}_k)
\]

\[
\leq H(\hat{V}_k, \hat{\hat{V}}_k, S_{k+1}) + I(S_{k+1}; \hat{V}_k)
\]

\[
= H(\hat{V}_k, S_{k+1}) + H(\hat{V}_k, \hat{\hat{V}}_k|S_{k+1}) + I(S_{k+1}; \hat{V}_k)
\]

\[
= H(\hat{V}_k, S_{k+1}) + I(S_{k+1}; \hat{V}_k) \quad \{ \text{since } \hat{V}_k \text{ is determined by } S_{k+1} \}
\]

\[
\leq \varphi(Pr(\hat{V}_k \neq V_k)) + I(S_{k+1}; \hat{V}_k)
\]

where the last step is Fano's inequality (see Sections 4.2.4 and 4.2.5). Hence, for large \( R \):

\[
-159-
\]
Let \( n' \) be the value of \( n \) for which the maximum is assumed in the above sum; so

\[
K \cdot K_{av} = K \cdot \sum_{c} \log(M_c) / (Nq) = \sum_{k=1}^{qK} H(V_k) / (Nq)
\]

\[
\leq \sum_{k=1}^{qK} I(S_{k+1}^{n'}; V_{k}) / (Nq) + \sum_{k=1}^{qK} \varphi(Pr\{\hat{V}_k \neq V_k\}) / (Nq)
\]

\[
\leq \max_{n} \sum_{k=1}^{qK} I(S_{k+1}^{n}; V_{k} | S_{k+1}^{n}) / q + \sum_{k=1}^{qK} \varphi(Pr\{\hat{V}_k \neq V_k\}) / (Nq).
\]

Let \( n' \) be the value of \( n \) for which the maximum is assumed in the above sum; so

\[
K \cdot K_{av} \leq \sum_{k=1}^{qK} I(S_{k+1}^{n'}; V_{k} | S_{k+1}^{n'}) / q + \sum_{k=1}^{qK} \varphi(Pr\{\hat{V}_k \neq V_k\}) / (Nq).
\]

Now

\[
I(S_{k+1}^{n'}; V_{k} | S_{k+1}^{n'}) = H(S_{k+1}^{n'} | S_{k+1}^{n'}) - H(S_{k+1}^{n'} | V_{k}, S_{k+1}^{n'})
\]

\[
\leq H(S_{k+1}^{n'} | S_{k+1}^{n'}) - H(S_{k+1}^{n'} | V_{k}, S_{k+1}^{n'}, S_{k}^{n'}, I_{k}(n'))
\]

\[
= H(S_{k+1}^{n'} | S_{k+1}^{n'}) - Pr\{I_{k}(n') = \emptyset\} \cdot H(S_{k}^{n'} | V_{k}, S_{k+1}^{n'}, S_{k}^{n'}, I_{k}(n') = \emptyset)
\]

\[
= H(S_{k+1}^{n'} | S_{k+1}^{n'}) - Pr\{I_{k}(n') = \emptyset\} \cdot H(S_{k}^{n'} | S_{k}^{n'}).
\]

The last equality follows from the fact that \( S_{k+1} = X_k \circ S_k \) and from the definition of a case 4—WM code. Substituting this and dividing by \( K \) yields

\[
K_{av} \leq \sum_{k} \left\{ H(S_{k+1}^{n'} | S_{k+1}^{n'}) - Pr\{I_{k}(n') = \emptyset\} \cdot H(S_{k}^{n'} | S_{k}^{n'}) \right\} / (Kq) + \varphi(Pr\{\hat{V}_k \neq V_k\}) / (Kq)
\]

\[
\leq \sum_{k} \left\{ (1 - Pr\{I_{k}(n') = \emptyset\}) \cdot H(S_{k}^{n'}) / (Kq) \right\} + 0(1/K) + \varphi(Pr\{\hat{V}_k \neq V_k\}) / (Kq)
\]

by convexity of \( \varphi(\cdot) \). This can be bounded from above by

\[
\sum_{k} \left\{ (1 - Pr\{I_{k}(n') = \emptyset\}) \cdot H(S_{k}^{n'}) / (Kq) \right\} + 0(1/K) + \varphi(Pr\{\hat{V}_k \neq V_k\}) / (Kq) / N
\]
The Write Unidirectional Memory

\[ \begin{align*}
= & \sum_k \{ H(S_{k+1}(n')) - \Pr[I_k(n')=\emptyset] \cdot H(S_k(n')) \}/(Kq) + O(1/K) + \\
& + \varphi(\sum_k \Pr[\hat{V}_k \notin V_k]/(Kq))/N \\
= & \sum_k \{ H(S_{k+1}(n')) - H(S_{k+1}(n')|I_k(n')) \}/(Kq) + O(1/K) + \\
& + \varphi(\sum_k \Pr[\hat{V}_k \notin V_k]/(Kq))/N \\
= & \sum_k \{ H(I_k(n')) - H(I_k(n')|S_{k+1}(n')) \}/(Kq) + O(1/K) + \\
& + \varphi(\sum_k \Pr[\hat{V}_k \notin V_k]/(Kq))/N.
\end{align*} \]

Let \( p_c := \Pr[I_k(n') = c] \) for \( k \equiv c \mod q \); by definition of the code, this number only depends on \( c \) and \( n' \). Define \( \beta_i^k := \Pr[S_k(n') = i] \) for all \( i \) and \( k \). As argued in the converse proof for case 3 (Section 4.2.5), the distribution of \( S_k(n') \) is described by a Markov chain. It should be noticed that we have exactly the same model here. Hence the number \( \beta_i^k \) tends to \( a_i^c \) if \( k \equiv c \mod q \), where the \( a_i^c \) satisfy (A.3). Hence, for large enough \( k \) with \( k \equiv c \mod q \):

\[ \begin{align*}
H(I_k(n')) - H(I_k(n')|S_{k+1}(n')) \\
\leq h(p_c) - a_c^{c+1} \cdot h(p_c/a_c^{c+1}) + \varepsilon \\
= h(p_c) - h(P) \cdot p_c/(1-P) + \varepsilon.
\end{align*} \]

Consequently

\[ \begin{align*}
\mathbb{E}_v \leq \sum_{c \in A} \left\{ h(p_c) - \frac{p_c \cdot h(P) + \varepsilon}{1-P} \right\}/q + O(1/K) + \varphi(\sum_k \Pr[\hat{V}_k \notin V_k]/(Kq))/N.
\end{align*} \]

This holds for all large \( K \), so we may take the limit for \( K \to \infty \). Since \( P_{err}(C) \leq \varepsilon \),

\[ \begin{align*}
\mathbb{E}_v \leq \sum_{c \in A} \left\{ h(p_c) - \frac{p_c \cdot h(P) + \varepsilon}{1-P} \right\}/q + \varphi(\varepsilon)/N.
\end{align*} \]

Letting \( \varepsilon \downarrow 0 \) gives the desired result. \( \Box \)
Chapter 4

Theorem (4.3.7) enables us to find $c_{4,av}$ by maximizing the rate sum over $\mathbb{Z}_q$. Again this was done by computer. As predicted, the maximum is assumed for a vector $(p_0, p_1, \ldots, p_{q-1})$ with $p_0 = p_1 = \cdots = p_{q-1}$. For $q > 2$ this follows from the computer results, but for $q = 2$, it is proved in Appendix C that

$$c_{4,av} = \max \{ h(p) - h(p^2)/(1+p) \mid 0 \leq p \leq 1 \} = 0.54589 \ldots$$

It follows that $c_{4,av}$ equals the upper bound on the capacity for case 4 period-1 codes found in [WIL89]. In other words, Willems' converse for period-1 codes also holds for the larger class of period-2 codes.

Table (4.3.8) lists $c_{4,av}$ for $i \in \{1, 3, 4\}$ and $q \leq 10$. Since $c_{2,av} = c_{1,av}$, this gives the average capacity for all cases. The table also lists the values of $(q-1)/q$, which can be achieved with the code described in the proof of Theorem (4.3.2).

<table>
<thead>
<tr>
<th>$q$</th>
<th>$c_{1,av}$</th>
<th>$c_{3,av}$</th>
<th>$c_{4,av}$</th>
<th>$(q-1)/q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.69424</td>
<td>0.69424</td>
<td>0.54589</td>
<td>0.50000</td>
</tr>
<tr>
<td>3</td>
<td>0.87915</td>
<td>0.87305</td>
<td>0.76072</td>
<td>0.66667</td>
</tr>
<tr>
<td>4</td>
<td>0.94678</td>
<td>0.94089</td>
<td>0.86572</td>
<td>0.75000</td>
</tr>
<tr>
<td>5</td>
<td>0.97523</td>
<td>0.97101</td>
<td>0.92239</td>
<td>0.80000</td>
</tr>
<tr>
<td>6</td>
<td>0.98811</td>
<td>0.98545</td>
<td>0.95462</td>
<td>0.83333</td>
</tr>
<tr>
<td>7</td>
<td>0.99419</td>
<td>0.99268</td>
<td>0.97345</td>
<td>0.85714</td>
</tr>
<tr>
<td>8</td>
<td>0.99713</td>
<td>0.99626</td>
<td>0.98456</td>
<td>0.87500</td>
</tr>
<tr>
<td>9</td>
<td>0.99858</td>
<td>0.99811</td>
<td>0.99112</td>
<td>0.88889</td>
</tr>
<tr>
<td>10</td>
<td>0.99929</td>
<td>0.99904</td>
<td>0.99496</td>
<td>0.90000</td>
</tr>
</tbody>
</table>

(4.3.8) Table. Values of $c_{i,av}$ for $i \in \{1, 3, 4\}$ and $q \leq 10$.

This concludes the chapter on WUMs over arbitrary alphabets. We have found the capacity region for three out of four cases, with a strong conjecture about the fourth one inspired by the known average capacity for this case. It remains an interesting open problem to find a proof for the converse in case 4.

Another problem is of course the construction of good codes for any of the
four cases. Some constructions have been found by the author, but these are basically time sharing codes that give only slight improvements over the trivial code with average rate \((q-1)/q\).

Appendix A

In this appendix we study the distribution of the WUM state \(S_{k+1} \in \mathcal{X}^N\) (notation as in Section 4.2.5 and 4.2.6) for fixed \(N\) and large values of \(k\), under the assumption that at each instant \(k\) with \(k \equiv c \mod q\), every \(c\)-filter \(x\) having \(p_c \cdot N\) \(c\)'s and \((1-p_c) \cdot N\) \(\square\)'s has the same probability of being written by the encoder. It should be noted that this is indeed the case with every code in the ensemble considered in the random coding proofs in both Section 4.2.5 and 4.2.6.

Let us abbreviate \(S_{k+1}^c\) by \(S_k^c\) and let \(I_k\) be the \(c\)-filter that transforms state \(S_k^c\) into \(S_{k+1}^c\). We consider a Markov chain similar to the one defined in Section 4.2.4. Its states are called \((c, s)\), \(c \in \mathcal{X}, 0 \leq s \leq q^N - 1\). The states \((c, s)\) correspond to all possibilities for \(S_k\) if \(k \equiv c \mod q\). The transition probabilities follow from the distribution of \(I_k\); the probability of going from state \((c, s) = S_k\) to state \((c+1, s') = S_{k+1}\) equals the probability that a \(c\)-filter \(I_k\) is chosen for which \(S_{k+1} = I_k \circ S_k^c\). We will show that the stationary probability distribution of this chain is such that \(S_k^c\) is typical (Definition (4.2.5.4)) with large probability.

First consider a single cell of the WUM, \(S_k(n)\). Let \(\beta_i^k := \Pr\{S_k(n) = i\}\) and \(k \equiv c \mod q\). Define \(p_c^c := 1-p_c\). By the definition of the code ensemble, we have for every code:

\[(A.1) \ Pr\{I_k(n) = c\} = p_c \quad \text{and} \quad Pr\{I_k(n) = \square\} = p_o^c\]
Chapter 4

The way in which the WUM is used implies

\[ \beta_{i+1}^{k+1} = p_{\square_i} \cdot \beta_i^k \quad \text{if } i \neq c, \]

\[ \beta_c^{k+1} = \beta_c^k + p_c (1 - \beta_c^k) = p_c + p_{\square_c} \cdot \beta_c^k. \]

Consider the Markov chain that can be derived from the original Markov chain by looking at the \(n\)th cell only; it has states \((c, i)\) where \(i\) is the value of \(S_k(n)\). This Markov chain has the following stationary distribution. Let \(a_i^c\) \((i \in \mathcal{A}, c \in \mathcal{A})\) satisfy

\[
\begin{align*}
\sum_{i \in \mathcal{A}} a_i^c &= 1, \\
\tag{A.2} a_{i+1}^{c+1} &= p_{\square_i} \cdot a_i^c \quad \text{for } i \neq c, \\
a_c^{c+1} &= p_c + p_{\square_c} \cdot a_c^c.
\end{align*}
\]

Then the stationary probability of state \((c, i)\) equals \(a_i^c / q\). In other words, for fixed \(c\), \(k \equiv c \mod q\) and \(k \to \infty\), we have \(\beta_i^k \to a_i^c\).

