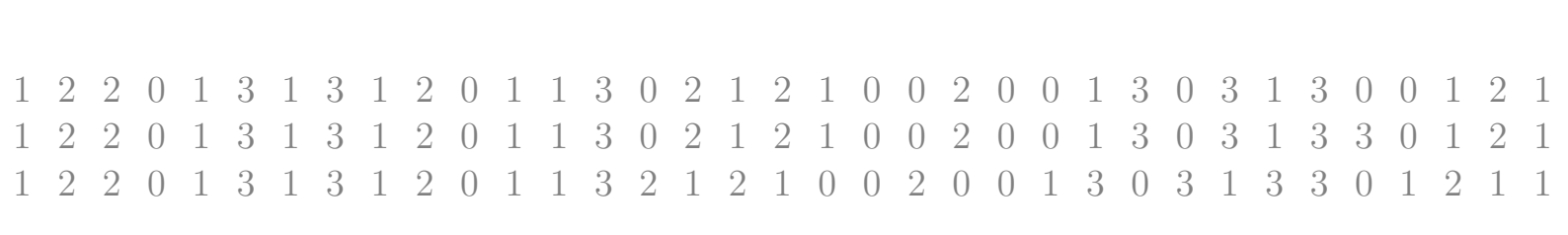
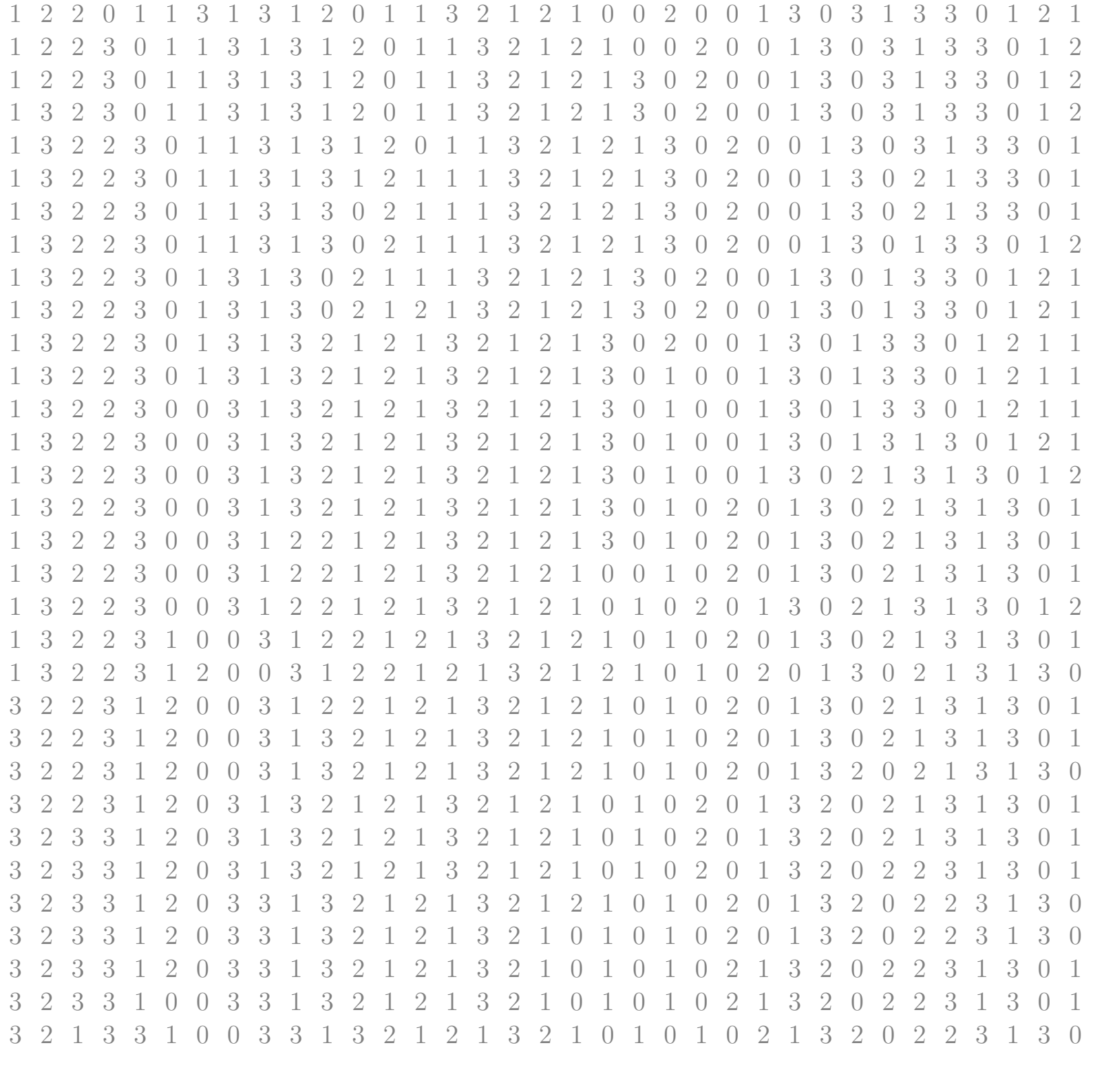


Bounds on the maximum size of deletion, insertion and substitution correcting codes

by

Ward Spee



Delft University of Technology

Master Thesis

Bounds on the maximum size of deletion, insertion and substitution correcting codes

by

W.J.P. (Ward) Spee

Thesis Committee:

Dr. ir. J.H. (Jos) Weber
Dr. R.J. (Robbert) Fokkink
Dr. J.A.M. (Joost) de Groot

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Section of Discrete Mathematics and Optimization of
Delft Institute of Applied Mathematics

Delft, The Netherlands,

May, 2023

Abstract

Coding techniques for correcting combinations of deletion, insertion and substitution errors are implemented in novel data storage applications. Error correcting codes have been well-studied for either substitution errors, or deletion and insertion (indel) errors, but the understanding of codes that correct combinations of these errors falls short. To achieve an efficient implementation in terms of redundancy, we are interested in the maximal size of a code that can correct t indels and s substitutions. Determining this maximal size in general is a difficult task, and thus in many instances we have to rely on bounds. In this thesis, we review existing bounds on the maximal size of both t-indel correcting codes and s-substitution correcting codes. Thereafter, we study how these bounds can be generalized to the setting of t-indel s-substitution correcting codes.

The main contributions of this thesis include two new explicit lower bounds, which are based on generalizations of the Gilbert-Varshamov bound. Furthermore, we show that the Singleton upper bound, several existing sphere-packing upper bounds and an upper bound based on matchings in hypergraphs can all be generalized to upper bounds on the maximal size of t-indel s-substitution correcting codes. Several of these bounds provide improvements upon existing results in literature. Moreover, we argue that the asymptotic redundancy of maximally sized t-indel s-substitution correcting codes lies between $(t+s)\log_q(n)$ and $2(t+s)\log_q(n) + o(\log_q(n))$.

Samenvatting

Tijdens het versturen of langdurig opslaan van digitale berichten bestaat de kans dat er fouten optreden in het bericht. Deze fouten zorgen er voor dat het ontvangen bericht niet identiek is aan het verstuurde bericht. In deze scriptie, ligt de focus op een drietal type fouten waarbij enkele symbolen in een bericht worden aangetast. Allereerst, kan een symbool wegvallen (deletion). Daarnaast kan een symbool veranderen naar een ander symbool (substitution). Ten derde, kan een symbool worden ontvangen terwijl het niet is verstuurd (insertion). Bijvoorbeeld, het woord 'raden' kan zo in drie stappen veranderen naar:

$$\text{raden} \to \text{rade} \to \text{rede} \to \text{trede}.$$

Om dergelijke fouten te kunnen verbeteren worden op geavanceerde wijze extra symbolen toegevoegd aan het bericht en mee verstuurd of opgeslagen. Door het analyseren van zowel het bericht als deze extra symbolen is de ontvanger meestal in staat om de fouten te verbeteren. Het toevoegen van deze extra symbolen zorgt voor redundantie omdat deze symbolen moeten worden verstuurd, maar geen informatie bevatten. Daarom is het wenselijk dat het aantal extra symbolen zo klein mogelijk is. Dit leidt tot het volgende vraagstuk: voor een gegeven verzameling met berichten, wat is het minimaal aantal extra symbolen dat kan worden toegevoegd aan deze berichten zodat het altijd mogelijk is om een bepaald aantal fouten te verbeteren? In veel gevallen is het alleen mogelijk om grenzen te bepalen voor het minimaal aantal extra benodigde symbolen. In andere woorden, dit aantal ligt tussen ondergrens x en bovengrens y.

In het geval dat er maar één type fout kan optreden dan is dit vraagstuk veelvuldig bestudeerd en zijn er meerdere grenzen bekend in de literatuur. Daarentegen is er minder bekend over de grenzen als er combinaties van het drietal fouten kunnen optreden. In deze scriptie worden bestaande grenzen voor enkele typen fouten geanalyseerd. Op basis daarvan wordt onderzocht hoe deze grenzen kunnen worden gegeneraliseerd of gecombineerd tot grenzen voor de combinatie van deze drie fouten. In het bijzonder worden meerdere nieuwe onder- en bovengrenzen aangetoond. In sommige gevallen vormen deze grenzen een verbetering ten opzichte van bestaande grenzen in de literatuur.

Preface

This thesis forms the final part of my master program in Applied Mathematics at Delft, University of Technology. I was first introduced to the topic of coding theory in a pre-university course called *Junior TU-Delft*, which sparked my interest in this field of mathematics. During my studies in Delft, the interest only increased after I followed courses and wrote my bachelor thesis in this field. For this reason, I approached Jos Weber to write a master thesis in coding theory as well.

The project was originally aimed towards DNA data storage, and constrained coding challenges arising from this application. After a preliminary literature review I noticed that little was stated in literature about the maximal cardinality of codes that correct combinations of deletion, insertion and substitution errors. Moreover, I realized that it is possible to generalize a well-known existing lower bound on maximum size of substitution correcting codes to this more general setting. Therefore, in consultation with Jos Weber, I changed the direction of the project and focused on bounds on the maximal size of codes that correct a combination of deletion, insertion and substitution errors.

In hindsight, this change in direction of the project has been a good choice, because it enabled me to tackle and solve certain problems that had not been solved before. This made the project both challenging and rewarding. Discussing these problems with my supervisor Jos Weber during the bi-weekly meetings and with Khaled Abdel-Ghaffar and Ludo Tolhuizen via (e-mail) contact, has been helpful and lead to many new ideas. Furthermore, it was interesting and informative to attend the *Workshop on Combinatorics in Digital Communication* in Eindhoven, and the *Symposium on Information Theory and Signal Processing in the Benelux* in Brussels. At the symposium in Brussels, I presented a poster about a key result from this thesis. Furthermore, I submitted a conference paper co-authored by Jos Weber.

Finally, this thesis project has shown me that I enjoy working on challenging mathematical problems, and discussing these problems in an academic setting. This is something I would like to pursue in my future career.

Acknowledgments

First and foremost, I would like to thank my supervisor Jos Weber for his guidance during the entire thesis project. During the bi-weekly meetings we discussed the progress, he gave constructive feedback on my work and provided helpful advice. I appreciated the freedom he gave me to determine my own path within the project. Therefore, I was able to choose the research topics that I found interesting to investigate. All in all, I greatly enjoyed working together.

During the research process Khaled Abdel-Ghaffar and Ludo Tolhuizen provided helpful and interesting comments on the contents of this thesis, for which I am very grateful as well. Moreover, I thank Robbert Fokkink and Joost de Groot for taking the time and effort to be part of the thesis committee.

Lists of terminology and notation

Term	Meaning
Alphabet	Finite set of symbols, i.e., $\mathcal{B}_q$.
Symbol	Element in $\mathcal{B}_q$.
Word	Element in $\mathcal{B}_q(n)$, i.e., a string/sequence/vector of symbols.
Empty word	Unique word of length zero denoted by Λ.
Run	Sequence of consecutive and identical symbols in a word that is not contained within a longer such sequence.
Code	Subset of $\mathcal{B}_q(n)$.
Codeword	Element of a code.
Hamming distance	The number of positions in which two words of equal length differ.
Redundancy	Loosely, the number of redundant symbols in an arbitrary codeword, i.e., $n - \log_q(\lvert\mathcal{C}\rvert)$.
Relative redundancy	Redundancy relative to the codeword length, i.e., $1 - \log_q(\lvert\mathcal{C}\rvert)/n$.
Deletion	Removal of a single symbol from a word.
Insertion	Addition of a single symbol into a word.
Substitution	Replacement of a single symbol by different symbol in a word.
Indel	Insertion or deletion.
Edit	Deletion, insertion or substitution.

Notation			Meaning
$\mathbb{Z}$	$:=$	$\bigcup_{i=0}^{\infty}\{-i, i\}$	Set of integers.
$\mathbb{Z}_{\geq a}$	$:=$	$\bigcup_{i=a}^{\infty}\{i\}$	Set of integers at least a.
S^n	$:=$	$S \times S \times \cdots \times S$	n times Cartesian product of the set S.
$\lvert S\rvert$	$\in$	$\mathbb{Z}_{\geq 0}$	Cardinality of a finite set S.
$\binom{a}{b}$	$:=$	$\frac{a!}{b!(a-b)!}$	Binomial coefficient. For $a < 0$, $b < 0$ or $b > n$, we use the convention that $\binom{a}{b} = 0$.
n	$\in$	$\mathbb{Z}_{\geq 1}$	Word length.
q	$\in$	$\mathbb{Z}_{\geq 2}$	Alphabet size.
t'	$\in$	$\{0, 1, ..., n\}$	Number of deletions.
t''	$\in$	$\mathbb{Z}_{\geq 0}$	Number of insertions.

Continued on the next page

t	$\in$	$\mathbb{Z}_{\geq 0}$	Number of indels.
s	$\in$	$\{0, 1, ..., n\}$	Number of substitutions.
τ	$\in$	$(0, 1)$	Number of indels relative to n.
σ	$\in$	$(0, 1)$	Number of substitutions relative to n.
r	$\in$	$\{1, 2, ..., n\}$	Number of runs.
d	$\in$	$\mathbb{Z}_{\geq 0}$	Minimum Hamming distance.
$\mathcal{B}_q$	$:=$	$\{0, 1, ..., q-1\}$	q-ary alphabet.
$\mathcal{B}_q(n)$	$:=$	$\{0, 1, ..., q-1\}^n$	Set of q-ary words of length n.
$\mathcal{B}_q^*$	$:=$	$\bigcup_{n=0}^{\infty}\{0, 1, ..., q-1\}^n$	Set of q-ary words of arbitrary length.
$\mathbf{x}, \mathbf{y}, \mathbf{z}$	$\in$	$\mathcal{B}_q(n)$	Generic q-ary words of length n.
Λ	$\in$	$\mathcal{B}_q(0)$	Empty word, the unique q-ary word of length 0.
$\mathbf{u}_{n,r}^1$	$\in$	$\mathcal{B}_2(n)$	Binary word with r runs, a first run of zeros with length $n-r+1$ and the remaining unit runs alternate between 0 and 1, e.g., $\mathbf{u}_{7,5}^1 = 0001010$.
$\mathcal{C}$	$\subseteq$	$\mathcal{B}_q(n)$	(Block) code, a set of fixed-length q-ary words.
$\mathbf{c}$	$\in$	$\mathcal{C}$	Codeword, element of a code.
$d(\mathbf{x}, \mathbf{y})$	$\in$	{0,1, ..., n}	Hamming distance between $\mathbf{x}, \mathbf{y} \in \mathcal{B}_q(n)$, i.e., the number of symbols in which $\mathbf{x}$ and $\mathbf{y}$ differ.
$r(\mathbf{x})$	$\in$	$\{1, ..., n\}$	Number of runs in $\mathbf{x} \in \mathcal{B}_q(n)$.
$\mathcal{R}_{n,q}(r)$	$:=$	$\{\mathbf{x} \in \mathcal{B}_q(n)$: $r(\mathbf{x}) = r\}$	Set of q-ary words of length n with precisely r runs.
$M_q(n, t, s)$	$:=$	$\max\{\lvert\mathcal{C}\rvert : \mathcal{C} \subseteq \mathcal{B}_q(n)$ s.t. $\mathcal{C}$ is a t-indel s-substitution correcting code$\}$	The maximum cardinality of a t-indel s-substitution correcting code.
$\mathcal{V}_{t',t'',s}(\mathbf{x})$	$\subseteq$	$\mathcal{B}_q(n-t'+t'')$	Set of words that can be obtained from $\mathbf{x} \in \mathcal{B}_q(n)$ by precisely t' deletions, t'' insertions and at most s substitutions.
$\mathcal{D}_t(\mathbf{x})$	$\subseteq$	$\mathcal{B}_q(n-t)$	Set of words that can be obtained from $\mathbf{x} \in \mathcal{B}_q(n)$ by precisely t deletions.
$\mathcal{I}_t(\mathbf{x})$	$\subseteq$	$\mathcal{B}_q(n+t)$	Set of words that can be obtained from $\mathbf{x} \in \mathcal{B}_q(n)$ by precisely t insertions.
$\mathcal{S}_s(\mathbf{x})$	$\subseteq$	$\mathcal{B}_q(n)$	Set of words that can be obtained from $\mathbf{x} \in \mathcal{B}_q(n)$ by at most s substitutions.
$H_q(p)$	$:=$	$p\log_q(q-1) - p\log_q(p) - (1-p)\log_q(1-p)$	The q-ary entropy function, defined for $p \in [0, 1-\frac{1}{q}]$ with $H_q(0) := 0$.
$H_q^*(p)$	$:=$	$H_q(\min\{x, 1-\frac{1}{q}\})$	Extended q-ary entropy function, for $p \in [0, \infty)$.
$R_q^+(\tau, \sigma)$	$:=$	$\limsup_{n\to\infty}\left(1-\frac{1}{n}\cdot \log_q(M_q(n, \lfloor\tau n\rfloor, \lfloor\sigma n\rfloor))\right)$	Superior asymptotic relative redundancy function.
$R_q^-(\tau, \sigma)$	$:=$	$\liminf_{n\to\infty}\left(1-\frac{1}{n}\cdot \log_q(M_q(n, \lfloor\tau n\rfloor, \lfloor\sigma n\rfloor))\right)$	Inferior asymptotic relative redundancy function.
$\mathcal{H}$	$:=$	$(V, \mathcal{E})$	Hypergraph with set of vertices V and set of hyperedges $\mathcal{E}$.
$\mathcal{M}$	$\subseteq$	$\mathcal{E}$	Matching, i.e., pairwise disjoint set of hyperedges.
$\nu(\mathcal{H})$	$\in$	$\mathbb{Z}_{\geq 0}$	Matching number, maximum size of a matching in $\mathcal{H}$.

Contents

1
Introduction

Digital communication and data storage have become omnipresent in today's society. Sending messages via e-mail, listening to music on wireless earbuds and storing photos on an SD-card are all examples of applications in which information is communicated from one place to another. Unfortunately, most communication and storage media are subject to noise and the message inevitably incurs occasional errors [1]. This results in (partial) loss or alterations of the message. The receiver of the message does not know whether the message contains errors. Therefore, the sender must employ error correction techniques so that the receiver is able to detect and even correct potential errors. Without such techniques, reliable communication and data storage are not possible.

A message is commonly expressed as a sequence of symbols. A well-known example is the ASCII scheme [2] in which sequences of eight bits are used to represent letters, numerals and other characters. In this thesis, we focus on the correction of errors that affect single symbols in a sequence. In particular, we consider symbols being substituted (changed) for different symbols, symbols being deleted (lost) and inserted (gained). For instance, suppose that the binary sequence 00110101 is sent. Due to noise, an insertion followed by a substitution and a deletion, may turn this sequence into

$$00110101 \to 001110101 \to 101110101 \to 10111010,$$

before the receiver obtains the message. It is important to state that the number of errors, error types and error positions within the message are all not known to the receiver. The receiver only knows the final sequence 10111010. It is the goal of the sender and receiver to come up with an efficient technique that allows the receiver to correct 10111010 to the original message 00110101.

Error-correction techniques have a long-standing interest because of their application in digital communication and data storage. 75 years ago Shannon [1] initiated research on this topic and laid the foundations for coding theory and in particular error correcting codes. Loosely speaking, error correcting codes cleverly add extra symbols to a message. These additional symbols enable the receiver to correct up to certain number of errors. Examples that show how error correcting codes can be constructed for different types of errors are given in Sections 1.2 & 1.6. The capability of correcting errors in a message comes at the cost of losing efficiency: the extra symbols are redundant, but should be sent as well. Obviously, we aim to use as little redundant symbols as possible in order

to improve the efficiency, while still being able to correct the errors. This raises the following question. Given a set of messages and a number t, what is the least number of redundant symbols that can be added to each message so that it is always possible to correct up to t errors in a message? Answering this question is a central problem in coding theory.

For substitution errors, this problem was first addressed around the 1950's by Hamming [3], Gilbert [4], Varshamov [5] and others. Their contributions focused mainly on substitution errors, because this type of error is common in practice. Since then, great progress has been made towards solving this problem for substitution errors [6]. However, it has become clear that providing a general and exact answer for this problem is challenging. For this reason, the exact answer is not known in many instances and we have to rely on lower and upper bounds.

For deletion and insertion errors, this problem has also been studied extensively (e.g., [7–10]), but not as much is known in comparison to substitution errors. In 1965, Levenshtein applied Shannon's framework in a seminal paper on codes that are able to correct combinations of deletion and insertion errors [7]. He showed an interesting property that a code which can correct any t deletions can also correct any t insertions, and vice versa. In other words, correcting a deletion can be done with the same code as correcting an insertion. We will provide a more precise statement and proof of this property in Chapter 2. This property shows that there is no reason to consider deletions and insertions separately. Therefore, they are often jointly called indels: an insertion or a deletion. Indels are particularly interesting because other types of errors can be formed by combining multiple indels. For instance, every $a \to b$ substitution can be obtained by two indels: a deletion of a followed by an insertion of b in the same position.

In this thesis, we consider the central problem in coding theory for codes that correct a combination of indels and substitutions. In the remainder of this introduction, we will motivate why the study of these codes is both theoretically and practically relevant. Thereafter, we will provide a more mathematically formal setting for indel and substitution correcting codes and the central problem in coding theory. This will allow us to formulate the research question of this thesis.

1.1 Motivation for studying indel and substitution correcting codes

The motivation for studying codes that correct combinations of indels and substitutions is twofold. Below, we briefly elaborate on these two points.

Firstly, the study of the central problem in coding theory for indel and substitution correcting codes is driven by the limited theoretical understanding of this problem. To the best of the author's knowledge, there are only several results within literature that address this problem for the combination of indels and substitutions. In [11], this is done by constructing non-trivial bounds on the size of indel and substitution correcting codes. Furthermore, Sima *et al.* constructed a class of such codes in [12] and showed how it can used to correct a combination of indels and substitutions. Later, Song *et*

al. provided improved constructions in terms of redundancy in [13] and [14]. These codes are constructed for a general number of indels and substitutions that need to be corrected. Other codes presented in [11, 15, 16] can correct only a fixed number of errors. This limited number of bounds on the size of codes and code constructions stands in pale contrast with the numerous results for both indel correcting codes and substitution correcting codes.

Secondly, novel data storage media require techniques that can correct indel and substitution errors for successful recovery of the stored information. An example of these novel techniques is given by DNA data storage. Inspired by the way in which DNA molecules store the biological information of organisms, researchers have successfully managed to artificially create DNA molecules and stored an entire book, eleven JPEG images and a JavaScript program [17] in these molecules. The inherent properties of DNA such as its long-life, non-volatility and unparalleled information density are advantageous for storing data in DNA [18]. Significant progress in the research and usability of DNA data storage systems over the last decade (see e.g., [19–24]) indicates that it may offer a viable alternative for traditional storage media in the future [25]. However, analysis shows that the DNA molecules occasionally incur combinations of deletion, insertion or substitution errors while stored [26]. Consequently, codes that can correct these errors efficiently are vital for this new technology to succeed. Concepts related to indel and substitution correcting codes are also used in language processing applications such as spell-checking and translation [27–29] and in another data storage application known as racetrack memory [30–32].

1.2 Example of indel and substitution correction

The following example introduces indel and substitution correction for DNA data storage.

Example 1.1. Suppose Alice wants to write a letter to Bob and store this letter in DNA so that Bob can read it in the far future. A DNA molecule consists of a long string of small units, called nucleobases. For each nucleobase there are four options: Adenine [A], Cytosine [C], Guanine [G] and Thymine [T]. In this way, a DNA molecule can be represented by a long sequence letters from the set $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$.

For writing the letter to Bob, Alice counts that she needs at most 64 different characters: 26 lowercase letters, 26 uppercase letters and 12 punctuation symbols. Then, she converts each character into a DNA sequence of length three as follows:

Character		DNA sequence
a	$\longleftrightarrow$	AAA,
b	$\longleftrightarrow$	AAC,
c	$\longleftrightarrow$	AAG,
	$\vdots$	
!	$\longleftrightarrow$	TTG,
.	$\longleftrightarrow$	TTT.

Notice that there exist $4^3 = 64$ DNA sequences of length three and thus each character can be converted. After Alice has written the letter, she converts each individual charac-

ter into a separate DNA sequence of length three. Each DNA sequence is then separately converted into a physical DNA molecule, and safely stored for a long time. Bob recovers these DNA molecules and reads back the DNA sequences. However, during this process some sequences may have undergone unwanted changes. For example, a stored sequence `CGA` might have turned into `CTA` (a substitution), into `CG` (a deletion) or into `CGGA` (an insertion). For simplicity, we assume that at most one error occurs within a sequence.

Bob does not know which DNA sequences Alice wrote, and has to rely fully on what he receives by reading the stored DNA molecules. The possibility of errors in the storage process means that Bob cannot be sure if the sequence that he recovers is identical to the sequence that was stored by Alice. Only if the length of a recovered sequence is not equal to three, he can be sure that an indel occurred. Nevertheless, if Bob recovers `CGGA` he has no way of knowing whether it originally was `CGG`, `CGA` or `GGA`. This makes it impossible for Bob to convert the DNA sequences back into characters and be sure that he recovers the original letter in a correct way.

In order to avoid this issue, Alice decides to convert each character into a DNA sequence of length nine. Alice uses the same conversion as before, but repeats each DNA sequence three times. For instance, this gives a conversion: $c \longleftrightarrow$ `AAGAAGAAG`. Again, Alice stores these sequences in DNA and they are later read back by Bob. Moreover, we assume that at most one error occurs per sequence, and that Bob is aware that Alice used this repetition procedure. We will argue that this enables Bob to correct any single error.

Namely, Bob knows that the original sequence must have been of the form `XYZXYZXYZ` for some $\mathtt{X}, \mathtt{Y}, \mathtt{Z} \in \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$. By counting the length of the recovered sequence Bob can identify whether a deletion (length 8), a substitution (length 9), or an insertion (length 10) occurred. In these cases, it is not hard to uniquely recover the original sequence. For instance, suppose Bob recovers the sequence `TCGTCGCG` of length 8, then he determines that a deletion occurred. Furthermore, the symbols 1 to 3 are identical to the symbols 4 to 6. By the assumption that at most one error occurred per sequence, Bob finds that the original sequence must have been `TCGTCGTCG`. The process of correcting an insertion is similar.

In case the recovered sequence has length 9, then either a substitution occurred or no error at all. Whenever this word is of the form `XYZXYZXYZ` then Bob concludes that no error has happened. Otherwise, he compares the symbols 1 to 3, 4 to 6 and 7 to 9. One of these three sub-sequences will differ from the other two as a result of the substitution. By the assumption that at most one error occurred per sequence, this allows Bob to also correct the substitution.

All in all, the repetition procedure makes sure that Bob can correct up to one error per sequence. Then, Bob can simply convert these corrected sequences back to the characters and read the original letter.

1.3 Mathematical description of indel and substitution correcting codes

Next, we introduce indel and substitution correcting codes in a mathematical setting. Furthermore, we discuss other terminology and notation that will be used in this thesis.

Consider the alphabet $\mathcal{B}_q := \{0, 1, ..., q-1\}$ consisting of $q \geq 2$ elements, called symbols. A sequence or string of these symbols is called a word. The length of a word is denoted by n and the size of the alphabet by q. The set of q-ary words of length n is denoted by $\mathcal{B}_q(n) := \{0, 1, ..., q-1\}^n$. By convention, $\mathcal{B}_q(0)$ consists of a unique word called the empty word and which is denoted by Λ. Furthermore, $\mathcal{B}_q^* := \cup_{n=0}^{\infty} \mathcal{B}_q(n)$ gives the collection of all q-ary words of arbitrary length[1]. For a finite set S we denote by $|S|$ the cardinality or size of S. For instance, it holds that $|\mathcal{B}_q(n)| = q^n$. For $\mathbf{x} \in \mathcal{B}_q(n)$ we write $\mathbf{x} = (x_1, x_2 \ldots, x_n)$ or shortly $\mathbf{x} = x_1 x_2 \cdots x_n$ for ease of notation. For two words $\mathbf{x}, \mathbf{y} \in \mathcal{B}_q(n)$, the number of positions in which $\mathbf{x}$ and $\mathbf{y}$ differ is called the Hamming distance, and it is denoted by $d(\mathbf{x}, \mathbf{y}) := |\{i \in \{1, ...n\} : x_i \neq y_i\}|$.

Example 1.2. The word $\mathbf{x} = (2, 3, 1, 0, 0, 1) = 231001$ is an element of the set $\mathcal{B}_4(6)$. Moreover, the set $\mathcal{B}_2(3)$ contains $2^3 = 8$ words and is given by

$$\mathcal{B}_2(3) = \{000, 001, 010, 011, 100, 101, 110, 111\}.$$

The Hamming distance between $00120, 31122 \in \mathcal{B}_4(5)$ is given by $d(00120, 31122) = 3$.

A run in a word $\mathbf{x} \in \mathcal{B}_q(n)$ is a sequence of consecutive and identical symbols from $\mathbf{x}$ that is not contained within a longer such sequence. The number of runs in $\mathbf{x}$ is denoted by $r(\mathbf{x})$. For $1 \leq r \leq n$, we denote the set of words with precisely r runs by

$$\mathcal{R}_{n,q}(r) := \{\mathbf{x} \in \mathcal{B}_q(n) : r(\mathbf{x}) = r\}.$$

The cardinality of $\mathcal{R}_{n,q}(r)$, i.e., the number of words in $\mathcal{B}_q(n)$ with r runs will be discussed in Section 2.2.

Example 1.3. Consider the word $\mathbf{y} = 02331 \in \mathcal{B}_4(5)$. Then, $\mathbf{y}$ contains the runs 0, 2, 33, 1 and satisfies $r(\mathbf{y}) = 4$. Remark that 3 is not considered to be a run in $\mathbf{y}$, because it is contained within the run 33.

A non-empty subset of words from $\mathcal{B}_q(n)$ is called a (block) code[2] and the elements in a code are called codewords. It follows that the size of any code is trivially bounded between 1 and q^n. Only the codewords of a code $\mathcal{C} \subseteq \mathcal{B}_q(n)$ are used to communicate information. In contrast, the words in $\mathcal{B}_q(n) \setminus \mathcal{C}$ are not used and are thus redundant. Hence, considering the ratio between $|\mathcal{C}|$ and $|\mathcal{B}_q(n) \setminus \mathcal{C}|$ gives an indication of the information capacity versus the redundancy of a code. Instead of considering $|\mathcal{C}|$ and $|\mathcal{B}_q(n)|$ directly, it is common to consider their ratio on a logarithmic scale. The (information) rate of a code is given by $\frac{1}{n}\log_q(|\mathcal{C}|)$. On the other hand, the relative redundancy of $\mathcal{C}$ is given by $1 - \frac{1}{n}\log_q(|\mathcal{C}|)$. Notice that both the information rate and the relative redundancy are bounded between 0 and 1, and that they add to 1.

[1]Notice that $\mathcal{B}_q$ and $\mathcal{B}_q(n)$ are not defined as a finite field and vector space, respectively, which is common in coding theory. This additional structure is not needed in this thesis.

[2]the prefix 'block' refers to the fact all codewords have equal length. In this thesis, we solely consider block codes and therefore we will simply use 'code', for brevity.