The solution of (A.2) (which is equivalent to (4.2.5.5)) is the following. For \(i, j \in \mathcal{A}\) and a \(q\)-tuple \((a_0, a_1, \ldots, a_{q-1})\), we define the 'cyclic product' \(\hat{\Pi}\) by

\[
\hat{\Pi}(a_i, a_j) := \begin{cases} 
a_i \cdot a_{i+1} \cdots a_j & \text{if } i \leq j \text{ and not } (i, j) = (0, q-1) \\
1 & \text{if } i \equiv j+1 \mod q \\
& \text{if } i \geq j+2 
\end{cases}
\]

Then it can be checked that

\[
\tag{A.3} a_i^c = p_i \cdot \hat{\Pi}(p_i^{c+1}, p_i^{c-1}) / (1 - p) \quad \text{with } P := \prod_{c \in \mathcal{A}} (p_c^c).
\]

Next, we look at two distinct cells, \(S_k(n_0)\) and \(S_k(n_1)\). For \(k \equiv c \mod q\) we define
\[ p_{cc} := \Pr\{X_k(n_0) = X_k(n_1) = c\}, \]
\[ p_{c\emptyset} := \Pr\{X_k(n_0) = c, X_k(n_1) = \emptyset\} \text{ and } p_{\emptyset c} \text{ analogously}, \]
\[ p_{cc}^c := \Pr\{X_k(n_0) = X_k(n_1) = \emptyset\}. \]

From the definition of the code, we find

\[
\begin{align*}
    p_{cc} &= \frac{(p_c^n)(p_c^{n-1})}{N(N-1)}, \\
p_{c\emptyset} &= p_{\emptyset c} = \frac{(p_c^n)(p_{\emptyset c}^n)}{N(N-1)}, \\
p_{cc}^c &= \frac{(p_{\emptyset c}^n)(p_{cc}^{n-1})}{N(N-1)}.
\end{align*}
\]

We can determine the stationary probability distribution of the pair \((S_k(n_0), S_k(n_1))\) from a set of equations similar to (A.2). For \(k \equiv c \mod q\) and \(k \to \omega\), we find:

\[ \Pr\{S_k(n_0) = i, S_k(n_1) = j\} \rightarrow a_{ij}^c, \]

where \(a_{ij}^c\) satisfies

\[
\begin{align*}
    \sum_{ij} a_{ij}^c &= 1, \\
a_{i+1,j}^c &= p_{\emptyset\emptyset} \cdot a_{ij}^c \quad \text{if } i \neq c, j \neq c, \\
a_{i+1,c}^c &= a_{i+c}^c = a_{i,c}^c \cdot p_{c\emptyset} + a_{i,c}^c \cdot p_{\emptyset c} \quad \text{if } i \neq c, \\
a_{c+1,c}^c &= p_{cc} + 2a_{c,c}^c \cdot p_{c\emptyset} + a_{cc}^c \cdot p_{\emptyset c^c}.
\end{align*}
\]

The solution of (A.5) turns out to be

\[
\begin{align*}
a_{ii}^c &= \frac{p_{\emptyset\emptyset}^i + 2a_{ij}^i \cdot p_{\emptyset\emptyset}^i \cdot \tilde{\Pi}(p_{\emptyset\emptyset}^i + 1, p_{\emptyset\emptyset}^{i-1})}{1 - p_{\emptyset\emptyset}}, \\
a_{ij}^c &= \frac{p_{\emptyset\emptyset}^i \cdot \tilde{\Pi}(p_{\emptyset\emptyset}^i + 1, p_{\emptyset\emptyset}^{i-1}) + a_{ij}^i \cdot p_{\emptyset\emptyset}^i \cdot \tilde{\Pi}(p_{\emptyset\emptyset}^i + 1, p_{\emptyset\emptyset}^{i-1})}{1 - p_{\emptyset\emptyset}} \quad \text{if } i \neq j,
\end{align*}
\]

- 165 -
where \( P^c_\infty \) := \( \prod_{c \in \mathcal{A}} (p^c_\infty) \).

Solution (A.6) holds for any simultaneous distribution of \((I_k(n_0), I_k(n_1))\) that can be expressed in \( p^c_{cc}, p^c_{co} \) and \( p^c_\infty \). Indeed, we only need some obvious relations like \( p^c_{cc} + p^c_{co} = p^c_c \), \( p^c_{co} + p^c_\infty = p^c_\infty \) and \( p^c_{cc} + 2p^c_{co} + p^c_\infty = 1 \), but (A.4) is not used. Taking \( p^c_{cc} := (p^c_c)^2, p^c_{co} := p^c_c \cdot p^c_\infty \) and \( p^c_\infty := (p^c_\infty)^2 \) gives a valid distribution, corresponding to independent \( I_k(n_0) \) and \( I_k(n_1) \), each with probability \( p^c_c \) of writing \( c \). In this case, the cells of \( S_k \) are also independent, so for all \( c, i \) and \( j \) (which can be checked from (A.6) and (A.3)).

Again we look at (A.4). It is easy to see that
\[
\begin{align*}
  P^c_{cc} &= (p^c_c)^2(1 + O(1/N)),
  P^c_{co} &= p^c_c \cdot p^c_\infty(1 + O(1/N)),
  p^c_\infty &= (p^c_\infty)^2(1 + O(1/N)).
\end{align*}
\]

Now we can apply some standard asymptotics in (A.6), like
\[
\begin{align*}
  P^c_\infty &= \prod_{c} (p^c_\infty) = \{\prod (p^c_\infty)\}^2 \cdot (1 + O(1/N)),
\end{align*}
\]
to find \( a^c_{i,j} = (a^c_i)(a^c_j)(1 + O(1/N)) \).

Let \( \delta_0 := \delta^4/N \) with \( \delta \) as defined in Sections 4.2.5 and 4.2.6. We can find a \( \mathcal{K}_N \) (depending on \( p_0, \ldots, p_{q-1}, N \) and \( \delta \)) such that for \( k \geq \mathcal{K}_N \) and \( k \equiv c \mod q \):

\[
\begin{align*}
  \forall i & \left| \Pr\{S_k(n) = i\} - a^c_i \right| < \delta_0, \\
  \forall i, j & \left| \Pr\{S_k(n_0) = i, S_k(n_1) = j\} - a^c_{i,j} \right| < \delta_0.
\end{align*}
\]

Then, with \( \mu^c_{k,i} := E(\phi(S_k(n) = i)) \):

\[
\begin{align*}
  \left| \mu^c_{k,i} - a^c_i \right| &< \delta_0, \\
  \text{var}(\phi(S_k(n) = i)) &\leq a^c_i (1 - a^c_i) + 2\delta_0, \\
  \text{cov}(\phi(S_k(n_0) = i), \phi(S_k(n_1) = i)) &= a^c_{i,i} - a^c_i \cdot a^c_i + 2\delta_0 = 0(1/N) + 2\delta_0.
\end{align*}
\]
The last inequality is Čebyšev's inequality. We can write this as

\[
\frac{1}{(\delta - \delta_0)^2 N^2} \sum_i \var \{ \varphi(S_k(n) = i) \} \\
= \frac{1}{(\delta - \delta_0)^2 N^2} \sum_i \var \{ \varphi(S_k(n) = i) \} + 2 \frac{2}{(\delta - \delta_0)^2 N^2} \sum_i \sum_{n_0 < n_1} \cov(\varphi(S_k(n_0) = i), \varphi(S_k(n_1) = i)) \\
\leq \frac{N}{(\delta - \delta_0)^2 N^2} \sum_i (a_i^C(1 - a_i^C) + 2\delta_0) + \frac{2N^2}{(\delta - \delta_0)^2 N^2} \var \{ \theta(1/N) + 2\delta_0 \} \\
= \frac{1}{(\delta - \delta_0)^2} \theta(1/N) + \frac{4q\delta_0}{(\delta - \delta_0)^2}.
\]

Since \( \delta \leq 1/2 \) and \( \delta_0 = \delta^4/N \), we find

\[
\frac{1}{(\delta - \delta_0)^2} < \frac{1}{3\delta^3} \quad \text{and} \quad \frac{4q\delta_0}{(\delta - \delta_0)^2} < \frac{1}{3\delta^3} \theta(1/N).
\]

Therefore, for \( k > \theta N \), \( \Pr\{S_k \text{ not typical} \} \) can be bounded by \( \frac{1}{\delta^3} \theta(1/N), \; N \to \infty. \)

**Appendix B**

We consider the function \( \mathcal{I}: [0,1)^q \to \mathbb{R}, \)

\[
(B.1) \quad \mathcal{I}(a) := \sum_{c \in \mathcal{A}} \sum_{i \neq c} a_i^C \cdot \frac{\frac{a_{i+1}^C}{a_i^C}}{h(a_{i+1}^C)}
\]

with \( a := (a_0^0, a_0^1, \ldots, a_{q-1}^1) \). We also define the region \( G, \)
In this appendix we will prove that

\[(B.3) \max \{ R(\alpha) | \alpha \in \mathcal{G} \} = q \cdot \log(\zeta_0),\]

with \(\zeta_0\) as in Definition (4.3.3). First we note that, using the definition of \(h(\cdot)\),

\[
R(\alpha) = \sum_{c} \sum_{i \neq c} \begin{cases} -a_i^{c+1}\log(a_i^{c+1}) + a_i^{c+1}\log(a_i^c) \\ (a_i^c - a_i^{c+1})\log(a_i^c - a_i^{c+1}) + (a_i^c - a_i^{c+1})\log(a_i^c) \end{cases}.
\]

By changing the order of summation, we obtain

\[(B.4) R(\alpha) = \sum_{i} \left( a_i^{i+1}\log(a_i^{i+1}) - a_i^{i}\log(a_i^{i}) \right) - \sum_{c \neq i} a_c^{c+1}\log(a_c^c - a_c^{c+1}).\]

We define the vector \(\gamma := (\gamma_0^1, \gamma_0^2, \ldots, \gamma_{q-1}^q)\) by

\[
\gamma_i^c := \begin{cases} a_i^c - a_i^{c+1} & \text{if } c \neq i \\ a_i^c & \text{if } c = i. \end{cases}
\]

This implies

\[a_i^c = \hat{\hat{\sum}}(\gamma_i^c, \gamma_i^i),\]

where \(\hat{\hat{\sum}}(\cdot)\) is the 'cyclic sum' defined in a way similar to the 'cyclic product' in Appendix A:

\[\hat{\hat{\sum}}(\gamma_i^c, \gamma_i^i) := \begin{cases} \gamma_i^c + \gamma_i^{c+1} + \ldots + \gamma_i^i & \text{if } c \leq i \\ \gamma_i^c + \gamma_i^{c+1} + \ldots + \gamma_i^{q-1} + \gamma_i^0 + \ldots + \gamma_i^i & \text{if } c > i. \end{cases}\]

Now (B.4) yields: \(R(\alpha) = R^*(\gamma)\), with

\[(B.5) R^*(\gamma) = \sum_{i} \left( \sum_{c} \gamma_i^c \cdot \log(\gamma_i^c) \right) - \sum_{c} \gamma_i^c \log(\gamma_i^c).\]

The set of restrictions \(\mathcal{G}\) in (B.2) translates to \(\mathcal{G}^*\),

---

Page 168
Now we have to find the maximum of \( \mathbf{x}^* \) over the region \( G^* \). Since \( G^* \) is a compact region, \( \mathbf{x}^* \) has a global maximum on \( G^* \). It is easy to see that points on the border (some \( \gamma_i^c = 0 \) or \( \gamma_i^C = 1 \) do not correspond to a global maximum: if \( \gamma_i^C = 0 \), the partial derivative \( \frac{\partial \mathbf{x}^*}{\partial \gamma_i^c} \) tends to \( +\infty \) for \( \gamma_i^C \downarrow 0 \), so the function value increases if we increase \( \gamma_i^C \) a little at the expense of some \( \gamma_i^d > 0 \). If \( \gamma_i^C = 1 \), it can be checked that \( \mathbf{x}^*(z) < q-1 \), whereas the global maximum is larger than this (cf. Table (4.3.8)). For the interior points we use the Lagrange multiplier method.

Let, for all \( c \),

\[
g_c(z) := \sum_{i \in \mathcal{I}} \hat{E}(\gamma_i^c, \gamma_i^d) - 1.
\]

Then we can write the restrictions in the form \( g_c(z) = 0 \). Therefore parameters \( \lambda_0, \lambda_1, \ldots, \lambda_{q-1} \) in \( \mathbb{R} \) exist such that

\[
\begin{align*}
(\text{B.7}) \quad & g_c(z) = 0 \quad \text{for all } c \\
& \frac{\partial (\mathbf{x}^* - \sum c \lambda_c g_c)}{\partial \gamma_j^d} = 0 \quad \text{for all } j \text{ and } d.
\end{align*}
\]

Computing partial derivatives, we find

\[
\frac{\partial g_c}{\partial \gamma_j^d} = \begin{cases} 
1 & \text{if } c \leq d \leq j \text{ or } d \leq j < c \text{ or } j < c \leq d \\
0 & \text{otherwise},
\end{cases}
\]

so (B.7) implies

\[
(\text{B.8}) \quad \log(\sum_c \gamma_j^c) - \log(\gamma_j^d) = \sum E(\lambda_{j+1}, \lambda_d)
\]

for all \( j \) and \( d \). Multiplying by \( \gamma_j^d \) and summing over \( j \) and \( d \) yields, with (B.5),
By (B.7), we have \( g_c(\pi) = 0 \), so

(B.9) \( \mathbf{E}^*(\pi) = \sum_c \lambda_c \).

From (B.8) we find for all \( j \) and \( d \)

\[
\gamma_j^d = (\sum_c \gamma_j^c) \cdot 2^{-\hat{E}(\lambda_{j+1}, \lambda_d)}.
\]

Summing over \( d \) yields

\[
\sum_d \gamma_j^d = (\sum_c \gamma_j^c) \cdot \sum_d 2^{-\hat{E}(\lambda_{j+1}, \lambda_d)},
\]
or

(B.10) \( \sum_d 2^{-\hat{E}(\lambda_{j+1}, \lambda_d)} = 1 \) for all \( j \).

Hence we find

\[
1 = \sum_d 2^{-\hat{E}(\lambda_{j}, \lambda_d)} = \sum_{d \neq j} 2^{-\hat{E}(\lambda_{j+1}, \lambda_d)} - \lambda_j + \hat{E}(\lambda_{j+1}, \lambda_d) + 2^{-\hat{E}(\lambda_{j}, \lambda_d)} - \lambda_j
\]

\[
= 2^{-\hat{E}(\lambda_{j+1}, \lambda_d)} + 1 \Rightarrow 2^{-\hat{E}(\lambda_{j+1}, \lambda_d)} = 2^{-\hat{E}(\lambda_{j}, \lambda_d)} - 2 \bar{\Lambda}_i + 1.
\]

By (B.10), this equals

\[
2^{-\hat{E}(\lambda_{j}, \lambda_d)} = 2^{-\hat{E}(\lambda_{j+1}, \lambda_d)} - \frac{\bar{\Lambda}_i}{2}.
\]

Therefore

(B.11) \( 2^{-\hat{E}(\lambda_{j}, \lambda_d)} = 2^{-\bar{\Lambda}_i} \) \( \Rightarrow 2^{-\bar{\Lambda}_i} = 1 \) for all \( j \).

Clearly (B.11) can hold only if all \( \lambda_j \) are equal. We define \( \lambda \) as the common value of all \( \lambda_j \): \( \lambda_0 = \lambda_1 = \ldots = \lambda \). Substituting this in (B.10) yields
The Write Unidirectional Memory

\[ \sum_{d} 2^{-(d-j)\lambda} = 1 \text{ for all } j, \]

where \((d-j)\) should be considered modulo \(q\), with values in \(\{1, 2, \ldots, q\}\). Hence

\[ \sum_{i=1}^{q} 2^{-i\cdot\lambda} = 1. \]

Setting \(x := 2^\lambda\), we get

\[ x^{-1} + x^{-2} + \ldots + x^{-q} = 1 \text{ or } x^q - x^{q-1} - x^{q-2} - \ldots - 1 = 0. \]

In other words,

\[ \frac{f(x)}{x - 1} = 0 \]

with \(f(\cdot)\) as defined in (4.3.3). The real roots of this equation are the ones among \(\zeta_0, \zeta_1, \ldots, \zeta_q\) that are unequal to 1, so the solutions for \(\lambda\) are the real ones \((\neq 0)\) among \(\log(\zeta_0), \log(\zeta_1), \ldots, \log(\zeta_q)\); notice that \(\log(\zeta_0)\) is among them by \((ii)\) of Lemma (4.3.4).

By (B.9), we have for every solution for \(\lambda\):

\[ F^*(x) = q \cdot \lambda. \]

Each of these corresponds to a stationary point of the function \(F^*\). Obviously, the global maximum is found for the largest one among them, \(\lambda = \log(\zeta_0)\). This proves that

\[ \max \{ F^*(x) \mid x \in G^* \} = q \cdot \log(\zeta_0) \]

which is equivalent to (B.3).
Appendix C

Here we prove that the average case 4 capacity, $c_{4, \text{av}}$, equals
\[
\max \{ h(p) - h(p^2)/(1+p) \mid 0 \leq p \leq 1 \}.
\]

From Theorem (4.3.7) and Definition (4.2.2.5), we see that
\[
c_{4, \text{av}} = \max \left\{ \frac{p_0 + p_1}{1 - (1-p_0)(1-p_1)} \cdot h((1-p_0)(1-p_1))/2 \mid 0 \leq p_0 \leq 1, 0 \leq p_1 \leq 1 \right\}.
\]
Letting $a := 1-p_0$, $\beta := 1-p_1$ and $f(a, \beta) := h(a) + h(\beta) - \frac{2 \cdot a \cdot \beta}{1-a \cdot \beta} h(a \cdot \beta)$, we find
\[
c_{4, \text{av}} = \max \{ f(a, \beta)/2 \mid 0 \leq a \leq 1, 0 \leq \beta \leq 1 \}.
\]

If we can show that
\[
\max \{ f(a, \beta) \mid 0 \leq a \leq 1, 0 \leq \beta \leq 1 \} = \max \{ f(a, a) \mid 0 \leq a \leq 1 \},
\]
then the proof is finished. First of all, the maximum of $f(a, \beta)$ is at least $f(0.2887, 0.2887) \approx 1.09177$. If $a = 0$, $a = 1$, $\beta = 0$ or $\beta = 1$, then $f(a, \beta) \leq 1$, so the maximum is not attained on the border of the region $[0,1]^2$. Hence the maximum corresponds to a point $(a, \beta)$ for which
\[
\frac{\partial f}{\partial a} = 0 \quad \text{and} \quad \frac{\partial f}{\partial \beta} = 0.
\]

Straightforward calculus shows
\[
(C.1) \quad \frac{\partial f}{\partial a} = \log \left( \frac{(1-a) \beta}{1-a \beta} \right) - \left( \frac{1-\beta}{1-a \beta} \right)^2 \log (a \beta)
\]
and $\frac{\partial f}{\partial \beta}$ is equal to $\frac{\partial f}{\partial a}$ with $a$ and $\beta$ interchanged. With $z := \frac{1-a}{1-a \beta}$ and $y := \frac{1-\beta}{1-a \beta}$, (C.1) yields
\[
(C.2) \quad \frac{\log (1-y)}{y} = \frac{\log (1-z)}{z} = h'(x) + h'(y).
\]
Let $g_1(x) := \frac{\log (1-x)}{x}$ and $g_2(x) := g_1(x) - h'(x)$. Then, with (C.2):
The Write Unidirectional Memory

(C.3) \( g_1(x) = g_1(y) \).

(C.4) \( h'(x) = g_2(y) \) and \( h'(y) = g_2(z) \).

Analysis of the functions \( g_1 \) and \( g_2 \) yields the following lemmas.

(C.5) Lemma. A number \( x_0 \in (0.71, 0.72) \) exists such that \( g_1 \) is monotonically increasing on \([0.5, x_0)\) and monotonically decreasing on \((x_0, 1)\). □

(C.6) Lemma. Function \( g_2 \) is monotonically increasing on \((0, 1)\) and \( g_2(z) < 0 \) for all \( z \in (0, 1) \). □

Because of (C.4) and Lemma (C.6), we have \( h'(x) < 0 \), so \( x > 0.5 \). Now assume \( x < x_0 \) (defined in Lemma (C.5)). Then, by (C.4) and Lemma (C.6):

\[
  h'(y) = g_2(z) \leq g_2(x_0) < g_2(0.72) < -2.18.
\]

Since \( h'(y) \) is decreasing in \( y \), this implies \( y > 0.819 \). With Lemma (C.5) we have

\[
  g_1(y) < g_1(0.819) < -3.676.
\]

This implies \( x < 0.6 \), since otherwise \( g_1(x) > g_1(0.6) > -3.673 \), violating (C.3). Again, we use (C.4) and Lemma (C.6):

\[
  h'(y) = g_2(z) < g_2(0.6) < -3.08.
\]

Therefore \( y > 0.894 \). But now find a contradiction with (C.3):

\[
  g_1(y) < g_1(0.894) < -4.05 < g_1(0.5) < g_1(z).
\]

We conclude that \( x \geq x_0 \). Similarly we must have \( y \geq x_0 \). Now the equality \( g_1(x) = g_1(y) \) together with the fact that \( g_1 \) is monotonic on \((x_0, 1)\) shows that \( x = y \). It is easy to check from the definitions of \( x \) and \( y \) that this implies \( a = \beta \).

This concludes the proof.
Chapter 5: Some Connections between Write Unidirectional Memories and Two-Way Channels

5.1. Introduction: the binary WUM and the BMC

In this final chapter, we investigate some possible links between the TWCs studied in Chapters 2 and 3 and the WUM studied in Chapter 4. This search for connections was initiated by some remarkable similarities in the results for the binary WUM and the Binary Multiplying Channel defined in Example (2.1.2). The most striking resemblance is the following. We recall that the Shannon outer bound \( G_0 \) for the BMC equals (see (2.2.10))

\[
G_0 = \{ (\beta \cdot h(\frac{1-a}{\beta}) , a \cdot h(\frac{1-\beta}{a})) \mid 0 \leq a \leq 1, 0 \leq \beta \leq 1, a + \beta \geq 1 \}.
\]

From Theorem (4.2.2.2) it follows that the capacity region \( C_{1,2} \) for cases 1 and 2 of the binary WUM equals

\[
C_{1,2} = \{ (a_1^0 \cdot h(a_1^0/a_1^0), a_0^1 \cdot h(a_0^0/a_0^1) \mid a_1^0 + a_0^1 = 1, a_1^1 + a_0^1 = 1, \\
\quad a_1^0 \geq a_0^0, a_1^1 \geq a_0^1 \}.
\]

(No 'co' is necessary, since it turns out that the region is convex.) If we take \( a_1^0 = a \) and \( a_1^1 = \beta \), we find that region \( C_{1,2} \) equals \( G_0 \). Furthermore Theorem (4.2.2.4) yields the following expression for the case 3 capacity region \( C_3 \):

\[
C_3 = \{ h(p_0) \cdot \{1 - \frac{p_0(1-p_1)}{1-(1-p_0)(1-p_1)}\}, h(p_1) \cdot \{1 - \frac{p_1(1-p_0)}{1-(1-p_0)(1-p_1)}\} \mid \\
\quad 0 \leq p_0 \leq 1, 0 \leq p_1 \leq 1 \}
\]
Some Connections

Summarizing, we see that the Shannon outer bound for the BMC equals the capacity region for the binary WUM in cases 1, 2 and 3. By looking more closely at the expressions for the outer bound and capacity region, respectively, we can show why this equality is not purely 'coincidental'.

As derived in Chapter 2, the BMC outer bound region consists of pairs \( \mathcal{H}(Y|I_1), \mathcal{H}(Y|I_0) \) for any distribution \( P(x_0, x_1) \). It is a well-known fact ([SCH83a]) that the points on the border of this region have \( P(0,0) = 0 \). In other words, if we use the graphical representation, the optimum resolution takes place in an L-shape \( \mathcal{A} \) as in Figure (5.1.1). Here \( p_0 := \Pr\{I_0 = 1| I_1 = 1, \text{ vertex } \mathcal{A} \} \) and \( p_1 := \Pr\{I_1 = 1| I_0 = 1, \text{ vertex } \mathcal{A} \} \).

\[
\mathcal{A} = \left\{ \left( \frac{p_1 \cdot h(p_0)}{1-(1-p_0)(1-p_1)}, \frac{p_0 \cdot h(p_1)}{1-(1-p_0)(1-p_1)} \right) \mid 0 \leq p_0 \leq 1, 0 \leq p_1 \leq 1 \right\}.
\]

Taking \( a = \frac{p_0}{p_0 + p_1 \cdot p_0 p_1} \) and \( \beta = \frac{p_1}{p_0 + p_1 \cdot p_0 p_1} \) shows that \( C_3 = \mathcal{G}_0 \).

(5.1.1) Figure. Outer bound resolution for the BMC.

The total weight of vertex \( \mathcal{A} \) is \( 1-(1-p_0)(1-p_1) \). Therefore we find, with \( a_0 := \Pr\{I_0 = 1| \mathcal{A} \} \) and \( a_1 := \Pr\{I_1 = 1| \mathcal{A} \} \):
Chapter 5

\[ 1 - p_0 = Pr\{I_0 = 0 | I_1 = 1, A\} = \frac{Pr\{I_0 = 0 | A\}}{Pr\{I_1 = 1 | A\}} = \frac{1 - a_0}{a_1} \]

and \( 1 - p_1 = \frac{(1 - a_1)}{a_0} \). Therefore

\begin{align*}
H(Y | I_1) &= a_1 \cdot h(p_0) \\
H(Y | I_0) &= a_0 \cdot h(p_1)
\end{align*}

with \( a_1 = \frac{p_1}{1 - (1 - p_0)(1 - p_1)} \), \( a_0 = \frac{p_0}{1 - (1 - p_0)(1 - p_1)} \).

Now we look at the binary WUM. A typical rate point in the capacity region looks like \((H(S_1 | S_0), H(S_2 | S_1))\), which follows from the proof in Section (4.2.4). The notation is the same as in Chapter 4: \( S_k \) denotes a state and \( I_k \) denotes a filter. If we define \( p_0 := Pr(I_0 = 0), \ p_1 := Pr(I_1 = 1), \ a_0 := Pr\{S_1 = 0\} \) and \( a_1 := Pr\{S_0 = 1\} \), then we find

\begin{align*}
H(S_1 | S_0) &= a_1 \cdot h(p_0), \\
H(S_2 | S_1) &= a_0 \cdot h(p_1).
\end{align*}

It remains to show that the \( a_k \) can be expressed in the \( p_k \) as in (5.1.2). To stress the similarity, we draw the distributions and relations between \( S_k \) and \( S_{k+1} \) in a figure like (5.1.1).

![Diagram](image)

(5.1.3) Figure. Two cycles for the binary WUM.
Some Connections

In the figure on the left, \( I_0 \) is written in the L-shape; on the right \( I_1 \) is written. Note that the lower right-hand corner in the figure is missing because a transition \( S_0 = 0 \rightarrow S_1 = 1 \) is not possible in a 0-cycle. From Figure (5.1.3), we see that \( a_0 \) can be found from the figure on the left: \( \Pr\{S_1 = 0\} = a_1 \cdot \Pr\{I_0 = 0\} + (1 - a_1) \), so \( a_0 = a_1 p_0 + 1 - a_1 \). Because of the periodicity of the binary WUM, \( S_2 \) must have the same distribution as \( S_0 \). Hence \( a_1 = a_0 \cdot \Pr\{I_1 = 1\} + 1 - a_0 = a_0 p_1 + 1 - a_0 \).

From these two equations, we find

\[
a_1 = \frac{p_1}{1 - (1 - p_0)(1 - p_1)}, \quad a_0 = \frac{p_0}{1 - (1 - p_0)(1 - p_1)}.
\]

This proves that the expression for a rate point in the capacity region of the WUM is equal to expression (5.1.2) for a rate point in \( G_0 \).

If we compare the computations for the BMC and the WUM, it can also be made plausible why this rate point is achievable for the WUM, whereas it is only an outer bound for the rate of the BMC.