Example 1.4. Let the code $\mathcal{C} \subset \mathcal{B}_3(6)$ of size 9 with codewords of length 6 be given by

$$\mathcal{C} = \{000000, 000111, 000222, 111000, 111111, 111222, 222000, 222111, 222222\}.$$

Note that knowing the first and fourth symbol in each codeword is sufficient to fully characterize a codeword, because the other symbols are solely copies of these two symbols. In other words, the first and fourth symbol carry information, while the other four symbols are redundant. Intuitively, this corresponds well the information rate of $\mathcal{C}$ being equal to $\frac{1}{n}\log_q(|C|) = \frac{1}{6}\log_3(9) = \frac{1}{3}$, while the relative redundancy of $\mathcal{C}$ equals $1 - \frac{1}{n}\log_q(|\mathcal{C}|) = 1 - \frac{1}{6}\log_3(9) = \frac{2}{3}$.

In this thesis, we consider three types of errors that may occur to a word $\mathbf{x} \in \mathcal{B}_q(n)$. Firstly, a deletion is an operation of the form $a \to \Lambda$ for $a \in \mathcal{B}_q$. A deletion of the symbol x_i in position $i \in \{1, ..., n\}$ of $\mathbf{x}$ results in the word $(x_1, \dots x_{i-1}, x_{i+1}, \dots x_n)$. The opposite operation of a deletion is an insertion. It is an operation of the form $\Lambda \to a$. An insertion of the symbol a between positions $i-1$ and i of $\mathbf{x}$ yields the word $(x_1, \dots x_{i-1}, a, x_i, \dots x_n)$. Lastly, a substitution is $a \to b$ type operation with $a, b \in \mathcal{B}_q$ and $a \neq b$. Hence, a substitution from x_i into b gives $(x_1, ..., x_{i-1}, b, x_{i+1}, ..., x_n)$. For example, the word $\mathbf{x} = 330112 \in \mathcal{B}_4(6)$ can be turned into

$$330112 \to 33012 \to 32012 \to 132012$$

by a deletion, substitution and an insertion. A deletion or an insertion is called an indel. In literature, (e.g., [13, 33]), the term 'edit' is commonly used for a multitude of errors that affect a single symbol. Therefore, in the context of this thesis, we refer to an edit as a deletion, insertion or a substitution.

Next, we are ready to give a formal definition of an indel and substitution correcting code. The following idea lies at the basis of this definition. Instead of using all q^n words in $\mathcal{B}_q(n)$ for communication or data storage, a code $\mathcal{C} \subset \mathcal{B}_q(n)$ is selected and only the words in $\mathcal{C}$ are used. Suppose the word $\mathbf{z} \in \mathcal{B}_q^*$ is received after some codeword $\mathbf{c} \in \mathcal{C}$ was sent. Moreover, assume that at most t errors occurred to $\mathbf{c}$. In case $\mathbf{c}$ is the only codeword in $\mathcal{C}$ from which $\mathbf{z}$ can be obtained by at most t edits, then $\mathbf{z}$ must have originated from $\mathbf{c}$. By ensuring that this property holds for all $\mathbf{z} \in \mathcal{B}_q^*$ it is always possible to correct at most t errors using the code $\mathcal{C}$.

Definition 1.5. For integers $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$, a code $\mathcal{C} \subseteq \mathcal{B}_q(n)$ is said to be a t-indel s-substitution correcting code if any q-ary word (not necessarily of length n) can be obtained from no more than one codeword of $\mathcal{C}$ by precisely t' deletions, t'' insertions and at most s substitutions, whenever $t' + t'' \leq t$.[3]

A 0-indel s-substitution correcting code is simply called an s-substitution correcting code. Analogously, a t-indel 0-substitution correcting code is called a t-indel correcting code.

[3]Given the indifference of deletions and insertions, we argue that the term t-indel s-substitution correcting code suits the capability of the code well. In literature, these codes are often called t-deletion s-substitution correcting codes, e.g., in [11, 13]. However, this ignores their capability of correcting insertions as well.

In Section 1.6 we will provide three concrete examples of t-indel s-substitution correcting codes. For each set of integers $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$ we are interested finding the maximal size of a t-indel s-substitution correcting code. This maximal size is denoted by,

$$M_q(n,t,s) := \max\{|\mathcal{C}| : \mathcal{C} \subseteq \mathcal{B}_q(n), \text{ s.t. } \mathcal{C} \text{ is a } t\text{-indel } s\text{-substitution correcting code}\}.$$

A t-indel s-substitution correcting code $\mathcal{C} \subseteq \mathcal{B}_q(n)$ that attains this size is called optimal. Such codes maximize the information rate and minimize the relative redundancy. In other words, these codes are most efficient in terms of redundancy while still being able to correct errors.

1.4 Problem statement and research question

In this thesis, the problem of determining the maximal size of t-indel and s-substitution correcting codes is studied. In other words, we aim to determine $M_q(n,t,s)$ or construct upper and lower bounds when computing the exact value is not viable.

This problem is what we previously reffered to as the central problem in coding theory. It has already been studied extensively for the specific cases of t-indel correcting codes and s-substitution correcting codes. This has lead to numerous bounds on $M_q(n,t,0)$ and $M_q(n,0,s)$ within literature, and to the exact value of these two quantities in highly specific cases. For an overview of existing bounds, see Section 4.1 and Chapter 5. In contrast, there is limited understanding of the more general problem for t-indel s-substitution correcting codes, as we have motivated in Section 1.1. Given that in many instances only bounds on $M_q(n,t,0)$ and $M_q(n,0,s)$ are known, it is unrealistic that exact values of $M_q(n,t,s)$ can be easily determined in general. Hence, we focus on constructing bounds for $M_q(n,t,s)$. This leads to the following research question:

How can existing bounds on the maximum size of t-indel correcting codes and s-substitution correcting codes be generalized to construct bounds on the maximum size of t-indel s-substitution correcting codes?

In short, we investigate how existing bounds on $M_q(n,t,0)$ and $M_q(n,0,s)$ can be used to construct bounds on $M_q(n,t,s)$.

1.5 Organisation of the thesis

In order to address the research question we employ the following organisation in this thesis. Firstly, we provide three concrete code constructions of t-indel s-substitution correcting codes in the next section. Chapters 2 & 3 deal with the properties and the cardinality of the set of words that can be reached from a word $\mathbf{x} \in \mathcal{B}_q(n)$ by precisely t' deletions, t'' insertions and at most s substitutions. These chapters do not answer the research question directly, but provide results that will be useful in the subsequent chapters. Next, Chapter 4 considers the research question aimed at lower bounds. Chapters 5 & 6 address the research question from the perspective of upper bounds. At last, in Chapter 7 we summarize the results of this thesis in order to answer the research question. Furthermore, several directions for future research are discussed.

1.6 Three code constructions

In this section, we discuss three code constructions for correcting various types of errors. We demonstrate how a code can be used to correct errors. The main idea of designing a code is making sure that the codewords are 'sufficiently different.' In case a certain number of errors occur, the resulting word should still 'resemble' the original word, but not any of the other codewords. An easy example of this principle is given by the following repetition code.

1.6.1 Repetition code

Let $n \geq 1$ and $q \geq 2$ be integers. Then the class of repetition codes is given by $\mathcal{C}_{n,q} = \{(a)^n : a \in \mathcal{B}_q\}$. For instance $\mathcal{C}_{5,4} = \{00000, 11111, 22222, 33333\}$. This class of codes has a high capability of correcting various types of errors. Up to $n-1$ deletions can be corrected by simply viewing which symbol is present in the resulting word. This follows because each codeword uses only a single symbol. For instance, if the word 22 is received it must have originated from 22222. Furthermore, up to $n-1$ insertions can also be corrected by counting which symbol occurs at least n times. Since the resulting word has a length of at most $2n-1$ the symbol $a \in \mathcal{B}_q$ for which this holds is unique. Hence, the original codeword must be $(a)^n$. For example, if 001001002 is received after sending a codeword from $\mathcal{C}_{5,4}$, then we note that that 0 occurs six times. Then, we decode to 00000. Using the repetition code $\mathcal{C}_{n,q}$ up to $\lfloor \frac{n-1}{2} \rfloor$ substitutions can be corrected. Since strictly less than half of the symbols are substituted, counting the symbol that occurs at least $\lceil \frac{n}{2} \rceil$ times indicates which codeword was used. We remark that $\mathcal{C}_{n,q}$ cannot correct these $n-1$ deletions, $n-1$ insertions and $\lfloor \frac{n-1}{2} \rfloor$ substitutions if they occur all together.

Nevertheless, it is also possible to correct combinations of deletions, insertions and substitutions using a suitable repetition code. For instance, any combination of at most one deletion, at most one insertion and at most one substitution can be corrected with $\mathcal{C}_{5,4}$. Namely, consider an arbitrary codeword from $\mathcal{C}_{5,4}$ consisting solely of the symbol $a \in \mathcal{B}_q$. In case one deletion occurred and possibly one insertion and one substitution, then the resulting word has length four or five and contains at least three times the symbol a. In case no deletion and possibly one insertion and one substitution occurred then the resulting word has length five or six and contains at least four times the symbol a. In both cases there is a unique symbol that occurs more often than the other symbols combined, which necessarily must be the symbol a. This uniquely indicates which codeword was used and allows the combination of errors to be corrected.

This example illustrates that $\mathcal{C}_{5,4}$ can correct many combinations of edits. Namely, we established that the $\mathcal{C}_{5,4}$ repetition code can correct either 4 deletions, or 4 insertions, or 2 substitutions, or the combination of 1 deletion, 1 insertion and 1 substitution.

The high capability of correcting errors comes at the cost of needing many redundant symbols. Only one out of the n symbols in a codeword contains information, while the others are copies and thus redundant. This leads to a large relative redundancy of $1 - \frac{1}{n}\log_q(|\mathcal{C}_{n,q}|) = 1 - \frac{1}{n}$. Hence, the repetition code can correct many errors at the cost of a high redundancy. The following two classes of codes can only correct a single substitution and indel, respectively, but need few redundant symbols.

1.6.2 Binary Hamming code

Next, we illustrate a class of single-substitution correcting codes that is well-known in coding theory: binary Hamming codes [3]. Hamming codes are commonly defined as a special class of linear codes [6], but we choose an alternative approach that requires no prior knowledge about linear codes. This approach is similar to the way in which Hamming codes were first described by Hamming [3] in 1950.

For the construction of Hamming codes, we first briefly introduce the concept of parity. For a q-ary word $\mathbf{x} \in \mathcal{B}_q(n)$ its parity is given by the least non-negative integer which is equal to $\sum_{i=1}^{n} x_i \mod q$. For instance, $\mathbf{x} = 0123210 \in \mathcal{B}_4(7)$ has a parity of 1, since $0+1+2+3+2+1+0 \equiv 9 \equiv 1 \mod 4$. Note that a single substitution in a word always changes the parity of that word. Using this observation it follows that the code $\{\mathbf{x} \in \mathcal{B}_q(n) : \sum_{i=1}^{n} x_i \equiv 0 \mod q\}$ can always detect one substitution error. Indeed, the parity of the corrupted word is non-zero whenever one substitution error occurs.

For the remainder of this construction we consider only $q = 2$. The key idea from Hamming in the code design is that by computing the parity on a subset of the positions instead of the entire word, it is possible to detect a single substitution in that subset. By carefully selecting multiple subsets on which to compute the parity, one is able to precisely pinpoint the position of the substitution. For binary words, the position of the substitution is sufficient for correction: a 0 must change into a 1, and vice versa.

In particular, we construct the binary Hamming code with codewords of length 15. Each codeword will contain 11 information symbols and 4 redundant symbols. This code is denoted by $Ham(15, 11)$. A word $\mathbf{x} \in \mathcal{B}_2(15)$ is codeword of $Ham(15, 11)$ if and only if it satisfies the following set of parity checks,

$$\begin{aligned}
x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} + x_{14} + x_{15} &\equiv 0 \mod 2,\\
x_4 + x_5 + x_6 + x_7 + x_{12} + x_{13} + x_{14} + x_{15} &\equiv 0 \mod 2,\\
x_2 + x_3 + x_6 + x_7 + x_{10} + x_{11} + x_{14} + x_{15} &\equiv 0 \mod 2,\\
x_1 + x_3 + x_5 + x_7 + x_9 + x_{11} + x_{13} + x_{15} &\equiv 0 \mod 2.
\end{aligned}$$

Note that x_1, x_2, x_4 and x_8 only occur in a single parity check. Hence, in a word $\mathbf{x} \in \mathcal{B}_2(15)$ the other eleven symbols can be chosen freely, while these four redundant symbol are fixed so that $\mathbf{x}$ satisfies the four parity checks. It follows that $Ham(15, 11)$ has a cardinality of $2^{15-4} = 2048$. The relative redundancy equals $1 - \frac{1}{15}\log_2(|Ham(15, 11)|) = \frac{4}{15}$ which corresponds to the fact that 4 out of the 15 symbols are redundant.

The code $Ham(15, 11)$ is able to correct up to one substitution error, as we will show below. The same four parity checks that are used to define codewords of $Ham(15, 11)$, can also be used to correct a potential substitution. These parity checks are chosen in such a way that the position of a potential error is narrowed down by checking the parity checks one by one. In the end, either the position of the substitution error is determined or it is concluded that no error occurred. Suppose $\mathbf{y} \in \mathcal{B}_2(15)$ is obtained from an unknown codeword of $Ham(15, 11)$ by at most one substitution. We perform the first parity check on $\mathbf{y}$ which determines whether an error occurred in the last eight positions of $\mathbf{y}$. In case $\sum_{i=8}^{15} y_i \equiv 1 \mod 15$, then there is an error in positions 8 to 15. Otherwise, we conclude that either an error occurred in the first seven positions, or

no error occurred at all. We proceed to the second parity which determines whether an error is present in positions 4, 5, 6, 7, 12, 13, 14, or 15. In case $\sum_{i=4}^{7} y_i + \sum_{i=12}^{15} y_i \equiv 1 \mod 15$, then an error occurred these positions. Combined with the conclusion from the first parity check, the error can be narrowed down to either the positions 4 to 7 or 12 to 15. If the second parity check for $\mathbf{y}$ equals 0, then we conclude that either the error occurred in the other positions or no error occurred at all. Again, combining this with the information of the first parity check narrows down the possible position of the error to 1,2,3 or 8, 9, 10, 11. The third and fourth parity check follow the same procedure.

After checking the four parities, either the position of the error is located, or it can be concluded that no error occurred (assuming there is at most one substitution error). Indeed, in case all parity checks yield 0 then there is no error, because the first parity check excludes the possibility of an error in positions 8 to 15. Subsequently, the following three parity checks exclude positions 4, 5, 6, 7, and 2, 3, and lastly 1. This leaves only the possibility that no error occurred. On the other hand, when at least one parity check is non-zero, an error occurred. Note that after the j-th parity check at most 2^{4-j} positions for the error remain possible, where $j \in \{1, 2, 3, 4\}$. Hence, after performing four parity checks the position of the error is located.

Example 1.6. Suppose that $\mathbf{y} = 101\ 0100\ 1101\ 0010$ is obtained from an unknown codeword $\mathbf{c} \in Ham(15, 11)$ by substituting at most one symbol. We aim to recover $\mathbf{c}$ from $\mathbf{y}$. The four parity checks yield

$$\begin{aligned} 1+1+0+1+0+0+1+0 &\equiv 0 \mod 2, \\ 0+1+0+0+0+0+1+0 &\equiv 0 \mod 2, \\ 0+1+0+0+0+1+1+0 &\equiv 1 \mod 2, \\ 1+1+1+0+1+1+0+0 &\equiv 1 \mod 2. \end{aligned}$$

The occurrence of at least one non-zero parity check indicates that a substitution error occurred. The first and second parity check being 0 indicate that the error does not occur in positions 8 to 15 and also not in 4 to 7 and 12 to 15, respectively. This leaves positions 1, 2 and 3 for the error. The third parity check being 1 shows that the error is in position 2, 3, 6, 7, 10, 11, 14 or 15. This narrows the position of the error further down to position 2 and 3. Similarly, the fourth parity check implies that the error is in the positions 1, 3, 5, 7, 9, 11, 13 or 15. This localizes the error in third position position. Hence, we carry out a $1 \to 0$ substitution in the third position of $\mathbf{y}$ and recover that $\mathbf{c} = 100\ 0100\ 1101\ 0010$. It can easily verified that $\mathbf{c}$ is a codeword of $Ham(15, 11)$ by checking that all four parity checks are equal to zero for $\mathbf{c}$.

1.6.3 Tenengolts' single indel correcting code

Lastly, we review Tenengolts' construction of a class of q-ary codes for correcting a single indel [34]. This construction is the q-ary generalization of a class of binary codes, called Varshamov-Tenengolts codes [15]. Levenshtein [7] showed that Varshamov-Tenengolts codes are capable of correcting a single indel. The following construction uses Levenshtein's reasoning to localize the position of a deletion and adds a simple parity check to determine the value of the deleted symbol.

For this construction, the set $\mathcal{B}_q(n)$ is partitioned into $q \cdot n$ classes. For each pair of integers $0 \leq \alpha \leq q-1$ and $0 \leq \beta \leq n-1$ we define the class $\mathcal{K}_{n,q}^{\alpha,\beta}$ as follows. A word $\mathbf{x} \in \mathcal{B}_q(n)$ belongs to $\mathcal{K}_{n,q}^{\alpha,\beta}$ if and only if $\mathbf{x} = x_1 x_2 \cdots x_n$ satisfies the following two congruences,

$$\sum_{i=1}^{n} x_i \equiv \alpha \mod q, \tag{1.1}$$

$$\sum_{i=1}^{n} (i-1) \cdot x'_i \equiv \beta \mod n, \tag{1.2}$$

where $\mathbf{x}' = x'_1 x'_2 \cdots x'_n$ is defined according to the rule $x'_1 = 1$ and

$$x'_i = \begin{cases} 1, \text{ if } x_i \geq x_{i-1}, \\ 0, \text{ if } x_i < x_{i-1}. \end{cases} \tag{1.3}$$

Remark that the classes are not necessarily of equal size [34]. For instance, $\mathcal{B}_3(5)$ contains $3^5 = 243$ words and is partitioned into 15 classes. Given that 15 is no divisor of 243, it follows that not all classes have equal size. A word in $\mathcal{B}_3(5)$ belongs to one and only one class, and thus there must be a class of size at least $\lceil \frac{243}{15} \rceil = 17$. In general, there exist parameters α and β such that $\mathcal{K}_{n,q}^{\alpha,\beta}$ has size of at least $\lceil \frac{q^n}{q \cdot n} \rceil$.

Each class $\mathcal{K}_{n,q}^{\alpha,\beta}$ forms a single-indel correcting code [34]. Here, we show how to correct a single deletion using $\mathcal{K}_{n,q}^{\alpha,\beta}$. By requiring that all codewords in $\mathcal{K}_{n,q}^{\alpha,\beta}$ satisfy the same two congruences, the value of the deleted symbol can be recovered via (1.1) and the position of the deletion via (1.2) and (1.3). Let $0 \leq \alpha \leq q-1$ and $0 \leq \beta \leq n-1$ be fixed and suppose that the sender transmits a codeword $\mathbf{x} \in \mathcal{K}_{n,q}^{\alpha,\beta}$. As a result of a single deletion, the word $\mathbf{y} = y_1 y_2 \cdots y_{n-1}$ is obtained by the receiver. Before correcting the deletion, we make several remarks.

The code parameters n, q, α and β as well as the word $\mathbf{y}$ are all known to the receiver. Therefore, the receiver can also compute $\mathbf{y}' = y'_1 y'_2 \cdots y'_{n-1}$ with $y'_1 = 1$ according to rule (1.3). In contrast, both $\mathbf{x}$ and $\mathbf{x}'$ are not known to the receiver. The receiver can directly detect that one deletion occurred by comparing length of $\mathbf{y}$ with n. For ease of notation, we denote by $j \in \{1, ..., n\}$ the position of the deletion in $\mathbf{x}$. That is, deleting symbol x_j from $\mathbf{x}$ yields $\mathbf{y}$. Note that j is unique up to the run to which x_j belongs. For instance, deleting the third or fifth symbol in 02111 both give 0211. Similarly, by $k \in \{1, ..., n\}$ we denote the position of the deletion in $\mathbf{x}'$, so that deleting x'_k from $\mathbf{x}'$ gives $\mathbf{y}'$.

This last statement actually requires a proof that $\mathbf{y}'$ can be obtained from $\mathbf{x}'$ by means of a single deletion. We claim that $\mathbf{y}'$ is obtained from $\mathbf{x}'$ by means of a single deletion in the run(s) of $\mathbf{x}'$ that include(s) x'_j and x'_{j+1}. Clearly, it does not matter which symbol from a run is deleted, because always the same word is obtained. From (1.3) it follows that $y'_i = x'_i$ for all $i < j$ and $y'_{i-1} = x'_i$ for all $i > j+1$. So the claim holds if $y'_j \in \{x'_j, x'_{j+1}\}$. In case $x'_j \neq x'_{j+1}$, then it trivially holds, because $\mathbf{x}'$ and $\mathbf{y}'$ are binary words. On the other hand, if $x'_j = x'_{j+1} = 1$, then the sequence x_{j-1}, x_j, x_{j+1} is non-decreasing according to rule (1.3). Therefore, it holds that $y'_j = 1$, since $y_{j-1} = x_{j-1} \leq x_{j+1} = y_j$. Analogously, we find $y'_j = 0$ whenever $x'_j = x'_{j+1} = 0$. Hence, the claim is true and also shows that it does not necessarily hold that $j = k$.

Now, we are ready to correct the deletion. The description of this process in [34] is quite brief, while it is rather complex. Therefore, we provide a more detailed description, and divide the process into the following four steps:

1. Recover the (q-ary) value of x_j.
2. Recover the (binary) value of x'_k.
3. Recover the position of k, and consequently recover $\mathbf{x}'$.
4. Recover the position of j using $\mathbf{x}'$ and the value of x_j. The position j and the value x_j jointly recover $\mathbf{x}$.

Step 1. The value of the deleted symbol can be reconstructed using (1.1). Notice that (1.1) acts as a parity check for the codewords. After the deletion, the parity of the received word is changed precisely by x_j. Since the parity of $\mathbf{x}$ is known to be α and the parity of $\mathbf{y}$ can be computed, x_j can be determined by assessing the difference between α and the parity of $\mathbf{y}$. Let S_1 be the smallest non-negative integer such that $S_1 = \alpha - \sum_{i=1}^{n-1} y_i \mod q$, then S_1 can be computed by the receiver. It holds that

$$x_j \equiv \sum_{i=1}^{n} x_i - \sum_{i=1}^{n-1} y_i \equiv \alpha - \sum_{i=1}^{n-1} y_i \equiv S_1 \mod q.$$

Since it holds that $0 \leq x_j \leq q-1$ and $0 \leq S_1 \leq q-1$, the previous congruence shows that $x_j = S_1$. The receiver has thus determined the value of the deleted symbol by computing S_1.

Step 2. Define $W = \sum_{i=1}^{n-1} y'_i$ and let S_2 be the smallest non-negative integer such that $S_2 \equiv \beta - \sum_{i=1}^{n-1}(i-1)y'_i \mod n$. Note that W and S_2 can be computed by the receiver. Intuitively, W denotes the number of ones in $\mathbf{y}$ and S_2 indicates how (1.2) is changed due to the deletion. Note that after the deletion of x'_k from $\mathbf{x}'$, the contribution of $(k-1)x'_k$ is removed from the summation in (1.2). Moreover, the symbols to the right of x'_k have shifted to the left by one in $\mathbf{y}'$ with respect to $\mathbf{x}'$. It follows that

$$S_2 \equiv \beta - \sum_{i=1}^{n-1}(i-1)y'_i \equiv \sum_{i=1}^{n}(i-1)x'_i - \sum_{i=1}^{n-1}(i-1)y'_i \equiv (k-1)x'_k + n_1 \mod n,$$

where n_1 denotes the number of ones in $\mathbf{x}'$ to the right of x'_k. Since both $0 \leq S_2 \leq n-1$ and $0 \leq (k-1)x'_k + n_1 \leq k-1+(n-k) = n-1$, it follows that $S_2 = (k-1)x'_k + n_1$. Recall that W and S_2 are both known to receiver, but n_1 is not.

This allows us to recover x'_k. Namely, it holds that $S_2 < W$ if and only if $x'_k = 0$, which is equivalent to stating that $S_2 \geq W$ if and only if $x'_k = 1$. In order to prove these equivalences, it suffices to show that $x'_k = 0$ implies $S_2 < W$ and $x'_k = 1$ implies $S_2 \geq W$. Indeed, if $x'_k = 0$, then it holds that $S_2 = n_1 < W$ because $x'_1 = 1$ by definition. Moreover, if $x'_k = 1$, then it holds that $S_2 = k - 1 + n_1 \geq W$. This holds because $\sum_{i=1}^{k-1} y'_i \leq k-1$ and $\sum_{i=k}^{n-1} y'_i = \sum_{i=k+1}^{n} x'_i = n_1$. We conclude that $x'_k = 0$ if $S_2 < W$, whereas $x'_k = 1$ if $S_2 \geq W$.

Step 3. For the recovery of $\mathbf{x}'$ it remains to determine the position of x'_k in $\mathbf{x}'$. In case $S_2 \geq W$, then it follows from the previous step that $x'_k = 1$ and $S_2 = k - 1 + n_1$.

Notice that $S_2 - W = k - 1 + n_1 - W$ denotes the number of zeros to the left of x'_k. This indicates where x'_k should be inserted in $\mathbf{y}'$. Namely, the receiver inserts a 1 in $\mathbf{y}'$ so that the number of zeros to left of the insertion is equal to $S_2 - W$.

In the other case that $S_2 < W$, then we know from the previous step that $x'_k = 0$ and $S_2 = n_1$. Recall that n_1 denotes the number of ones in $\mathbf{x}'$ to the right of x'_k. Therefore, the receiver inserts a 0 in $\mathbf{y}'$ so that the number of ones to right of the insertion is equal to S_2. Hence, we recovered $\mathbf{x}'$ in both cases.

Step 4. We recover the position of the deletion in $\mathbf{x}$, i.e., the index j. Recall the claim which states that $\mathbf{y}'$ is obtained from $\mathbf{x}'$ by means of a single deletion in the run(s) of $\mathbf{x}'$ that include(s) x'_j and x'_{j+1}. Let h denote the run in $\mathbf{x}'$ that contains x'_k, then this claim implies that x'_j lies in the h-th or $(h-1)$-th run in $\mathbf{x}'$. Note that rule (1.3) implies that a binary run in $\mathbf{x}'$ of length l corresponds to a non-decreasing or strictly decreasing sequence of symbols in $\mathbf{x}$ of length $l+1$. Here, we ignore the first 1 in $\mathbf{x}'$. For instance, with $\mathbf{z} = 1233013210$ and $\mathbf{z}' = 1111011000$, then the first run '111' in $\mathbf{z}'$ (ignoring the first 1) corresponds to the non-decreasing sequence '1233'. Similarly, '0' corresponds to '30', '11' to '013' and '000' to '3210'. Indeed, the binary runs in $\mathbf{z}'$ are one shorter than the lengths of the corresponding monotonic sequences in $\mathbf{z}$.

Recall that we know $\mathbf{y}$ and that we recovered $\mathbf{x}'$. Next, consider the lengths of the h-th and $(h-1)$-th run in $\mathbf{x}'$. If we compare these lengths to the lengths of the corresponding monotonic sequences in $\mathbf{y}$, then one of these sequences must be one short due to the deletion. Hence, x_j must be inserted in that monotonic sequence. The position of the insertion of x_j within that sequence is determined by the value of x_j, which is computed in step 1, and the fact that this sequence is monotonic.

To summarize, we give a concrete example of this class of codes and the decoding algorithm.

Example 1.7. Let $n = 9$, $q = 4$, $\alpha = 2$ and $\beta = 4$, so that we consider the code $\mathcal{K}_{9,4}^{2,4}$. The word $\mathbf{x} = 302331110$ and its associated word $\mathbf{x}' = 101110110$ satisfy

$$\sum_{i=1}^{9} x_i \equiv 3+0+2+3+3+1+1+1+0 \equiv 14 \equiv 2 \mod 4,$$

$$\sum_{i=1}^{9} (i-1)x'_j \equiv 0\cdot 1 + 1\cdot 0 + 2\cdot 1 + 3\cdot 1 + 4\cdot 1 + 5\cdot 0 + 6\cdot 1 + 7\cdot 1 + 8\cdot 0 \equiv 22 \equiv 4 \mod 9.$$

Hence, $\mathbf{x}$ is a codeword of $\mathcal{K}_{9,4}^{2,4}$.

Suppose that the word $\mathbf{y} = 30231110$ is received, and that the receiver knows that it originated from a word in $\mathcal{K}_{9,4}^{2,4}$ and that at most one deletion occurred. However, the receiver does not know $\mathbf{x}$ nor $\mathbf{x}'$. By checking the length of $\mathbf{y}$ it is immediately clear that one deletion occurred. Next, we show how the four aforementioned steps can be used to recover $\mathbf{x}$.

Step 1. We compute S_1 given by,

$$S_1 \equiv 2 - (3+0+2+3+1+1+1+0) \equiv 2 - 11 \equiv 3 \mod 4,$$

and conclude that the deleted symbol is a '3'.

Step 2. The received word $\mathbf{y} = 30231110$ yields the associated word $\mathbf{y}' = 10110110$, according to rule (1.3). Hence, it holds that $W = 5$. The parameter S_2 satisfies

$$S_2 \equiv 4 - (0 \cdot 1 + 1 \cdot 0 + 2 \cdot 1 + 3 \cdot 1 + 4 \cdot 0 + 5 \cdot 1 + 6 \cdot 1 + 7 \cdot 0) \equiv 4 - 16 \equiv 6 \mod 9$$

which gives $S_2 = 6$. Since $S_2 > W$ we conclude that $x'_k = 1$. In other words, $\mathbf{x}'$ can be obtained from $\mathbf{y}'$ by inserting a '1' (in a yet unknown position).

Step 3. From the previous step, we know that we need to insert a '1' into $\mathbf{y}'$, and that $S_2 > W$. This '1' is inserted so that the number of zeros to the left of this insertion equals $S_2 - W = 6 - 5 = 1$. Hence, we recover that $\mathbf{x}' = 101110110$.

Step 4. The recovered word $\mathbf{x}' = 101110110$ has runs '0', '111', '0', '11' and '0', where we ignore the first 1. The word $\mathbf{y} = 30231110$ consists of the non-decreasing and strictly decreasing sequences '30', '023', '31', '111' and '01'. Notice that the second sequence '023' has the same length as its corresponding binary run '111'. This means that this sequence is short by one. Hence, the symbol '3' must be inserted in the sequence '023'. By definition, this sequence must be non-decreasing also after the insertion, and thus we obtain '0233'.

All in all, we find that $\mathbf{x} = 302331110$, and thus the correction is successful.

2

Deletion, insertion and substitution set

In this chapter we set out to discuss the set of words that can be reached from a q-ary word by means of precisely t' deletions, t'' insertions and at most s substitutions. For $\mathbf{x} \in \mathcal{B}_q(n)$, this set is denoted by $\mathcal{V}_{t',t'',s}(\mathbf{x})$. In particular, we formally define this set and discuss how this set is related to t-indel s-substitution correcting codes in Section 2.1. In Section 2.2, we consider how the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ and the number of runs in $\mathbf{x}$ are related. The next chapter is dedicated to the cardinality of this set.

The contents of these two chapters do not answer the research question directly. More so, they provide the tools for answering these questions in subsequent chapters. The set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ is important in this regard because it lies at the heart of the definition of t-indel s-substitution correcting codes. Therefore, properties of the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ and results on its cardinality are often needed in order to derive bounds on the maximum cardinality of t-indel s-substitution correcting codes.

2.1 Description of $\mathcal{V}_{t',t'',s}(\mathbf{x})$ and its basic properties

Let us start by giving a formal definition of the set of words that can be reached from a q-ary word by a certain number of deletions, insertions and substitutions.

Definition 2.1. Let $n \geq 1$, $q \geq 2$, $t', t'' \geq 0$ and $s \geq 0$ be integers such that $n - t' + t'' \geq 0$. For a q-ary word $\mathbf{x} \in \mathcal{B}_q(n)$, let $\mathcal{V}_{t',t'',s}(\mathbf{x}) \subseteq \mathcal{B}_q(n - t' + t'')$ denote the set of words that can be reached from $\mathbf{x}$ by precisely t' deletions, t'' insertions and at most s substitutions (not necessarily in that order)[1].

Moreover, define the deletion set by $\mathcal{D}_{t'}(\mathbf{x}) = \mathcal{V}_{t',0,0}(\mathbf{x})$, the insertion set by $\mathcal{I}_{t''}(\mathbf{x}) = \mathcal{V}_{0,t'',0}(\mathbf{x})$ and the substitution set by $\mathcal{S}_s(\mathbf{x}) = \mathcal{V}_{0,0,s}(\mathbf{x})$.

Note that t' deletions reduce the length of a word by t', t'' insertions increase the length by t'', whereas s substitutions leave the length unaffected. Therefore, for a q-ary word $\mathbf{x}$ of length n, the words in the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ have length $n - t' + t''$. In this definition and throughout the sequel of this thesis, we use the following conventions. Firstly, we use $\mathcal{V}_{0,0,0}(\mathbf{x}) = \{\mathbf{x}\}$. Secondly, for $n - t' + t'' = 0$ and $\mathbf{x} \in \mathcal{B}_q(n)$, we set $\mathcal{V}_{t',t'',s}(\mathbf{x}) = \{\Lambda\}$ and $|\mathcal{V}_{t',t'',s}(\mathbf{x})| = 1$, where we recall that Λ denotes the empty word of length 0.

In Appendix B, a Python script is given that can be used to compute the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ for concrete words and parameters. In the following example, we illustrate how to determine the words in the set $\mathcal{V}_{2,1,1}(2221)$.

Example 2.2. Consider the word $2221 \in \mathcal{B}_3(4)$ and the parameters $t' = 2$, $t'' = 1$ and $s = 1$. It holds that $n = 4$ and $q = 3$. In this example, we compute the set $\mathcal{V}_{2,1,1}(2221)$. The words in this set have length $n - t' + t'' = 3$. Note that there exist $q^{n-t'+t''} = 27$ ternary words of length 3. As a first step, we consider the words that are obtained via the specific order of t' deletions followed by t'' insertions and lastly by at most s substitutions. These words clearly form a subset of $\mathcal{V}_{2,1,1}(2221)$.

After two deletions from 2221 it holds that $\mathcal{D}_2(2221) = \{21, 22\}$. Indeed, it is possible to delete either twice a symbol 2 from 2221 resulting in 21, or delete the symbol 1 and one 2 giving 22 which gives 22. Next, we compute

$$\begin{aligned} \mathcal{I}_1(21) &= \{021, 121, 221, 201, 211, 210, 212\}, \\ \mathcal{I}_1(22) &= \{022, 122, 222, 202, 212, 220, 221\}. \end{aligned}$$

The union of these two sets gives all words that can be obtained from 2221 by two deletions followed by one insertion. This union is given by

$$\mathcal{I}_1(21) \cup \mathcal{I}_1(22) = \{021, 022, 121, 122, 201, 202, 210, 211, 212, 220, 221, 222\}.$$

Lastly, we consider the single substitution. That means that we compute $\mathcal{S}_1(\mathbf{y})$ for all $\mathbf{y} \in \mathcal{I}_1(21) \cup \mathcal{I}_1(22)$, and again take the union of these sets. For brevity, we mention only

$$\begin{aligned} \bigcup_{\mathbf{y} \in \mathcal{I}_1(21) \cup \mathcal{I}_1(22)} \mathcal{S}_1(\mathbf{y}) = \{&001, 002, 010, 011, 012, 020, 021, 022, 101, 102, 110, 111, 112 \\ &120, 121, 122, 200, 201, 202, 210, 211, 212, 220, 221, 222\}. \end{aligned} \tag{2.1}$$

[1] For ease of remembering, the subscripts in $\mathcal{V}_{t',t'',s}(\mathbf{x})$ are ordered alphabetically: deletions, insertions and substitutions.

It is not hard to verify that each of these 25 words in this set can be obtained by at most one substitution from a word in $\mathcal{I}_1(21) \cup \mathcal{I}_1(22)$. For instance, the word 001 can be obtained from 201 by a $0 \to 2$ substitution of the first symbol. The only two words in $\mathcal{B}_3(3)$ that are not in this set are 000 and 100. Indeed, these two words cannot be reached from $\mathcal{I}_1(21) \cup \mathcal{I}_1(22)$ by at most one substitution.

To summarize, we have constructed the set of words that can be reached from 2221 by precisely two deletions, one insertions and at most one substitutions, in that specific order of edits. This set is given by (2.1). According to Definition 2.1, we still need to verify whether additional words (i.e., 100 and/or 000) are contained in $\mathcal{V}_{2,1,1}(2221)$. These words might be obtained by considering the edits in a different order. Fortunately, we will show that is not the case, and that in fact we have found all elements in $\mathcal{V}_{2,1,1}(2221)$.

The aforementioned definition of the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ does not impose the order in which the deletions, insertions and substitutions are considered. The following lemma shows that the order does not matter. Every order of the edits (e.g., first all deletions, followed by the insertions and lastly the substitutions) leads to the same set. In other words, specifying a particular order of the edits does not lead to a more restricted set of words. Therefore, we are free to choose any order of the edits.

This property has been proven in [35, Lem. 1] for deletions and insertions only. It has been stated for deletions, insertions as well as substitutions in [36, Sec. 2], but without a proof nor a reference. Here, we extend the proof of [35] to also include substitutions, and thus provide a proof for the statement in [36].

Lemma 2.1. *Let $n \geq 1$ and $q \geq 2$ be integers and $\mathbf{x} \in \mathcal{B}_q(n)$. Let $t'_1, t''_1, s_1, t'_2, t''_2, s_2 \geq 0$ be integers such that $n - t'_1 + t''_1 \geq 1$ and $n - t'_1 - t'_2 + t''_1 + t''_2 \geq 1$. Then, the following holds*[2]*,*

$$\mathcal{V}_{t'_2,t''_2,s_2}(\mathcal{V}_{t'_1,t''_1,s_1}(\mathbf{x})) = \mathcal{V}_{t'_1+t'_2,t''_1+t''_2,s_1+s_2}(\mathbf{x}).$$

Proof. Firstly, we observe that the requirements $n - t'_1 + t''_1 \geq 1$ and $n - t'_1 - t'_2 + t''_1 + t''_2 \geq 1$ imply that the set $\mathcal{V}_{t'_2,t''_2,s_2}(\mathcal{V}_{t'_1,t''_1,s_1}(\mathbf{x}))$ is well-defined. It suffices to show the following six properties, which together will prove the lemma. For $a, b \geq 1$, we claim that $\mathcal{D}_1(\mathcal{I}_1(\mathbf{x})) = \mathcal{I}_1(\mathcal{D}_1(\mathbf{x}))$, $\mathcal{D}_1(\mathcal{S}_1(\mathbf{x})) = \mathcal{S}_1(\mathcal{D}_1(\mathbf{x}))$, $\mathcal{I}_1(\mathcal{S}_1(\mathbf{x})) = \mathcal{S}_1(\mathcal{I}_1(\mathbf{x}))$, $\mathcal{D}_a(\mathcal{D}_b(\mathbf{x})) = \mathcal{D}_{a+b}(\mathbf{x})$, $\mathcal{I}_a(\mathcal{I}_b(\mathbf{x})) = \mathcal{I}_{a+b}(\mathbf{x})$ and $\mathcal{S}_a(\mathcal{S}_b(\mathbf{x})) = \mathcal{S}_{a+b}(\mathbf{x})$. By repeatedly applying these properties the result follows. The latter three properties follow directly from Definition 2.1, so we prove solely the first three properties. In what follows, the superscripts $^+$ and $^-$ denote that the variables are elements of $\mathcal{B}_q(n+1)$ and $\mathcal{B}_q(n-1)$ respectively, and an omission thereof represents variables in $\mathcal{B}_q(n)$.

To this end, let $\mathbf{y} \in \mathcal{D}_1(\mathcal{I}_1(\mathbf{x}))$ which means that there exists some $\mathbf{z}^+ \in \mathcal{I}_1(\mathbf{x})$ such that $\mathbf{y} \in \mathcal{D}_1(\mathbf{z}^+)$. Hence, $\mathbf{x}, \mathbf{y} \in \mathcal{D}_1(\mathbf{z}^+)$ and thus there exists some word $\mathbf{z}^- \in \mathcal{D}_2(\mathbf{z}^+)$ such that $\mathbf{z}^- \in \mathcal{D}_1(\mathbf{x})$ and $\mathbf{z}^- \in \mathcal{D}_1(\mathbf{y})$. In turn, this implies that $\mathbf{y} \in \mathcal{I}_1(\mathbf{z}^-)$ and $\mathbf{y} \in \mathcal{I}_1(\mathcal{D}_1(\mathbf{x}))$. This shows that $\mathcal{D}_1(\mathcal{I}_1(\mathbf{x})) \subseteq \mathcal{I}_1(\mathcal{D}_1(\mathbf{x}))$. The converse inclusion follows from considering a similar argument in the reverse order and its proof is therefore omitted.

Next, let $\mathbf{y}^- \in \mathcal{D}_1(\mathcal{S}_1(\mathbf{x}))$ then there exists some $\mathbf{z} \in \mathcal{B}_q(n)$ such that $d(\mathbf{x}, \mathbf{z}) \leq 1$ and $\mathbf{y}^- \in \mathcal{D}_1(\mathbf{z})$ where d denotes the Hamming distance. Let i denote the leftmost position that can be deleted from $\mathbf{z}$ to yield $\mathbf{y}^-$. Define the word $\mathbf{z}^-$ as the word which is obtained

[2]Hereforth, we will use the slight abuse of notation: $\mathcal{V}_{t'_2,t''_2,s_2}(\mathcal{V}_{t'_1,t''_1,s_1}(\mathbf{x})) = \bigcup_{\mathbf{y} \in \mathcal{V}_{t'_1,t''_1,s_1}(\mathbf{x})} \mathcal{V}_{t'_2,t''_2,s_2}(\mathbf{y})$.

from $\mathbf{x}$ by performing a deletion on position i. Since the deletions are performed on the same positions, it follows that $d(\mathbf{z}^-, \mathbf{y}^-) \leq 1$. Hence, $\mathbf{z}^-$ satisfies both $d(\mathbf{z}^-, \mathbf{y}^-) \leq 1$ and $\mathbf{z}^- \in \mathcal{D}_1(\mathbf{x})$. This shows that $\mathbf{y}^- \in \mathcal{S}_1(\mathcal{D}_1(\mathbf{x}))$ and $\mathcal{D}_1(\mathcal{S}_1(\mathbf{x})) \subseteq \mathcal{S}_1(\mathcal{D}_1(\mathbf{x}))$. Again, the proof of the converse inclusion follows similarly and is left out.

Lastly, let $\mathbf{y}^+ \in \mathcal{I}_1(\mathcal{S}_1(\mathbf{x}))$ which implies that there exists some $\mathbf{z} \in \mathcal{B}_q(n)$ such that $d(\mathbf{x}, \mathbf{z}) \leq 1$ and $\mathbf{y}^+ \in \mathcal{I}_1(\mathbf{z})$. Define $\mathbf{z}^+$ as the word that is obtained by performing the same insertion on $\mathbf{x}$ as the insertion that transforms $\mathbf{z}$ into $\mathbf{y}^+$. In case multiple insertions can transform $\mathbf{z}$ into $\mathbf{y}^+$, the leftmost insertion is taken for uniqueness. Since both the placement and symbols of the two insertions in $\mathbf{x}$ and $\mathbf{z}$ are equal, we find also that $d(\mathbf{z}^+, \mathbf{y}^+) \leq 1$. Combined with $\mathbf{z}^+ \in \mathcal{I}_1(\mathbf{x})$ this gives $\mathbf{y}^+ \in \mathcal{S}_1(\mathcal{I}_1(\mathbf{z}))$. We conclude that $\mathcal{I}_1(\mathcal{S}_1(\mathbf{x})) \subseteq \mathcal{S}_1(\mathcal{I}_1(\mathbf{x}))$. In a similar way the proof of the converse inclusion can be obtained and it is thus omitted. □

So far, we have formally described the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ and shown that the order of the edits in this set can be chosen freely. Next, we make the connection between this set and t-indel s-substitution correcting codes more concrete. Namely, we will state five equivalent ways to characterize t-indel s-substitution correcting codes in terms of the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$. Before doing so, we treat two simple properties that will be useful for proving this result. The first property deals with reversing the order of the edits.

Lemma 2.2 ([36])**.** *Let $n \geq 1$, $q \geq 2$, $t', t'' \geq 0$ and $s \geq 0$ be integers such that $n - t' + t'' \geq 1$. Let $\mathbf{x} \in \mathcal{B}_q(n)$ and $\mathbf{y} \in \mathcal{B}_q(n - t' + t'')$, then $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{x})$ if and only if $\mathbf{x} \in \mathcal{V}_{t'',t',s}(\mathbf{y})$.*

Proof. For a single edit the statement clearly holds. Namely, a deletion can be reversed by an insertion and vice versa. Moreover, an $a \to b$ substitution is reversable by $b \to a$ substitution. In other words, $\mathbf{y} \in \mathcal{D}_1(\mathbf{x})$ if and only if $\mathbf{x} \in \mathcal{I}_1(\mathbf{y})$ and $\mathbf{y} \in \mathcal{S}_1(\mathbf{x})$ if and only if $\mathbf{x} \in \mathcal{S}_1(\mathbf{y})$.

Suppose that $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{x})$, then we will show that $\mathbf{x} \in \mathcal{V}_{t'',t',s}(\mathbf{y})$. Beware that the order of t' and t'' is reversed in the previous statement. Consider a sequence of edits that turns $\mathbf{x}$ into $\mathbf{y}$ using t' deletions, t'' insertions and s substitutions. We apply this sequence of edits to $\mathbf{y}$ in reverse order. That is, the first edit in the sequence that turns $\mathbf{x}$ into $\mathbf{y}$ will be the last edit in the sequence that turns $\mathbf{y}$ into $\mathbf{x}$, et cetera. Moreover, we also reverse the type of edits: a deletion is carried out instead of an insertion, an insertion instead of a deletion and an $a \to b$ substitution instead of a $b \to a$ substitution. Using the observations from the first paragraph, it follows that $\mathbf{x}$ can be obtained from $\mathbf{y}$ by precisely t'' deletions, t' insertions and at most s substitutions, i.e., $\mathbf{x} \in \mathcal{V}_{t'',t',s}(\mathbf{y})$. For symmetry reasons, the converse statement follows analogously. □

The second property was stated by Cullina and Kiyavash in [35, Lem. 1], and will be helpful in the next proof. For completeness, we restate their short proof.

Lemma 2.3 ([35], Lemma 1)**.** *Let $n \geq 1$, $q \geq 2$ and $0 \leq t \leq n$ be integers, and let $\mathbf{x}, \mathbf{y} \in \mathcal{B}_q(n)$. Then it holds that $\mathcal{D}_t(\mathbf{x}) \cap \mathcal{D}_t(\mathbf{y}) \neq \emptyset$ if and only if $\mathcal{I}_t(\mathbf{x}) \cap \mathcal{I}_t(\mathbf{y}) \neq \emptyset$.*

Proof. Suppose that $\mathcal{D}_t(\mathbf{x}) \cap \mathcal{D}_t(\mathbf{y})$ is non-empty and let $\mathbf{z}^- \in \mathcal{D}_t(\mathbf{x}) \cap \mathcal{D}_t(\mathbf{y})$. By Lemma 2.2 it holds that $\mathbf{z}^- \in \mathcal{D}_t(\mathbf{x})$ and $\mathbf{y} \in \mathcal{I}_t(\mathbf{z}^-)$. In turn, this jointly gives $\mathbf{y} \in \mathcal{I}_t(\mathcal{D}_t(\mathbf{x}))$. By Lemma 2.1 the order of the edits can be reversed which gives $\mathbf{y} \in \mathcal{D}_t(\mathcal{I}_t(\mathbf{x}))$. Thus there exists a $\mathbf{z}^+ \in \mathcal{I}_t(\mathbf{x})$ such that $\mathbf{y} \in \mathcal{D}_t(\mathbf{z}^+)$. Rearranging the last relation gives $\mathbf{z}^+ \in \mathcal{I}_t(\mathbf{y})$

by Lemma 2.2. Hence, we find $\mathbf{z}^+ \in \mathcal{I}_t(\mathbf{x}) \cap \mathcal{I}_t(\mathbf{y})$. We conclude that $\mathcal{I}_t(\mathbf{x}) \cap \mathcal{I}_t(\mathbf{y})$ is non-empty as well. The converse statement follows analogously. □

Now we are ready to show the connection between the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ and t-indel s-substitution correcting codes. The next lemma collects several similar results from e.g., [11, Lem. 2], [13, Sec. II], [35, Lem. 2] and [37, Cor. 2]. None of these results state the following lemma in full, but collectively they imply the following lemma. We provide a proof of the entire statement below.

Lemma 2.4. *Let $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$ be integers, and let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be a code. Then, the following statements are equivalent:*

1. *$\mathcal{C}$ is a t-indel s-substitution correcting code.*
2. *$\mathcal{V}_{t',t'',s}(\mathbf{c}_1) \cap \mathcal{V}_{t',t'',s}(\mathbf{c}_2) = \emptyset$ for all distinct codewords $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$, and for all integers $t', t'' \geq 0$ such that $t' + t'' \leq t$.*
3. *$\mathcal{V}_{t,0,s}(\mathbf{c}_1) \cap \mathcal{V}_{t,0,s}(\mathbf{c}_2) = \emptyset$ for all distinct codewords $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$.*
4. *$\mathcal{V}_{0,t,s}(\mathbf{c}_1) \cap \mathcal{V}_{0,t,s}(\mathbf{c}_2) = \emptyset$ for all distinct codewords $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$.*
5. *$\mathbf{c}_2 \notin \mathcal{V}_{t,t,2s}(\mathbf{c}_1)$ for all distinct $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$.*

Proof. We show five implications in a circular manner which proves the equivalence of all statements. Recall that by definition $\mathcal{C}$ is called a t-indel s-substitution correcting code if any q-ary word (not necessarily of length n) can be obtained from no more than one codeword of $\mathcal{C}$ by precisely t' deletions, t'' insertions and at most s substitutions, whenever $t' + t'' \leq t$.

(1 ⇒ 2). Suppose that $\mathcal{C} \subseteq \mathcal{B}_q(n)$ is a t-indel s-substitution correcting code. Let $\mathbf{c}_1$ and $\mathbf{c}_2$ be two distinct codewords of $\mathcal{C}$ and $t', t'' \geq 0$ be integers such that $t' + t'' \leq t$. Suppose for contradiction that there exists some $\mathbf{z} \in \mathcal{V}_{t',t'',s}(\mathbf{c}_1) \cap \mathcal{V}_{t',t'',s}(\mathbf{c}_2)$, then $\mathbf{z}$ can be obtained from two codewords from $\mathcal{C}$ by t' deletions, t'' insertions and s or fewer substitutions. This violates the definition of the t-indel s-substitution correcting code $\mathcal{C}$. Hence, this proves the first implication.

(2 ⇒ 3). This implication is immediate when setting $t' = t$ and $t'' = 0$.

(3 ⇒ 5). Let $\mathbf{c}_1$ and $\mathbf{c}_2$ be two distinct codewords of $\mathcal{C}$ and assume that $\mathcal{V}_{t,0,s}(\mathbf{c}_1) \cap \mathcal{V}_{t,0,s}(\mathbf{c}_2) = \emptyset$. Moreover, assume for contradiction that $\mathbf{c}_2 \in \mathcal{V}_{t,t,2s}(\mathbf{c}_1)$. This implies that $\mathbf{c}_2$ can be obtained from $\mathbf{c}_1$ by precisely t deletions, t insertions and $k \leq 2s$ substitutions. By Lemma 2.1 it follows that this can be done in any ordering of the edits. In particular, it follows that there exists a $\mathbf{z} \in \mathcal{B}_q(n-t)$ such that $\mathbf{z}$ can be reached by from $\mathbf{c}_1$ by t deletions and $\lceil \frac{k}{2} \rceil \leq s$ substitutions. Moreover, $\mathbf{c}_2$ can be obtained from $\mathbf{z}$ by t insertions and $k - \lceil \frac{k}{2} \rceil = \lfloor \frac{k}{2} \rfloor \leq s$ substitutions. All in all, it follows that $\mathbf{z} \in \mathcal{V}_{t,0,s}(\mathbf{c}_1)$ and $\mathbf{c}_2 \in \mathcal{V}_{0,t,s}(\mathbf{z})$, which gives $\mathbf{z} \in \mathcal{V}_{t,0,s}(\mathbf{c}_1) \cap \mathcal{V}_{t,0,s}(\mathbf{c}_2)$ according to Lemma 2.2. However, this contradicts the assumption that this intersection is empty. Hence, the third statement implies the fifth statement of this lemma.

(5 ⇒ 4). Let $\mathbf{c}_1$ and $\mathbf{c}_2$ be two distinct codewords of $\mathcal{C}$ and assume $\mathbf{c}_2 \notin \mathcal{V}_{t,t,2s}(\mathbf{c}_1)$. Assume for contradiction that there exists a $\mathbf{z} \in \mathcal{V}_{0,t,s}(\mathbf{c}_1) \cap \mathcal{V}_{0,t,s}(\mathbf{c}_2)$. Then, by Lemma 2.2, it holds that $\mathbf{c}_2 \in \mathcal{V}_{t,0,s}(\mathbf{z})$. However, this implies together with $\mathbf{z} \in \mathcal{V}_{0,t,s}(\mathbf{c}_1)$ that $\mathbf{c}_2 \in \mathcal{V}_{t,0,s}(\mathcal{V}_{0,t,s}(\mathbf{c}_1))$. From Lemma 2.1 it follows that $\mathbf{c}_2 \in \mathcal{V}_{t,t,2s}(\mathbf{c}_1)$ which contradicts the first assumption.

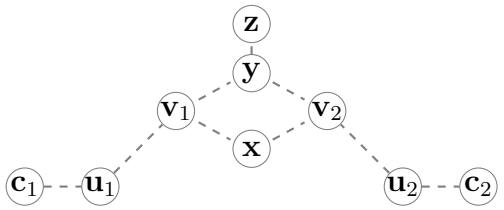


Figure 2.1: Schematic representation of the variables in the proof of implication (*4* ⇒ *1*) in Lemma 2.4. On the right, the lengths of the variables are given. Horizontal lines indicate that these variables can be obtained from each other by substitutions. Diagonal and vertical lines imply that the bottom variable can be obtained from the top variable via deletions, or vice versa through insertions.

(*4* ⇒ *1*). Assume that $\mathcal{V}_{0,t,s}(\mathbf{c}_1) \cap \mathcal{V}_{0,t,s}(\mathbf{c}_2) = \emptyset$ for all distinct codewords $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$. Let $t', t'' \geq 0$ be integers such that $t' + t'' \leq t$, and $\mathbf{x} \in \mathcal{B}_q(n - t' + t'')$ be an arbitrary word. Then, we will show by contradiction that $\mathbf{x}$ cannot be obtained from more than one codeword by precisely t' deletions, t'' insertions and at most s substitutions. Clearly, any q-ary word $\mathbf{x}'$ of different length than $\mathbf{x}$ cannot be reached by a codeword in $\mathcal{C}$ in such a way. In order to keep track of the variables in the following paragraph, we refer to Figure 2.1 for a schematic representation.

To this end, suppose for contradiction that there exist $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$ such that $\mathbf{x}$ can be reached by both $\mathbf{c}_1$ and $\mathbf{c}_2$ using precisely t' deletions, t'' insertions and at most s substitutions. By Lemma 2.1, this implies that $\mathbf{x} \in \mathcal{D}_{t'}(\mathcal{I}_{t''}(\mathcal{S}_s(\mathbf{c}_1)))$ and $\mathbf{x} \in \mathcal{D}_{t'}(\mathcal{I}_{t''}(\mathcal{S}_s(\mathbf{c}_2)))$. Thus there exist $\mathbf{u}_1 \in \mathcal{S}_s(\mathbf{c}_1)$, $\mathbf{u}_2 \in \mathcal{S}_s(\mathbf{c}_2)$ and $\mathbf{v}_1 \in \mathcal{I}_{t''}(\mathbf{u}_1)$, $\mathbf{v}_2 \in \mathcal{I}_{t''}(\mathbf{u}_2)$ such that $\mathbf{x} \in \mathcal{D}_{t'}(\mathbf{v}_1)$ and $\mathbf{x} \in \mathcal{D}_{t'}(\mathbf{v}_2)$. Then by Lemma 2.3 it follows that there exists a $\mathbf{y} \in \mathcal{B}_q(n + t' + t'')$ such that $\mathbf{y} \in \mathcal{I}_{t'}(\mathbf{v}_1)$ and $\mathbf{y} \in \mathcal{I}_{t'}(\mathbf{v}_2)$. Thus we have the relations $\mathbf{y} \in \mathcal{I}_{t'}(\mathbf{v}_1)$ and $\mathbf{v}_1 \in \mathcal{I}_{t''}(\mathbf{u}_1)$ which combine to $\mathbf{y} \in \mathcal{I}_{t'+t''}(\mathbf{u}_1)$. Analogously, it also holds that $\mathbf{y} \in \mathcal{I}_{t'+t''}(\mathbf{u}_2)$. If we invoke the relations $\mathbf{u}_1 \in \mathcal{S}_s(\mathbf{c}_1)$ and $\mathbf{u}_2 \in \mathcal{S}_s(\mathbf{c}_2)$ and Lemma 2.1 again, we find that $\mathbf{y} \in \mathcal{V}_{0,t'+t'',s}(\mathbf{c}_1) \cap \mathcal{V}_{0,t'+t'',s}(\mathbf{c}_2)$. As a last step, insert $t - (t'' - t') \geq 0$ times the zero symbol in front of $\mathbf{y}$ to obtain the word $\mathbf{z}$. Then, it holds that $\mathbf{z} \in \mathcal{V}_{0,t,s}(\mathbf{c}_1) \cap \mathcal{V}_{0,t,s}(\mathbf{c}_2)$, but this contradicts the first assumption that this intersection is empty. This proves the last implication. □

From the previous lemma we observe the following when we set $s = 0$. In that case, we consider t-indel correcting codes. Note that the third statement intuitively states that $\mathcal{C}$ can correct t deletions, and the fourth statement that $\mathcal{C}$ can correct t insertions. In other words, this proves again a fundamental property from Levenshtein [7] that a (binary) code capable of correcting only deletions is also able to correct equally many insertions. In fact, Lemma 2.4 is slightly more general because it considers codes over a q-ary alphabet instead of binary codes, and it also includes substitutions.

The previous lemma provides a concrete tool for checking that a set $\mathcal{C} \subseteq \mathcal{B}_q(n)$ is a t-indel s-substitution correcting code.

Example 2.3. Recall the repetition code $\mathcal{C}_{n,q} = \{(a)^n : a \in \mathcal{B}_q\}$ from Subsection 1.6.1. For instance, $\mathcal{C}_{4,5}$ is given by $\{0000, 1111, 2222, 3333, 4444\}$. Let $\mathbf{c}_i$ in $\mathcal{C}_{4,5}$ be the codeword that consists solely of the symbol $i \in \mathcal{B}_5$, then it holds that $\mathcal{D}_3(\mathbf{c}_i) = \{i\}$. Hence, Lemma 2.4 shows that $\mathcal{C}_{4,5}$ is a 3-indel correcting code.

Moreover, note that $\mathcal{V}_{1,0,1}(\mathbf{c}_i)$ consists of all words of length 3 that contain at least twice the symbol i. As a result, $\mathcal{V}_{1,0,1}(\mathbf{c}_i)$ and $\mathcal{V}_{1,0,1}(\mathbf{c}_j)$ are disjoint whenever $i \neq j$. Again using Lemma 2.4 shows that $\mathcal{C}_{4,5}$ is also a single-indel single-substitution correcting code.

Observe that each substitution can be obtained by two indels; a deletion followed by an insertion. Therefore, a code that can correct two indels can also correct one substitution. For this reason, after we established that $\mathcal{C}_{4,5}$ is 3-indel correcting code in the previous example, we could have directly concluded that $\mathcal{C}_{4,5}$ is also a single-indel single-substitution correcting code. This idea is captured in the following lemma.

Lemma 2.5 ([13], Section 1)**.** *Let $n \geq 1$, $q \geq 2$, $t \geq 0$ and $s \geq 0$ be integers such that $t+2s \leq n$. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be a $(t+2s)$-indel correcting code. Then, it holds that $\mathcal{C}$ is also a t-indel s-substitution correcting code. As a consequence, $M_q(n, t+2s, 0) \leq M_q(n, t, s)$.*

Proof. According to Lemma 2.4, the code $\mathcal{C}$ satisfies $\mathcal{V}_{t+s,s,0}(\mathbf{c}) \cap \mathcal{V}_{t+s,s,0}(\mathbf{c}') = \emptyset$ for all distinct $\mathbf{c}, \mathbf{c}' \in \mathcal{C}$, because $\mathcal{C}$ be a $(t+2s)$-indel correcting code. Using the observation that a substitution can be obtained by a deletion followed by an insertion, it is immediate that $\mathcal{V}_{t,0,s}(\mathbf{x}) \subseteq \mathcal{V}_{t+s,s,0}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{B}_q(n)$. This implies that $\mathcal{V}_{t,0,s}(\mathbf{c}) \cap \mathcal{V}_{t,0,s}(\mathbf{c}') = \emptyset$ for all distinct $\mathbf{c}, \mathbf{c}' \in \mathcal{C}$. Therefore, we find that $\mathcal{C}$ is a t-indel s-substitution correcting code, using again Lemma 2.4.

Let $\mathcal{C}' \subseteq \mathcal{B}_q(n)$ be a $(t+2s)$-indel correcting code of maximal size. Then, we have $|\mathcal{C}'| = M_q(n, t+2s, 0)$ and that $\mathcal{C}'$ is also t-indel s-substitution correcting code by the observations in the previous paragraph. Obviously, the size of $\mathcal{C}'$ is at most $M_q(n, t, s)$. Therefore, it holds that $M_q(n, t+2s, 0) = |\mathcal{C}'| \leq M_q(n, t, s)$. □

In contrast, the converse statement of Lemma 2.5 does not hold. This is not surprising, since there are many words that can be obtained from one deletion and one insertion, but not from one substitution. Therefore, it is often more difficult to correct two arbitrary indels than one arbitrary substitution. This is shown in the following counterexample.

Counterexample 2.4. Consider the binary code $\mathcal{C} = \{010, 101\} \subseteq \mathcal{B}_2(3)$. It holds that

$$\begin{aligned} \mathcal{S}_1(010) \cap \mathcal{S}_1(101) &= \{010, 110, 000, 011\} \cap \{101, 001, 111, 100\} = \emptyset, \\ \mathcal{V}_{1,1,0}(010) \cap \mathcal{V}_{1,1,0}(101) &= \{000, 001, 010, 100, 011, 101, 110\} \cap \{001, 010, 100, 011, 101, 110, 111\} \\ &= \{001, 010, 100, 011, 101, 110\}. \end{aligned}$$

This shows that $\mathcal{C}$ is a 1-substitution correcting code by Lemma 2.4, because the first intersection is empty. On the other hand, $\mathcal{C}$ is no 2-indel correcting code, because $\mathcal{V}_{1,1,0}(010) \cap \mathcal{V}_{1,1,0}(101)$ is clearly non-empty.

This counterexample illustrates that the condition to be a 2-indel correcting code is more stringent than the condition to be a 1-substitution correcting code. As a result, there are instances in which a 2-indel correcting code of maximal size is strictly smaller than a 1-substitution correcting code of maximal size, as the following example shows.

Example 2.5. In this example, we show that $M_2(15, 2, 0) \ll M_2(15, 0, 1)$. In other words, we provide an instance in which 2-indel correcting codes are far from optimal within the class of 1-substitution correcting codes.

Recall the binary Hamming code $Ham(15, 11)$ from Subsection 1.6.2 consisting of 2^{11} words of length 15. We have shown that $Ham(15, 11)$ is a single-substitution correcting code. Obviously, $Ham(15, 11)$ is at most as large as the largest binary single-substitution correcting code with codewords of length 15. As a result, it holds that[3] $M_2(15, 0, 1) \geq 2^{11} = 2048$.