In the BMC situation, user \( i \) cannot obtain information about the other user's input if he sends \( I_i = 0 \). Indeed, this causes an 'erasure' for his output \( Y_i \). If both users send the 'erasure causing' symbol 0 simultaneously, neither user receives any information. Hence it is not surprising that the input pair \( (0, 0) \) should be avoided in optimum transmission. Clearly we cannot always avoid this pair in a strategy: if the two users are independent (which they are, initially), then the pair \( (0, 0) \) will occur with positive probability. Because of this, achievability of the rate in \( G_0 \) cannot be guaranteed.

For the WUM, something similar holds. If \( S_0 = 0 \), no information can be stored in the 0-cycle: an 'erasure' occurs in the output pair \( (S_0, S_1) \). Similarly, if \( S_1 = 1 \), we have an erasure in the output \( (S_2, S_1) \) of the 1-cycle. Note, however, that the erasures cannot occur in two consecutive cycles: if we have an erasure
Chapter 5

after the 0-cycle, then \( S_1 = 0 \), so a positive amount of information \( H(I_2) \) can be stored in the subsequent 1-cycle. Hence the 'double erasure' occurring for the BMC if \((0,0)\) is sent never occurs for the WUM, so the rate can be achieved here.

Another interesting similarity between the BMC and the WUM should be mentioned here. This involves coding for the binary case 3 WUM and for the BMC without feedback. (We will explain later what we mean by 'without feedback'.)

The best code known for the WUM in case 3, having error probability 0, achieves rate 0.52832 as mentioned in Section 4.1.3. Its parameters satisfy \( M_0 = M_1 = N = 3 \). The encoding functions for this code are the following (cf. [COH88]).

\[
\begin{align*}
    f_0(0) &= (0 0 \square), & f_0(1) &= (0 \square 0), & f_0(2) &= (\square 0 0), \\
    f_1(0) &= (1 1 \square), & f_1(1) &= (1 \square 1), & f_1(2) &= (\square 1 1).
\end{align*}
\]

The reader may check that the pair \((S_0, S_{k+1})\) always determines the filter \( I_k \) used. This code is symmetric (see Section 4.1.2); therefore we can translate the encoding functions \( f_0 \) and \( f_1 \) to one function \( f \), by replacing 0 by 1 and \( \square \) by 0 in the range of \( f_0 \) and replacing \( \square \) by 1 in the range of \( f_1 \). Now '1' means 'write a symbol in this position' and '0' means 'leave this position unchanged'. We find

\[ f: \{0,1,2\} \rightarrow \{0,1\}^3, \]

\[ f(0) = (1 1 0), \quad f(1) = (1 0 1), \quad f(2) = (0 1 1). \]

Now we will show that the same code can be used for the BMC without feedback. 'No feedback' means that the two users are not allowed to look at previous output symbols \( Y_0, Y_1, \ldots, Y_{n-1} \) before transmitting input symbol \( I_i, n \) \((i \in \{0,1\}, \ n \in \mathbb{N})\). Therefore a code for the BMC without feedback has just two
encoding functions, \( f_0 \) and \( f_1 \) (as opposed to the \( 2^N \) functions that are needed if feedback is allowed, see Definition (2.1.6)):

\[ f_i : \{0,1,2,\ldots,\mathcal{M}_i - 1\} \rightarrow \{0,1\}^N, \]

where \( \mathcal{M}_i \) is the number of messages for user \( i \) and \( N \) is the codeword length. The decoding functions are the same as in the case with feedback. It follows from [SHA61] that the capacity region of the BMC without feedback coincides with the Shannon inner bound region \( \mathcal{G}_i \). In fact, this holds for any TWC. (Also, compare [SAL88].) However, all codes that have been found up to now obtain rates not even close to the border of \( \mathcal{G}_i \).

The best code constructed up to this moment was found by Benschop [BEN90]. It has error probability 0, codeword length 3 and each user has three messages. Its encoding functions \( f_0 \) and \( f_1 \) are equal, say \( f = f_0 = f_1 \), where \( f \) is given by (5.1.4). It can be seen from Table (5.1.5) that both users are able to determine the other user's message by looking at their own message and the output sequence, \( Y \).

\[
\begin{array}{c|cccc}
\mathcal{I}_0 & \mathcal{I}_1 & 110 & 101 & 011 \\
\hline
110 & 110 & 100 & 010 \\
101 & 100 & 101 & 001 \\
011 & 010 & 001 & 011 \\
\end{array}
\]

(5.1.5) Table. Codewords and output for the BMC without feedback.

Hence we have the surprising fact that the code given by (5.1.4) seems to
be optimum both for the zero-error WUM in case 3 and for the zero-error BMC without feedback! Unfortunately, no other (nontrivial) codes for the case 3–WUM or for the BMC without feedback could be found, so it is not clear whether this implies any connection between codes for the two channels in general. This might be an interesting topic for future research. In the rest of this chapter, however, we will go further into the first similarity noticed: that of the BMC outer bound being equal to the WUM capacity region.

5.2. The q–ary WUM and generalizations of the BMC

5.2.1. The q–ary ‘two–way channel’

Now that we have noticed a connection between the binary WUM and the BMC, we can try to transfer this connection to WUMs over higher alphabets, as studied in Section 4.2, and a possible generalization of the BMC. We have seen that the rate $H(S_c | S_{c-1})$, obtained in the $(c-1)$–cycle of the binary WUM, can be linked to the optimum information rate over the BMC from user $c+1$ to user $c$, $H(Y_I | X_c)$. (Note that $c+1$ should be considered modulo 2) 4. If we want to generalize this to an alphabet $A = \{0,1,\ldots,q-1\}$, then we must find a link between the rate in the $(c-1)$–cycle of the $q$–ary WUM and the rate from user $c+1$ (modulo $q$) to user $c$ over some yet to define ‘generalized BMC’.

---

4 It may seem more logical to call this the information rate from user $c-1$ to user $c$. This does not make any difference in the binary case, but after generalization to a larger alphabet, it will turn out that this alternative does not represent the connection with the WUM as clearly as the other one. Since the only difference between the two possibilities is the direction of the information flow, this does not affect the theory in any way.
The above implies that such a generalized BMC, say $\mathcal{K}$, necessarily has $q$ users and that, for all $c \in \mathcal{A}$, user $c$ wants to know the message transmitted by user $c+1$ and at the same time he wants to transmit a message to user $c-1$. For this reason, we will call such a channel a $(q$--ary) ring channel.

Let user $c$'s input alphabet be $\mathcal{X}_c$, let his output alphabet be $\mathcal{Y}_c$. If user $c$'s input is $I_c$ and he receives output $Y_c$, then $I_c$ and $Y_c$ should tell him something about $I_{c+1}$. Hence we will assume that $Y_c$ depends on $I_c$ and $I_{c+1}$ only, in a deterministic way. Therefore, generalizing Definition (2.1.3), we assume the existence of $q$ channel functions, $f_{\mathcal{K},0}$, $f_{\mathcal{K},1}$, \ldots, $f_{\mathcal{K},q-1}$, with

$$f_{\mathcal{K},c} : \mathcal{X}_{c+1} \times \mathcal{X}_c \rightarrow \mathcal{Y}_c$$

and $f_{\mathcal{K},c}(I_{c+1}, I_c)$ is the output $Y_c$ for user $c$ if the input pair for user $c+1$ and $c$ is $(I_{c+1}, I_c)$. Now we can depict the ring channel $\mathcal{K}$ as in Figure (5.2.1.1).

(5.2.1.1) Figure. A ring channel $\mathcal{K}$ for $q$ users.

Note that we do not consider a specific channel with fixed functions $f_{\mathcal{K},c}$. This is because we do not know the best choice for these functions yet; the BMC can be generalized in many ways, as will be shown in the next sections. We could even consider a larger class of $q$--user channels, e.g. by allowing the output for each user to depend on all $q$ input symbols. However, this generalization does not seem to have the spirit of the $q$--ary WUM: the rate of a cycle in the $q$--ary WUM depends on two consecutive states and therefore it is more natural to look at a
Chapter 5

channel where only 'consecutive' (neighboring) users interact with each other.

Another reason for not specifying the channel yet is the fact that we have to derive an equivalent to the BMC outer bound for the ring channel, because we intend to show a link between the \( q \)-ary WUM capacity and the outer bound for the 'generalized BMC'. Since Shannon [SHA61] proved his inner and outer bound for TWCs in general, we will do the same and generalize Theorem (2.2.4) to \( q \)-ary ring channels in general. First we generalize the definitions of a code, error probability, achievability and capacity (cf. Section 2.1) to ring channels.

(5.2.1.2) Definition. Consider a \( q \)-ary ring channel \( \mathcal{X} \) as described above. A code for \( \mathcal{X} \) with parameters \( (N, W_0, W_1, \ldots, W_{q-1}) \) consists of \( q \cdot N \) encoding functions \( f_{c,n} \) \((c \in A, n \in \{0,1,\ldots,N-1\})\) and \( q \) decoding functions \( g_c \), \( c \in A \). Function \( f_{c,n} \):

\[
f_{c,n} : \{0,1,\ldots,N-1\} \times Y_c^n \rightarrow X_c,
\]

maps message \( W_c \) and received sequence \( Y_c^n \) for user \( c \) into his next channel input \( I_{c,n} \). The decoding function \( g_c \):

\[
g_c : \{0,1,\ldots,N_{c-1}\} \times Y_c^n \rightarrow \{0,1,\ldots,N_{c+1-1}\},
\]
defines how user \((c+1)\)'s message is estimated by user \( c \):

\[
\hat{W}_{c+1} = g_c(W_c, Y_c^n).
\]

(5.2.1.3) Definition. For a code with parameters \( N \) and \( W_c \), \( c \in A \), the rate tuple \( R = (R_0, R_1, \ldots, R_{q-1}) \) is defined by

\[
R_c := \log(W_c)/N, \quad c \in A.
\]

(5.2.1.4) Definition. A rate tuple \( R \) is achievable if for all \( \varepsilon > 0 \), a code with parameters \( N \) and \( W_c \) \((c \in A)\) exists, having error probabilities \( P_{e,c} \) defined as

\[
\Pr\{\hat{W}_c \neq W_c\} \text{ for all } c, \text{ such that }
\]

\[
\log(W_c)/N > R_c - \varepsilon \quad \text{and} \quad P_{e,c} < \varepsilon \quad \text{for all } c \in A.
\]
Now we can state the equivalent of Shannon's theorem for ring channels.

(5.2.1.5) Theorem. Let, for a given ring channel,
\[ G_i := \{ x \in \mathbb{R}^q \mid \forall c \ [0 \leq x_c \leq I(I_{c};Y_{c-1} | I_{c-1})] \} \]
for any input distribution
\[ P(x_0, x_1, \ldots, x_{q-1}) = \prod_c P(x_c) \]
and
\[ G_d := \{ x \in \mathbb{R}^q \mid \forall c \ [0 \leq x_c \leq I(I_{c};Y_{c-1} | I_{c-1})] \} \]
for any \( P(x_0, x_1, \ldots, x_{q-1}) \).

Then all points in \( G_i \) are achievable and points outside \( G_d \) are not achievable. \( \square \)

Proof: We start with the proof of the fact that all points in \( G_i \) are achievable. Let
\( \varepsilon > 0 \) and consider a product distribution over \((I_0, I_1, \ldots, I_{q-1})\), i.e.,
\[ P_{I_0, I_1, \ldots, I_{q-1}}(x_0, x_1, \ldots, x_{q-1}) = P_{I_0}(x_0) \cdot P_{I_1}(x_1) \cdots P_{I_{q-1}}(x_{q-1}). \]

Define \( r_c := I(I_{c};Y_{c-1} | I_{c-1}) \) for all \( c \). If we can prove that \((r_0, r_1, \ldots, r_{q-1})\) is an achievable rate, then, by time sharing, all points in \( G_i \) are achievable. Let \( N \in \mathbb{N} \) and define \( N_c := \lfloor 2^{N(r_c - \varepsilon)} \rfloor \) for all \( c \). We will consider an ensemble of codes, randomly constructed, each with parameters \((N, N_0, N_1, \ldots, N_{q-1})\). If we can show that at least one of these codes has \( P_{e,c} < \varepsilon \) for all \( c \), then this proves the achievability. In this proof we are going to use a well-known lemma on typicality, which can be found in any textbook on information theory. For the sake of completeness, we state this lemma (quoted from [ELG80]) below.

(5.2.1.6) Lemma. Consider a set of discrete random variables, \( z^{(1)}, z^{(2)}, \ldots, z^{(k)} \) with a probability distribution \( P(z^{(1)}, z^{(2)}, \ldots, z^{(k)}) \). Consider sequences of length \( N \) of these random variables, with distribution
\[ P(z^{(1)}_n, z^{(2)}_n, \ldots, z^{(k)}_n) = \prod_{n=0}^{N-1} P(z^{(1)}_n, z^{(2)}_n, \ldots, z^{(k)}_n). \]

For \( \delta > 0 \), define the \( \delta \)-typical set \( A_\delta(z^{(1)}, z^{(2)}, \ldots, z^{(k)}) \) by
Chapter 5

\[ A_\delta(z^{(1)}, z^{(2)}, \ldots, z^{(k)}) := \{ (z^{(1)}, z^{(2)}, \ldots, z^{(k)}) \mid \forall S \subseteq \{z^{(1)}, z^{(2)}, \ldots, z^{(k)}\} \left[ |\log P(\xi)/N - H(S)| < \delta \right] \} \]

and let \( A_\delta(S) \) denote the restriction of this set to the coordinates in \( S \). Then, if \( S_1 \) and \( S_2 \) are subsets of \( \{z^{(1)}, z^{(2)}, \ldots, z^{(k)}\} \), the following holds.