On the other hand, we mention already the following upper bound on the maximum size of a t-indel correcting code

$$M_2(n, t, 0) \leq \frac{2^{n+t}}{\sum_{i=0}^{t} \binom{n+t}{i}},$$

which will be discussed in Lemma 5.4. In particular, it holds for binary 2-indel correcting codes that $M_2(15, 2, 0) \leq \frac{2^{17}}{1+17+136} \approx 851.12$. This implies that any 2-indel correcting code contains at most 851 words, since a code size is necessarily integer-valued. It follows that $M_2(15, 2, 0) \leq 851$. All in all, it holds clearly that $M_2(15, 2, 0) \ll M_2(15, 0, 1)$.

This example illustrates that we should not limit the search for a 1-substitution correcting code of maximum size to the class of 2-indel correcting codes. For the same reason, we should not limit the search for an optimal t-indel s-substitution correcting code to the class of $(t + 2s)$-indel correcting codes, whenever $s \geq 1$. For the purpose of this thesis, it motivates why it makes sense to consider bounds on $M_q(n, t, s)$ and not restrict our attention to (existing) bounds on $M_q(n, t + 2s, 0)$.

2.2 Relation between $\mathcal{V}_{t',t'',s}(\mathbf{x})$ and the number of runs of $\mathbf{x}$

In this section we explore the number of runs in an arbitrary word $\mathbf{x} \in \mathcal{B}_q(n)$ in relation with the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$. Recall that a run in $\mathbf{x}$ is a subsequence with identical and consecutive symbols that is not contained within a longer such run. For example, the word 211130 consists of the four runs 2, 111, 3 and 0. The subsequence 11 is not considered a run, because it is contained within the longer subsequence 111.

Next, we answer a question that will repeatedly arise in this thesis. Given that $\mathbf{x} \in \mathcal{B}_q(n)$ has $r(\mathbf{x})$ runs, how does the number of runs change after applying a certain number of deletions, insertions and/or substitutions. Intuitively, we show that $r(\mathbf{y})$ cannot be very different from $r(\mathbf{x})$ when $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{x})$ and t', t'' and s are small. More specifically, we bound the difference between $r(\mathbf{x})$ and $r(\mathbf{y})$ for $\mathbf{x} \in \mathcal{B}_q(n)$ and $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{x})$ in terms of t', t'' and s.

The following lemma is a generalization of [11, Claim 1] in which the statement was made for a single deletion and s substitutions only. However, neither a proof nor a reference for this claim was provided in [11]. Here, we generalize their claim to an arbitrary number of t' deletions, t'' insertions and s substitutions. A proof of this more general statement can be found in Appendix A.1.

[3] In fact, $M_2(15, 0, 1) = 2048$ which follows from the Hamming bound in Lemma 5.3. Namely, it states that $M_2(15, 0, 1) \leq \frac{2^{15}}{1+15} = 2048$. However, this is not relevant for this example.

Lemma 2.6. *Let $n \geq 1$, $q \geq 2$, $t' \geq 0$, $t'' \geq 0$ and $s \geq 0$ be integers such that $n - t' + t'' \geq 1$. Let $\mathbf{x} \in \mathcal{B}_q(n)$ and $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{x})$, then the following holds,*

$$r(\mathbf{x}) - 2(t' + s) \leq r(\mathbf{y}) \leq r(\mathbf{x}) + 2(t'' + s).$$

The bounds in this lemma are tight and cannot be improved in general. Namely, there exist words $\mathbf{x} \in \mathcal{B}_q(n)$ and $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{x})$ that attain these bounds with equality. The following example provides concrete words $\mathbf{x}$ and $\mathbf{y}$ for which this is the case.

Example 2.6. Let $q = 2$, $t, s \geq 0$, and consider the binary alternating word $\mathbf{x} = 0101 \cdots 010$ of length $2(t + s) + 1$. Clearly, it holds that $r(\mathbf{x}) = 2(t + s) + 1$, and that $\mathbf{x}$ consists of $t + s + 1$ zeros and $t + s$ ones. By deleting the first t ones from $\mathbf{x}$ followed by substituting the remaining s ones to zeros, we obtain the all-zero word $\mathbf{y} = (0)^{t+s+1}$. By construction, it follows that $\mathbf{y} \in \mathcal{V}_{t,0,s}(\mathbf{x})$. Given that $r(\mathbf{y}) = 1$, the words $\mathbf{x}$ and $\mathbf{y}$ satisfy the lower bound on $r(\mathbf{x})$ from Lemma 2.6 with equality.

By considering this example in reverse, i.e., by turning $\mathbf{y}$ into $\mathbf{x}$ using t insertions and s substitutions it follows that the upper bound from Lemma 2.6 is tight as well.

Before concluding this chapter we count the number of words in $\mathcal{B}_q(n)$ with precisely r runs. Recall that we denote the set of these words by $\mathcal{R}_{n,q}(r)$. It is obvious that each word contains at least one run and at most n. Hence, we consider the cardinality of $\mathcal{R}_{n,q}(r)$ for $1 \leq r \leq n$. The following result follows from a simple counting argument.

Lemma 2.7 ([38], Section II)**.** *Let $n \geq 1$, $q \geq 2$ and $1 \leq r \leq n$ be integers. The number of words in $\mathcal{B}_q(n)$ with precisely r runs is given by*

$$|\mathcal{R}_{n,q}(r)| = q\binom{n-1}{r-1}(q-1)^{r-1}.$$

Proof. For words of length n, the first position always starts a new run and from the remaining $n-1$ positions, $r-1$ positions are chosen where a new run can start. Therefore, there are $\binom{n-1}{r-1}$ configurations that indicate where new runs start. Moreover, note that the symbols of a run may not be equal to that of the previous run, as otherwise the runs would form one single run together. Hence, the first run can take all q possible symbols, whereas the remaining runs have only $q - 1$ options. It follows from these observations that $|\mathcal{R}_{n,q}(r)| = q\binom{n-1}{r-1}(q-1)^{r-1}$. □

In Figure 2.2 the number of q-ary words with length 25 and r runs is depicted on a logarithmic scale for each $1 \leq r \leq 25$. For $q = 2$, we observe a symmetry of the graph around $r = 13$. More specifically, we observe that there are equally many binary words with r runs as with $n - r + 1$ runs, where $1 \leq r \leq n$ and $n = 25$. Although this follows directly from the relation $\binom{n-1}{r-1} = \binom{n-1}{n-r}$, it can also intuitively be explained by the following observations. Note that if we know the positions of the runs within a binary word $\mathbf{x}$, then there are only two options for $\mathbf{x}$. By additionally specifying the value of $x_1 \in \mathcal{B}_2$ only one option for $\mathbf{x}$ remains. For instance, suppose positions 1, 2, 3 and 4, 5 and 6, 7 form the runs in a word $\mathbf{x}$, then either $\mathbf{x} = 0001100$ or 1110011. Based on the value of x_1 the specific word is fixed. The positions of the runs can either be specified using the $r - 1$ out of the $n - 1$ positions where a run can start, or the $n - r$ positions out of $n - 1$ that do not start a run. Here, we ignore the first position in a word that

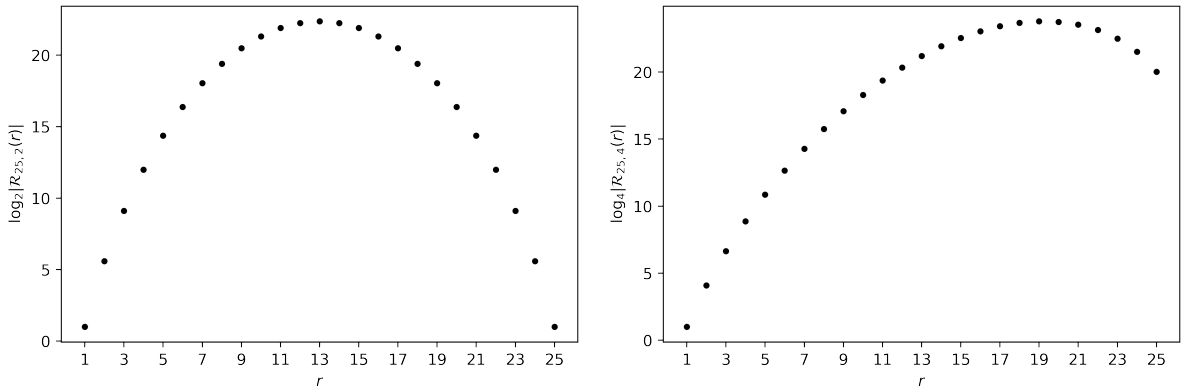


Figure 2.2: The number of words of length $n = 25$ with $1 \leq r \leq n$ runs is shown logarithmically for alphabet sizes $q = 2$ (left) and $q = 4$ (right).

necessarily starts a run. This explains why there are equally many binary words with r runs as with $n - r + 1$ runs.

This is certainly not the case for non-binary words. In Figure 2.2, this can be observed for $q = 4$ because the graph is no longer symmetric around $r = 13$. When $q \geq 3$, it does not suffice to know the positions of the runs and the value of the first run in order to determine which word is considered. Knowing the value of a run does not determine the value of a neighbouring run, because there are $q - 1 \geq 2$ options for the value of the neighbouring run.

3

Cardinality of the deletion, insertion and substitution set

This section sets out to discuss the cardinality of $\mathcal{V}_{t',t'',s}(\mathbf{x})$. For an arbitrary word $\mathbf{x} \in \mathcal{B}_q(n)$ and general parameters t', t'' and s, counting the number of elements in the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ is a non-trivial task. As recent as 2019, Abu-Sini and Yaakobi [39] stated the following about this task: "to the best of our knowledge, finding the size of the ball $\mathcal{V}_{t',t'',s}(\mathbf{x})$ has not been studied before and it is a challenging problem by itself". Only for highly specific sets of parameters or words the cardinality of the deletion, insertion and substitution set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ is known. In the other cases we must rely on bounds.

Similar to the previous chapter, this chapter is not meant to answer the research question from Section 1.4 directly. The results in this chapter will be useful in subsequent chapters where we aim to answer this question.

3.1 Cardinality of $\mathcal{S}_s(\mathbf{x})$ and $\mathcal{I}_t(\mathbf{x})$

By only considering substitutions, we first restrict our attention to the set $\mathcal{S}_s(\mathbf{x}) = \mathcal{V}_{0,0,s}(\mathbf{x})$ for general $s \geq 0$. It is rather straightforward to count the number of elements in this set for any $\mathbf{x} \in \mathcal{B}_q(n)$. Indeed, for a word $\mathbf{x} \in \mathcal{B}_q(n)$ of length n there are $\binom{n}{i}$ ways in which the positions for precisely $i \in \{1, ..., n\}$ substitutions can be chosen. Each substitution can turn a symbol into one of $q-1$ different symbols. It follows that there are $\binom{n}{i}(q-i)^i$ distinct words in $\mathcal{B}_q(n)$ that can be obtained from $\mathbf{x}$ by means of precisely i substitutions. Hence, the number of words that can be obtained by substituting at most s symbols is given by

$$|\mathcal{S}_s(\mathbf{x})| = |\mathcal{V}_{0,0,s}(\mathbf{x})| = \sum_{i=0}^{s} \binom{n}{i} (q-1)^i, \tag{3.1}$$

for each $\mathbf{x} \in \mathcal{B}_q(n)$. Notice that $|\mathcal{S}_s(\mathbf{x})|$ depends on $\mathbf{x}$ only through the parameters n and q and not on the structure of the word $\mathbf{x}$. In other words, the cardinality of $\mathcal{S}_s(\mathbf{x})$ is equal for all $\mathbf{x} \in \mathcal{B}_q(n)$.

For insertions solely, the cardinality of the set $\mathcal{I}_t(\mathbf{x})$ is less obvious to determine than for substitutions, but still an explicit formula can be found. According to [35] and [38], the following lemma was first proven by Levenshtein in [40]. Since the author has not been able to retrieve this Russian article, we refer to a brief proof from [41]. We present a more detailed version of this proof in Appendix A.2.

Lemma 3.1 ([40])**.** *For integers $n \geq 1$, $q \geq 2$ and $t \geq 0$ the cardinality of the set $\mathcal{I}_t(\mathbf{x})$ is given by*

$$|\mathcal{I}_t(\mathbf{x})| = |\mathcal{V}_{0,t,0}(\mathbf{x})| = \sum_{i=0}^{t} \binom{n+t}{i} (q-1)^i,$$

for each $\mathbf{x} \in \mathcal{B}_q(n)$.

The previous lemma shows that the cardinality of $\mathcal{I}_t(\mathbf{x})$ is equal for all $\mathbf{x} \in \mathcal{B}_q(n)$. We have also observed that the same property holds for the set $\mathcal{S}_s(\mathbf{x})$. This suggests that this might also be true for the set of words in $\mathcal{B}_q(n+t)$ that can be reached from $\mathbf{x}$ by t insertions and at most s substitutions, i.e., the set $\mathcal{V}_{0,t,s}(\mathbf{x})$. Surprisingly, the following example shows that is not true in general.

Counterexample 3.1. Consider the binary words $\mathbf{x} = 00 \in \mathcal{B}_2(2)$ and $\mathbf{y} = 01 \in \mathcal{B}_2(2)$. After applying one insertion and at most one substitution, we obtain

$$\mathcal{V}_{0,1,1}(\mathbf{x}) = \{000, 001, 010, 100, 011, 101, 110\},$$
$$\mathcal{V}_{0,1,1}(\mathbf{y}) = \{000, 001, 010, 100, 011, 101, 110, 111\}.$$

Note that 111 cannot be obtained from $\mathbf{x} = 00$ by one substitution and one insertion. Therefore it holds that $111 \notin \mathcal{V}_{0,1,1}(\mathbf{x})$. This example shows that the cardinality of the set $\mathcal{V}_{0,t,s}(\mathbf{z})$ depends on the word $\mathbf{z} \in \mathcal{B}_q(n)$ and is not only determined by the parameters n, q, t and s. The cardinality of $\mathcal{V}_{0,1,1}(\mathbf{z})$ for arbitrary $\mathbf{z} \in \mathcal{B}_q(n)$ will be discussed in Section 3.3.

3.2 Cardinality of $\mathcal{D}_t(\mathbf{x})$

Next, we consider the cardinality of the set $\mathcal{D}_t(\mathbf{x})$. In contrast to the cardinalities of $\mathcal{I}_t(\mathbf{x})$ and $\mathcal{S}_s(\mathbf{x})$ which are known and easy to compute for each $\mathbf{x} \in \mathcal{B}_q(n)$ and $t, s \geq 0$, the cardinality of $\mathcal{D}_t(\mathbf{x})$ is more difficult to determine. The main reason for this complexity is that $|\mathcal{D}_t(\mathbf{x})|$ depends not only on the parameters t, n and q, but also on the structure of the word $\mathbf{x}$. The following example investigates the cardinality of $\mathcal{D}_t(\mathbf{x})$ for various words $\mathbf{x} \in \mathcal{B}_q(n)$ and values of t, and it will show that $|\mathcal{D}_t(\mathbf{x})|$ indeed depends on the structure of $\mathbf{x}$.

Example 3.2. Let $t = 1$ and consider the words $\mathbf{x} = 001122$, $\mathbf{y} = 111022$ and $\mathbf{z} = 010212$ in $\mathcal{B}_3(6)$. Observe that after a single deletion it holds that

$$\begin{aligned} \mathcal{D}_1(\mathbf{x}) &= \{01122, 00122, 00112\}, \\ \mathcal{D}_1(\mathbf{y}) &= \{11022, 11122, 11102\}, \\ \mathcal{D}_1(\mathbf{z}) &= \{10212, 00212, 01212, 01012, 01020, 01022\}. \end{aligned}$$

Although the parameters t, n and q are equal, the cardinalities of $\mathcal{D}_1(\mathbf{x})$ and $\mathcal{D}_1(\mathbf{z})$ differ. Notice that each word in $\mathcal{D}_1(\mathbf{x})$ can be uniquely determined by specifying the run from which a symbol is deleted. For instance, deleting either the third or the fourth symbol from $\mathbf{x}$ both yield 00122, because the third and fourth symbol belong to the same run. It follows that $|\mathcal{D}_1(\mathbf{x})| = 3 = r(\mathbf{x})$. Analogously, we find that $|\mathcal{D}_1(\mathbf{y})| = 3 = r(\mathbf{y})$ and $|\mathcal{D}_1(\mathbf{z})| = 6 = r(\mathbf{z})$. In general, the property $|\mathcal{D}_1(\mathbf{w})| = r(\mathbf{w})$ holds for all $\mathbf{w} \in \mathcal{B}_q(n)$.

For multiple deletions, the number of runs of an arbitrary word $\mathbf{w} \in \mathcal{B}_q(n)$ is no longer sufficient to characterize $|\mathcal{D}_t(\mathbf{w})|$. Namely, $\mathbf{x}$ and $\mathbf{y}$ both have three runs, but it holds that

$$\begin{aligned} \mathcal{D}_2(\mathbf{x}) &= \{1122, 0022, 0011, 0122, 0112, 0012\}, \\ \mathcal{D}_2(\mathbf{y}) &= \{1022, 1112, 1122, 1102, 1110\}. \end{aligned}$$

Nevertheless, the number of runs may still give an indication of the size of $|\mathcal{D}_t(\mathbf{w})|$. Loosely speaking, $|\mathcal{D}_t(\mathbf{w})|$ is small if $\mathbf{w}$ has few runs, whereas it is large if $\mathbf{w}$ contains many runs. For example, it holds that $\mathcal{D}_2(000000) = \{0000\}$ and

$$\mathcal{D}_2(\mathbf{z}) = \{1022, 0102, 1021, 0112, 1212, 0101, 0021, 0121, 0012, 0122, 1012, 0022, 0212\},$$

while these words contain one and six runs, respectively.

Example 3.2 shows that knowing the parameters n, q and t is not sufficient to determine $|\mathcal{D}_t(\mathbf{x})|$ for $\mathbf{x} \in \mathcal{B}_q(n)$. Moreover, it shows that the number of runs in a word $\mathbf{x} \in \mathcal{B}_q(n)$ is related to the cardinality of $\mathcal{D}_t(\mathbf{x})$, but is also not sufficient to characterize $|\mathcal{D}_t(\mathbf{x})|$ for $t \geq 2$. In particular, it is argued that for $t = 1$ it holds that

$$|\mathcal{D}_1(\mathbf{x})| = r(\mathbf{x}) \tag{3.2}$$

for all $\mathbf{x} \in \mathcal{B}_q(n)$. This has already been established by Levenshtein in [7].

For $2 \leq t \leq 5$ the exact cardinality of $|\mathcal{D}_t(\mathbf{x})|$ has been determined by Mercier *et al.* [42], who provided an analytic expression based on the structure of a word $\mathbf{x} \in \mathcal{B}_q(n)$. In theory, this process can be used for arbitrary t, but the authors of [42] noted that this is not feasible in practice, because "the terms become increasingly complex and the size of the formulas grows exponentially with t". Therefore, we must rely on upper and lower bounds for $t \geq 6$.

All subsequent bounds are based on the idea that the number of runs of a word is highly related to the cardinality of $\mathcal{D}_t(\mathbf{x})$. This idea was also observed in Example 3.2. In 1966, Levenshtein [7] showed the following bounds,

$$\binom{r(\mathbf{x}) - t + 1}{t} \leq |\mathcal{D}_t(\mathbf{x})| \leq \binom{r(\mathbf{x}) + t - 1}{t}, \tag{3.3}$$

for all $\mathbf{x} \in \mathcal{B}_2(n)$ and $0 \leq t \leq n$. For a proof of these bounds we refer to [7]. Although Levenshtein's paper considered solely words over the binary alphabet, these bounds hold for arbitrary values of $q \geq 2$ as well according to [38]. For $t = 1$, the two bounds agree and this provides a different proof for Equation (3.2).

In 2002, Hirschberg and Regnier [43] improved Levenshtein's lower bound for all $t \geq 2$ by showing

$$\sum_{i=0}^{t} \binom{r(\mathbf{x}) - t}{i} \leq |\mathcal{D}_t(\mathbf{x})|, \tag{3.4}$$

for all $\mathbf{x} \in \mathcal{B}_q(n)$. The fact that this is an improvement for $t \geq 2$ follows from the observation

$$\binom{r(\mathbf{x}) - t + 1}{t} = \binom{r(\mathbf{x}) - t}{t} + \binom{r(\mathbf{x}) - t}{t - 1} = \sum_{i=t-1}^{t} \binom{r(\mathbf{x}) - t}{i} \leq \sum_{i=0}^{t} \binom{r(\mathbf{x}) - t}{i}.$$

Here, we used the recurrence $\binom{a}{b} = \binom{a-1}{b} + \binom{a-1}{b-1}$ for binomial coefficients, and omit the terms for which $0 \leq i \leq t - 2$ from the summation. We refer to [43] for a proof of this bound. In 2015, this lower bound was even further improved by Liron and Langberg [44]. However, their bound applies only to binary words. For this reason, we do not consider their lower bound in detail.

Lastly, we consider the cardinality of $\mathcal{D}_t(\mathbf{x})$ averaged over all $\mathbf{x} \in \mathcal{B}_q(n)$. Recall that $\mathbf{x} \in \mathcal{I}_t(\mathbf{y})$ if and only if $\mathbf{y} \in \mathcal{D}_t(\mathbf{x})$ from Lemma 2.2. Using this observation, it was shown in [38] that the average cardinality of $\mathcal{D}_t(\mathbf{x})$ is given by

$$\begin{aligned}
\frac{1}{q^n} \sum_{\mathbf{x} \in \mathcal{B}_q(n)} |\mathcal{D}_t(\mathbf{x})| &= \frac{1}{q^n} \sum_{\mathbf{y} \in \mathcal{B}_q(n-t)} |\mathcal{I}_t(\mathbf{y})| \\
&= \frac{q^{n-t}}{q^n} \sum_{i=0}^{t} \binom{(n-t)+t}{i} (q-1)^i, \\
&= \frac{1}{q^t} \sum_{i=0}^{t} \binom{n}{i} (q-1)^i,
\end{aligned} \tag{3.5}$$

where we used Lemma 3.1 for the cardinality of $|\mathcal{I}_t(\mathbf{y})|$ for all $\mathbf{y} \in \mathcal{B}_q(n-t)$.

3.3 Cardinality of $\mathcal{V}_{1,0,1}(\mathbf{x})$ and $\mathcal{V}_{0,1,1}(\mathbf{x})$

In this section, we consider the cardinalities of $\mathcal{V}_{1,0,1}(\mathbf{x})$ and $\mathcal{V}_{0,1,1}(\mathbf{x})$ for arbitrary $\mathbf{x} \in \mathcal{B}_q(n)$. In both cases, an analytic formula for the cardinality of these sets is known, but the formula for $\mathcal{V}_{0,1,1}(\mathbf{x})$ holds only for binary words. The cardinalities of these sets are useful for deriving bounds on single-indel single-substitution correcting codes.

We consider first a single deletion and a single substitution, i.e., the cardinality of the set $\mathcal{V}_{1,0,1}(\mathbf{x})$. The expression for $|\mathcal{V}_{1,0,1}(\mathbf{x})|$ for a q-ary word $\mathbf{x}$ was stated without a proof in [11, Lem. 2]. The authors referred to a proof of [39, Thrm. 2], but the statement in [39] and its proof only hold for binary words. We present a proof of this formula for general alphabet size $q \geq 2$ which extends the ideas from [39]. However, this proof is rather lengthy and does not provide many new insights that are needed for the remainder of this thesis. Hence, we have relegated the proof to Appendix A.3.

Lemma 3.2. *Let $n \geq 1$ and $q \geq 2$ be integers and $\mathbf{x} \in \mathcal{B}_q(n)$. Then, the following holds,*

$$|\mathcal{V}_{1,0,1}(\mathbf{x})| = L^{n,q}_{1,0,1}(r(\mathbf{x})) := \begin{cases} (n-1)(q-1)+1 & \text{if } r(\mathbf{x}) = 1, \\ r(\mathbf{x})\left((n-2)(q-1)-1\right)+q+2 & \text{if } r(\mathbf{x}) \geq 2. \end{cases}$$

Example 3.3. Let $\mathbf{x} = 00000$ and $\mathbf{y} = 01201$ both be words in $\mathcal{B}_3(5)$. Then, it holds that $r(\mathbf{x}) = 1$ and $r(\mathbf{y}) = 5$. The previous lemma gives that

$$\begin{aligned} |\mathcal{V}_{1,0,1}(\mathbf{x})| &= (5-1)(3-1)+1 = 9, \\ |\mathcal{V}_{1,0,1}(\mathbf{y})| &= 5((5-2)(3-1)-1)+5 = 5 \cdot 5 + 5 = 30. \end{aligned}$$

We remark that it is important to state that $q = 3$. For instance, both words $\mathbf{x}$ and $\mathbf{y}$ could also be viewed as words in $\mathcal{B}_8(5)$, i.e., with $q = 8$. In that case it holds that

$$\begin{aligned} |\mathcal{V}_{1,0,1}(\mathbf{x})| &= (5-1)(8-1)+1 = 29, \\ |\mathcal{V}_{1,0,1}(\mathbf{y})| &= 5((5-2)(8-1)-1)+10 = 5 \cdot 20 + 10 = 110. \end{aligned}$$

It is not surprising that these sets are much larger than before, since for $q = 8$ a single substitution can turn a symbol into seven (instead of two) different other symbols.

Observe that the cardinality of $\mathcal{V}_{1,0,1}(\mathbf{x})$ depends on $\mathbf{x}$ only via the parameters n, q and $r(\mathbf{x})$. For fixed parameters n and q, the expression for $|\mathcal{V}_{1,0,1}(\mathbf{x})|$ in Lemma 3.2 can be seen as a function $L^{n,q}_{1,0,1}$ in terms of $r = r(\mathbf{x})$. In subsequent chapters, it will be useful to know that $L^{n,q}_{1,0,1}$ is non-decreasing in r for $n \geq 3$ and $q \geq 2$. Indeed, we have

$$L^{n,q}_{1,0,1}(1) = (n-1)(q-1)+1 \leq 2(n-2)(q-1)+q = L^{n,q}_{1,0,1}(2).$$

Moreover, it holds that $(n-2)(q-1)-1 \geq 0$ for $n \geq 3$ and $q \geq 2$. It follows that $L^{n,q}_{1,0,1}(r) \leq L^{n,q}_{1,0,1}(r+1)$ for all $r \geq 2$.

Next, we move towards a single insertion and a single substitution. Abu-Sini and Yaakobi [39] derived a formula for $|\mathcal{V}_{0,1,1}(\mathbf{x})|$ for $\mathbf{x} \in \mathcal{B}_2(n)$. This formula holds for binary words only.

Lemma 3.3 ([36], Theorem 10)**.** *Let $n \geq 1$ be an integer and let $\mathbf{x} \in \mathcal{B}_2(n)$ be a binary word with $r = r(\mathbf{x})$ runs of lengths $l_1, l_2, \dots l_r$. The number of words that can be obtained from $\mathbf{x}$ by one insertion and one substitution is given by*

$$|\mathcal{V}_{0,1,1}(\mathbf{x})| = (n+2)^2 - 2 - \sum_{i=1}^{r} \frac{l_i(l_i+5)}{2}$$

For a proof of this lemma, we refer to [36]. For the purpose of this thesis, we are highly interested in a formula of $|\mathcal{V}_{0,1,1}(\mathbf{x})|$ that holds for $\mathbf{x} \in \mathcal{B}_q(n)$ and arbitrary $q \geq 2$. However, to the best of our knowledge, such a formula has not been stated in literature. The authors of [36] do not consider the case $q \geq 2$, and it is not clear to the author of this thesis how the proof of Lemma 3.3 can be extended to $q \geq 2$. In the following example we make the expression from Lemma 3.3 more concrete for the binary alternating word.

Example 3.4. Consider the binary alternating word $\mathbf{x} = 1010\cdots10 \in \mathcal{B}_2(n)$ for even $n \geq 2$ and $\mathbf{x} = 1010\cdots1 \in \mathcal{B}_2(n)$ for odd $n \geq 1$. In both cases, it holds that $\mathbf{x}$ has $r(\mathbf{x}) = n$ runs, which are all of length 1. It follows that

$$|\mathcal{V}_{0,1,1}(\mathbf{x})| = (n+2)^2 - 2 - \sum_{i=1}^{n} \frac{1 \cdot (1+5)}{2} = n^2 + 4n + 4 - 2 - 3n = n^2 + n + 2.$$

In contrast to the set $\mathcal{V}_{1,0,1}(\mathbf{x})$, the cardinality of the set $\mathcal{V}_{0,1,1}(\mathbf{x})$ cannot be characterized by the parameters n and $r(\mathbf{x})$ only. Additional information on the lengths of the individual runs in $\mathbf{x}$ is needed to fully determine the cardinality of this set. In subsequent chapters, we do not always know the lengths of all individual runs. Instead, we often only know the parameters n and $r = r(\mathbf{x})$ about the structure of $\mathbf{x} \in \mathcal{B}_2(n)$. Therefore, we cannot apply Lemma 3.3 in this setting.

Fortunately, we establish a work-around in the following lemma. Namely, this lemma provides a lower bound on $|\mathcal{V}_{0,1,1}(\mathbf{x})|$ in terms of n and $r = r(\mathbf{x})$. To this end, define the binary word $\mathbf{u}^1_{n,r}$ of length n consisting of r runs as follows. The binary word $\mathbf{u}^1_{n,r}$ is defined such that its first run has length $n - r + 1$ and contains only zero symbols. The remaining $r - 1$ runs of $\mathbf{u}^1_{n,r}$ are unit runs which alternate between 1 and 0. More specifically, the i-th run consists of zeros if $i \equiv 1 \mod 2$ and it consists of ones if $i \equiv 0 \mod 2$. For example, $\mathbf{u}^1_{8,4} = 00000101$. Indeed, the word $\mathbf{u}^1_{8,4}$ contains 4 runs, a first run of length $n - r + 1 = 5$ which is followed by $r - 1 = 3$ unit runs. To the best of the author's knowledge the following lemma has not been stated before in literature.

Lemma 3.4. *Let $n \geq 1$ be an integer. For all binary words $\mathbf{x} \in \mathcal{B}_2(n)$ with $r = r(\mathbf{x})$ runs, it holds that*

$$|\mathcal{V}_{0,1,1}(\mathbf{x})| \geq |\mathcal{V}_{0,1,1}(\mathbf{u}^1_{n,r})| = L^{n,2}_{0,1,1}(r) := -\frac{1}{2}r^2 + (n + \frac{1}{2})r + \frac{1}{2}(n^2 + n + 4).$$

Moreover, for fixed n, the function $L^{n,q}_{0,1,1}(r)$ is increasing in r on the set $\{1, ..., n\}$.

Proof. Firstly, we state a claim and show that this claim implies $|\mathcal{V}_{0,1,1}(\mathbf{x})| \geq |\mathcal{V}_{0,1,1}(\mathbf{u}^1_{n,r})|$. Let $\mathbf{x} \in \mathcal{B}_2(\mathbf{x})$ be an arbitrary binary word with r runs of lengths $l_1, ..., l_r$. From Lemma 3.3 it follows that $|\mathcal{V}_{1,0,1}(\mathbf{x})|$ is invariant under permutations of the runs in $\mathbf{x}$. Therefore, without loss of generality we can assume that l_1 is the longest run, i.e., $l_1 \geq l_j$ for all

$2 \leq j \leq r$. Moreover, we can also assume without loss of generality that the first run consists of zeros. Otherwise, we consider $\overline{\mathbf{x}}$ instead, and note that $|\mathcal{V}_{1,0,1}(\overline{\mathbf{x}})| = |\mathcal{V}_{1,0,1}(\mathbf{x})|$. Here $\overline{\mathbf{x}}$ denotes the binary complement of $\mathbf{x}$.

Next, we are able to state and prove the claim. Let $2 \leq j \leq r$ be an integer and $\mathbf{x}^*$ be the binary word obtained from $\mathbf{x}$ by shortening the j-th run by one and extending the first run by one, then it holds that $|\mathcal{V}_{0,1,1}(\mathbf{x})| \geq |\mathcal{V}_{0,1,1}(\mathbf{x}^*)|$. Apart from the first and the j-th run, the lengths of the other runs remain unchanged. Therefore, it holds that

$$\begin{aligned}|\mathcal{V}_{0,1,1}(\mathbf{x})| - |\mathcal{V}_{0,1,1}(\mathbf{x}^*)| &= -\frac{1}{2}l_1(l_1+5) - \frac{1}{2}l_j(l_j+5) + \frac{1}{2}(l_1+1)(l_1+6) + \frac{1}{2}(l_j-1)(l_j+4) \\ &= \frac{1}{2}(-5l_1 - 5l_j + 7l_1 + 3l_j + 2) \\ &= l_1 - l_j + 1 \geq 0,\end{aligned}$$

where we used that $l_1 \geq l_j$ and Lemma 3.3. This proves the claim. By repeatedly applying the claim to runs for which $l_j \geq 2$ and $j \geq 2$, the first inequality of the lemma follows. Indeed, the first run becomes strictly longer after each step, while the other runs become shorter. This process necessarily terminates at the word $\mathbf{u}^1_{n,r}$ after a finite number of iterations.

Secondly, we show that $|\mathcal{V}_{0,1,1}(\mathbf{u}^1_{n,r})| = L^{n,2}_{0,1,1}(r)$ by applying Lemma 3.3. Note that for $\mathbf{u}^1_{n,r}$ it holds that $l_1 = n - r + 1$ and $l_2 = l_3 = \cdots = l_r = 1$. Therefore, we find

$$\begin{aligned}|\mathcal{V}_{0,1,1}(\mathbf{u}^1_{n,r})| &= (n+2)^2 - 2 - \frac{1}{2}(n-r+1)(n-r+6) - 3(r-1) \\ &= n^2 + 4n + 2 - \frac{1}{2}(n^2 + r^2 + 6 - 2nr - 7r + 7n) - 3r - 3 \\ &= -\frac{1}{2}r^2 + (n + \frac{1}{2})r + \frac{1}{2}(n^2 + n + 4).\end{aligned}$$

For fixed n, the function $L^{n,2}_{0,1,1}(r)$ is a concave quadratic polynomial in r which is symmetric around $n + \frac{1}{2}$. Hence, it is increasing in r on the set $\{1, ..., n\}$. □

3.4 Cardinality of $\mathcal{V}_{t',t'',s}(\mathbf{x})$

As stated before, finding a general expression for cardinality of $\mathcal{V}_{t',t'',s}(\mathbf{x})$ is a challenging task. In this section, we review and derive various steps in this direction.

3.4.1 Exact results

For completeness, we mention two exact results on the cardinality of $\mathcal{V}_{t',t'',s}(\mathbf{x})$ in literature. Firstly, a general expression for the cardinality of $\mathcal{V}_{1,1,0}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{B}_q(n)$ was derived by Sala and Dodeleck in [45]. The set of words that can be reached by one deletion and one insertion from $\mathbf{x} \in \mathcal{B}_q(n)$ is of lesser importance in this thesis, since we mainly focus on the combination of indels and substitutions.

Secondly, we consider the parameters $t' = 1$, $t'' = 0$ and $s \geq 0$ and binary words $\mathbf{x} \in \mathcal{B}_2(n)$. In this setting, an exact expression for $|\mathcal{V}_{1,0,s}(\mathbf{x})|$ for all $\mathbf{x} \in \mathcal{B}_2(n)$ has been provided by Abu-Sini and Yaakobi in [36]. From this expression it is apparent that

$|\mathcal{V}_{1,0,s}(\mathbf{x})|$ only depends $\mathbf{x}$ via the parameters n and $r(\mathbf{x})$. We state that the outline of the proof of this result resembles the proof of Lemma 3.2 on the cardinality of $\mathcal{V}_{1,0,1}(\mathbf{x})$. Here, we omit a proof because it is lengthy and this result is not of great importance for the sequel of this thesis. For a detailed proof, we refer to [36].

3.4.2 Lower bound on the cardinality of $\mathcal{V}_{t,0,s}(\mathbf{x})$

In this section we derive a novel non-asymptotic lower on the cardinality of $\mathcal{V}_{t,0,s}(\mathbf{x})$ for general $t, s \geq 0$ and $\mathbf{x} \in \mathcal{B}_q(n)$. To the best of the author's knowledge this bound has not been stated before in literature and even forms the only non-trivial lower bound $|\mathcal{V}_{t,0,s}(\mathbf{x})|$ that holds for general $t, s \geq 0$.

Lemma 3.5. *Let $n \geq 1$, $0 \leq t \leq \frac{n}{2}$ and $0 \leq s \leq \frac{n}{2}$ be integers. Let $\mathbf{x} \in \mathcal{B}_q(n)$ be a word for which its number of runs $r = r(\mathbf{x})$ satisfies $\lfloor \frac{r}{2} \rfloor \geq s$ and $\lfloor \frac{r}{2} \rfloor \geq t$. Then, it holds that*

$$\sum_{i=0}^{s} \binom{\lfloor \frac{r}{2} \rfloor}{i} (q-1)^i \cdot \sum_{j=0}^{t} \binom{\lceil \frac{r}{2} \rceil - t}{j} \leq |\mathcal{V}_{t,0,s}(\mathbf{x})|.$$

Proof. For this proof, we will use the following idea. We consider only the words which are obtained by at most s substitutions in the first $\lfloor \frac{r}{2} \rfloor$ runs and precisely t deletions in the remaining runs. This is possible since, $t \leq \lfloor \frac{r}{2} \rfloor \leq n - \lceil \frac{r}{2} \rceil$. Naturally, a lower bound on the number of words constructed in this way also forms a lower bound on $|\mathcal{V}_{t,0,s}(\mathbf{x})|$.

Let $\mathbf{x}^1$ be the word which consists of the first $\lfloor \frac{r}{2} \rfloor$ runs of $\mathbf{x}$, and $\mathbf{x}^2$ be the word which consists of the last $r - \lfloor \frac{r}{2} \rfloor = \lceil \frac{r}{2} \rceil$ runs of $\mathbf{x}$. In other words, $\mathbf{x}$ is the concatenation of $\mathbf{x}^1$ and $\mathbf{x}^2$. Denote by n_1 and n_2 the lengths of $\mathbf{x}^1$ and $\mathbf{x}^2$, respectively. It holds that $n_1 \geq \lfloor \frac{r}{2} \rfloor \geq s$ and $n_2 \geq \lceil \frac{r}{2} \rceil \geq t$, since each run has a length of at least one. By concatenating a word $\mathbf{u} \in \mathcal{S}_s(\mathbf{x}^1)$ with a word $\mathbf{v} \in \mathcal{D}_t(\mathbf{x}^2)$, we obtain the word[1] $(\mathbf{u}|\mathbf{v}) \in \mathcal{V}_{t,0,s}(\mathbf{x})$. Each such distinct pair of words $\mathbf{u}, \mathbf{v}$ yields a distinct word in $\mathcal{V}_{t,0,s}(\mathbf{x})$ and thus there exist at least $|\mathcal{S}_s(\mathbf{x}^1)| \cdot |\mathcal{D}_t(\mathbf{x}^2)|$ words in $\mathcal{V}_{t,0,s}(\mathbf{x})$. As a result, we get

$$\begin{aligned}
|\mathcal{V}_{t,0,s}(\mathbf{x})| &\geq |\mathcal{S}_s(\mathbf{x}^1)| \cdot |\mathcal{D}_t(\mathbf{x}^2)| \\
&\geq \sum_{i=0}^{s} \binom{n_1}{i} (q-1)^i \cdot \sum_{j=0}^{t} \binom{n_2 - t}{j} \\
&\geq \sum_{i=0}^{s} \binom{\lfloor \frac{r}{2} \rfloor}{i} (q-1)^i \cdot \sum_{j=0}^{t} \binom{\lceil \frac{r}{2} \rceil - t}{j},
\end{aligned}$$

where we used (3.1) for the cardinality $\mathcal{S}_s(\mathbf{x}^1)$ and (3.4) as a lower bound on $|\mathcal{D}_t(\mathbf{x}^2)|$. □

The lower bound in this lemma counts only words which are obtained in a very specific way. This results in a weak bound of $|\mathcal{V}_{t,0,s}(\mathbf{x})|$. This is also illustrated by the following example.

[1] By the notation $(\mathbf{u}|\mathbf{v})$ we denote the concatenation of the words $\mathbf{u} \in \mathcal{B}_q(n_1)$ and $\mathbf{v} \in \mathcal{B}_q(n_2)$ into the concatenated word in $\mathcal{B}_q(n_1 + n_2)$. For instance, let $\mathbf{u} = 1023 \in \mathcal{B}_4(4)$ and $\mathbf{v} = 302 \in \mathcal{B}_4(3)$, then $(\mathbf{u}|\mathbf{v}) = 1023302 \in \mathcal{B}_4(7)$.

Example 3.5. Let $n = 7, q = 4$ and $t = s = 1$. For these parameters, note that the previous lemma only applies to the words in $\mathcal{B}_q(n)$ with $r \geq 2$ runs. Consider the words $\mathbf{x} = 000111$, $\mathbf{y} = 1102333$ and $\mathbf{z} = 0123012$ in $\mathcal{B}_4(7)$. These words have $r(\mathbf{x}) = 2$, $r(\mathbf{y}) = 4$ and $r(\mathbf{z}) = 7$ runs, respectively. The lower bound from Lemma 3.5 gives

$$|\mathcal{V}_{1,0,1}(\mathbf{x})| \geq \sum_{i=0}^{1} \binom{\lfloor \frac{r(\mathbf{x})}{2} \rfloor}{i} 3^i \cdot \sum_{j=0}^{1} \binom{\lceil \frac{r(\mathbf{x})}{2} \rceil - 1}{j} = (1+3) \cdot (1+1) = 8,$$

$$|\mathcal{V}_{1,0,1}(\mathbf{y})| \geq \sum_{i=0}^{1} \binom{\lfloor \frac{r(\mathbf{y})}{2} \rfloor}{i} 3^i \cdot \sum_{j=0}^{1} \binom{\lceil \frac{r(\mathbf{y})}{2} \rceil - 1}{j} = (1+6) \cdot (1+1) = 14,$$

$$|\mathcal{V}_{1,0,1}(\mathbf{z})| \geq \sum_{i=0}^{1} \binom{\lfloor \frac{r(\mathbf{z})}{2} \rfloor}{i} 3^i \cdot \sum_{j=0}^{1} \binom{\lceil \frac{r(\mathbf{z})}{2} \rceil - 1}{j} = (1+9) \cdot (1+3) = 40.$$

On the other hand, recall that Lemma 3.2 yields the exact cardinalities. This gives $|\mathcal{V}_{1,0,1}(\mathbf{x})| = r(\mathbf{x})((n-2)(q-1)-1) + q + 2 = 2 \cdot 14 + 6 = 34$. Similarly, we find that $|\mathcal{V}_{1,0,1}(\mathbf{y})| = 62$ and $|\mathcal{V}_{1,0,1}(\mathbf{z})| = 104$. This shows that for the words $\mathbf{x}$ $\mathbf{y}$ and $\mathbf{z}$, the lower bound performs rather weak.

The main purpose of this lower bound lies in the asymptotic regime. Loosely speaking, we will use this bound to show that for words with sufficiently many runs, the cardinality of $\mathcal{V}_{t,0,s}(\mathbf{x})$ is asymptotically equal to $\mathcal{O}(n^{t+s})$. This will be addressed in Section 6.8.

3.5 q-ary entropy function

In this section we introduce the q-ary entropy function and discuss its properties. This function plays an important role to bound the size of the Hamming sphere of radius r, which is given by the quantity $\sum_{i=0}^{r} \binom{n}{i}(q-1)^i$. This quantity arises often when determining the cardinality of $\mathcal{V}_{t',t'',s}(\mathbf{x})$. For instance, recall that this term occurs in the cardinalities of the sets $\mathcal{S}_s(\mathbf{x})$ and $\mathcal{I}_t(\mathbf{x})$ (after some appropriate scaling). For deriving asymptotic results it will be crucial to bound these terms using the q-ary entropy function.

Definition 3.6. Let $q \geq 2$ be an integer, then the q-ary entropy function $H_q : [0, 1-\frac{1}{q}] \to \mathbb{R}$ is defined by

$$H_q(x) = x \log_q(q-1) - x \log_q(x) - (1-x) \log_q(1-x),$$

where $H_q(0) = 0$ for continuity. Moreover, define the extended q-ary entropy function $H_q^* : [0, \infty) \to \mathbb{R}$ by

$$H_q^*(x) = H_q(\min\{x, 1 - \frac{1}{q}\}).$$

Figure 3.1 shows a plot of H_q for various parameters q. Based on this figure it is not surprising that on the interval $[0, 1-\frac{1}{q}]$ the q-ary entropy function has the following basic and useful properties. According to [6, Sec. 4.5], it holds that H_q is continuous, non-negative, $\cap$-concave, invertible and attains its minimum value of 0 at $x = 0$ and its maximum value of 1 at $x = 1 - \frac{1}{q}$.

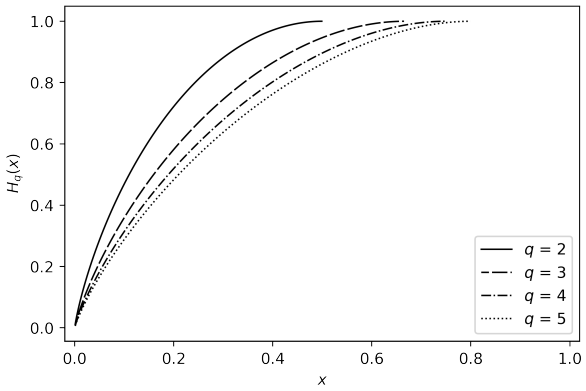


Figure 3.1: The q-ary entropy function for $2 \leq q \leq 5$.

The following lemma gives a lower and upper bound on the size of the Hamming sphere in terms of the q-ary entropy function. For a proof of these bounds we refer to [6].

Lemma 3.6 ([6], Lemma 4.7 & 4.8)**.** *Let $n \geq 1, q \geq 2$ and $r \geq 0$ be integers such that $0 \leq \frac{r}{n} \leq 1 - \frac{1}{q}$. Then the following holds,*

$$\frac{1}{n+1} q^{nH_q(\frac{r}{n})} \leq \sum_{i=0}^{r} \binom{n}{i} (q-1)^i \leq q^{nH_q(\frac{r}{n})}.$$

The bounds in this lemma can be sharpened according to [6], but for the purpose of this thesis these bounds suffice. In the asymptotic regime the following result will be useful.

Lemma 3.7 ([9], Equation (4))**.** *For each $\lambda \in (0, 1)$ the following holds,*

$$\lim_{n \to \infty} \frac{1}{n} \log_q \left(\sum_{i=0}^{\lfloor \lambda n \rfloor} \binom{n}{i} (q-1)^i \right) = H_q^*(\lambda).$$

For a proof of this result we refer to [9].

4

Lower bounds for t-indel s-substitution correcting codes

This chapter sets out to derive lower bounds on the maximal size of t-indel s-substitution correcting codes. In line with our research question, we first review two existing lower bounds on $M_q(n, 0, s)$ and $M_q(n, t, 0)$ in Section 4.1. Thereafter, these bounds are generalized to form a novel lower bound on the maximal size of t-indel s-substitution correcting codes in Section 4.2. Next, we take a closer look at the derivation of this novel bound, and identify an improvement. This leads to a second novel lower bound on $M_q(n, t, s)$ in Section 4.3. The lower bounds on $M_q(n, t, s)$ are then compared in a non-asymptotic setting in Section 4.4. Lastly, we derive two asymptotic results on $M_q(n, t, s)$ using the lower bounds from this Chapter.

4.1 Two existing lower bounds for s-substitution correcting codes and t-indel correcting codes

For s-substitution correcting codes, we consider first the Gilbert-Varshamov lower bound on $M_q(n, 0, s)$. This bound was proven by Gilbert [4] and later independently by Varshamov [5]. Here, we restate this result and prove it using the original argument from Gilbert. This proof uses a sphere-covering argument and is arguably the most common proof of this result.

Lemma 4.1 ([4,5] Gilbert-Varshamov bound)**.** *For integers $n \geq 1$, $q \geq 2$ and $0 \leq s \leq n$, the following gives a lower bound on $M_q(n, 0, s)$,*

$$M_q(n, 0, s) \geq \frac{q^n}{\sum_{i=0}^{2s} \binom{n}{i}(q-1)^i}.$$

Proof. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be an s-substitution correcting code of maximal size. For all $\mathbf{c} \in \mathcal{C}$ consider the substitution spheres $\mathcal{S}_{2s}(\mathbf{c})$. Recall that $\mathcal{S}_{2s}(\mathbf{c})$ contains the words in $\mathcal{B}_q(n)$ that can be obtained from $\mathbf{c}$ by at most $2s$ substitutions.

Observe that for each $\mathbf{x} \in \mathcal{B}_q(n)$ there exists a codeword $\mathbf{c} \in \mathcal{C}$ with $\mathbf{x} \in \mathcal{S}_{2s}(\mathbf{c})$ and equivalently $\mathbf{c} \in \mathcal{S}_{2s}(\mathbf{x})$. In order to prove this statement, assume by contradiction that there exists some $\mathbf{x} \in \mathcal{B}_q(n)$ such that $\mathcal{S}_{2s}(\mathbf{x}) \cap \mathcal{C} = \emptyset$. This implies that $\mathbf{c} \notin \mathcal{S}_{2s}(\mathbf{x})$ for all $\mathbf{c} \in \mathcal{C}$. Therefore, $\mathcal{C} \cup \{\mathbf{x}\}$ is also an s-substitution correcting code. Namely, this follows directly from Lemma 2.4 which states that $\mathcal{C} \cup \{\mathbf{x}\}$ is an s-substitution correcting code if and only if $\mathbf{c}_1 \notin \mathcal{S}_{2s}(\mathbf{c}_2)$ for all $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C} \cup \{\mathbf{x}\}$. However, this contradicts the maximality of $\mathcal{C}$ and proves the statement that for each $\mathbf{x} \in \mathcal{B}_q(n)$, there exists a codeword $\mathbf{c} \in C$ with $\mathbf{x} \in \mathcal{S}_{2s}(\mathbf{c})$.

This statement implies that each $\mathbf{x} \in \mathcal{B}_q(n)$ is covered by some $\mathcal{S}_{2s}(\mathbf{c})$ for at least one $\mathbf{c} \in \mathcal{C}$. In other words, the union of sets $\mathcal{S}_{2s}(\mathbf{c})$ over all $\mathbf{c} \in \mathcal{C}$ is a subset of $\mathcal{B}_q(n)$. This union is necessarily also a subset of $\mathcal{B}_q(n)$ and thus they must be equal. It follows that

$$q^n = |\mathcal{B}_q(n)| = \left| \bigcup_{\mathbf{c} \in C} \mathcal{S}_{2s}(\mathbf{c}) \right| \leq \sum_{\mathbf{c} \in C} |\mathcal{S}_{2s}(\mathbf{c})| = |\mathcal{C}| \sum_{i=0}^{2s} \binom{n}{i}(q-1)^i. \tag{4.1}$$

Here, we used that $|\mathcal{S}_{2s}(\mathbf{c})| = \sum_{i=0}^{2s} \binom{n}{i}(q-1)^i$ for all $\mathbf{c} \in \mathcal{C}$, (cf. (3.1)). By noting that $\mathcal{C}$ is maximal, i.e., $|\mathcal{C}| = M_q(n, 0, s)$, and by rewriting (4.1), the result of this lemma follows. □

This sphere-covering proof of the Gilbert-Varshamov bound is an existence proof. It does not suggest how to actually construct an s-substitution correcting code with a size that meets (or exceeds) the lower bound. The Gilbert-Varshamov bound does not always provide a strong lower bound, as illustrated by the following example.

Example 4.1. In this example, we are interested in bounding $M_2(15, 0, 1)$ from below. The aforementioned Gilbert-Varshamov lower bound gives

$$M_2(15, 0, 1) \geq \frac{2^{15}}{\sum_{i=0}^{2} \binom{15}{i}(2-1)^i} = \frac{32768}{1 + 15 + 105} \approx 270.8.$$

This implies that there exists a single-substitution correcting code of size at least 271, because a code size is necessarily integer-valued. However, recall the binary single-substitution Hamming code with words of length 15 from Subsection 1.6.2. We established that $Ham(15, 11)$ has a cardinality of $2^{15-11} = 2048$. In other words, this code construction shows that $M_2(15, 0, 1) \geq 2048$, which greatly exceeds the lower bound of 271 by Gilbert and Varshamov.

In the proof of the Gilbert-Varshamov bound the set $\mathcal{B}_q(n)$ is covered with the spheres $\mathcal{S}_{2s}(\mathbf{c})$ centered around the codewords $\mathbf{c} \in \mathcal{C}$. A natural question is whether a similar argument can be used to derive a lower bound for t-indel correcting codes. Recall that a code $\mathcal{C}$ is a t-indel correcting code if and only if $\mathbf{c}_1 \notin \mathcal{V}_{t,t,0}(\mathbf{c}_2)$ for all $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$. Therefore we may ask, is possible to cover $\mathcal{B}_q(n)$ with the sets $\mathcal{V}_{t,t,0}(\mathbf{c})$ and deduce a bound in the same way?

The short answer to this question is no. However, it is possible to adapt the argument in order to derive a bound. Namely, we have implicitly used in (4.1) of Lemma 4.1 that the sets $\mathcal{S}_{2s}(\mathbf{c})$ are all of equal size. As we have seen in Chapter 3, this is not the case for the sets $\mathcal{V}_{t,t,0}(\mathbf{c})$. Therefore, the argument in the proof of Lemma 4.1 cannot be repeated for t-indel correcting codes. Fortunately, Tolhuizen [46] provided an alternative strategy to prove the Gilbert-Varshamov bound of Lemma 4.1 that does not rely on the fact that the sets $\mathcal{S}_{2s}(\mathbf{c})$ are of equal size. This allowed Levenshtein [38] to apply Tolhuizen's strategy to the sets $\mathcal{V}_{t,t,0}(\mathbf{c})$ in order to derive a lower bound on $M_q(n, t, 0)$. In [38], only a brief outline of the proof has been provided. We give a detailed proof below based on this outline and the strategy by Tolhuizen.

Lemma 4.2 ([38], Theorem 1)**.** *For integers $n \geq 1$, $q \geq 2$ and $0 \leq t \leq n$, the following gives a lower bound on $M_q(n, t, 0)$,*

$$M_q(n, t, 0) \geq \frac{q^{n+t}}{\left(\sum_{i=0}^{t} \binom{n}{i}(q-1)^i\right)^2}.$$

Proof. This proof is divided into two major steps. First, we aim to apply the main result from Tolhuizen [46] in order to show that

$$M_q(n, t, 0) \geq \frac{q^n}{\frac{1}{q^n}\sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,t,0}(\mathbf{x})|}. \tag{4.2}$$

Secondly, we show that the denominator in this expression is upper bounded by

$$q^{-t}\left(\sum_{i=0}^{t} \binom{n}{i}(q-1)^i\right)^2.$$

Together, these steps show the desired result. For the first step, let $(\mathcal{X}, \rho)$ be a metric space and $d \geq 1$ an integer. The main result of [46] states that there exists a subset (i.e., a code) $\mathcal{C} \subseteq \mathcal{X}$ that satisfies $\rho(c, c') \geq d$ for all distinct $c, c' \in \mathcal{C}$ and $|\mathcal{C}|V_{d-1}^{avr} \geq |\mathcal{X}|$. Here, V_{d-1}^{avr} denotes the average size of the spheres $V_{d-1}(x) = \{y \in \mathcal{X} : \rho(x, y) \leq d-1\}$ over all $x \in \mathcal{X}$. For a proof of this result, we refer to [46]. In Lemma 4.3 we will prove a similar statement using a graph-theoretical argument. Hence, we omit a proof here.

Next, we apply this result to the context of t-indel correcting codes. Let $\mathcal{X}$ be given by $\mathcal{B}_q(n)$ and ρ denote the deletion/insertion metric, which equals the minimum number of deletions and insertions needed to translate $\mathbf{x}$ into $\mathbf{y}$. Given that $\mathcal{X} = \mathcal{B}_q(n)$ we will write $\mathbf{x} \in \mathcal{X}$ instead of $x \in \mathcal{X}$, et cetera. In Section I of [7], Levenshtein showed that ρ is indeed a metric and that $\mathcal{C}$ is a t-indel correcting code if and only if $\rho(\mathbf{c}, \mathbf{c}') \geq 2t+1$ for distinct codewords $\mathbf{c}, \mathbf{c}' \in \mathcal{C}$. All together, this implies that there exists a t-indel correcting code $\mathcal{C} \subseteq \mathcal{X} = \mathcal{B}_q(n)$ for which

$$|\mathcal{C}| \geq \frac{|\mathcal{X}|}{V_{2t}^{avr}(\mathbf{x})} = \frac{|\mathcal{X}|}{\frac{1}{|\mathcal{X}|}\sum_{\mathbf{x}\in\mathcal{X}} |V_{2t}(\mathbf{x})|} = \frac{q^n}{\frac{1}{q^n}\sum_{\mathbf{x}\in\mathcal{B}_q(n)} |V_{2t}(\mathbf{x})|}.$$

Naturally, this t-indel correcting code $\mathcal{C}$ satisfies $|\mathcal{C}| \leq M_q(n,t,0)$. Hence, in order to conclude the first step we will show that $V_{2t}(\mathbf{x}) = \mathcal{V}_{t,t,0}(\mathbf{x})$.

To this end, observe that for $\mathbf{x}, \mathbf{y} \in \mathcal{B}_q(n)$ the following statements are equivalent; it holds that $\rho(\mathbf{x}, \mathbf{y}) \leq 2t$ if and only if $\mathbf{y} \in \mathcal{V}_{t,t,0}(\mathbf{x})$. Indeed, suppose that $\mathbf{y} \in \mathcal{V}_{t,t,0}(\mathbf{x})$ then it follows directly from the definitions of ρ and $\mathcal{V}_{t,t,0}(\mathbf{x})$ that $\rho(\mathbf{x}, \mathbf{y}) \leq 2t$. Conversely, suppose that $\rho(\mathbf{x}, \mathbf{y}) \leq 2t$. For the words $\mathbf{x}$ and $\mathbf{y}$ of the same length, the minimal number of deletions and insertions that turn $\mathbf{x}$ into $\mathbf{y}$ must contain an equal number of t' deletions and t' insertions, where $t' \leq t$. Hence, $\mathbf{x}$ can be turned into $\mathbf{y}$ by t' deletions and t' insertions. Clearly, $\mathbf{x}$ can also be turned into $\mathbf{y}$ by t deletions and t insertions. This can be done inserting $t - t'$ zeros in front of $\mathbf{x}$ and deleting these zeros again, followed by turning $\mathbf{x}$ into $\mathbf{y}$ by t' deletions and t' insertions. This shows the equivalence. By the definition of the set $V_{2t}(\mathbf{x})$, this equivalence implies that $V_{2t}(\mathbf{x}) = \mathcal{V}_{t,t,0}(\mathbf{x})$. This concludes the first step and shows (4.2).

For the second step, note that each $\mathbf{y} \in \mathcal{V}_{t,t,0}(\mathbf{x})$ can be reached from $\mathbf{x}$ by first deleting t symbols from $\mathbf{x}$ followed by inserting t symbols according to Lemma 2.1. Therefore, it holds for all $\mathbf{x} \in \mathcal{B}_q(n)$ that

$$|\mathcal{V}_{t,t,0}(\mathbf{x})| \leq \sum_{\mathbf{y}\in\mathcal{D}_t(\mathbf{x})} |\mathcal{I}_t(\mathbf{y})|. \tag{4.3}$$

Recall that the cardinality of $\mathcal{I}_t(\mathbf{y})$ is given by

$$|\mathcal{I}_t(\mathbf{y})| = \sum_{i=0}^{t} \binom{(n-t)+t}{i} (q-1)^i, \tag{4.4}$$

for all $\mathbf{y} \in \mathcal{B}_q(n-t)$ according to Lemma 3.1. Moreover, the cardinality of $\mathcal{D}_t(\mathbf{x})$ averaged over all $\mathbf{x} \in \mathcal{B}_q(n)$ was given in (3.5) by

$$q^{-n} \sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{D}_t(\mathbf{x})| = q^{-t} \sum_{i=0}^{t} \binom{n}{i} (q-1)^i. \tag{4.5}$$

This allows us to derive the following bound on the average of $|\mathcal{V}_{t,t,0}(\mathbf{x})|$ over all $\mathbf{x} \in \mathcal{B}_q(n)$,

$$\begin{aligned} q^{-n} \sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,t,0}(\mathbf{x})| &\overset{(4.3)}{\leq} q^{-n} \sum_{\mathbf{x}\in\mathcal{B}_q(n)} \sum_{\mathbf{y}\in\mathcal{D}_t(\mathbf{x})} |\mathcal{I}_t(\mathbf{y})| \\ &\overset{(4.4)}{=} q^{-n} \sum_{\mathbf{x}\in\mathcal{B}_q(n)} \sum_{\mathbf{y}\in\mathcal{D}_t(\mathbf{x})} \sum_{i=0}^{t} \binom{n}{i} (q-1)^i \\ &= q^{-n} \sum_{i=0}^{t} \binom{n}{i} (q-1)^i \sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{D}_t(\mathbf{x})| \\ &\overset{(4.5)}{=} q^{-t} \left(\sum_{i=0}^{t} \binom{n}{i} (q-1)^i \right)^2. \end{aligned}$$

In the third step we used that the expression $\sum_{i=0}^{t} \binom{n}{i}(q-1)^i$ does not depend on both $\mathbf{x}$ and $\mathbf{y}$. Moreover, we used that $\sum_{\mathbf{y}\in\mathcal{D}_t(\mathbf{x})} 1 = |\mathcal{D}_t(\mathbf{x})|$. This proves the bound from the second step and finalizes the over-all proof. □

Example 4.2. In this example we bound $M_3(11, 1, 0)$ from below. The previous lower bound by Levenshtein gives

$$M_3(11, 1, 0) \geq \frac{3^{11+1}}{(\binom{11}{0}(3-1)^0 + \binom{11}{1}(3-1)^1)^2} = \frac{3^{12}}{23^2} \approx 1004.61.$$

Therefore, there exists a single-indel correcting code $\mathcal{C} \subseteq \mathcal{B}_3(11)$ of size at least 1005. On the other hand, recall from Tenengolts' construction of single-indel correcting codes in Subsection 1.6.3 that there exists such a code of size $\lceil \frac{q^n}{q \cdot n} \rceil = \lceil \frac{3^{11}}{3 \cdot 11} \rceil = 5369$. This implies that $M_3(11, 1, 9) \geq 5369$. Hence, Levenshtein's bound is rather weak for this specific set of parameters.

Remark 4.3. It is not hard to adapt the proof of Lemma 4.2 to the setting of s-substitution correcting codes. Namely, consider the metric space $(\mathcal{X}, d)$ where $\mathcal{X} = \mathcal{B}_q(n)$ and d denotes the Hamming distance function. In the context of the first step in the proof of Lemma 4.2, let $V_{2s}(\mathbf{x}) := \{\mathbf{y} \in \mathcal{X} : d(\mathbf{x}, \mathbf{y}) \leq 2s\}$. Hence, $V_{2s}(\mathbf{x})$ consists of all words in $\mathcal{B}_q(n)$ that differ in at most $2s$ positions from $\mathbf{x}$. This means they can be reached from $\mathbf{x}$ by at most $2s$ substitutions. Therefore, it holds that $V_{2s}(\mathbf{x}) = \mathcal{S}_{2s}(\mathbf{x})$ and we find that

$$M_q(n, 0, s) \geq \frac{|\mathcal{X}|}{V_{2s}^{avg}(\mathbf{x})} = \frac{q^n}{\frac{1}{q^n}\sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{S}_{2s}(\mathbf{x})|} = \frac{q^n}{\sum_{i=0}^{2s} \binom{n}{i}(q-1)^i}.$$

Here, we used that the sets $\mathcal{S}_{2s}(\mathbf{x})$ have equal size for all $\mathbf{x} \in \mathcal{B}_q(n)$ according to (3.1). For brevity, we do not consider all details and refer to [46] for a detailed proof. The main purpose of this remark is to convince the reader that this strategy (with more details) leads to the an alternative proof of the Gilbert-Varshamov bound.

To summarize, we proved the Gilbert-Varshamov bound for s-substitution correcting codes using the original sphere-covering argument. Based on the fact that the sets

$\mathcal{V}_{t,t,0}(\mathbf{x})$ are not of equal size for all $\mathbf{x} \in \mathcal{B}_q(n)$, we established that this sphere-covering argument does not easily extend to t-indel correcting codes. For this reason, we reviewed a different strategy by Tolhuizen and Levenshtein. This strategy leads to the lower bound from Lemma 4.2 for t-indel correcting codes as well as the Gilbert-Varshamov bound for s-substitution correcting codes.