(i) \( P(A_\delta(S_1)) \geq 1 - \delta \)

(ii) \( |A_\delta(S_1)| = 2^N \cdot H(S_1) + o(N), \ N \to \infty \)

(iii) If \( \xi \in A_\delta(S_1): P(\xi) = 2^{-N(H(S_1) - \delta) + o(N)}, N \to \infty \)

(iv) If \( (\xi_1, \xi_2) \in A_\delta(S_1, S_2): P(\xi_1 | \xi_2) = 2^{-N(H(S_1 | S_2) - 2\delta) + o(N), N \to \infty.} \)

Now we define the code ensemble. Let \( \delta < \min\{\varepsilon/(2q), \varepsilon/8\} \). For \( \xi \in \mathcal{A} \) and \( w_c \in \{0, 1, \ldots, W_c - 1\} \), we randomly choose words \( I_c, w_c \in \mathcal{X}_c^N \) according to

\[ P_{I} (I_{c}, w_{c}) = \prod_{n=0}^{N-1} P_{I_c} (z_{c,n}). \]

We write this as \( P_{I_c} (I_{c}) \). Every code in the ensemble will correspond to a choice of these words, \( \{I_{c}, w_{c}\} \). For a fixed choice \( \{I_{c}, w_{c}\} \), the encoding function \( f_{c,n} \) maps message \( w_c \) and output sequence \( y_c^n \) to the input letter \( z_{c,w_n} \) (that is, the \( n \)th letter in \( z_{c,w_c} \)). Decoding function \( g_{c} \) maps message \( w_c \) and output \( y_c = y_c^n \) to the unique index \( \hat{w}_{c+1} \) for which \( (z_{c+1}, \hat{w}_{c+1}, z_c, w_c, y_c) \in A_\delta(I_{c+1}, I_c, Y_c) \) (see the definition in Lemma (5.2.1.6)). We abbreviate \( A_\delta(I_{c+1}, I_c, Y_c) \) by \( A_\delta^\prime \). If no such \( \hat{w}_{c+1} \) exists, or if more than one \( \hat{w}_{c+1} \) exists, then an error is declared.

Now that we have defined how the random code ensemble is constructed, we proceed with bounding the error probability averaged over all codes. We have

\[ \sum_{\text{code } \{\xi_c, w_c\}} P_r\{\text{code } = \{\xi_c, w_c\}\} \cdot \sum_{\xi_c \mid \hat{y}_c \neq Y_c} P_r\{\hat{y}_c \neq Y_c \mid \text{code } \{\xi_c, w_c\}\} \]
Some Connections

Here we assume, w.l.o.g., that \( w_c = 0 \) and \( w_{c-1} = 0 \). Let \( y := f_{X_{c-1}}(z, z') \). The above expression can be bounded by

\[
\sum_{c \in A} \sum_{z \in X_c} \sum_{z' \in X_{c-1}} \Pr[I_{c, c-1} = \{z, z', \hat{\omega}_c \neq 0\}] \cdot \{\phi((z, z', y) \notin \Lambda_\delta) + \\
\phi(\exists \hat{\omega}_c \neq 0 [(z_c, \hat{\omega}_c, z', y) \notin \Lambda_\delta]) \} 
\]

\[
\leq \sum_{c} \delta + \sum_{c} (N_c - 1) \sum_{z, \hat{z}, z', y} \sum_{z' \in X_{c-1}} \Pr[I_{c} (z) \cdot P_{X_c, c} (\hat{z}) \cdot P_{X_{c-1}} (z') \cdot \{\phi((z, z', y) \notin \Lambda_\delta)\}.
\]

The inequality in the first term follows from (i) of Lemma (5.2.1.6) and the inequality in the second term follows from the union bound. Because \( \delta < \epsilon/(2q) \), we can bound the above by

\[
\epsilon/2 + \sum_{c} N_c \sum_{z, \hat{z}, z', y} \sum_{z' \in X_{c-1}} \Pr[I_{c} (z) \cdot P_{X_c} (\hat{z}) \cdot P_{X_{c-1}} (z') \cdot \{\phi((z, z', y) \notin \Lambda_\delta)\}.
\]

\[
\leq \epsilon/2 + \sum_{c} N_c \|I_\delta\| \cdot 2^{-N} \left( H(X_c) + H(X_{c-1}) + H(Y_{c-1} | X_{c-1}) - 4\delta \right) + o(N)
\]

(1) \leq \epsilon/2 + \sum_{c} N_c \|I_\delta\| \cdot 2^{-N} \left( H(X_c) + H(X_{c-1}) + H(Y_{c-1} | X_{c-1}) - 4\delta \right) + o(N)

(2) \leq \epsilon/2 + \sum_{c} N_c \|I_\delta\| \cdot 2^{-N} \left( H(X_c) + H(X_{c-1}) + H(Y_{c-1} | X_{c-1}) - 4\delta \right) + o(N), \quad N \to \infty.

Here (1) follows from (iii) and (iv) of Lemma (5.2.1.6) and (2) follows from the definition of \( N_c \) and by (ii) of Lemma (5.2.1.6). Since \( I_c \) and \( I_{c-1} \) are independent in the distribution \( P_{X_0, X_1, \ldots, X_{q-1}} \), and \( I_c \) and \( I_{c-1} \) determine \( Y_{c-1} \), we have

\[
I(X_c; Y_{c-1} | X_{c-1}) + H(X_c, Y_{c-1}) - H(X_c) - H(X_{c-1}) = 0.
\]

\[= H(Y_{c-1} | X_{c-1}) + H(X_c | X_{c-1}) - H(X_c) - H(X_{c-1}) - H(Y_{c-1} | X_{c-1}) = 0.
\]
Therefore $\sum_c P_{e,c}$ averaged over all codes can be bounded by

$$\frac{\epsilon}{2} + \sum_c 2^N(-\epsilon + 4\delta) + o(N), \ N \to \infty$$

$$\leq \frac{\epsilon}{2} + \sum_c 2^{-N\epsilon}/2 + o(N), \ N \to \infty$$

$$< \epsilon \text{ if } N \text{ is sufficiently large.}$$

It follows that at least one code for the ring channel exists, having the desired rate, such that $\sum_c P_{e,c} < \epsilon$. This concludes the proof of the achievability of all rates in $G_i$.

Now we prove the converse. Consider an $(N, N_0, N_1, \ldots, N_{q-1})$ code with $P_{e,c} < \epsilon$ for all $c$. Firstly, for fixed $c$, we have Fano's inequality (see Section 4.2.4) stating that a function $\varphi(\cdot)$ exists such that $\varphi(\epsilon)/N \to 0$ if $\epsilon \to 0$ and

$$H(V_c | \hat{V}_c) < \varphi(\epsilon).$$

Secondly, for all $c$:

$$\log(N_c) = H(V_c) = H(V_c | V_{c-1})$$

since $V_c$ and $V_{c-1}$ are independent. Furthermore

$$(5.2.1.7) \ H(V_c | V_{c-1}) = H(V_c | V_{c-1}) - H(V_c | V_{c-1}, Y_c^N, V_{c-1}^N) + H(V_c | V_{c-1}, Y_c^N, V_{c-1}^N).$$

The third term on the right-hand side of (5.2.1.7) satisfies

$$H(V_c | V_{c-1}, Y_c^N, V_{c-1}^N) \leq H(V_c | V_{c-1}, Y_c^N, V_{c-1}^N)$$

$$\leq H(V_c | V_{c-1}, Y_c^N, V_{c-1}^N) + H(V_c | V_{c-1}, Y_c^N, V_{c-1}^N)$$

since $\hat{V}_c = g_{c-1}(V_{c-1}, Y_{c-1}^N)$. Because of Fano's inequality, this yields

$$H(V_c | V_{c-1}, Y_c^N, V_{c-1}^N) \leq \varphi(\epsilon).$$

The first two terms on the right-hand side of (5.2.1.7) amount to
Some Connections

\[ H(V_c | V_{c-1}) - H(V_c | V_{c-1}, Y^n_{c-1}) = I(V_c; Y^n_{c-1} | V_{c-1}) \]

\[ \leq H(Y^n_{c-1} | V_{c-1}) \]

\[ = \sum_n H(Y_{c-1, n} | V_{c-1}, Y^n_{c-1}) \{ \text{by the chain rule} \} \]

\[ \leq \sum_n H(Y_{c-1, n} | V_{c-1}, Y^n_{c-1}, Y^n_{c-1, n}) \]

\[ = \sum_n H(Y_{c-1, n} | V_{c-1}, Y^n_{c-1}, X_{c-1, n}) \]

because \( X_{c-1, n} = f_{c-1}(V_{c-1}, Y^n_{c-1}) \). We can bound this as

\[ \sum_n H(Y_{c-1, n} | V_{c-1}, Y^n_{c-1}, X_{c-1, n}) \]

\[ \leq \sum_n H(Y_{c-1, n} | X_{c-1, n}) \]

\[ = \sum_n H_c(n), \]

if we define \( H_c(n) \) as \( H(Y_{c-1, n} | X_{c-1, n}) \) (= \( I(Y_{c-1, n} | X_{c-1, n}) \)). It should be noted that \( H(n) \in G_0 \), where

\[ H(n) = (H_0(n), H_1(n), \ldots, H_{q-1}(n)). \]

We have shown

\[ \log(M_c) \leq \sum_n H_c(n) + \varphi(\varepsilon) \]

and therefore, the rate of the code is bounded (componentwise) as

\[ \left( \frac{\log(M_0)}{N}, \ldots, \frac{\log(M_{q-1})}{N} \right) \leq \sum_n H(n)/N + (\varphi(\varepsilon), \varphi(\varepsilon), \ldots, \varphi(\varepsilon))/N. \]

Now \( \sum_n H(n)/N \in G_0 \), since \( G_0 \) is convex, and \( (\varphi(\varepsilon), \varphi(\varepsilon), \ldots, \varphi(\varepsilon))/N \to \bar{Q} \) as \( \varepsilon \to 0 \).

Hence an achievable rate cannot be an interior point of the complement of \( G_0 \). \( \Box \)

In the following sections, we look at a few examples of ring channels. We apply Theorem (5.2.1.5) to determine the outer bound region for each channel and we investigate whether this region is equal to the \( q \)-ary WUM capacity region.
5.2.2. The $q$-ary BMC

Perhaps the most obvious generalization of the binary multiplying channel is the '$q$-ary BMC', defined as follows.

(5.2.2.1) Definition. Let $q \in \mathbb{N}$, $\mathcal{A}:= \{0, 1, \ldots, q-1\}$. The $q$-ary BMC is a $q$-ary ring channel $\mathcal{K}$ as described in Section 5.2.1, with

$$X_c = \{0, 1\}, \mathcal{X}_c = \{0, 1\} \text{ and } f_{X, c}(X_{c+1}, X_c) = I_{c+1} \cdot I_c \text{ for all } c \in \mathcal{A}. \quad \Box$$

Let $G_i$ and $G_o$ be the inner and outer bounds for this channel, respectively, as defined in Theorem (5.2.1.5). We find the following relations.

(5.2.2.2) $G_i = \max \{ \mathbb{R} \in \mathbb{R}^q \mid 0 \leq \mathbb{R} \leq a_{c-1} h(a_c), \text{ for } a_c \in [0, 1], \ c \in \mathcal{A} \}$.

(5.2.2.3) $G_o = \max \{ \mathbb{R} \in \mathbb{R} \mid 0 \leq \mathbb{R} \leq Pr\{I_{c-1}=1\} \cdot h(Pr\{I_c=1|I_{c-1}=1\}) \}$ for any distribution $P(x_0, x_1, \ldots, x_{q-1})$.

This follows from the fact that $h(I_c|I_{c-1}) = Pr\{I_{c-1}=1\} \cdot h(Pr\{I_c=1|I_{c-1}=1\})$ for the $q$-ary BMC. For the inner bound, this can be further simplified to $Pr\{I_{c-1}=1\} \cdot h(Pr\{I_c=1\})$ since $I_c$ and $I_{c-1}$ are independent in this case. Define

$$g_i := \max \{ \sum_c g_c/q \mid (g_0, g_1, \ldots, g_{q-1}) \in G_i \} \quad \text{and}$$

$$g_o := \max \{ \sum_c g_c/q \mid (g_0, g_1, \ldots, g_{q-1}) \in G_o \}.$$ 

For $q = 2$, we know that $g_i = 0.61695$ and $g_o = 0.69424$ (see Section 2.2). If $G_o$ equals the WUM capacity for case 1, 2 or 3, then we must have $g_o = 0.87915$ or $g_o = 0.87305$ for $q = 3$, according to Table (4.3.8). However, we find the following.

(5.2.2.4) Theorem. For $q = 3$, the $q$-ary BMC has $g_o = 0.69424$. \quad \Box

Proof: see below Lemma (5.2.2.6).
This shows that the \( q \)-ary BMC does not have the property we were looking for, namely that its outer bound coincides with the capacity region of the \( q \)-ary WUM. Nevertheless it is interesting to see that the outer bound for this channel does not increase with increasing \( q \), at least for \( q = 2 \) and \( q = 3 \). In fact, the author even proved that \( \ell_o = (3/4) \cdot h(2/3) = 0.68872 \) for \( q = 4 \), so the bound really seems to decrease in \( q \). We do not give the proof of the statement for \( q = 4 \), since it is similar to (but much longer than) the proof for \( q = 3 \).