These similarities raise the natural question whether this strategy can be generalized to obtain a lower bound for t-indel s-substitution correcting codes as well. In the following section this question will be addressed.

4.2 Gilbert-Varshamov inspired lower bound on $M_q(n, t, s)$

In this section, we derive a novel Gilbert-Varshamov inspired lower bound for t-indel s-substitution correcting codes. First, we formulate an implicit Gilbert-Varshamov bound in Lemma 4.3. Subsequently, we obtain an explicit lower bound for t-indel s-substitution correcting codes in Theorem 4.4. The statement of Lemma 4.3 resembles the first step in the proof of Lemma 4.2, whereas Theorem 4.4 is similar to the second step in this proof. To the best of the author's knowledge, both results have not been stated before within literature.

The proof of the following lemma is based on the discussion in Sections I and II of [46] by Tolhuizen. This statement is a straightforward adaptation thereof to suit the setting of t-indel s-substitution correction codes.

Lemma 4.3. *Let $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$ be integers. The following gives a lower bound on $M_q(n, t, s)$,*

$$M_q(n, t, s) \geq \frac{q^n}{\mathcal{V}^{avr}_{t,t,2s}}, \tag{4.6}$$

where $\mathcal{V}^{avr}_{t,t,2s} = q^{-n} \sum_{\mathbf{x} \in \mathcal{B}_q(n)} |\mathcal{V}_{t,t,2s}(\mathbf{x})|$.

Proof. The idea of this proof is to translate the problem of finding a large code in the setting of coding theory, to the problem of finding a large clique[1] in a graph in the setting of graph theory. For the latter problem, we apply Turán's theorem which states that a graph with sufficiently many edges contains a large clique, which in turn induces a large code.

Define the undirected graph $G = (V, E)$ without loops or double edges as follows. Let $V = \mathcal{B}_q(n)$ be the set of nodes of G. Two distinct nodes $\mathbf{x}$ and $\mathbf{y}$ from V are joined by an edge if $\mathbf{x} \notin \mathcal{V}_{t,t,2s}(\mathbf{y})$. This is well-defined because $\mathbf{x} \notin \mathcal{V}_{t,t,2s}(\mathbf{y})$ if and only if $\mathbf{y} \notin \mathcal{V}_{t,t,2s}(\mathbf{x})$ by Lemma 2.2. Intuitively, two vertices $\mathbf{x}$ and $\mathbf{y}$ are joined by an edge if and only if they can both belong to the same code. Then, the number of nodes of G

[1] A clique of a graph G is an induced subgraph of G that is complete. In other words, all pairs of vertices in a clique are connected by an edge. See Figure 4.1.

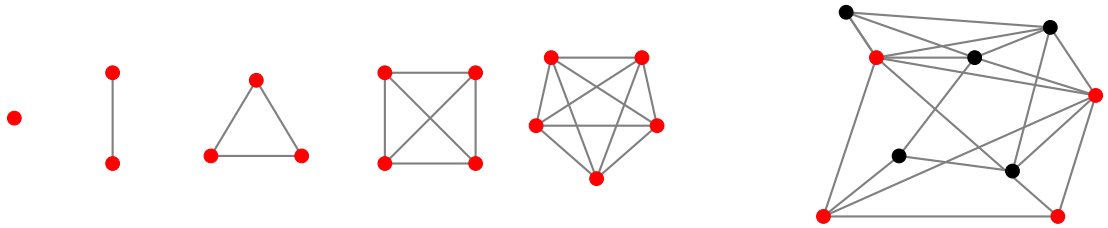


Figure 4.1: On the left, complete graphs with 1 to 5 vertices. On the right, a clique of size 4 (in red) within a larger graph.

equals $|V| = q^n$ and the number of edges equals

$$\begin{aligned}
|E| &= \frac{1}{2}\sum_{\mathbf{x}\in V}(|V \setminus \mathcal{V}_{t,t,2s}(\mathbf{x})|) \\
&= \frac{1}{2}\sum_{\mathbf{x}\in V}(|V| - |\mathcal{V}_{t,t,2s}(\mathbf{x})|) \\
&= \frac{1}{2}q^{2n} - \frac{1}{2}\sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,t,2s}(\mathbf{x})| \\
&= \frac{1}{2}q^n(q^n - \mathcal{V}^{avr}_{t,t,2s}),
\end{aligned}$$

where the first equality follows from the fact that each $\mathbf{x} \in V$ has $|V \setminus \mathcal{V}_{t,t,2s}(\mathbf{x})|$ incident edges. Therefore, summing $|V \setminus \mathcal{V}_{t,t,2s}(\mathbf{x})|$ over all nodes in $\mathbf{x} \in V$ equals $2|E|$ because each edge is counted twice. Observe that from the definition of the edges in G and Lemma 2.4 it follows that a clique of size k in G corresponds to a t-indel s-substitution correcting code of size k.

Turán's theorem [46, Sec. II] implies that the graph $G = (V, E)$ contains a clique of size k whenever k is an integer such that $2 \leq k \leq n$ and

$$|E| > \frac{k-2}{2(k-1)}|V|^2. \tag{4.7}$$

By substituting the cardinalities of V and E into the expression above and dividing by $\frac{1}{2}q^n$, this condition can be rewritten as

$$q^n - \mathcal{V}^{avr}_{t,t,2s} > \frac{k-2}{k-1}q^n = q^n - \frac{1}{k-1}q^n,$$

which in turn is equivalent to

$$k < 1 + \frac{q^n}{\mathcal{V}^{avr}_{t,t,2s}}.$$

Note that the integer $k^* := \lceil \frac{q^n}{\mathcal{V}^{avr}_{t,t,2s}} \rceil$ satisfies this last condition. In turn, k^* also satisfies (4.7). Hence, we conclude that there exists a clique in G of size k^*, which implies that there exists a t-indel s-substitution correcting code of size k^*. By the definition of $M_q(n,t,s)$ we find $M_q(n,t,s) \geq k^* \geq \frac{q^n}{\mathcal{V}^{avr}_{t,t,2s}}$, as desired. □

The previous proof depends on a version of Turán's theorem as stated in [46, Sec. II]. A slightly stronger version of Turán's theorem can be found in [47] together with a short proof by induction on k. In the following section, we will take a closer look at Turán's theorem and therefore we omit a proof here.

Example 4.4. In this example, we bound $M_3(11,1,1)$ from below using Lemma 4.3. Since n and q are rather small in this example, we can compute $\mathcal{V}^{avr}_{1,1,2}$ using the script in Appendix B. For each $\mathbf{x} \in \mathcal{B}_3(11)$ we determine $|\mathcal{V}_{1,1,2}(\mathbf{x})|$ and get for instance,

$$|\mathcal{V}_{1,1,2}(00000000000)| = 1563,$$
$$|\mathcal{V}_{1,1,2}(01201201201)| = 20427.$$

As an average result over all $\mathbf{x} \in \mathcal{B}_3(11)$, we obtain $\mathcal{V}^{avr}_{1,1,2} = \frac{1}{3^{11}} \sum_{\mathbf{x}\in\mathcal{B}_3(11)} |\mathcal{V}_{1,1,2}(\mathbf{x})| \approx 13023.0569$. Using Lemma 4.3 we obtain the following lower bound,

$$M_3(11,1,1) \geq \frac{3^{11}}{13023.0569} \approx 13.60.$$

Since $M_3(11,1,1)$ is integer-valued, this result gives $M_3(11,1,1) \geq 14$. Note that knowing $\mathcal{V}^{avr}_{1,1,2}$ up to four decimals is sufficient, because the numerical lower bound on $M_3(11,1,1)$ is rounded up to arrive at a final result.

Given that the size of $\mathcal{B}_q(n)$ grows exponentially in n, computing this average is impractical for large n and q with the script from Appendix B. In order to be able to use the lower bound in the previous lemma for all parameters n, q, t and s, we employ an upper bound for $\mathcal{V}^{avr}_{t,t,2s}$. This makes the bound explicit at the cost of obtaining a weaker bound.

Theorem 4.4. *For integers $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$, the following gives a lower bound on $M_q(n,t,s)$,*

$$M_q(n,t,s) \geq \frac{q^{n+t}}{\left(\sum_{i=0}^{t} \binom{n}{i}(q-1)^i\right)^2 \left(\sum_{i=0}^{2s} \binom{n-t}{i}(q-1)^i\right)}. \tag{4.8}$$

Proof. We claim that size of $\mathcal{V}_{t,t,2s}(\mathbf{x})$ averaged over all $\mathbf{x} \in \mathcal{B}_q(n)$ can be bounded by

$$\mathcal{V}^{avg}_{t,t,2s} = q^{-n} \sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,t,2s}(\mathbf{x})| \leq q^{-t} \left(\sum_{i=0}^{t} \binom{n}{i}(q-1)^i\right)^2 \left(\sum_{i=0}^{2s} \binom{n-t}{i}(q-1)^i\right).$$

In that case, the result of this theorem follows immediately from applying this upper bound to Lemma 4.3. Therefore, this proof is limited to proving this claim. In what follows, a superscript $^-$ will denote a variable in $\mathcal{B}_q(n-t)$, whereas an omission thereof means that the variable is an element of $\mathcal{B}_q(n)$.

To this end, recall that each element in $\mathcal{V}_{t,t,2s}(\mathbf{x})$ can be reached from $\mathbf{x} \in \mathcal{B}_q(n)$ by first deleting precisely t symbols, followed by substituting at most $2s$ symbols and lastly inserting exactly t symbols. Hence, it follows that

$$|\mathcal{V}_{t,t,2s}(\mathbf{x})| = \left| \bigcup_{\mathbf{y}^-\in\mathcal{D}_t(\mathbf{x})} \bigcup_{\mathbf{z}^-\in\mathcal{S}_{2s}(\mathbf{y}^-)} \mathcal{I}_t(\mathbf{z}^-) \right| \leq \sum_{\mathbf{y}^-\in\mathcal{D}_t(\mathbf{x})} \sum_{\mathbf{z}^-\in\mathcal{S}_{2s}(\mathbf{y}^-)} |\mathcal{I}_t(\mathbf{z}^-)|. \tag{4.9}$$

In order to evaluate the right-hand side of this expression, recall that the cardinalities of the sets $\mathcal{I}_t(\mathbf{x}^-)$ and $\mathcal{S}_{2s}(\mathbf{x}^-)$ for any $\mathbf{x}^- \in \mathcal{B}_q(n-t)$ are given by

$$|\mathcal{I}_t(\mathbf{x}^-)| = \sum_{i=0}^{t} \binom{(n-t)+t}{i} (q-1)^i, \tag{4.10}$$

$$|\mathcal{S}_{2s}(\mathbf{x}^-)| = \sum_{i=0}^{2s} \binom{n-t}{i} (q-1)^i. \tag{4.11}$$

Furthermore, the cardinality of $\mathcal{D}_t(\mathbf{x})$ averaged over all $\mathbf{x} \in \mathcal{B}_q(n)$ is given by

$$\frac{1}{q^n} \sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{D}_t(\mathbf{x})| = \frac{1}{q^t} \sum_{i=0}^{t} \binom{n}{i} (q-1)^i. \tag{4.12}$$

in accordance with (3.5). By carefully taking into account the lengths of the words in the following expression we find

$$\begin{aligned}
\frac{1}{q^n} \sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,t,2s}(\mathbf{x})| &\overset{(4.9)}{\leq} \frac{1}{q^n} \sum_{\mathbf{x}\in\mathcal{B}_q(n)} \sum_{\mathbf{y}^-\in\mathcal{D}_t(\mathbf{x})} \sum_{\mathbf{z}^-\in\mathcal{S}_{2s}(\mathbf{y}^-)} |\mathcal{I}_t(\mathbf{z}^-)| \\
&\overset{(4.10)\&(4.11)}{=} \frac{1}{q^n} \sum_{\mathbf{x}\in\mathcal{B}_q(n)} \sum_{\mathbf{y}^-\in\mathcal{D}_t(\mathbf{x})} \left(\sum_{i=0}^{2s} \binom{n-t}{i} (q-1)^i \right) \left(\sum_{i=0}^{t} \binom{n}{i} (q-1)^i \right) \\
&= \frac{1}{q^n} \left(\sum_{i=0}^{2s} \binom{n-t}{i} (q-1)^i \right) \left(\sum_{i=0}^{t} \binom{n}{i} (q-1)^i \right) \sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{D}_t(\mathbf{x})| \\
&\overset{(4.12)}{=} \frac{1}{q^t} \left(\sum_{i=0}^{t} \binom{n}{i} (q-1)^i \right)^2 \left(\sum_{i=0}^{2s} \binom{n-t}{i} (q-1)^i \right).
\end{aligned}$$

This proves the claim and consequently concludes the proof. □

Theorem 4.4 provides a lower bound on the size of a maximal t-indel s-substitution correcting code. Notice that both the Gilbert-Varshamov bound on $M_q(n,0,s)$ of Lemma 4.1 and the Levenshtein's bound on $M_q(n,t,0)$ from Lemma 4.2 are implied by Theorem 4.4 by setting $t=0$ and $s=0$, respectively.

Example 4.5. In Example 4.4 we established that $M_3(11,1,1) \geq 14$ using Lemma 4.3. Next, we compute a lower bound on $M_3(11,1,1)$ based on Theorem 4.4, which gives,

$$M_3(11,1,1) \geq \frac{3^{12}}{\left(\binom{11}{0} + \binom{11}{1}\cdot 2\right)^2 \cdot \left(\binom{10}{0} + \binom{10}{1}\cdot 2 + \binom{10}{2}\cdot 4\right)} = \frac{3^{12}}{23^2 \cdot 201} \approx 4.9981.$$

This implies that $M_3(11,1,1) \geq 5$. It is not surprising that this result is weaker than the previous result from Example 4.4. In Theorem 4.4 we used a non-tight bound on $\mathcal{V}^{avg}_{1,1,2}$ instead of the exact value which leads to this deterioration.

Next, we aim to improve the lower bounds on $M_q(n,t,s)$ from this section.

4.3 Improving the Gilbert-Varshamov inspired lower bound on $M_q(n, t, s)$

In this section, we take a closer look at the strategy that was used to derive the bounds in the previous section. In doing so, we identify an improvement for the lower bound on $M_q(n, t, s)$ from Lemma 4.3. The following results are inspired by a discussion by Sala *et al.* in [48]. They argue how to derive lower bounds on $M_q(n, t, 0)$ that improve upon Levenshtein's bound from Lemma 4.2. Here, we adapt their argument to the setting of t-indel s-substitution correcting codes, and derive a lower bound on $M_q(n, t, s)$.

Recall that for the lower bound on $M_q(n, t, s)$ of Lemma 4.3 the following idea is used. Instead of trying to find a large code which leads to a lower bound on $M_q(n, t, s)$, the problem is translated to a graph-theoretical setting. By defining an appropriate graph G, a t-indel s-substitution correcting code $\mathcal{C} \subseteq \mathcal{B}_q(n)$ corresponds to a clique in G of size $|\mathcal{C}|$. Thereafter, Turán's theorem is used to show that there exists a large clique in G. This implies that there exists a t-indel s-substitution correcting code of equal size.

We note the following about this strategy. In essence, the strength of the lower bound on $M_q(n, t, s)$ is determined by how well Turán's theorem is able to find a large clique in the graph G. In other words, if there is a method to show that there is an even larger clique in G, then this results in a better lower bound on $M_q(n, t, s)$ as well. With this in mind, we make a side step to discuss cliques in graphs.

4.3.1 Intermezzo - maximally sized clique of a graph

We are interested in finding the size of a large clique within a general graph, and preferably of a largest clique. Recall that a clique in a graph G is an induced subgraph of G that is complete, see Figure 4.1. Obviously, the size of such a clique depends of the structure of the graph. For example, in Figure 4.2 the graphs G_1 and G_2 contain both equally many vertices and edges. However, the size of maximal cliques differs per graph.

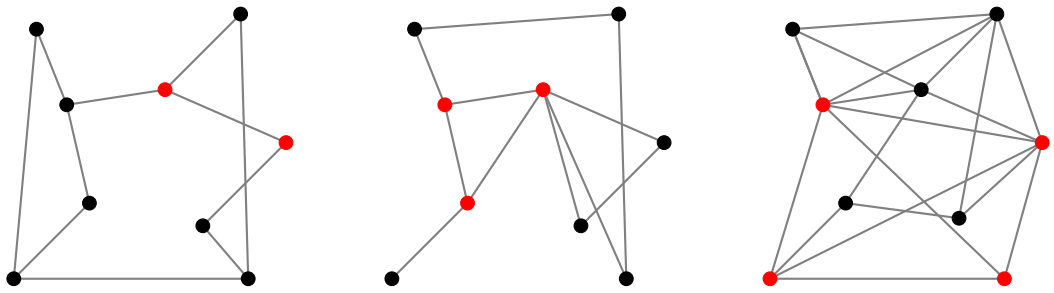


Figure 4.2: Two graphs G_1 and G_2 with 9 vertices and 11 edges, and one graph G_3 with 9 vertices and 20 edges. For each graph, a maximal size clique is indicated in red. For instance, the indicated clique in G_1 is maximal, since there is no triplet of vertices in G_1 that are all connected by an edge. None of the maximal cliques are unique.

At this point, we recall Turán's theorem [46]. Let $G=(V,E)$ be a graph and $2 \leq k \leq |V|$ be an integer such that

$$|E| > \frac{k-2}{2(k-1)}|V|^2 \iff \frac{2|E|}{|V|^2} > 1 - \frac{1}{k-1} \iff \frac{|V|^2}{|V|^2 - 2|E|} > k-1, \quad (4.13)$$

then Turán's theorem states that G contains a clique of size k.

Example 4.6. In this example, we investigate what can be concluded about the size of the cliques in the graphs from Figure 4.2 based on Turán's theorem. For $G_i=(V_i,E_i)$ with $i \in \{1,2,3\}$ it holds that

$$\frac{|V_j|^2}{|V_j|^2 - 2|E_j|} = \frac{9^2}{9^2 - 2\cdot 11} \approx 1.37, \text{ for } j \in \{1,2\},$$
$$\frac{|V_3|^2}{|V_3|^2 - 2|E_3|} = \frac{9^2}{9^2 - 2\cdot 20} \approx 1.98.$$

Given that these values are all smaller than 2, we conclude based on Turán's theorem that all three graphs contain a clique of size 2. This result is optimal for G_1 because G_1 does not contain a clique of size three (or larger). On the other hand, it is clearly suboptimal for the graphs G_2 and G_3.

A reason why Turán's theorem does not always provide the size of a largest clique is because it only considers the parameters $|V|$ and $|E|$ about the structure of the graphs. Indeed, observe that (4.13) only contains $|V|$ and $|E|$ as the information on $G=(V,E)$. Obviously, these two parameters are not sufficient to characterize the full and complex structure of a graph. As a result, it is also not possible in general to conclude what the largest size of a clique in G is based only on $|V|$ and $|E|$.

Therefore, we visit another result by Caro [49] and Wei [50] on the existence of large cliques in graphs. This result considers the degree of each vertex, instead of only graph parameters $|V|$ and $|E|$. The degree of a vertex $v \in V$ is denoted by $deg(v)$, and it is defined as the number of edges incident to v, i.e., the number of edges that use v as an endpoint. For example, in Figure 4.2 the vertices in the left-bottom of the graphs G_1, G_2 and G_3 have a degree of 3, 1 and 4, respectively. The following result can be proven using induction on $|V|$, as was originally done by Caro [49]. For a full proof we refer to Appendix A.4.

Theorem 4.5 ([49,50] Caro & Wei)**.** *Let $G=(V,E)$ be a simple graph, then G contains a clique of size*

$$\sum_{v \in V} \frac{1}{|V| - deg(v)}. \quad (4.14)$$

Let us revisit Figure 4.2 in relation to the Caro-Wei theorem.

Example 4.7. In this example, we consider the graph G_3 from Figure 4.2. Note that this graph has no vertices of degree at most two, four vertices of degree three, one vertex of degree four, two vertices of degree five and two vertices of degree six. Hence, Theorem 4.5 gives that G_3 has clique of size at least

$$\sum_{v \in V} \frac{1}{|V| - deg(v)} = 4 \cdot \frac{1}{9-3} + \frac{1}{9-4} + 2 \cdot \frac{1}{9-5} + 2 \cdot \frac{1}{9-6} \approx 2.03.$$

Given that the size of a clique is necessarily integer-valued, we conclude based on the Caro-Wei theorem that G_3 has a clique of size three. This is still sub-optimal, since G_3 has a clique of size four. Nevertheless, it forms a strict improvement over the result implied by Turán's theorem, see Example 4.6.

It is clear that the degrees of the vertices contain more information about the structure of a graph than only the parameters $|V|$ and $|E|$. Hence, it might not be surprising that the Caro-Wei theorem improves Turán's theorem.

Lemma 4.6 ([48, 51])**.** *Let $G = (V, E)$ be a simple graph, then it holds that*

$$\frac{|V|^2}{|V|^2 - 2|E|} \leq \sum_{v \in V} \frac{1}{|V| - deg(v)}. \tag{4.15}$$

Consequently, the Caro-Wei theorem implies Turán's theorem of (4.13)*.*

For a proof of this lemma we refer to Appendix A.5. For completeness, we mention that there exist even stronger results than the Caro-Wei theorem on the size of a maximal clique within literature, e.g., [52, 53]. These results require even more details about the structure of the graph. For example, the results in [52, 53] are based on the number of certain sub-graphs within a graph. However, we have to keep in mind that for our purpose these graph-theoretical results need to translated to coding-theoretical results. In this case, we note that this specific information about the sub-graphs in terms of codes is not known (to us). Therefore, we do not consider these results in this thesis.

Next, we return our attention to the setting of t-indel s-substitution correcting codes.

4.3.2 Improved lower bound using the Caro-Wei theorem

In this subsection, we present another lower bound on the maximal size of a t-indel s-substitution correcting code. This result will improve upon the bound from Lemma 4.3. The reason for this improvement is that we use the same derivation as in Lemma 4.3, but we use the result from Caro-Wei instead of from Turán. To the best of the author's knowledge, the following bound has not been stated before within literature.

Lemma 4.7. *Let $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$ be integers. The following gives a lower bound on $M_q(n, t, s)$,*

$$M_q(n, t, s) \geq \sum_{\mathbf{x} \in \mathcal{B}_q(n)} \frac{1}{|\mathcal{V}_{t,t,2s}(\mathbf{x})|}.$$

Proof. For the proof of this lemma, we follow a similar reasoning as in Lemma 4.3. Consider the same simple graph $G = (V, E)$ as defined in Lemma 4.3. That is, let $V = \mathcal{B}_q(n)$ be the set of vertices. Two distinct vertices $\mathbf{x}, \mathbf{y} \in \mathcal{B}_q(n)$ are joined by an edge if and only if $\mathbf{x} \notin \mathcal{V}_{t,t,2s}(\mathbf{y})$. Intuitively, two vertices are joined by an edge precisely if both vertices can be elements of the same t-indel s-substitution correcting code. From this definition of G, it follows that the degree of a vertex $\mathbf{x} \in \mathcal{B}_q(n)$ is given by

$$deg(\mathbf{x}) = |\mathcal{B}_q(n) \setminus \mathcal{V}_{t,t,2s}(\mathbf{x})| = q^n - |\mathcal{V}_{t,t,2s}(\mathbf{x})|.$$

In this setting, Theorem 4.5 implies that G contains a clique of size

$$\sum_{\mathbf{x}\in\mathcal{B}_q(n)} \frac{1}{|\mathcal{B}_q(n)| - deg(\mathbf{x})} = \sum_{\mathbf{x}\in\mathcal{B}_q(n)} \frac{1}{q^n - (q^n - |\mathcal{V}_{t,t,2s}(\mathbf{x})|)} = \sum_{\mathbf{x}\in\mathcal{B}_q(n)} \frac{1}{|\mathcal{V}_{t,t,2s}(\mathbf{x})|}. \quad (4.16)$$

Recall from Lemma 4.3 that a clique of size k in G corresponds to a t-indel s-substitution correcting code of size k. Therefore, we conclude that there exists a t-indel s-substitution correcting code with a size as given by (4.16). Naturally, the size of this code forms a lower bound on $M_q(n,t,s)$, which finalizes our proof. □

Example 4.8. In Example 4.4, we found that $M_3(11,1,1) \geq 14$ according to Lemma 4.3. Here, we investigate whether Lemma 4.7 improves upon this result. To this end, we computed the cardinality of $\mathcal{V}_{1,1,2}(\mathbf{x})$ for every $\mathbf{x} \in \mathcal{B}_3(11)$ using the script in Appendix B. Using these cardinalities and Lemma 4.7 we find,

$$M_3(11,1,1) \geq \sum_{\mathbf{x}\in\mathcal{B}_3(10)} \frac{1}{|\mathcal{V}_{1,1,2}(\mathbf{x})|} = 14.38.$$

It follows that $M_3(11,1,1) \geq 15$. Therefore, Lemma 4.7 gives a strictly greater lower bound than Lemma 4.3.

The previous example shows that there is an instance in which Lemma 4.7 gives a strictly better bound than Lemma 4.3. For general parameters, it holds that Lemma 4.7 performs at least as well as Lemma 4.3.

Lemma 4.8. *Let $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$ be integers. Then, the following holds*

$$\frac{q^n}{\mathcal{V}^{avr}_{t,t,2s}} = \frac{q^n}{q^{-n}\sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,t,2s}(\mathbf{x})|} \leq \sum_{\mathbf{x}\in\mathcal{B}_q(n)} \frac{1}{|\mathcal{V}_{t,t,2s}(\mathbf{x})|}.$$

Consequently, Lemma 4.7 offers a lower bound that is at least as strong as the bound in Lemma 4.3.

Proof. The (first) equality in this lemma holds trivially, because $\mathcal{V}^{avr}_{t,t,2s}$ is given by $q^{-n}\sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,t,2s}(\mathbf{x})|$, (cf. Lem. 4.3). For the inequality in this lemma, define the reciprocal function $f : (0,\infty) \to (0,\infty)$ given by $f(x) = \frac{1}{x}$. Note that f is convex on the entire interval $(0,\infty)$, because the second derivative $f''(x) = \frac{2}{x^3}$ is non-negative on $(0,\infty)$. Using the convexity of f is follows directly that

$$\begin{aligned}
\frac{1}{q^n}\sum_{\mathbf{x}\in\mathcal{B}_q(n)} \frac{1}{|\mathcal{V}_{t,t,2s}(\mathbf{x})|} &= \frac{1}{q^n}\sum_{\mathbf{x}\in\mathcal{B}_q(n)} f(|\mathcal{V}_{t,t,2s}(\mathbf{x})|) \\
&\geq f\Big(\frac{1}{q^n}\sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,t,2s}(\mathbf{x})|\Big) \\
&= \frac{q^n}{\sum_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,t,2s}(\mathbf{x})|}.
\end{aligned}$$

By multiplying the last chain of (in)equalities with q^n, the result of this lemma follows. □

Without a concrete expression for $|\mathcal{V}_{t,t,2s}(\mathbf{x})|$ for all $\mathbf{x} \in \mathcal{B}_q(n)$ the aforementioned lower bound is implicit. For small n and q it is possible to evaluate the bound using a script such as in Appendix B, but this is impractical for large n and q. Unfortunately, as discussed in Chapter 3 a general expression for $|\mathcal{V}_{t,t,2s}(\mathbf{x})|$ is not known to us. Therefore, we must rely on an upper bound in order to get an explicit result. In doing so, we obtain the following result. As far as the author is aware, the following lower bound on $M_q(n,t,s)$ has not been stated before within literature.

Theorem 4.9. *For integers $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$, the following gives a lower bound on $M_q(n,t,s)$,*

$$M_q(n,t,s) \geq \sum_{r=1}^{n} \frac{q\binom{n-1}{r-1}(q-1)^{r-1}}{\binom{r+t-1}{t}\left(\sum_{i=0}^{t}\binom{n}{i}(q-1)^i\right)\left(\sum_{i=0}^{2s}\binom{n-t}{i}(q-1)^i\right)}. \tag{4.17}$$

Proof. First, we claim that the cardinality of $\mathcal{V}_{t,t,2s}(\mathbf{x})$ can be upper bounded by

$$|\mathcal{V}_{t,t,2s}(\mathbf{x})| \leq \binom{r(\mathbf{x})+t-1}{t}\left(\sum_{i=0}^{t}\binom{n}{i}(q-1)^i\right)\left(\sum_{i=0}^{2s}\binom{n-t}{i}(q-1)^i\right).$$

Thereafter, we will apply Lemma 4.7 together with this claim to arrive at the desired lower bound on $M_q(n,t,s)$.

In order to prove this claim, recall that each element in $\mathcal{V}_{t,t,2s}(\mathbf{x})$ can be reached from $\mathbf{x} \in \mathcal{B}_q(n)$ by first deleting precisely t symbols, followed by substituting at most $2s$ symbols and lastly inserting exactly t symbols, according to Lemma 2.1. Hence, it follows that

$$|\mathcal{V}_{t,t,2s}(\mathbf{x})| = \left|\bigcup_{\mathbf{y}^- \in \mathcal{D}_t(\mathbf{x})}\bigcup_{\mathbf{z}^- \in \mathcal{S}_{2s}(\mathbf{y}^-)} \mathcal{I}_t(\mathbf{z}^-)\right| \leq \sum_{\mathbf{y}^- \in \mathcal{D}_t(\mathbf{x})}\sum_{\mathbf{z}^- \in \mathcal{S}_{2s}(\mathbf{y}^-)} |\mathcal{I}_t(\mathbf{z}^-)|. \tag{4.18}$$

In order to evaluate the right-hand side of this expression, recall that the cardinalities of the sets $\mathcal{I}_t(\mathbf{x}^-)$ and $\mathcal{S}_{2s}(\mathbf{x}^-)$ for any $\mathbf{x}^- \in \mathcal{B}_q(n-t)$ are given by

$$|\mathcal{I}_t(\mathbf{x}^-)| = \sum_{i=0}^{t}\binom{(n-t)+t}{i}(q-1)^i,$$
$$|\mathcal{S}_{2s}(\mathbf{x}^-)| = \sum_{i=0}^{2s}\binom{n-t}{i}(q-1)^i.$$

Furthermore, it holds that $|\mathcal{D}_t(\mathbf{x})| \leq \binom{r(\mathbf{x})+t-1}{t}$ for all $\mathbf{x} \in \mathcal{B}_q(n)$ according to the upper bound in (3.3). Here, $r(\mathbf{x})$ denotes the number of runs in the word $\mathbf{x}$. By combining these results, we find

$$\begin{aligned}|\mathcal{V}_{t,t,2s}(\mathbf{x})| &\overset{(4.18)}{\leq} \sum_{\mathbf{y}^- \in \mathcal{D}_t(\mathbf{x})}\sum_{\mathbf{z}^- \in \mathcal{S}_{2s}(\mathbf{y}^-)} |\mathcal{I}_t(\mathbf{z}^-)| \\ &= \binom{r(\mathbf{x})+t-1}{t}\left(\sum_{i=0}^{t}\binom{n}{i}(q-1)^i\right)\left(\sum_{i=0}^{2s}\binom{n-t}{i}(q-1)^i\right).\end{aligned}$$

So the claim holds true. Next, we apply this claim to Lemma 4.7, and obtain

$$\begin{aligned} M_q(n,t,s) &\geq \sum_{\mathbf{x}\in\mathcal{B}_q(n)} \frac{1}{|\mathcal{V}_{t,t,2s}(\mathbf{x})|} \\ &\geq \sum_{\mathbf{x}\in\mathcal{B}_q(n)} \frac{1}{\binom{r(\mathbf{x})+t-1}{t}\left(\sum_{i=0}^{t}\binom{n}{i}(q-1)^i\right)\left(\sum_{i=0}^{2s}\binom{n-t}{i}(q-1)^i\right)} \\ &= \sum_{r=1}^{n} \frac{q\binom{n-1}{r-1}(q-1)^{r-1}}{\binom{r+t-1}{t}\left(\sum_{i=0}^{t}\binom{n}{i}(q-1)^i\right)\left(\sum_{i=0}^{2s}\binom{n-t}{i}(q-1)^i\right)}. \end{aligned}$$

where we used Lemma 2.7 to count the number of words with r runs in $\mathcal{B}_q(n)$. The last chain of (in)equalities concludes the proof. □

Example 4.9. In this example, we continue with finding a lower bound for $M_3(11,1,1)$. In this regard, the previous theorem states that

$$M_3(11,1,1) \geq \sum_{r=1}^{11} \frac{3\cdot\binom{10}{r-1}\cdot 2^{r-1}}{r\cdot\left(\binom{11}{0}+\binom{11}{1}\cdot 2\right)\cdot\left(\binom{10}{0}+\binom{10}{1}\cdot 2+\binom{10}{2}\cdot 4\right)} = \sum_{r=1}^{11}\frac{3\cdot\binom{10}{r-1}\cdot 2^{r-1}}{r\cdot 4623} \approx 5.22.$$

It follows that Theorem 4.9 gives $M_3(11,1,1)\geq 6$. Let us compare this result, to what has been found with other lower bounds in this chapter. This result is strictly stronger than $M_3(11,1,1)\geq 5$ from Example 4.5 which was found using the explicit result of Theorem 4.4. So we find that the explicit bound with Caro-Wei improves the explicit bound with Turán for this set of parameters. Moreover, the result from this example weaker than $M_3(11,1,1)\geq 15$ which was found in Example 4.8 with Lemma 4.7. This is not surprising, since Theorem 4.9 was derived using this stronger implicit lemma.