For the inner bound, it is easy to see that \( \ell_i \geq 0.61695 \) for all \( q \) by taking \( a_c = 0.70350 \) for all \( c \) in (5.2.2.2). Computer search showed that \( \ell_i \) is equal to 0.61695 for \( q \in \{ 3, 4 \} \). For larger \( q \), there are too many parameters involved to be certain that the computer program found the global maximum over the region \( G_i \). The above observations lead us to the following conjecture.

\[(5.2.2.5) \text{Conjecture.} \text{ For all } q \geq 2, \text{ the } q\text{-ary BMC has} \]
\[\ell_i = 0.61695 \text{ and } \ell_o \leq 0.69424. \]

Now we give the proof of Theorem (5.2.2.4). We will use some properties of the entropy function in this proof that will also be applicable in Section 5.2.3. Therefore we will state these properties in a lemma.

\[(5.2.2.6) \text{Lemma.} \text{ Define the function } \rho \text{ by} \]
\[\rho: [0,1] \times [0,1] \to [0,1], \]
\[\rho(x,y) := (x+y) \cdot h \left( \frac{x}{x+y} \right) \text{ if } x+y > 0 \text{ and } \rho(0,0) := 0. \]

The following holds for \( \rho \).

\[(i) \quad \rho(x,y) \text{ is nondecreasing both in } x \text{ and in } y. \]

\[(ii) \quad \text{For all } x_1, x_2, y_1 \text{ and } y_2 \text{ in } [0,1]: \]
\[\rho \left( \left( x_1 + x_2 \right)/2, \left( y_1 + y_2 \right)/2 \right) \geq \rho(x_1, y_1)/2 + \rho(x_2, y_2)/2. \]
Chapter 5

Proof: (i) Since \( \rho(x,y) = -x \cdot \log(x) - y \cdot \log(y) + (x+y) \cdot \log(x+y) \), we have

\[
\frac{\partial \rho(x,y)}{\partial x} = \log\left(\frac{x+y}{x}\right) \geq 0 \text{ for } x > 0.
\]

Therefore \( \rho(x,y) \) is nondecreasing in \( x \). Because \( \rho(x,y) = \rho(y,x) \), it immediately follows that \( \rho(x,y) \) is nondecreasing in \( y \) as well.

(ii) For given \( x_1, x_2, y_1 \) and \( y_2 \) (not all 0), let \( \theta := \frac{x_1 + y_1}{x_1 + x_2 + y_1 + y_2} \). Then \( \theta \in [0,1] \). Since it is well-known that \( h(\cdot) \) is a convex function, we find

\[
h\left(\theta \cdot \frac{x_1}{x_1 + y_1} + (1-\theta) \cdot \frac{x_2}{x_2 + y_2}\right) \geq \theta \cdot h\left(\frac{x_1}{x_1 + y_1}\right) + (1-\theta) \cdot h\left(\frac{x_2}{x_2 + y_2}\right).
\]

Multiplying both sides of this inequality by \( \frac{x_1 + x_2 + y_1 + y_2}{2} \) yields

\[
\rho\left(\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2}\right) \geq \rho(x_1, y_1)/2 + \rho(x_2, y_2)/2.
\]

\[
\square
\]

Proof of Theorem (5.2.2.4): Consider a probability distribution \( P \) over \( I_0, I_1, I_2 \).

Let \( P(1\bullet) := P(100) + P(101) + P(110) + P(111) \), so \( P(1\bullet) = \Pr\{I_0 = 1\} \) for the given distribution. Similarly we define \( P(11\bullet) \) and other probabilities. Next, we define \( E(P) := (E_0(P) + E_1(P) + E_2(P))/3 \), with

\[
E_0(P) := P(**1) \cdot h\left(\frac{P(11\bullet)}{P(**1)}\right),
\]

\[
E_1(P) := P(1\bullet) \cdot h\left(\frac{P(11\bullet)}{P(1\bullet)}\right),
\]

\[
E_2(P) := P(1\bullet) \cdot h\left(\frac{P(11\bullet)}{P(1\bullet)}\right).
\]

It follows from the definition of \( C_o \) that \( C_o = \max \{E(P) \mid \text{distribution } P\} \). If \( P \) is a distribution for which this maximum is achieved, then it can be seen that distribution \( P^* \), defined by

\[
P^*(z_0, z_1, z_2) := P(z_2, z_0, z_1)
\]

also achieves the maximum (since \( E_0(P^*) = E_1(P) \), et cetera). Applying Lemma (5.2.2.6), (ii) with \( z_1 := P(1\bullet), y_1 := P(0\bullet), x_2 := P^*(1\bullet) \) and \( y_2 := P^*(0\bullet) \),

\[
-190-
\]
we find

\[ L_0((P+P^*)/2) \geq L_0(P)/2 + L_0(P^*)/2. \]

Since similar relations hold for \( L_1(\cdot) \) and \( L_2(\cdot) \), we obtain

\[ L((P+P^*)/2) \geq L(P)/2 + L(P^*)/2 = L(P). \]

Therefore, distribution \((P+P^*)/2\) also achieves the maximum. This argument shows that we may assume for the optimum distribution \( P \):

\[ P(z_0, z_1, z_2) = P(z_2, z_0, z_1) \text{ for all } (z_0, z_1, z_2) \in \{0,1\}^3. \]

Hence we have reduced the number of parameters \( P(z_0, z_1, z_2) \) from 8 to 4: we have \( P(000), P(001) \) (which is equal to \( P(010) \) and \( P(100) \)), \( P(011) \) (equal to \( P(101) \) and \( P(110) \)) and \( P(111) \). These parameters must satisfy

\[ P(000) + 3P(001) + 3P(011) + P(111) = 1. \]

Now \( L(P) \) can be written as

\[ L(P) = (P(001) + 2P(011) + P(111)) \cdot h\left(\frac{P(011) + P(111)}{P(001) + 2P(011) + P(111)}\right). \]

Clearly \( L(P) \) is maximum when \( P(000) = 0 \). If \( P(001) > 0 \) for the optimum distribution, then we can define the following distribution \( P^* \):

\[ P^*(z_0, z_1, z_2) := P(z_0, z_1, z_2) \text{ for } (z_0, z_1, z_2) \notin \{(001), (011)\}; \]
\[ P^*(001) := 0, \quad P^*(011) := P(011) + P(001). \]

Obviously \( P^* \) satisfies \( P^*(000) + 3P^*(001) + 3P^*(011) + P^*(111) = 1 \), and

\[ L(P^*) = (2P(001) + 2P(011) + P(111)) \cdot h\left(\frac{P(001) + P(011) + P(111)}{2P(001) + 2P(011) + P(111)}\right). \]

If we write \( z := P(011) + P(111) \) and \( y := P(001) + P(011) \), then the above yields (with \( \rho \) as defined in Lemma (5.2.2.6)):

\[ L(P) = \rho(z, y) \quad \text{and} \quad L(P^*) = \rho(z + P(001), y). \]

Because of (i) of Lemma (5.2.2.6), we find \( L(P^*) \geq L(P) \). Therefore we may
Chapter 5

assume \( P(001) = 0 \) in the optimum distribution \( P \). The function \( R(P) \) reduces to

\[
R(P) = (2P(011) + P(111)) \cdot h\left(\frac{P(011) + P(111)}{2P(011) + P(111)}\right)
\]

\[
= (1 - P(011)) \cdot h\left(\frac{P(011)}{1 - P(011)}\right).
\]

Here we used \( 3P(011) + P(111) = 1 \) and \( h(p) = h(1-p) \). Now it is easy to verify
(as in the binary case) that the maximum of \( R(P) \) is taken for \( P(011) = 0.27639, P(111) = 0.17082 \)
and that the maximum value is 0.69424. \( \square \)

5.2.3. A \( q \)-ary ring channel resembling the WUM

In the previous section we have seen that the \( q \)-ary BMC is not the
generalization of the BMC that we hoped to find. In fact, if we compare
expression (4.2.2.1) and expression (5.2.2.3), we could have expected that the
outer bound \( G_o \) of the \( q \)-ary BMC does not equal the capacity region of the WUM
in case 1 and 2. Since each component of a rate tuple in capacity region \( C_1 \) is the
sum of \( q-1 \) terms and each component of a point in \( G_o \) consists of just one term, it
is not surprising that the regions are equal for \( q = 2 \) but not for larger values of \( q \).

Let us try to find a ring channel \( K \) for which the outer bound expression
\( I(I_c;Y_{c-1}|I_{c-1}) \) (as found in Theorem (5.2.1.5)) can be written as the sum of
\( q-1 \) terms, like

\[
(5.2.3.1) \quad R_c = \sum_{i \in \mathcal{A}\{c\}} a_i^c \cdot h\left[\frac{a_{i+1}^c}{a_i^c}\right].
\]

For every deterministic ring channel, we have

\[
I(I_c;Y_{c-1}|I_{c-1}) = H(Y_{c-1}|I_{c-1})
\]

\[
 = \sum_{i \in \mathcal{A}_{c-1}} P_r(I_{c-1} = i) \cdot H(Y_{c-1}|I_{c-1} = i).
\]

Since the WUM rate point has \( R_c = H(S_{c+1}|S_c) \) (see the discussion in the
introduction to Section 5.2.1), we know that \( I_{c-1} \) should play the role of \( S_{c} \), the
state of the WUM before the $c$–cycle, and $Y_{c-1}$ should correspond to $S_{c+1}$, the state after the $c$–cycle. Therefore it seems promising to take $X_c = \Lambda$ for all $c$. Comparing the expression for $H(Y_{c-1} | I_{c-1})$ to (5.2.3.1), we see that the term $H(Y_{c-1} | I_{c-1} = i)$ with $i = c$ should not contribute to the entropy and that all other terms for $i \neq c$ should be of the form

$$P r\{I_{c-1} = i\} \cdot h\left(\frac{P r\{Y_{c-1} = j \text{ and } I_{c-1} = i\}}{P r\{I_{c-1} = i\}}\right),$$

where $Y_{c-1}$ can assume two values (conditional to $I_{c-1}$).

These considerations lead us to two possible suggestions for the channel functions $f_{K,c}$. We will call the corresponding ring channels $\mathcal{K}(1)$ and $\mathcal{K}(2)$, respectively.

(5.2.3.2) Definition. For $q \in \mathbb{N}$ and $\Lambda := \{0,1, \ldots, q-1\}$, we define the $q$–ary ring channel $\mathcal{K}(1)$ with alphabets $X_c$ and $Y_c$ and channel functions $f_{\mathcal{K}(1),c}$ by

$$X_c := \Lambda, \ Y_c := \{0,1\},$$

$$f_{\mathcal{K}(1),c}(I_{c+1},I_c) := \phi(I_c \neq c+1 \text{ and } I_{c+1} = c+1).$$

(5.2.3.3) Definition. For $q \in \mathbb{N}$ and $\Lambda := \{0,1, \ldots, q-1\}$, we define the $q$–ary ring channel $\mathcal{K}(2)$ with alphabets $X_c$ and $Y_c$ and channel functions $f_{\mathcal{K}(2),c}$ by

$$X_c := \Lambda, \ Y_c := \{0,1\},$$

$$f_{\mathcal{K}(2),c}(I_{c+1},I_c) := \phi(I_c \neq c+1 \text{ and } I_{c+1} = I_c).$$

It should be noted that, for $q = 2$, these two channels have the same partition pattern as the BMC (cf. Section 2.5) and therefore their inner and outer bounds coincide with the Shannon bounds for the BMC.
For each of these channels, we now derive the expression $H(Y_{c-1}|I_{c-1})$ occurring in the outer bound. Let $H(1)$ and $H(2)$ be the expressions for channels $K(1)$ and $K(2)$, respectively. We find

\[ H(1) = \sum_{i \neq c} Pr\{Y_{c-1} = i\} \cdot h(Pr\{Y_{c-1} = 1|I_{c-1} = i\}) \]
\[ = \sum_{i \neq c} Pr\{I_{c-1} = i\} \cdot h(Pr\{I_{c} = c|I_{c-1} = i\}), \]
\[ H(2) = \sum_{i \neq c} Pr\{I_{c-1} = i\} \cdot h(Pr\{I_{c} = 1|I_{c-1} = i\}) \]
\[ = \sum_{i \neq c} Pr\{I_{c-1} = i\} \cdot h(Pr\{I_{c} = i|I_{c-1} = i\}). \]

It is quite easy to show that the outer bound region of channel $K(1)$ cannot be equal to the WUM capacity region for values of $q$ larger than 2. This follows from the theorem below.

(5.2.3.4) Theorem. If we define $\mathcal{G}_i$ and $\mathcal{G}_o$ as the 'maximum average rates' over the inner and outer bound regions $\mathcal{G}_i$ and $\mathcal{G}_o$ for the channel $K(1)$, similar to the definitions in Section 5.2.2, then we have

$\mathcal{G}_i = \mathcal{G}_o = 1$ for all $q > 2$. \(\square\)

Proof: It follows from the expression for $H(1)$ that $\mathcal{G}_o \leq 1$, since this implies $R_c \leq Pr\{I_{c-1} \neq c\} \leq 1$. For a rate point in the inner bound, we have

$R_c = \sum_{i \neq c} Pr\{I_{c-1} = i\} \cdot h(Pr\{I_{c} = c\}).$

If we take $Pr\{I_{c} = c\} = 1/2$ and $Pr\{I_{c} = c+2 \ (mod\ q)\} = 1/2$, then it is easy to see that $R_c = 1$ for all $c$. Hence rate $(1,1,\ldots,1)$ is achievable for $q > 2$. \(\square\)

This theorem shows that channel $K(1)$ is a generalization of the BMC which has a quite surprising behavior. For $q = 2$, its capacity region is unknown, but for all values larger than 2, the channel is completely trivial: its capacity
region is equal to the unit hypercube in \( q \) dimensions. Furthermore, the rate \((1,1,\ldots,1)\) can be achieved using an error-free code with \( N = 2 \) and \( N_c = 2 \) for all \( c \). The latter follows from the proof of Theorem (5.2.3.4): if user \( c \) transmits \( c \) or \( c+2 \), depending on the message, then user \( c-1 \) can always tell the difference.

We now concentrate on channel \( \mathcal{K}(2) \) to see whether the outer bound for this channel is equal to the cases 1 and 2 – capacity region for the WUM. We define \( \mathcal{G}_i \) and \( \mathcal{G}_o \) as above, but this time for \( \mathcal{K}(2) \). Computer optimization yielded the results listed in Table (5.2.3.5). For comparison, this table also lists the maximum average rate \( c_{1,av} \) for the case 1–WUM, as defined in (4.3.1). These values are the same as in Table (4.3.8).