In the previous example we have seen an instance in which Theorem 4.9 is strictly stronger than Theorem 4.4. Remark that it is a priori unclear whether this holds in general. Namely, a stronger result was used in the proof of Theorem 4.9, namely Lemma 4.7 instead of Lemma 4.3. However, the upper bound on $|\mathcal{V}_{t,t,2s}(\mathbf{x})|$ from Theorem 4.9 might be weaker than the bound on $\mathcal{V}^{avg}_{t,t,2s}$ that was used in Theorem 4.4. In the next section, we will show that this is indeed the case. In other words, we show that there exist also instances in which Theorem 4.4 outperforms Theorem 4.9.

4.4 Non-asymptotic comparison of the lower bounds

In the previous sections, we established various lower bounds on $M_q(n,t,s)$. Here, we compare these lower bounds in a non-asymptotic setting. We also include the lower bound on $M_q(n,t,0)$ from Lemma 4.2 in the comparison. This is possible, since any $(t+2s)$-indel correcting code is also a t-indel s-substitution correcting code according to Lemma 2.5. Therefore, lower bounds on $M_q(n,t+2s,0)$ also form lower bounds on $M_q(n,t,s)$ and can be compared as well.

The lower bounds on $M_q(n,t,s)$ behave exponentially in n, and thus tend to become large even for relatively small n and q. For this reason, we compare the relative redundancy of an optimal code, i.e., $1-\frac{1}{n}\log_q(M_q(n,t,s))$ instead of $M_q(n,t,s)$. The relative redundancy of any code lies between 0 and 1, since the size of a code $\mathcal{C}\subseteq\mathcal{B}_q(n)$ is trivially bounded between 1 and q^n. Note that the lower bounds on $M_q(n,t,s)$ turn into upper

bounds on $1 - \frac{1}{n}\log_q(M_q(n,t,s))$. Consider Figure 4.3 in which several bounds from this chapter are compared. Lemmas 4.3 & 4.7 are only compared for $q = 2$ and small n, because computation times for these bounds are long. The graphs in this figure indicate the upper bounds on $1 - \frac{1}{n}\log_q(M_q(n,t,s))$. In other words, the exact (and unknown) value of $1 - \frac{1}{n}\log_q(M_q(n,t,s))$ lies in the region indicated in grey. The graphs show a continuous behavior in n, but they should obviously only be considered for integer values of n. Let us highlight some observations about this figure.

1. We observe that the graphs from Lemma 4.2, Theorem 4.4 and Theorem 4.9 are equal to 1 for small n, and then show a (mostly) decreasing behavior in n. The sets of parameters for which a graph equals 1 correspond to the trivial bound $1 - \frac{1}{n}\log_q(M_q(n,t,s)) \leq 1$ or equivalently $M_q(n,t,s) \geq 1$. Hence, for sufficiently large n, all bounds become non-trivial. The decreasing behavior of the graphs for large n could be expected. Namely, for fixed q, t and s and increasing n, the number of errors decreases relative to n. Therefore, it is reasonable that the number of redundant symbol needed to correct these errors decreases as well relative to n.

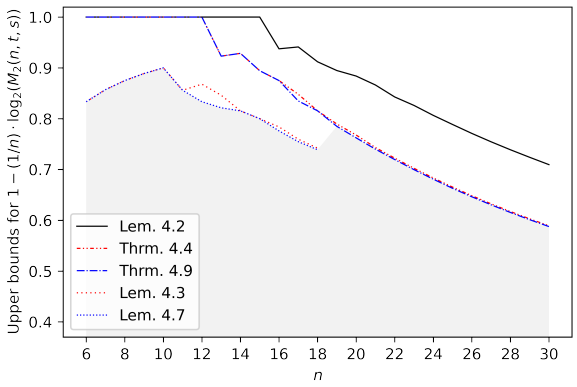
2. We observe that both Theorem 4.4 and Theorem 4.9 outperform the existing lower bound from Lemma 4.2 for the given sets of parameters, except for a few instances. Although Lemma 4.2 and Theorem 4.4 have been derived using analogous reasoning, this figure shows that different results can be obtained with both bounds. The improvements from Theorem 4.4 compared to Lemma 4.2 show that it was worth the effort to reconsider this lemma in terms of t-indel s-substitution correcting codes and derive the novel bound of Theorem 4.4.

3. It is apparent that the explicit bounds from Theorem 4.4 and Theorem 4.9 are almost alike numerically. On a logarithmic scale these differences seem small, but they can be significant on a linear scale. For instance, Theorem 4.4 gives $M_2(30,1,1) \geq 5126$ and $M_4(20,2,1) \geq 3917$, while Theorem 4.9 gives $M_2(30,1,1) \geq 5297$ and $M_4(20,2,1) \geq 3671$, respectively. It might be surprising that there are instances in which Theorem 4.4 provides a better bound than Theorem 4.9. A reason for this behavior was given in the last paragraph of the previous section.

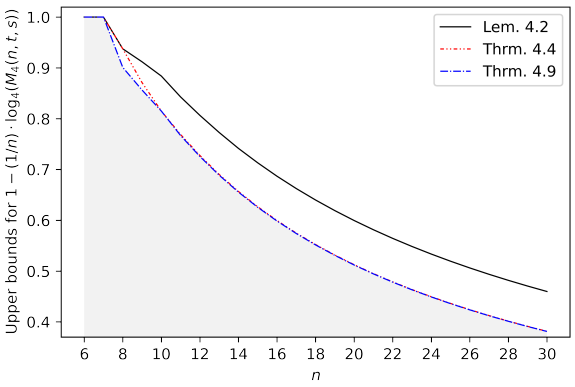
4. In Figure 4.3, the bounds from Lemma 4.3 and 4.7 have been compared as well for $q = 2$. We have computed the cardinality of $\mathcal{V}_{1,1,2}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{B}_q(n)$ using the script in Appendix B, and evaluated the respective bounds. It is not surprising that these results improve upon the bounds from Theorem 4.4 and Theorem 4.9, respectively. Indeed, both theorems have been derived using these implicit lemmas, and are thus weaker by construction. It is also not surprising that Lemma 4.7 performs at least as well as Lemma 4.3, because in Lemma 4.8 we showed that this holds in general. The increasing behaviour for small n can be explained as follows. These cases correspond to the bound $M_2(n,t,s) \geq 2$, or equivalently, $1 - \frac{1}{n}\log_2(M_2(n,t,s)) \leq 1 - \frac{1}{n}$, which is indeed increasing in n for $n \geq 6$.

5. We note that in view of computation time, computing $|\mathcal{V}_{t,t,2s}(\mathbf{x})|$ for all $\mathbf{x} \in \mathcal{B}_q(n)$ and all sets of parameters n, q, t and s is not viable using the script in Appendix B. Indeed, the number of words in $\mathcal{B}_q(n)$ is q^n, and the sizes of the set $|\mathcal{V}_{t,t,2s}(\mathbf{x})|$ grow polynomially in n. Therefore, the computation times grow exponentially in

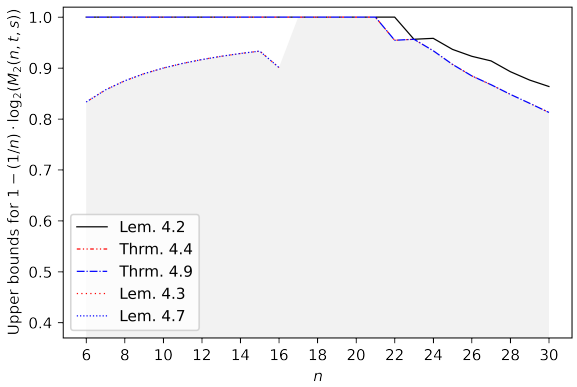
n. Hence, evaluating Lemma 4.3 and 4.7 is only possible for a limited number of parameters. Although Theorem 4.4 and Theorem 4.9 provide weaker results, these bounds have the advantage that they are explicit.



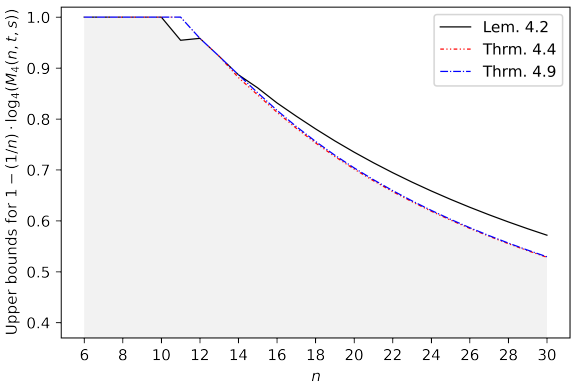
(a) $q = 2$, $t = 1$ and $s = 1$.



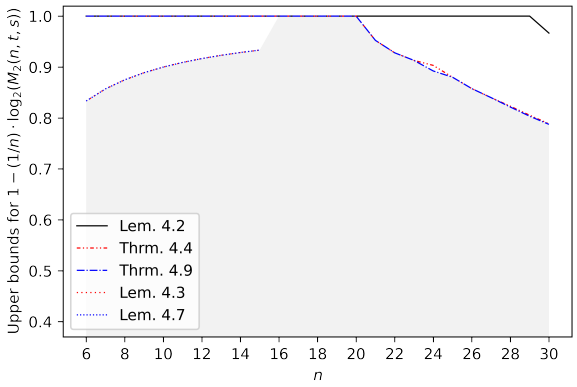
(b) $q = 4$, $t = 1$ and $s = 1$.



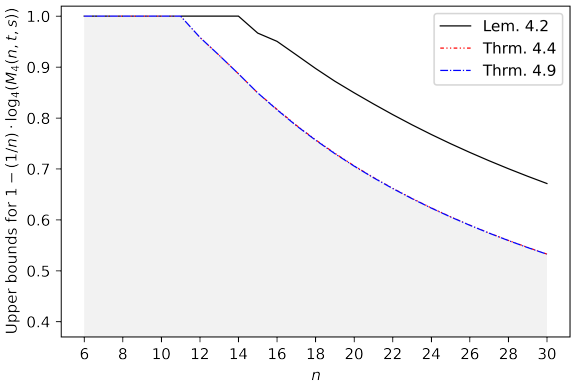
(c) $q = 2$, $t = 2$ and $s = 1$.



(d) $q = 4$, $t = 2$ and $s = 1$.



(e) $q = 2$, $t = 1$ and $s = 2$.



(f) $q = 4$, $t = 1$ and $s = 2$.

Figure 4.3: Comparison of the upper bounds on $1 - \frac{1}{n}\log_q(M_q(n,t,s))$ for various sets of parameters. The true values of $1 - \frac{1}{n}\log_q(M_q(n,t,s))$ lie in the grey regions. These upper bounds are based on the lower bounds on $M_q(n,t,s)$ in this chapter. For Lemma 4.2 we use that $M_q(n,t,s) \leq M_q(n,t+2s,0)$.

4.5 Asymptotic implications

In this section, we consider several asymptotic implications of the lower bounds on $M_q(n,t,s)$ from this chapter. Firstly, we show that the asymptotic redundancy of an optimal t-indel s-substitution correcting code is at most logarithmic in n. Secondly, we consider the asymptotic behavior of Theorem 4.4 when t and s grow linearly with n.

Consider the asymptotic setting where q, t and s are fixed integers and $n \to \infty$. Then, we use Theorem 4.4 to investigate the redundancy of an optimal t-indel s-substitution correcting code. In particular, we will show that the maximal size of a t-indel s-substitution correcting code has an asymptotic redundancy of at most $(2t+2s)\log_q(n) + o(\log_q(n))$. To this end, consider the following result.

Theorem 4.10. *Let $q \geq 2$ be an integer. For integers $t, s \geq 0$ such that $s + t \geq 1$, the following holds,*

$$\limsup_{n\to\infty} \frac{n - \log_q(M_q(n,t,s))}{(2t+2s)\log_q(n)} \leq 1.$$

The proof is stated in Appendix A.6 for brevity. The following statement is immediate from the previous theorem.

Corollary 4.11. *A maximal size t-indel s-substitution correcting code has an asymptotic redundancy of at most $(2t+2s)\log_q(n) + o(\log_q(n))$.*

Let us put this result into perspective with regard to other results in literature. In [7], Levenshtein showed two asymptotic bounds on $M_2(n,t,s)$ which imply that a binary t-indel s-substitution correcting code of maximal size has an asymptotic redundancy between $(t+s)\log_2(n) + o(\log_2(n))$ and $2(t+s)\log_2(n) + o(\log_2(n))$. Our result, extends this asymptotic upper bound to q-ary codes as well. To the best of our knowledge, this was not done before. Moreover, we note that it has been achieved using a different technique. Levenshtein's results are proven using asymptotic bounds on $M_2(n,t,s)$. On the other hand, Corollary 4.11 uses the non-asymptotic lower bound on $M_q(n,t,s)$ from Theorem 4.4.

In [54], Sima *et al.* presented t-indel correcting codes with an asymptotic redundancy of $4t\log_q(n)$. According to Song *et al.* [13], their construction is also able to correct substitutions as long as the total number of edits does not exceed t. In other words, they constructed t-indel s-substitution correcting codes with an asymptotic redundancy of $4(t+s)\log_q(n)$. In [13], Song *et al.* improved this result by presenting a code construction that achieves an asymptotic redundancy of $(4t + 4s - 1 - \lfloor \frac{2s-1}{q} \rfloor)\log_q(n)$. To the best of our knowledge, this code construction of a t-indel s-substitution correcting code has the smallest asymptotic redundancy within literature. Although our result does not induce a concrete code construction, it shows that there is a gap in terms of asymptotic redundancy between the best code construction in literature, and what is optimally possible. Indeed, note that $4t + 4s - 1 - \lfloor \frac{2s-1}{q} \rfloor \geq 4t + 3s - 1 > 2t + 2s$ for $q \geq 2$ and $t, s \geq 1$. A reason for this gap could be the following. The constructions in [13, 54] not only aim to optimize the relative redundancy of a code, but are also designed with fast and efficient encoding and decoding schemes. This might come at the cost of needing more redundancy.

In the previous asymptotic results, the parameters t and s are considered fixed. It is also interesting to consider the asymptotic setting in which t and s behave linearly in n, and n tends to infinity. This also corresponds to the more realistic case that the error rates do not depend on the length of the codeword. Let $\tau, \sigma \in [0,1]$ and set $t = \tau n$ and $s = \sigma n$. In this case, we define the (superior) asymptotic relative redundancy function

$$R_q^+(\tau,\sigma) := \limsup_{n\to\infty} \left\{ 1 - \frac{1}{n}\log_q(M_q(n, \lfloor \tau n \rfloor, \lfloor \sigma n \rfloor)) \mid n \in \mathbb{Z}_{\geq 1} \right\} \tag{4.19}$$

The reason that this function is defined using a limit superior instead of simply a limit is to ensure that (4.19) is well-defined. In this setting, consider the following asymptotic result. Recall that H_q^* denotes the extended q-ary entropy function, see Section 3.5.

Theorem 4.12. *Let $q \geq 2$ be an integer and $\tau, \sigma \in (0,1)$. Then, it holds that*

$$R_q^+(\tau,\sigma) \leq -\tau + 2H_q^*(\tau) + (1-\tau)H_q^*(\frac{2\sigma}{1-\tau}).$$

Proof. By applying Theorem 4.4 to (4.19), it readily follows that

$$\begin{aligned}
R_q^+(\tau,\sigma) &\leq \limsup_{n\to\infty} 1 - \frac{1}{n}\log_q\left(\frac{q^{n+\tau n}}{\left(\sum_{i=0}^{\tau n}\binom{n}{i}(q-1)^i\right)^2\left(\sum_{i=0}^{2\sigma n}\binom{n-\tau n}{i}(q-1)^i\right)}\right) \\
&= -\tau + \limsup_{n\to\infty}\frac{2}{n}\log_q\left(\sum_{i=0}^{\tau n}\binom{n}{i}(q-1)^i\right) \\
&\quad + \limsup_{n\to\infty}\frac{1}{n}\log_q\left(\sum_{i=0}^{2\sigma n}\binom{n-\tau n}{i}(q-1)^i\right) \\
&= -\tau + \limsup_{n\to\infty}\frac{2}{n}\log_q\left(\sum_{i=0}^{\tau n}\binom{n}{i}(q-1)^i\right) \\
&\quad + \limsup_{n'\to\infty}\frac{1-\tau}{n'}\log_q\left(\sum_{i=0}^{\frac{2\sigma}{1-\tau}n'}\binom{n'}{i}(q-1)^i\right) \\
&= -\tau + 2H_q^*(\tau) + (1-\tau)H_q^*(\frac{2\sigma}{1-\tau}),
\end{aligned}$$

where we used the change of variables $n = \frac{n'}{1-\tau}$. Moreover, we used Lemma 3.7 to evaluate both limit superiori. □

Note that it is not possible to obtain a better result on the asymptotic relative redundancy function based on Theorem 4.4. Indeed, in the proof of Theorem 4.12 we applied the bound from Theorem 4.4, and thereafter all subsequent steps are equalities.

Example 4.10. Let $q = 2$, $\tau = \frac{1}{25}$ and $\sigma = \frac{1}{50}$. Loosely speaking, this corresponds to each type of error (deletions, insertions and substitutions) occurring at most once in every 50 symbols. In this case, Theorem 4.4 gives that

$$R_2^-\left(\tfrac{1}{25}, \tfrac{1}{50}\right) \leq -\tfrac{1}{25} + 2 \cdot H_2^*\left(\tfrac{1}{25}\right) + (1 - \tfrac{1}{25}) \cdot H_2^*\left(\tfrac{1}{24}\right) \approx 0.7645.$$

This result implies that asymptotically at most 76.45% of the symbols have to be redundant when correcting up to three errors (one of each type) per 50 symbols.

5

Existing upper bounds for t-indel correcting codes and s-substitution correcting codes

This chapter discusses several existing non-asymptotic upper bounds for t-indel correcting codes and s-substitution correcting codes. The results are to a lesser extent meant as a literature review that provides a complete overview of the existing results. More so, the aim of this section will be to review various methods and collect ideas from the proofs of these bounds. By identifying similarities in the derivations of bounds for t-indel correcting codes and s-substitution correcting codes, we aim to find ways to generalize these bounds to t-indel s-substitution correcting codes. In the next chapter, we utilize these similarities in order to construct upper bounds on $M_q(n, t, s)$.

In particular, Section 5.1 discusses two Singleton bounds both for s-substitution correcting codes and t-indel correcting codes. Next, three sphere-packing bounds are discussed in Section 5.2. In Section 5.3, an upper bound on t-indel correcting codes is discussed based on matchings in hypergraphs. These upper bounds are compared in a non-asymptotic setting in Section 5.4.

5.1 Two Singleton upper bounds

In this section we discuss two Singleton bounds, one upper bound for s-substitution correcting codes [55] and one upper bound for t-indel correcting codes [56]. The derivations of both bounds rely on the same ideas.

Lemma 5.1 ([55], Singleton bound for s-substitution correcting codes)**.** *For integers $n \geq 1$, $q \geq 2$ and $0 \leq s \leq \frac{n}{2}$ the following gives an upper bound on $M_q(n, 0, s)$,*

$$M_q(n, 0, s) \leq q^{n-2s}.$$

Proof. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be an s-substitution correcting code of maximal size. Consider the code $\mathcal{C}^- \subseteq \mathcal{B}_q(n-2s)$ that is obtained from $\mathcal{C}$ by deleting the first $2s$ symbols from all codewords in $\mathcal{C}$. We claim that each pair of distinct codewords $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$ corresponds to a pair of distinct codewords $\mathbf{c}_1^-, \mathbf{c}_2^- \in \mathcal{C}^-$, that are obtained from $\mathbf{c}_1$ and $\mathbf{c}_2$, respectively, by deleting the first $2s$ symbols. In order to prove this claim, let $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$ then it holds that $\mathcal{S}_s(\mathbf{c}_1) \cap \mathcal{S}_s(\mathbf{c}_2) = \emptyset$ by Lemma 2.4, because $\mathcal{C}$ is an s-substitution correcting code. It follows that $\mathbf{c}_1$ and $\mathbf{c}_2$ differ in at least $2s+1$ positions, as otherwise the intersection is non-empty. As a result, after $2s$ deletions, the words $\mathbf{c}_1^-$ and $\mathbf{c}_2^-$ must differ in at least 1 position. Hence, they are distinct and thus the claim holds.

This claim gives that $\mathcal{C}$ and $\mathcal{C}^-$ have the same number of elements. Obviously, it holds that $|\mathcal{C}^-| \leq q^{n-2s}$, because the codewords in $\mathcal{C}^-$ have length $n-2s$ and there exist q^{n-2s} words in $\mathcal{B}_q(n-2s)$. By combining the previous observations, it follows that

$$M_q(n, 0, s) = |\mathcal{C}| = |\mathcal{C}^-| \leq q^{n-2s},$$

since $\mathcal{C}$ was chosen to be maximal. This concludes the proof. □

Example 5.1. In this example, we bound $M_2(15, 0, 1)$ from above using the Singleton bound. This upper bound yields,

$$M_2(15, 0, 1) \leq 2^{15-2 \cdot 1} = 2^{13} = 8192.$$

On the other hand, recall that the binary Hamming code $Ham(15, 11)$ from Subsection 1.6.2 has size $2^{11} = 2048$. Together, these results imply that the size of $Ham(15, 11)$ lies at most a factor of four away from the size of an optimal code. Namely, it holds that $2^{11} \leq M_2(15, 0, 1) \leq 2^{13}$.

The previous Singleton bound is a rather simple bound. This results in a weak bound for some particular sets of parameters. For instance, the previous example shows that $M_2(15, 0, 1) \leq 2^{13}$, whereas we will see in the next section that this upper bound can be significantly improved.

The simplicity of this bound does not make it weak in all cases. The Singleton bound for substitution errors is tight for $q \geq n$. There exists an extensive class of s-substitution correcting codes, called Reed-Solomon codes [57], with a size that attains the Singleton bound and with $q \geq n$. These codes have been widely applied in practice [6]. One of the reasons that Reed-Solomon codes are appealing for practical use is that they attain the Singleton bound. In other words, these codes are optimal in terms of redundancy: there do not exist other codes with the same parameters n and q that need fewer redundant symbols in order to correct s substitutions.

A similar argument as in Lemma 5.1 can be used to bound $M_q(n,t,0)$ from above.

Lemma 5.2 ([56], Singleton bound for t-indel correcting codes)**.** *For integers $n \geq 1$, $q \geq 2$ and $0 \leq t \leq n$, the following gives an upper bound on $M_q(n,t,0)$,*

$$M_q(n,t,0) \leq q^{n-t}.$$

Proof. This statement can be proven analogously to Lemma 5.1. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be an t-indel correcting code of maximal size. In that case, Lemma 2.4 shows that $\mathcal{D}_t(\mathbf{c}_1) \cap \mathcal{D}_t(\mathbf{c}_2) = \emptyset$ for any the distinct $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$. Consider the code $\mathcal{C}^- \subseteq \mathcal{B}_q(n-t)$ that is obtained from $\mathcal{C}$ by deleting the first t symbols from all codewords in $\mathcal{C}$. The remainder of the proof is analogous to the proof of Lemma 5.1 and it is therefore omitted for brevity. □

Example 5.2. In this example we recall the class of n-repetition codes $\mathcal{C}_{n,q} = \{(a)^n : a \in \mathcal{B}_q\}$ of size $|\mathcal{C}_{n,q}| = q$, for integers $n \geq 1$ and $q \geq 2$. In Subsection 1.6.1, we established that $\mathcal{C}_{n,q}$ is an $(n-1)$-indel correcting code. From this fact, it follows that $M_q(n, n-1, 0) \geq q$. In this setting, the Singleton bound for $(n-1)$-indel correcting codes yields

$$M_q(n, n-1, 0) \leq q^{n-(n-1)} = q.$$

All in all, we conclude that $M_q(n, n-1, 0) = q$ for all $n \geq 1$ and $q \geq 2$. Moreover, the class of q-ary n-repetition codes is optimal in the set of $(n-1)$-indel correcting codes.

In the previous example we verified that the Singleton bound for t-indel correcting codes is tight for $t = n-1$. It is not hard to show that Lemma 5.2 is tight in the edge cases $t = 0$ and $t = n$ as well. Indeed, for $t = 0$ it holds that $M_q(n,0,0) \leq q^n$ and the code $\mathcal{C} = \mathcal{B}_q(n)$ is a 0-indel correcting code of size q^n, which shows $M_q(n,0,0) = q^n$. This code $\mathcal{C}$ is a 0-indel correcting code because $\mathcal{D}_0(\mathbf{c}) \cap \mathcal{D}_0(\mathbf{c}') = \{\mathbf{c}\} \cap \{\mathbf{c}'\} = \emptyset$ for distinct $\mathbf{c}, \mathbf{c}' \in \mathcal{C}$. Similarly, the code $\mathcal{C} = \{(0)^n\}$ is an n-indel correcting code of size 1, while the Singleton bound yields $M_q(n,n,0) \leq q^{n-n} = 1$. Therefore, it holds that $M_q(n,n,0) = 1$.

As recent as in 2023, it was shown in [56] by that these three trivial cases ($t = 0$, $t = n-1$ and $t = n$) are the only cases in which the Singleton bound for t-indel correcting codes is tight. For $1 \leq t \leq n-2$, a strictly better upper bound was provided in [56]: $M_q(n,t,0) \leq \frac{1}{2}(q^{n-t} + q^{n-t-1})$. The right-hand side of this expression is clearly strictly smaller than q^{n-t} for $1 \leq t \leq n-2$. This improved bound holds for all integers $n \geq 1$ and $q \geq 2$ [56]. This is a rather surprising result, because it implies that there is no analog for the Reed-Solomon codes within the set of indel correcting codes, i.e., a class of t-indel correcting codes that attains the Singleton bound for $1 \leq t \leq n-2$.

We conclude this section by remarking that the two Singleton upper bounds are seemingly alike. This suggest that an analogous argument may also yield an upper bound for t-indel s-substitution correcting codes. In Subsection 6.2, this is shown to be possible and a Singleton bound for t-indel s-substitution correcting codes is derived.

5.2 Three sphere-packing upper bounds

In this section one existing upper bound for s-substitution correcting codes and two existing upper bounds for t-indel correcting codes are discussed. The derivations of these bounds are all based on the concept of sphere-packing.

In short, the sphere-packing concept works as follows. Each codeword induces a sphere of words around it which cannot intersect with the spheres of other codewords. Hence, finding a large code corresponds to packing many disjoint spheres in the space $\mathcal{B}_q(n)$. Clearly, the combined sizes of these spheres cannot exceed the number of words in $\mathcal{B}_q(n)$. Otherwise there would be a word in $\mathcal{B}_q(n)$ that is contained in at least two spheres, which is not allowed. By relating the sizes of the spheres with the size of $\mathcal{B}_q(n)$, this leads to an upper bound on the maximum code size.

Remark 5.3. Sphere-packing should not be confused with the sphere-covering techniques from Chapter 4. With sphere-packing we aim to 'pack' as many non-overlapping spheres in the set $\mathcal{B}_q(n)$ as possible. As a result, not all elements $\mathcal{B}_q(n)$ are necessarily contained in a sphere. Sphere-packing leads to an upper bound on the number of spheres and consequently also to an upper bound on the maximal size of codes.

In contrast, with sphere covering we aim to 'cover' each element of $\mathcal{B}_q(n)$ by at least one sphere. This means that the spheres are allowed to overlap. The restricting property of sphere-covering is that the centers of each sphere, i.e. the codewords, cannot be within any other sphere. Sphere-covering leads to a lower bound on the maximal size of a code.

5.2.1 Sphere-packing with substitutions

In 1950, Hamming [3] used the concept of sphere-packing in order to derive an upper bound on the maximal size of binary s-substitution correcting codes. His bound can be easily generalized to q-ary codes [6], as follows.

Lemma 5.3 ([3, 6], Hamming bound)**.** *For integers $n \geq 1$, $q \geq 2$ and $0 \leq s \leq n$, the following gives an upper bound on $M_q(n, 0, s)$,*

$$M_q(n, 0, s) \leq \frac{q^n}{\sum_{i=0}^{s} \binom{n}{i}(q-1)^i}.$$

Proof. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be an s-substitution correcting code of maximal size. For any codeword $\mathbf{c} \in \mathcal{C}$, consider the set given by $\mathcal{S}_s(\mathbf{c})$. From Lemma 2.4 it follows that these sets are disjoint, i.e., $\mathcal{S}_s(\mathbf{c}) \cap \mathcal{S}_s(\mathbf{c}') = \emptyset$ because $\mathcal{C}$ is an s-substitution correcting code. This implies that the code $\mathcal{C}$ gives rise to a disjoint union of sets centered around the codewords in $\mathcal{C}$. Naturally, the number of words in this union cannot exceed the total number of words in $\mathcal{B}_q(n)$. Hence, the following bound is obtained,

$$\sum_{\mathbf{c}\in\mathcal{C}} |\mathcal{S}_s(\mathbf{c})| = \left| \bigcup_{\mathbf{c}\in\mathcal{C}} \mathcal{S}_s(\mathbf{c}) \right| \leq |\mathcal{B}_q(n)| = q^n, \tag{5.1}$$

where we used that the spheres $\mathcal{S}_s(\mathbf{c})$ for $\mathbf{c} \in \mathcal{C}$ are disjoint. Recall the size of $\mathcal{S}_s(\mathbf{c})$ equals $\sum_{i=0}^{s} \binom{n}{i}(q-1)^i$ for each $\mathbf{c} \in \mathcal{C}$. Moreover, we recall that $\mathcal{C}$ was chosen to be

maximal and hence it holds that $|\mathcal{C}| = M_q(n, 0, s)$. It follows that

$$\sum_{\mathbf{c}\in\mathcal{C}} |\mathcal{S}_s(\mathbf{c})| = |\mathcal{C}| \cdot \sum_{i=0}^{s} \binom{n}{i}(q-1)^i = M_q(n, 0, s) \cdot \sum_{i=0}^{s} \binom{n}{i}(q-1)^i. \tag{5.2}$$

The result follows from combining (5.1) and (5.2), followed by rearranging terms. □

Example 5.4. This example forms a continuation of Example 5.1, where we established that $2^{11} \leq M_2(15, 0, 1) \leq 2^{13}$. The lower bound follows from the fact that the $Ham(15, 11)$ code has size 2^{11}. The upper bound follows from the Singleton bound for substitution correcting codes. Using the Hamming bound, we obtain

$$M_2(15, 0, 1) \leq \frac{2^{15}}{\binom{15}{0} + \binom{15}{1}} = \frac{2^{15}}{1 + 15} = \frac{2^{15}}{2^4} = 2^{11} = 2048.$$

Therefore, we conclude that $M_2(15, 0, 11) = 2^{11}$. It follows that the Hamming bound does not only improve upon the Singleton bound, but it also shows that $Ham(15, 11)$ is optimal.