<table>
<thead>
<tr>
<th>( q )</th>
<th>( \mathcal{G}_i )</th>
<th>( \mathcal{G}_o )</th>
<th>( c_{1,av} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.61695</td>
<td>0.69424</td>
<td>0.69424</td>
</tr>
<tr>
<td>3</td>
<td>0.73731</td>
<td>0.87915</td>
<td>0.87915</td>
</tr>
<tr>
<td>4</td>
<td>0.80298</td>
<td>0.94678</td>
<td>0.94678</td>
</tr>
</tbody>
</table>

(5.2.3.5) Table. Values of \( \mathcal{G}_i \), \( \mathcal{G}_o \) and \( c_{1,av} \) for \( q = 2, 3 \) and 4.

Table (5.2.3.5) suggests that the outer bound \( \mathcal{G}_o \) for \( \mathcal{K}(2) \) is equal to the WUM capacity region \( C_1 \), at least for the maximum average rate point and for \( q \leq 4 \). As in Section 5.2.2, computer search could not be performed (i.e., the results were unreliable) for larger values of \( q \) because of the large number of parameters. Indeed, for \( q = 4 \), the probability distribution \( P(x_0, x_1, x_2, x_3) \) already consists of \( 4^4 = 256 \) parameters! Computer search was possible only because this number of variables could be reduced to 57, by applying some techniques that are described below. For \( q \geq 5 \), not even this reduction made computations feasible.
Chapter 5

Now we will show how the number of parameters in the probability distribution can be reduced when searching for an optimum rate point. Let \( q \geq 2 \) and define \( P(i\cdots\cdot i) := \Pr\{X_0 = i\} \), etc., analogous to the definition in the proof of Theorem (5.2.2.4). Now the rate point in \( G_o \) corresponding to this distribution, say \( \mathbb{R}(P) = (\mathbb{R}_0(P), \mathbb{R}_1(P), \ldots, \mathbb{R}_{q-1}(P)) \), can be written as

\[
(5.2.3.6) \quad \mathbb{R}_c(P) = \sum_{i \neq c} P(\cdots i \cdots \cdot i \cdots \cdot) h(P(\cdots i \cdots \cdot i \cdots \cdot)) / P(\cdots i \cdots \cdot i \cdots \cdot)
\]

where the \( i \) in \( P(\cdots i \cdots \cdot i \cdots \cdot) \) is in coordinate position \( c-1 \) and the two \( i \)’s in \( P(\cdots i \cdots \cdots) \) are in positions \( c-1 \) and \( c \).

First of all, if we want to find the maximum average rate \( \theta_o \), we can apply exactly the same reasoning as used in the proof of Theorem (5.2.2.4) to show that the optimum \( P(\cdot) \) satisfies

\[
P(x_0, x_1, x_2, \ldots, x_{q-1}) = P(x_{q-1} + 1, x_0 + 1, x_1 + 1, \ldots, x_{q-2} + 1)
\]

for all \((x_0, x_1, x_2, \ldots, x_{q-1}) \in \Lambda^q\). For \( q = 4 \), this reduced the number of parameters from 256 to 70. Further reduction to 57 parameters is possible by Theorem (5.2.3.7) below, stating that, for an optimum rate point \( \mathbb{R}(P) \) in \( G_o \), we may assume

\[
P(\cdots c, c+1, \cdots) = 0 \quad \text{if the } c+1 \text{ is in coordinate position } c.
\]

(5.2.3.7) Theorem. Let \( P \) be a distribution such that \( \mathbb{R}(P) \in G_o \). Then there is a distribution \( P^* \) such that \( \mathbb{R}(P^*) \geq \mathbb{R}(P) \) (componentwise inequality) and

\[
P^*(\cdots c, c+1, \cdots) = 0
\]

if the \( c+1 \) is in coordinate position \( c \).

Proof: Let \( P \) be a distribution with \( P(1\cdots\cdots0) > 0 \). We will define a distribution \( P^* \) for which \( \mathbb{R}(P^*) \geq \mathbb{R}(P) \) with strict inequality and \( P^*(1\cdots\cdots0) = 0 \). Let, for all \((z_0, z_1, \cdots, z_{q-1}) \in \Lambda^q\),
Some Connections

\( P^*(x_0, x_1, \cdots, x_{q-1}) := P(x_0, x_1, \cdots, x_{q-1}) \) if \((x_0, x_{q-1}) \notin \{(1,0), (0,0)\}, \)

\( P^*(1, x_1, \cdots, x_{q-2}, 0) := 0, \)

\( P^*(0, x_1, \cdots, x_{q-2}, 0) := P(0, x_1, \cdots, x_{q-2}, 0) + P(1, x_1, \cdots, x_{q-2}, 0). \)

Clearly, \( P^* \) is a probability distribution. It satisfies

\[ P^*(\cdots \ast i \cdots) = P(\cdots \ast i \cdots) \quad \text{if the } i \text{ is not in position 0} \]

and

\[ P^*(\cdots \ast i \cdots) = P(\cdots \ast i \ast i \cdots) \]

if the pair of \( i \)'s are not in positions \((0,1)\) or \((q-1,0)\). By (5.2.3.6), this yields

\[ k_c(P^*) = k_c(P) \quad \text{for } c \notin \{0,1\}. \]

For \( c = 0 \), we find

\[ k_0(P^*) = \sum_{i \neq 0} P^*(\cdots \ast i \cdots) h(P^*(\cdots \ast i \cdots). \]

It is easy to check that

\[ P^*(\cdots \ast i \cdots) = P(\cdots \ast i \cdots) \quad \text{and} \quad P^*(i \ast \cdots i) = P(i \ast \cdots i) \]

for \( i \neq 0 \). Therefore \( k_0(P^*) = k_0(P) \). Finally,

\[ k_1(P^*) = \sum_{i \neq 1} P^*(i \ast \cdots i) h(P^*(i \ast \cdots i)). \]

We have

\[ P^*(i \ast \cdots i) = P(i \ast \cdots) \quad \text{and} \quad P^*(i i \ast \cdots i) = P(i i \ast \cdots) \quad \text{for } i \neq 0, i \neq 1, \]

\[ P^*(0 \ast \cdots) = P(0 \ast \cdots) + P(1 \ast \cdots 0), \]

\[ P^*(00 \ast \cdots) = P(00 \ast \cdots) + P(10 \ast \cdots 0). \]

It follows that

\[ k_1(P^*) = \sum_{i \neq 1} P^*(i \ast \cdots i) h(P^*(i \ast \cdots i)) = \sum_{i \neq 0,1} P(i \ast \cdots i) h(P(i i \ast \cdots) + P^*(i \ast \cdots i)). \]

\[ + (P(0 \ast \cdots) + P(1 \ast \cdots 0)) h(P(00 \ast \cdots) + P(10 \ast \cdots 0)). \]
Chapter 5

Define

\[ x := P(0\cdots0), \quad \Delta x := P(1\cdots0) - P(0\cdots0), \]
\[ y := P(0\cdots0) - P(0\cdots0), \quad \Delta y := P(1\cdots0) - P(0\cdots0). \]

Then, by Lemma (5.2.2.6):

\[
(P(0\cdots0) + P(1\cdots0)) h \left( \frac{P(0\cdots0) + P(1\cdots0)}{P(0\cdots0) + P(1\cdots0)} \right) = \rho(x+\Delta x, y+\Delta y)
\]
\[
> \rho(x, y) = P(0\cdots0) h \left( \frac{P(0\cdots0)}{P(0\cdots0)} \right).
\]

Therefore

\[
\mathcal{K}_1(P^*) \geq \sum_{i \neq 1} P(i\cdots) h \left( \frac{P(i\cdots)}{P(i\cdots)} \right) = \mathcal{K}_1(P).
\]

This proves that \( \mathcal{K}(P^*) \geq \mathcal{K}(P) \) with strict inequality. If \( P^* \) is such that \( P^*(\cdots c, c+1, \cdots) > 0 \) for some \( c \neq 0 \), then the above argument can be repeated to change \( P^* \) to a new distribution \( P^{**} \) having \( P^{**}(\cdots c, c+1, \cdots) = 0 \) and \( \mathcal{K}(P^{**}) \geq \mathcal{K}(P^*) \), etcetera. This proves the theorem. \( \Box \)

Theorem (5.2.3.7) is one more indication that \( \mathcal{G}_\varnothing \) and \( \mathcal{C}_1 \) may be equal for all \( q \), because of the following. Let \( \mathcal{K} \in \mathcal{G}_\varnothing \) correspond to a distribution for which

(5.2.3.8) \( P(\cdots) = 0 \) if \( j \neq c, i \neq j \) and the \( j \) is in position \( c \).

Hence, for this distribution, \( X_c = j \) implies \( X_{c-1} = j \) if \( j \neq c \). Now define

\[ a_i^c := Pr \{ X_{c-1} = i \} \text{ for } i \in \mathcal{A}, \ c \in \mathcal{A}. \]

Then, because of property (5.2.3.8), we have for \( i \neq c \):

\[ Pr \{ X_{c-1} = i \text{ and } X_c = i \} = Pr \{ X_c = i \} = a_i^{c+1}. \]

Consequently,

\[ K_c = \sum_{i \in \mathcal{A} \setminus \{c\}} a_i^c \cdot h(a_i^{c+1}/a_i^c), \]

\( - 198 - \)
Some Connections

\[ \sum_{i \in A} a_i^c = 1 \text{ for all } c \text{ and } a_i^c \geq a_i^{c+1} \text{ for } i \neq c. \]

This shows that \( K \in C_1 \), by Definition (4.2.2.1). Therefore, if (5.2.3.8) would hold for all points in \( G_o \), we would have \( G_o \subset C_1 \). Now Theorem (5.2.3.7) states that, at least for \( i = c \) and \( j = c+1 \), (5.2.3.8) holds for all 'good' rate points in \( G_o \).

Summarizing, we have found that the optimum points in \( G_o \) have some of the properties of rate points in \( C_1 \). Because of this, we conjecture that the outer bound region of \( K(2) \) and the case 1-capacity region of the WUM are indeed equal and hence, that \( K(2) \) is the \( q \)-ary ring channel we were looking for. We have not found a proof for this statement yet, but computer search appears to confirm the conjecture.

Our search for a generalization of the BMC with the aforementioned property has lead us to some new examples of multi-user channels. Our study of \( q \)-ary ring channels, as defined in this chapter, may be a step towards a more general theory of Shannon-like 'q-way channels'.
List of Symbols

The following notational conventions are used in this thesis.

Sets are denoted with script capitals: \( \mathcal{A}, \mathcal{C}, \mathcal{G}, \mathcal{F}, \mathcal{R}, \mathcal{V} \). A bar over such a symbol denotes the closure of the set: \( \bar{\mathcal{F}}, \bar{\mathcal{V}} \).

Constants are denoted with italic capitals: \( \mu, \eta, \xi, \epsilon \), and indices are denoted with italic minuscules: \( c, i, j, k, n \).

Random variables and their realizations are denoted with italic capitals and the corresponding lower case letters, respectively: \( \xi, \omega, \xi, \xi, \eta, \xi, \xi \).

A vector of length \( N \) is denoted as \( \mathbf{x}^N \) or \( (x_0, x_1, \ldots, x_{N-1}) \) or \( (x(0), x(1), \ldots, x(N-1)) \).

If \( N = 0 \), this notation represents the empty sequence.

\( \mathbb{N}, \mathbb{R} \) and \( \mathbb{C} \) denote the sets of natural, real and complex numbers, respectively.

Here \( \mathbb{N} := \{1, 2, \ldots\} \).

Let \( \mathcal{I} \) and \( \mathcal{Y} \) be random variables over finite sets \( \mathcal{I} \) and \( \mathcal{Y} \) with distribution \( P_{\mathcal{I} \mathcal{Y}} \) and marginals \( P_{\mathcal{I}} \) and \( P_{\mathcal{Y}} \). We denote the entropy of \( \mathcal{I} \) as \( H(\mathcal{I}) \):

\[
H(\mathcal{I}) := \sum_{z \in \mathcal{I}} -P_{\mathcal{I}}(z) \log(P_{\mathcal{I}}(z)).
\]

The average mutual information between \( \mathcal{I} \) and \( \mathcal{Y} \) is denoted as \( I(\mathcal{I}; \mathcal{Y}) \), with

\[
I(\mathcal{I}; \mathcal{Y}) := \sum_{(z, y) \in \mathcal{I} \times \mathcal{Y}} P_{\mathcal{I} \mathcal{Y}}(z, y) \log \left( \frac{P_{\mathcal{I} \mathcal{Y}}(z, y)}{P_{\mathcal{I}}(z)P_{\mathcal{Y}}(y)} \right).
\]

In these expressions, the base of the logarithm is 2.

The following is a list of the most important symbols used in this thesis. The page number refers to the page where the symbol is introduced. Symbols not occurring in this list are defined in the text at the point where they are used.
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\diamond$</td>
<td>45, 116</td>
</tr>
<tr>
<td>$\circ$</td>
<td>117</td>
</tr>
<tr>
<td>$\vdash$</td>
<td>136</td>
</tr>
<tr>
<td>$a$</td>
<td>167</td>
</tr>
<tr>
<td>$a^c_i$</td>
<td>127, 140, 164</td>
</tr>
<tr>
<td>$a^c_{ij}$</td>
<td>165</td>
</tr>
<tr>
<td>$c_i^c(n)$</td>
<td>135, 146</td>
</tr>
<tr>
<td>$\beta_c^k(n)$</td>
<td>134</td>
</tr>
<tr>
<td>$\lambda^c_i$</td>
<td>168</td>
</tr>
<tr>
<td>$\zeta_0$</td>
<td>153</td>
</tr>
<tr>
<td>$\eta_n$</td>
<td>74</td>
</tr>
<tr>
<td>$\theta_i$, $\Theta_i$</td>
<td>24</td>
</tr>
<tr>
<td>$\lambda_c$</td>
<td>169</td>
</tr>
<tr>
<td>$\Pi(\cdot, \cdot)$</td>
<td>164</td>
</tr>
<tr>
<td>$\rho(\cdot, \cdot)$</td>
<td>189</td>
</tr>
<tr>
<td>$\sigma_i(\cdot)$</td>
<td>24</td>
</tr>
<tr>
<td>$\Sigma$</td>
<td>168</td>
</tr>
<tr>
<td>$\phi(\cdot)$</td>
<td>21</td>
</tr>
<tr>
<td>$\psi(\cdot)$</td>
<td>136</td>
</tr>
<tr>
<td>$a_t$</td>
<td>58</td>
</tr>
<tr>
<td>$A_\delta(\cdot)$</td>
<td>183</td>
</tr>
<tr>
<td>$A$</td>
<td>7, 29, 35, 125</td>
</tr>
<tr>
<td>$A_{n,i}$</td>
<td>36</td>
</tr>
<tr>
<td>$b(x)$</td>
<td>111</td>
</tr>
<tr>
<td>$b(M)$</td>
<td>98</td>
</tr>
<tr>
<td>$b_t$</td>
<td>76</td>
</tr>
<tr>
<td>$\bar{B}$</td>
<td>57</td>
</tr>
<tr>
<td>$B(M)$</td>
<td>67</td>
</tr>
<tr>
<td>$B, B(M)$</td>
<td>68</td>
</tr>
<tr>
<td>$co$</td>
<td>12</td>
</tr>
<tr>
<td>$C_i$</td>
<td>122</td>
</tr>
<tr>
<td>$C_i, av$</td>
<td>152</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi_i(\cdot)$</td>
<td>138, 147</td>
</tr>
<tr>
<td>$\phi^c_k(n)$</td>
<td>75</td>
</tr>
<tr>
<td>$C_i$</td>
<td>121, 127</td>
</tr>
<tr>
<td>$C, C_K$</td>
<td>11</td>
</tr>
<tr>
<td>$d(\cdot, \cdot)$</td>
<td>14</td>
</tr>
<tr>
<td>$\theta$</td>
<td>76</td>
</tr>
<tr>
<td>$f(z)$</td>
<td>153, 178</td>
</tr>
<tr>
<td>$f(n)$, $f^0(n)$,...</td>
<td>98</td>
</tr>
<tr>
<td>$f^c(\cdot)$</td>
<td>118, 120, 126</td>
</tr>
<tr>
<td>$f_{i,n}(\cdot)$</td>
<td>10, 182</td>
</tr>
<tr>
<td>$f^c_K(\cdot, \cdot)$</td>
<td>8, 19</td>
</tr>
<tr>
<td>$F(n)$</td>
<td>42</td>
</tr>
<tr>
<td>$F^0(n)$</td>
<td>45</td>
</tr>
<tr>
<td>$F^0, F^0(n)$</td>
<td>49</td>
</tr>
<tr>
<td>$F, F(n)$</td>
<td>42</td>
</tr>
<tr>
<td>$F^0, F^0(n)$</td>
<td>45</td>
</tr>
<tr>
<td>$g_c(\cdot)$</td>
<td>118, 120, 126, 169</td>
</tr>
<tr>
<td>$g_{i,i}(\cdot, \cdot)$</td>
<td>10, 182</td>
</tr>
<tr>
<td>$h_i, h_o$</td>
<td>188</td>
</tr>
<tr>
<td>$g_i$</td>
<td>12, 188</td>
</tr>
<tr>
<td>$g_i(N)$</td>
<td>13</td>
</tr>
<tr>
<td>$h(\cdot)$</td>
<td>15</td>
</tr>
<tr>
<td>$H(\cdot), H(\cdot, \cdot)$</td>
<td>12</td>
</tr>
<tr>
<td>$\bar{H}$</td>
<td>57</td>
</tr>
<tr>
<td>$I(\cdot; \cdot)$</td>
<td>12</td>
</tr>
<tr>
<td>$I_i(\cdot)$</td>
<td>24</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>6, 181</td>
</tr>
<tr>
<td>$k(N)$</td>
<td>13</td>
</tr>
<tr>
<td>$l_t$</td>
<td>58</td>
</tr>
<tr>
<td>$\mu_0(c), \mu_1(c)$</td>
<td>137, 147</td>
</tr>
<tr>
<td>$\mu(A)$</td>
<td>79</td>
</tr>
</tbody>
</table>
# Subject Index

**Achievable**, 11, 68, 121, 127, 182  
alphabet, 7, 33, 70, 125  
input, 6, 181  
output, 6, 181  
alternating, 119  
area, 21  

**Basic interval**, 24, 25, 70  
basic rectangle, 24, 48  
Benschop, 179  
Berger, 17, 19  
binary multiplying channel, 8, 174  
block length, 10, 125  
BMC, 8, 14, 98-109  
generalized, 180  
q-ary, 188-191  
bootstrapping, 16, 64-66, 81  
Borden, 123, 128  
breadth, 57  

**Capacity**, 1  
average, 152, 172  
region, 2, 11, 31, 95, 121, 127  
zero-error, 97-98  
Čebyšev’s inequality, 167  
cell, 116, 125  
channel, 1  
function, 8, 181, 193  
restricted, 18  
T-, 8, 17, 37  
ternary, 18, 30, 110  
closure, 42  
code, 10, 46, 182  
fixed length, 60, 95  
variable length, 62  
zero-error, 48, 60, 99  
codeword, 142  
length, 10, 62, 96  
Cohen, 124  
compression, 70, 79-80  
connected regions, 28, 38  
construction, 75, 103, 123  
converse, 124, 133, 145, 159, 186  
convex, 11, 12, 86, 145, 187  
copies of tree, 44, 56, 72, 91  
correlated source coding, 69  
cybercycle, 115, 125  

**Decoder**, 9, 117-118  
decoding function, 10, 118, 126, 182  
dependence balance bound, 17  
depth, 42  
derived channel, 13, 22  
deterministic, 8  
distance, 14, 42  
Dueck, 16  

**Encoder**, 9, 117-118  
coding function, 10, 118, 126, 182  
Engels, 33  
entropy, 12, 50  
equivalence, 32, 129  
erasure, 177-178  
error probability, 10, 83, 121, 140, 182  
eventually error-free, 130  
exponential convergence, 57  
Fano, 136, 146, 159, 186  
fate of a cell, 156-158  
feedback, 7, 178-179  
filter, 116, 125  
Gaal, 32  
generalized defects, 155  
grapical representation, 3, 19, 21  
grid, 47, 60, 92  

**Hagelbarger code**, 64, 103  
Han, 16  
Hazewindus, 68  
height, 57  
Hekstra, 17  
hole, 116  

**Information theory**, 1  
inner bound, 12, 19  
input, 7, 24, 181  
interval, 19  
invisible vertex, 66  
Jacobs, 32, 110  
Jelinek, 16  
jointly typical (j.t.), 140, 148  

**Kuznetsov**, 125, 155  

**Labeling**, 129-131  
Lagrange multiplier method, 169  
limiting region, 14, 88  
Lormans, 110  
L-shape, 54, 104, 177

---

- 203 -
References


References


References


– 207 –


Samenvatting

Dit proefschrift behandelt twee onderwerpen uit de informatietheorie, te weten tweewegkanalen (TWC's) en schrijf-eenrichtings-geheugens (WUM's).

Een TWC modelleert communicatie in twee richtingen: twee gebruikers willen elkaar gelijktijdig informatie sturen. Het kanaal dat zij gebruiken produceert voor elke gebruiker een (discrete) output die afhankt van beide inputs. Het kanaal is geheugenloos, maar elke gebruiker mag zijn volgende input afhangen van eerdere inputs en outputs. We onderzoeken hoe efficiënt dit kanaal gebruikt kan worden, d.w.z. we zoeken het capaciteitsgebied.

In het proefschrift brengen we enige 'loose' resultaten uit het verleden samen onder één theorie. We gebruiken de zogenaamde 'grafische representatie', die tweeweg-communicatie herleidt tot partitionering van het eenheidsvierkant. Dit was reeds bekend voor een voorbeeld van een TWC, het binaire 'and' kanaal (BMC), maar deze methode blijkt bruikbaar voor een grote klasse van kanalen. Hiermee is het mogelijk goede ondergrenzen aan capaciteitsgebieden af te leiden.

Een WUM is een model van een herschrijfbare optische plaat. Op zo'n plaat staat binaire informatie en in een schrijfCyclus kunnen symbolen slechts in één richting veranderd worden: ofwel nullen kunnen in enen veranderen (over de hele plaat; we noemen dit een 1-Cyclus), of enen kunnen in nullen veranderen (een 0-Cyclus). We nemen aan dat 0- en 1-cycli elkaar afwisselen. Ook hier vragen we naar de capaciteit: we willen zo veel mogelijk informatie opslaan op een plaat.

We onderscheiden vier gevallen, afhankelijk van de vraag of de schrijver en/of de lezer de toestand van de plaat kent voor het schrijven. Voor elk van deze vier gevallen bepalen we de capaciteit. Voor een aantal van deze gevallen was de capaciteit al bekend; in dit proefschrift generaliseren we deze resultaten naar een algemener type WUM, namelijk de WUM over een alfabet met q symbolen. We hebben nu q verschillende (cyclisch doorlopen) schrijfCyclus. Ook hiervoor bepalen we het capaciteitsgebied, in drie van de vier gevallen. Voor geval 4 (schrijver noch lezer kent de oude toestand) geven we een binnengrens waarvan we vermoeden dat hij gelijk is aan de capaciteit.

Tenslotte leggen we een verband tussen TWC's en WUM's. Omdat het capaciteitsgebied voor twee gevallen van de binaire WUM gelijk is aan een fundamentele buitengrens aan de capaciteit van het BMC, zoeken we een generalisatie van dit fenomeen over een willekeurig alfabet. Dit geeft aanleiding tot gegeneraliseerde TWC's voor q gebruikers, waarvoor we enige eigenschappen afleiden.
Curriculum Vitae

Ineke van Overveld was born on June 25, 1962 in Dongen, The Netherlands. After graduating from the John F. Kennedy Atheneum in Dongen, in 1980, she studied mathematics at the Eindhoven University of Technology. She received her M.Sc. degree (cum laude) in February 1986.

During the period March 1986 – September 1990 she has been with the Electrical Engineering Department of the Eindhoven University of Technology as a research assistant. In October 1990 she joined the Institute for Perception Research in Eindhoven, where she is currently working on processing and perception of images.
Stellingen

bij het proefschrift

"On the Capacity Region for
Deterministic Two-Way Channels and
Write-Unidirectional Memories"

W. M. C. J. van Overveld

Een $d$-dimensionale product code met geschikte afmetingen, $n_1, n_2, \ldots, n_d$, waarbij de codesymbolen cyclisch gerangschikt zijn, kan niet alleen bursts ter lengte $n_1$ corrigeren, zoals aangetoond door Bahl en Chien, maar is tevens in staat bursts ter lengte $n_1 \cdot n_2 \cdot \ldots \cdot n_s$ te corrigeren voor $s \leq d-2$. Bovendien kan er een eenvoudig decodeeralgoritme gegeven worden voor het geval dat $d = 3$.


Bij vaste–lengte strategieën voor een tweewegkanaal neemt zowel het aantal parameters in de inputverdeling als het aantal locale maxima van de ratefunctie exponentieel toe in de bloklengte. Om deze reden zijn numerieke optimaliseringsmethoden in de praktijk niet bruikbaar bij het zoeken naar een globaal maximum.

Dit proefschrift, §3.6.

Voor WUM codes met alfabetgrootte $q$ voor geval 2 zoals gedefinieerd in dit proefschrift, maar waarbij de volgorde van de cycli door de encoder beslist mag worden (zoals voor $q = 2$ is beschreven door Borden), is de maximale rate voor kleine bloklengte aanzienlijk groter dan bij het cyclisch doorlopen van de schrijfcycli. Als $l(N)$ de maximale rate is bij bloklengte $N$, dan geldt:

$$l(1) \geq \log(q)/N, \quad l(2) \geq \log(2q-1)/N, \quad l(3) \geq \log(4q-3)/N, \quad l(4) \geq \log(6q-5)/N.$$ 

Voor $q = 2$ treedt hierin gelijkheid op.


Dit proefschrift, hoofdstuk 4.

Beschouw de volgende generalisatie van Tiersma's vermoeden:

"Voor $k \in \mathbb{N}$ bevat elke 0/1 matrix met $4k$ enen een submatrix met $2k$ enen". Als de matrix minstens $2k$ niet–lege rijen bevat, dan is deze bewering waar en is de submatrix te vormen door een aantal rijen uit de matrix te schrappen. In zijn algemeenheid echter is de vraag of het schrappen van rijen kan leiden tot de gewenste submatrix een NP–volledig probleem.


Laat $M \in \mathbb{N}$, $N \in \mathbb{N}$ en $a \in [0,1/2]$ met $M \leq \left\lfloor \frac{(1-a)N}{a} \right\rfloor$. De constructie van bepaalde binaire symmetrische WUM codes voor geval 2 (cf. Simonyi) met rate $\log(M)/N$ is equivalent met de constructie van een "goede $M$-kleuring" (cf. Berge) van de hypergraaf $(X,E)$ met 

$$X := \{x \in \{0,1\}^N | \text{wt}(x_i) = aN\}, \quad E := \{e_i \mid i = 1,2,\ldots,\left\lfloor \frac{N}{a} \right\rfloor\},$$

en $e_i := \{x_j \in X | (x_i,x_j) = aN\}$ waarbij $\overline{x_i}$ het complement van $x_i$ is.


Het door verscheidene auteurs opgemerkte verband tussen de afname in (informa-tietheoretische) entropie van een systeem en de hiermee gepaard gaande energie-dissipatie (d.w.z. de toename in thermodynamische entropie) is te beschouwen als een omzetting van "verwarring" in "verwarming".


Het alfabet dat "Cyrillisch" genoemd wordt, draagt deze naam niet terecht omdat het een duidelijk andere oorsprong heeft dan het door Cyrillus ontworpen schrift.


Het aantal malen dat een vrouwelijke promovendus aan een technische universiteit bij vergissing voor vakgroepsecretaresse aangewezen wordt is wellicht te verminderen door meer mannen als secretaresse in dienst te nemen.