5.2.2 Sphere-packing with insertions

Next, we move towards indel correcting codes. More specifically, we consider t-indel correcting codes from the perspective of codes that correct solely insertions. The following upper bound on $M_q(n, t, 0)$ is based on the same reasoning as the Hamming bound.

Lemma 5.4 ([9], Theorem 1)**.** *For integers $n \geq 1$, $q \geq 2$ and $t \geq 1$, the following gives an upper bound on $M_q(n, t, 0)$,*

$$M_q(n, t, 0) \leq \frac{q^{n+t}}{\sum_{i=0}^{t} \binom{n+t}{i}(q-1)^i}.$$

Proof. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be a t-indel correcting code of maximal size, i.e., $|\mathcal{C}| = M_q(n, t, 0)$. We proceed with a similar argument as for the Hamming bound of Lemma 5.3. To each codeword $\mathbf{c} \in \mathcal{C}$, we associate the set $\mathcal{I}_t(\mathbf{c})$ consisting of all words that can be reached from $\mathbf{c}$ by precisely t insertions. Given that $\mathcal{C}$ is a t-indel correcting code, Lemma 2.4 implies that $\mathcal{I}_t(\mathbf{c}) \cap \mathcal{I}_t(\mathbf{c}') = \emptyset$ for all distinct $\mathbf{c}, \mathbf{c}' \in \mathcal{C}$. Naturally, these insertion sets are subsets of $\mathcal{B}_q(n + t)$. Hence, it holds that $\cup_{\mathbf{c}\in\mathcal{C}} \mathcal{I}_t(\mathbf{c}) \subseteq \mathcal{B}_q(n + t)$. It follows that

$$\sum_{\mathbf{c}\in\mathcal{C}} |\mathcal{I}_t(\mathbf{c})| = \left| \bigcup_{\mathbf{c}\in\mathcal{C}} \mathcal{I}_t(\mathbf{c}) \right| \leq |\mathcal{B}_q(n + t)| = q^{n+t},$$

where we used that the sets $\mathcal{I}_t(\mathbf{c})$ are disjoint. Furthermore, recall that

$$|\mathcal{I}_t(\mathbf{c})| = \sum_{i=0}^{t} \binom{n + t}{i}(q-1)^i,$$

for all $\mathbf{c} \in \mathcal{C}$. By combining the previous results, we find

$$|\mathcal{C}| \sum_{i=0}^{t} \binom{n + t}{i}(q-1)^i = \sum_{\mathbf{c}\in\mathcal{C}} |\mathcal{I}_t(\mathbf{c})| \leq q^{n+t}.$$

Lastly, note that $|\mathcal{C}| = M_q(n, t, 0)$, since $\mathcal{C}$ was chosen to be maximal. Rearranging the terms in the last line yields the desired result. □

Example 5.5. In this example, we compute a concrete upper bound for $M_3(10, 1, 0)$ based on the previous lemma. Recall that $M_3(10, 1, 0)$ denotes the maximum size of a ternary single-indel correcting code with codewords of length 10. Lemma 5.4 gives

$$M_3(10, 1, 0) \leq \frac{3^{11}}{\binom{11}{0}(3-1)^0 + \binom{11}{1}(3-1)^1} = \frac{3^{11}}{1 \cdot 1 + 11 \cdot 2} = \frac{3^{11}}{23} \approx 7702.04.$$

It follows that $M_3(10, 1, 0) \leq 7702$. In order to put this result into perspective, note that the Singleton bound gives $M_3(10, 1, 0) \leq 3^{10-1} = 19683$, which is significantly worse. Moreover, recall Tenengolts' construction for single indel correcting codes from Subsection 1.6.3. Using this construction, we found that there exists a single indel correcting code in $\mathcal{B}_q(n)$ of size $\lceil \frac{q^n}{q \cdot n} \rceil$. In the setting of this example, we obtain the lower bound $M_3(10, 1, 0) \geq \lceil \frac{3^{10}}{3 \cdot 10} \rceil = 1969$. All in all, it holds that $1969 \leq M_3(10, 1, 0) \leq 7702$.

5.2.3 Sphere-packing with deletions

Lemma 5.4 uses the sphere-packing technique by correcting only insertions. An obvious next step is to consider sphere-packing from the perspective of correcting solely deletions. However, in this case we run into the same issue as in the previous chapter. Namely, the sets $\mathcal{D}_t(\mathbf{x})$ are not of equal size for all $\mathbf{x} \in \mathcal{B}_q(n)$ which makes the argument more complicated.

It is possible to simply use the previous argument, but this leads to the Singleton bound. This can be explained as follows. For insertions, $|\mathcal{I}_t(\mathbf{x})|$ does not depend on $\mathbf{x} \in \mathcal{B}_q(n)$. Therefore, in the proof of Lemma 5.4 we were able to use that

$$|\mathcal{C}| \sum_{i=0}^{t} \binom{n+t}{i} (q-1)^i = \sum_{\mathbf{c} \in \mathcal{C}} |\mathcal{I}_t(\mathbf{c})| \leq |\mathcal{B}_q(n+t)| = q^{n+t}$$

in order to derive a bound. Notice that for the first equality it is crucial that $|\mathcal{I}_t(\mathbf{c})|$ does not depend on $\mathbf{c}$, and is always equal to $\sum_{i=0}^{t} \binom{n+t}{i}(q-1)^i$. For deletions, the sets $\mathcal{D}_t(\mathbf{c})$ are not of equal size, and thus we can only rely on

$$|\mathcal{C}| \min_{\mathbf{c} \in \mathcal{C}} |\mathcal{D}_t(\mathbf{c})| \leq \sum_{\mathbf{c} \in \mathcal{C}} |\mathcal{D}_t(\mathbf{c})| \leq |\mathcal{B}_q(n-t)| = q^{n-t}.$$

This follows from an analogous argument as in Lemmas 5.3 & 5.4 which will be made precise in the next proof. Hence, we need an expression for $\min_{\mathbf{c} \in \mathcal{C}} |\mathcal{D}_t(\mathbf{c})|$. Notice that this minimum could potentially be equal to 1 whenever $(0)^n$ is a codeword of $\mathcal{C}$, because $|\mathcal{D}_t((0)^n)| = 1$. In that case, the previous bound simplifies to the Singleton bound, $M_q(n, t, 0) = |\mathcal{C}| \cdot 1 \leq q^{n-t}$. Given that it is not known which particular codewords belong to the code $\mathcal{C}$, we cannot use a stronger result.

With this in mind, Levenshtein modified the sphere-packing argument in order to find a different bound [38]. Intuitively, he used the following idea. Recall that[1] the words $\mathbf{x} \in \mathcal{B}_q(n)$ with 'few' runs give rise to a small set $\mathcal{D}_t(\mathbf{x})$. Hence, the codewords with few runs cause that $\min_{\mathbf{c} \in \mathcal{C}} |\mathcal{D}_t(\mathbf{c})|$ is small. By excluding these words Levenshtein is able to obtain a better bound because it guarantees that $\min_{\mathbf{c} \in \mathcal{C}} |\mathcal{D}_t(\mathbf{c})|$ is not small. Lastly, he adds a correctional term to account for these excluded words.

[1] For example, this was observed in Example 3.2.

In [38], Levenshtein provided only a very brief outline of the proof. A compact proof was given in [9]. Here, we state a more detailed version of the latter proof.

Lemma 5.5 ([38], Theorem 2)**.** *For integers $n \geq 1$, $q \geq 2$ and $1 \leq t \leq n$, the following gives an upper bound on $M_q(n, t, 0)$ for each integer r such that $\max\{1, t-1\} \leq r \leq n$,*

$$M_q(n,t,0) \leq \frac{q^{n-t}}{\sum_{i=0}^{t} \binom{r+1-t}{i}} + q \sum_{i=1}^{r} \binom{n-1}{i-1} (q-1)^{i-1}.$$

Proof. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be a maximum size t-indel correcting code. The idea of the proof is to partition $\mathcal{B}_q(n)$ into two clusters $\mathcal{A}^-$ and $\mathcal{A}^+$ that contain the words with at most r runs and at least $r+1$ runs respectively. The bound on the cardinality of $\mathcal{C}$ is then obtained by deriving separate bounds for $|\mathcal{C} \cap \mathcal{A}^-|$ and $|\mathcal{C} \cap \mathcal{A}^+|$. For $|\mathcal{C} \cap \mathcal{A}^-|$ a trivial bound is used, and $|\mathcal{C} \cap \mathcal{A}^+|$ is bounded using a sphere-packing argument.

Trivially, $|\mathcal{A}^-|$ is an upper bound for the number of codewords from $\mathcal{C}$ in $\mathcal{A}^-$, i.e, $|\mathcal{C} \cap \mathcal{A}^-|$. Using Lemma 2.7, which counts the number of words with a given number of runs, the following bound is obtained

$$|\mathcal{C} \cap \mathcal{A}^-| \leq |\mathcal{A}^-| = q \sum_{i=1}^{r} \binom{n-1}{i-1} (q-1)^{i-1}. \tag{5.3}$$

For a codeword $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+$, consider the set $\mathcal{D}_t(\mathbf{c})$. Lemma 2.4 implies that these sets are pairwise disjoint, because $\mathcal{C}$ is a t-indel correcting code. Clearly, the union of the sets $\mathcal{D}_t(\mathbf{c})$ for $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+$ cannot exceed the total number of words in $\mathcal{B}_q(n-t)$. Therefore, we deduce the following bound,

$$\sum_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} |\mathcal{D}_t(\mathbf{c})| = \left| \bigcup_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} \mathcal{D}_t(\mathbf{c}) \right| \leq |\mathcal{B}_q(n-t)| = q^{n-t},$$

where we used that the sets $\mathcal{D}_t(\mathbf{c})$ do not intersect for $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+$.

By the definition of $\mathcal{A}^+$, it follows for all $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+$ that $r(\mathbf{c}) \geq r+1$. From (3.4), we recall the lower bound $|\mathcal{D}_t(\mathbf{cc})| \geq \sum_{i=0}^{t} \binom{r(\mathbf{c})-t}{i}$ for any $\mathbf{x} \in \mathcal{B}_q(n)$. Consequently, it holds that

$$\min_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} |\mathcal{D}_t(\mathbf{c})| \geq \min_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} \sum_{i=0}^{t} \binom{r(\mathbf{c})-t}{i} \geq \sum_{i=0}^{t} \binom{r+1-t}{i}. \tag{5.4}$$

Note that r is chosen so that $r+1-t \geq 0$, and thus the expression on the right-hand side of (5.4) is at least equal to one, because we use the convention that $\binom{a}{0} = 1$ for all $a \geq 0$. This avoids division by zero in denominator in first term of this lemma. Next, we find that

$$\sum_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} |\mathcal{D}_t(\mathbf{c})| \;\geq\; |\mathcal{C} \cap \mathcal{A}^+| \cdot \min_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} |\mathcal{D}_t(\mathbf{c})| \;\geq\; |\mathcal{C} \cap \mathcal{A}^+| \cdot \sum_{i=0}^{t} \binom{r+1-t}{i}. \tag{5.5}$$

By combining the previous results and by noting that $\mathcal{C}$ was chosen to be maximal, we obtain the desired result. Namely,

$$\begin{aligned} M_q(n,t,s) &= |\mathcal{C}| \\ &= |\mathcal{C}\cap\mathcal{A}^+| + |\mathcal{C}\cap\mathcal{A}^-| \\ &\overset{(5.5)\&(5.3)}{\leq} \frac{\sum_{\mathbf{c}\in\mathcal{C}\cap\mathcal{A}^+}|\mathcal{D}_t(\mathbf{c})|}{\sum_{i=0}^{t}\binom{r+1-t}{i}} + q\sum_{i=1}^{r}\binom{n-1}{i-1}(q-1)^{i-1} \\ &\overset{(5.4)}{\leq} \frac{q^{n-t}}{\sum_{i=0}^{t}\binom{r+1-t}{i}} + q\sum_{i=1}^{r}\binom{n-1}{i-1}(q-1)^{i-1}. \end{aligned}$$

This concludes the proof. □

The previous lemma offers a family of upper bounds. For each parameter r, we get a different bound. Therefore, this bound should be minimized over the parameter r in order to get the strongest bound. For general parameters n, q and t it is not a priori clear which r yields the strongest bound. In a non-asymptotic setting we compute the optimal bound for various sets of parameters n, q and t in Section 5.4 and will compare it to other result in this chapter.

Example 5.6. In this example, we bound $M_3(10,1,0)$ from above using Lemma 5.5. Note that for each $1 \leq r \leq 10$ we get a different bound:

$$\begin{aligned} M_3(10,1,0) &\leq \left\lfloor \frac{3^9}{\sum_{i=0}^{1}\binom{r}{i}} \right\rfloor + 3\sum_{i=1}^{r}\binom{9}{i-1}2^{i-1} \\ &= \left\lfloor \frac{3^9}{r+1} \right\rfloor + 3\sum_{i=1}^{r}\binom{9}{i-1}2^{i-1} \\ &:= T_1(r) + T_2(r). \end{aligned}$$

The first term can be rounded down, because it bounds the number of codewords with at least r runs, which is integer-valued. Note that $T_2(r)$ is also an integer for each r. The following table lists these upper bounds for all $1 \leq r \leq 10$.

r	1	2	3	4	5	6	7	8	9	10
$T_1(r)$	9841	6561	4920	3936	3280	2811	2460	2187	1968	1789
$T_2(r)$	3	57	489	2505	8553	20649	36777	50601	57513	59049
$T_1(r)+T_2(r)$	9844	6618	<u>5409</u>	6441	11833	23460	39237	52788	59481	60838

From this table it follows that the smallest upper bound is achieved at $r = 3$. In that case, we find $M_3(10,1,0) \leq 5409$. This improves upon the bound $M_3(10,1,0) \leq 7702$ which was found in Example 5.5 using Lemma 5.4.

It is worth to remark that for $r = 10$ the term $T_2(r)$ counts the number in $\mathcal{B}_3(10)$ with 10 of fewer runs, i.e., all words in $\mathcal{B}_3(10)$. In this particular case, the upper bound from Lemma 5.5 performs worse than the trivial bound $M_3(10,1,0) \leq |\mathcal{B}_3(10)| = 3^{10}$.

To recap, we observe that the three main upper bounds of this section are all derived using a sphere-packing approach. Depending on the specific setting, i.e., substitutions,

insertions, or deletions, a different bound is obtained. Nevertheless, the similarities in the derivations of these bounds suggest that it is possible to generalize the sphere-packing approach to the setting of t-indel s-substitution correcting codes. This generalization to upper bounds on $M_q(n, t, s)$ will be addressed in Sections 6.3 and 6.4.

5.3 Hypergraph and matching upper bound

In this section, we visit a strategy to derive an upper bound for t-indel correcting codes from Kulkarni and Kiyavash [8]. This strategy uses the concepts of hypergraphs and matchings. Before stating these results, we first provide some background on these concepts which we sourced from [8].

A hypergraph is a generalization of a simple graph in which the edges are subsets of the vertices of arbitrary size instead of size two. More specifically, a hypergraph $\mathcal{H} = (V, \mathcal{E})$ is a tuple of a finite set of vertices V and a collection $\mathcal{E}$ of non-empty subsets of V, called hyperedges, which together span V. The definition of hyperedges as sets allows notions such as disjoint hyperedges, or the intersection/union of hyperedges. A matching in $\mathcal{H}$ is a collection of pairwise disjoint hyperedges $E_1, ..., E_j \in \mathcal{E}$. The maximum integer $j \geq 1$ for which such a collection exists, is called the matching number and is denoted by $\nu(\mathcal{H})$. In other words, $\nu(\mathcal{H})$ denotes the size of the largest matching in $\mathcal{H}$. To each hypergraph $\mathcal{H}$, a $|V| \times |\mathcal{E}|$ matrix A can be associated which is the called an adjacency matrix of $\mathcal{H}$. The rows of A are indexed by the vertices $x_1, ..., x_{|V|}$ and the columns by the hyperedges $E_1, ..., E_{|\mathcal{E}|}$ and the entries are defined by $A_{i,j} = 1$ if $x_i \in E_j$ and 0 otherwise.

The hypergraph and matching strategy to derive an upper bound on $M_q(n, t, 0)$ is rather lengthy. For this reason, we divide this strategy into the following three steps.

1. An upper bound on the maximum size of a matching in a hypergraph is found using an integer linear programming approach. In other words, the matching number of a general hypergraph $\mathcal{H}$, i.e. $\nu(\mathcal{H})$, is bounded from above.

2. The maximum size of a matching in $\mathcal{H}$ is related to the maximum size of a t-indel correcting code in $\mathcal{B}_q(n)$. As a result, the bound from the first step can be used to obtain an upper bound on $M_q(n, t, 0)$. This upper bound on $M_q(n, t, 0)$ is implicit, because it depends on the cardinality of $\mathcal{D}_t(\mathbf{x})$ which is not known in general.

3. The upper bound on $M_q(n, t, 0)$ is made explicit using a lower bound on $|\mathcal{D}_t(\mathbf{x})|$ at the cost of obtaining a weaker bound.

Step 1. The maximum size of a matching in a hypergraph is bounded from above in the next lemma. The idea of the proof is to formulate the matching number $\nu(\mathcal{H})$ of a general hypergraph $\mathcal{H}$ in terms of the optimal value of an integer linear program. Then, a feasible point in the dual of a linear programming relaxation of this integer linear program is constructed. The objective value of this dual feasible point gives an upper bound on the optimal value of the primal program. In turn, this yields the desired upper bound on $\nu(\mathcal{H})$. A complete proof of the next lemma can be found in Appendix A.7.

Lemma 5.6 ([8], Lem. 2.4)**.** *Let $\mathcal{H} = (V, \mathcal{E})$ be a hypergraph. Let $\mathbf{w} = (w(x))_{x \in V}$ be a real-valued vector that satisfies the following two conditions,*

1. *$w(x) \geq 0$ for all $x \in V$,*
2. *$\sum_{x \in E} w(x) \geq 1$ for all $E \in \mathcal{E}$.*

Then, it holds that $\nu(\mathcal{H}) \leq \mathbf{1}^\top \mathbf{w} = \sum_{x \in V} w(x)$. Here, $\mathbf{1}$ denotes the all-one column-vector of the appropriate length.

The previous lemma offers an upper bound on the maximum size of a matching in a general hypergraph. This bound relies on the construction of a vector $\mathbf{w}$ which satisfies the aforementioned two conditions. Hence, different vectors $\mathbf{w}$ can yield different bounds.

Step 2. Next, we relate the problem of finding a maximum size matching to the problem of finding a maximum size t-indel correcting code. Then we use this relation and the bound from the previous lemma to derive an upper bound on $M_q(n, t, 0)$.

Lemma 5.7 ([8], Theorem 4.1)**.** *For integers $n \geq 1$, $q \geq 2$ and $t \geq 0$ such that $2t < n$, the following gives an upper bound on $M_q(n, t, 0)$,*

$$M_q(n, t, 0) \leq \sum_{\mathbf{y} \in \mathcal{B}_q(n-t)} \frac{1}{|\mathcal{D}_t(\mathbf{y})|}$$

Proof. The aim of this proof will be to translate the problem of bounding the maximum size of a t-indel correcting code to the equivalent problem of bounding the size of a maximal matching in a hypergraph. Then, we apply the previous lemma to find an upper bound on $M_q(n, t, 0)$. In what follows, $\mathbf{x}$ will denote a word of length n, whereas $\mathbf{y}$ denotes a word of length $n - t$.

Define the following hypergraph $\mathcal{H}^{\mathcal{D}}_{n,q,t} := \{\mathcal{B}_q(n-t), \{\mathcal{D}_t(\mathbf{x}), \mathbf{x} \in \mathcal{B}_q(n)\}\}$. The vertices of $\mathcal{H}^{\mathcal{D}}_{n,q,t}$ are given by the words in $\mathcal{B}_q(n-t)$ and the hyperedges are formed by the sets $\mathcal{D}_t(\mathbf{x})$. This hypergraph is well-defined because the words in $\mathcal{D}_t(\mathbf{x})$ are elements of $\mathcal{B}_q(n-t)$ for $\mathbf{x} \in \mathcal{B}_q(n)$. Then, we will show that $M_q(n, t, 0) = \nu(\mathcal{H}^{\mathcal{D}}_{n,q,t})$. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be a t-indel correcting code. By Lemma 2.4 this is equivalent to stating that the sets in $\mathcal{M} := \{\mathcal{D}_t(\mathbf{c}), \mathbf{c} \in \mathcal{C}\} \subseteq \mathcal{E}$ are disjoint. In turn, this holds if and only if $\mathcal{M}$ is a matching in $\mathcal{H}^{\mathcal{D}}_{n,q,t}$ by the definition of a matching. Clearly, it holds that $|\mathcal{M}| = |\mathcal{C}|$. Therefore, a t-indel correcting code in $\mathcal{B}_q(n)$ induces a matching in $\mathcal{H}^{\mathcal{D}}_{n,q,t}$ of equal size and vice versa. Since this also holds for maximum sized codes and matchings, it follows that $M_q(n, t, 0) = \nu(\mathcal{H}^{\mathcal{D}}_{n,q,t})$. This translates the problem of upper bounding the maximum size of a t-indel correcting code, to finding an upper bound for $\nu(\mathcal{H}^{\mathcal{D}}_{n,q,t})$.

Next, we apply Lemma 5.6 for the hypergraph $\mathcal{H}^{\mathcal{D}}_{n,q,t}$. This means that we construct a vector $\mathbf{w} = (w(\mathbf{y}))_{\mathbf{y} \in \mathcal{B}_q(n-t)}$ that satisfies the conditions: 1) $\mathbf{w} \geq \mathbf{0}$ and 2) $\sum_{\mathbf{y} \in \mathcal{D}_t(\mathbf{x})} w(\mathbf{y}) \geq 1$ for all $\mathbf{x} \in \mathcal{B}_q(n)$. Define the vector $\mathbf{w}^*(\mathbf{y}) := \frac{1}{|\mathcal{D}_t(\mathbf{y})|}$. This vector is well-defined because the words in $\mathcal{D}_t(\mathbf{y})$ have length $n - 2t > 0$ since $\mathbf{y} \in \mathcal{B}_q(n - 2t)$. The conclusion of Lemma 5.6 gives the desired upper bound

$$M_q(n, t, 0) = \nu(\mathcal{H}^{\mathcal{D}}_{n,q,t}) \leq \sum_{\mathbf{y} \in \mathcal{B}_q(n-t)} w^*(\mathbf{y}) = \sum_{\mathbf{y} \in \mathcal{B}_q(n-t)} \frac{1}{|\mathcal{D}_t(\mathbf{y})|}.$$

Therefore, it remains to be shown that $\mathbf{w}^*$ satisfies the two conditions. To this end, we first note that $\mathbf{w}^* \geq \mathbf{0}$ holds, because $|\mathcal{D}_t(\mathbf{y})| \geq 1$ for all $\mathbf{y} \in \mathcal{B}_q(n-t)$ and thus

$w^*(\mathbf{y}) = \frac{1}{|\mathcal{D}_t(\mathbf{y})|}$ is strictly positive for each $\mathbf{y} \in \mathcal{B}_q(n-t)$. The second constraint is also satisfied, because it holds that

$$\sum_{\mathbf{y}\in\mathcal{D}_t(\mathbf{x})} w^*(\mathbf{y}) = \sum_{\mathbf{y}\in\mathcal{D}_t(\mathbf{x})} \frac{1}{|\mathcal{D}_t(\mathbf{y})|} \geq \sum_{\mathbf{y}\in\mathcal{D}_t(\mathbf{x})} \frac{1}{|\mathcal{D}_t(\mathbf{x})|} = 1,$$

where we used the monotonicity property, $|\mathcal{D}_t(\mathbf{y})| \leq |\mathcal{D}_t(\mathbf{x})|$ if $\mathbf{y} \in \mathcal{D}_t(\mathbf{x})$ [8, Lem. 4.1]. Hence, $\mathbf{w}^*$ satisfies the aforementioned two conditions which concludes the proof. □

The previous bound is implicit in its current form, because we do not have an exact expression for $|\mathcal{D}_t(\mathbf{x})|$ for all $\mathbf{x} \in \mathcal{B}_q(n)$ for general $t \geq 1$, see Chapter 3. On the other hand, when $t = 1$ it holds that $|\mathcal{D}_1(\mathbf{x})| = r(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{B}_q(n)$. In that case, it is possible to make the upper bound from the previous lemma explicit. To this end, consider the following example.

Example 5.7. In this example, we continue with deriving an upper bound for $M_3(10, 1, 0)$. So far, we established in previous examples that $1969 \leq M_3(10, 1, 0) \leq 5409$.

For $n \geq 1$, $q \geq 2$ and $t = 1$, Lemma 5.7 gives the following upper bound

$$M_q(n, 1, 0) \leq \sum_{\mathbf{y}\in\mathcal{B}_q(n-1)} \frac{1}{|\mathcal{D}_t(\mathbf{y})|} = \sum_{\mathbf{y}\in\mathcal{B}_q(n-1)} \frac{1}{r(\mathbf{y})} = \sum_{r=1}^{n-1} \frac{q\binom{n-2}{r-1}(q-1)^{r-1}}{r}.$$

In the last step, we used that there are $q\binom{n-2}{r-1}(q-1)^{r-1}$ words in $\mathcal{B}_q(n-1)$ with precisely r runs, according to Lemma 2.7. In particular, this yields

$$M_3(10, 1, 0) \leq \sum_{r=1}^{9} \frac{3 \cdot \binom{8}{r-1} \cdot 2^{r-1}}{r} \approx 3280.33.$$

Hence, we conclude that $M_3(10, 1, 0) \leq 3280$ which improves upon the Singleton bound from Lemma 5.2 as well as the best sphere-packing upper bound from Section 5.2.

Step 3. In this third and last step, we make the bound on $M_q(n, t, 0)$ explicit for general $t \geq 1$ using a lower bound for $|\mathcal{D}_t(\mathbf{y})|$ where $\mathbf{y} \in \mathcal{B}_q(n-t)$. In [8], this is done with a lower bound from Liron and Langberg [44]. However, we note that to the best of our understanding their bound was meant for $q = 2$ only, whereas it was applied for $q \geq 2$ in [8]. Therefore, we present an alternative explicit upper bound in the following the corollary which holds for general q, using the lower bound from (3.4).

Corollary 5.8. *For integers $n \geq 1$, $q \geq 2$ and $t \geq 0$ such that $2t < n$, the following gives an upper bound on $M_q(n, t, 0)$,*

$$M_q(n, t, 0) \leq \sum_{k=1}^{n-t} \frac{q\binom{n-t-1}{k-1}(q-1)^{k-1}}{\max\{1, \sum_{i=0}^{t}\binom{k-t}{i}\}}.$$

Proof. From Lemma 2.7 it is known that the number of words in $\mathcal{B}_q(n)$ with precisely k runs equals $q\binom{n-1}{k-1}(q-1)^{k-1}$. Moreover, recall from (3.4) the lower bound

$$|\mathcal{D}_t(\mathbf{y})| \geq \sum_{i=0}^{t} \binom{r(\mathbf{y}) - t}{i} \tag{5.6}$$

which holds for all $\mathbf{y} \in \mathcal{B}_q(n-t)$. Note that for $r(\mathbf{y}) < t$ this lower bound yields zero, because of our convention that $\binom{a}{b} = 0$ whenever $a < 0$. In order to avoid division by zero, we also use the trivial lower bound $|\mathcal{D}_t(\mathbf{y})| \geq 1$. Combining these results with the upper bound from Lemma 5.7 gives

$$M_q(n,t,0) \leq \sum_{\mathbf{y}\in\mathcal{B}_q(n-t)} \frac{1}{|\mathcal{D}_t(\mathbf{y})|} = \sum_{k=1}^{n-t} \sum_{\substack{\mathbf{y}\in\mathcal{B}_q(n-t);\\ r(\mathbf{y})=k}} \frac{1}{|\mathcal{D}_t(\mathbf{y})|} \leq \sum_{k=1}^{n-t} \frac{q\binom{n-t-1}{k-1}(q-1)^{k-1}}{\max\{1, \sum_{i=0}^{t}\binom{k-t}{i}\}},$$

which concludes the proof. □

Example 5.8. We compute the upper bounds on $M_3(10,1,0)$ and $M_3(12,2,0)$ based on Corollary 5.8. Note that the denominator in the expression of Corollary 5.8 simplifies to $\sum_{i=0}^{t}\binom{r(\mathbf{y})-t}{i} = r(\mathbf{y})$ for $t = 1$. Therefore, we obtain the same upper bound $M_3(10,1,0) \leq 3280$ as in Example 5.7. On the other hand, using Corollary 5.8 we find

$$\begin{aligned} M_3(12,2,0) &\leq \sum_{k=1}^{10} \frac{3\cdot 2^{k-1}\cdot\binom{9}{k-1}}{\max\{1, \sum_{i=0}^{2}\binom{k-2}{i}\}} \\ &= \frac{3\cdot 1}{1} + \frac{6\cdot 9}{1} + \frac{12\cdot 36}{2} + \sum_{k=4}^{10} \frac{3\cdot 2^{k-1}\cdot\binom{9}{k-1}}{1+k-2+\frac{1}{2}(k-2)(k-3)} \\ &= 273 + \sum_{k=4}^{10} \frac{3\cdot 2^{k-1}\cdot\binom{9}{k-1}}{\frac{1}{2}(k^2-3k+4)} \approx 4656.85. \end{aligned}$$

Hence, it follows that $M_3(12,2,0) \leq 4656$. The first three terms in the previous summation have been considered separately in order to carefully evaluate the maximum.

Before concluding this section, we remark that only deletions were used in the derivation of the bound on $M_q(n,t,0)$. Indeed, the hypergraph $\mathcal{H}^{\mathcal{D}}_{n,q,t}$ as defined in Lemma 5.7 is based on the sets $\mathcal{D}_t(\mathbf{x})$ for $\mathbf{x} \in \mathcal{B}_q(n)$. It is natural to ask whether a similar strategy also leads to an upper bound using only insertions or substitutions, i.e., the sets $\mathcal{I}_t(\mathbf{x})$ or $\mathcal{S}_s(\mathbf{x})$, respectively. This is possible in both cases. However, it was stated in [8] that in the case of insertions the same bound as in Lemma 5.4 is obtained. Indeed, by considering the hypergraph $\mathcal{H}^{\mathcal{I}}_{n,q,t} = \{\mathcal{B}_q(n+t), \{\mathcal{I}_t(\mathbf{x}), \mathbf{x} \in \mathcal{B}_q(n)\}\}$ and by setting $w(\mathbf{y}) = \frac{1}{|\mathcal{I}_t(\mathbf{y})|}$ the following bound can be found using analogous reasoning,

$$M_q(n,t,0) \leq \sum_{\mathbf{y}\in\mathcal{B}_q(n+t)} \frac{1}{|\mathcal{I}_t(\mathbf{y})|} = \frac{q^{n+t}}{\sum_{i=0}^{t}\binom{n+t}{i}(q-1)^i}.$$

For substitutions this leads to the Hamming bound of Lemma 5.3,

$$M_q(n,0,s) \leq \sum_{\mathbf{y}\in\mathcal{B}_q(n)} \frac{1}{|\mathcal{S}_s(\mathbf{y})|} = \frac{q^n}{\sum_{i=0}^{s}\binom{n}{i}(q-1)^i},$$

if we set $\mathcal{H}^{\mathcal{S}}_{n,q,s} = \{\mathcal{B}_q(n), \{\mathcal{S}_s(\mathbf{x}), \mathbf{x} \in \mathcal{B}_q(n)\}\}$ and $w(\mathbf{y}) = \frac{1}{|\mathcal{S}_s(\mathbf{y})|}$ [8]. We conclude that for insertions and substitutions, the hypergraph and matching strategy and sphere-packing strategies lead to the same upper bounds. The reason that different bounds are found in the case of deletions, is because the sets $\mathcal{D}_t(\mathbf{x})$ are not of equal size.

The fact that the hypergraph and matching strategy also leads to upper bounds using insertions and substitutions indicates that a similar strategy might work to derive an upper bound on $M_q(n,t,s)$ as well. This generalization to the setting of t-indel s-substitution correcting codes will be considered in Section 6.5.

5.4 Non-asymptotic comparison of the upper bounds for t-indel correcting codes

In this section, the upper bounds for t-indel correcting codes from this chapter are compared in a non-asymptotic setting. For an insightful comparison, we consider bounds on relative redundancy of an optimal code $1 - \frac{1}{n}\log_q(M_q(n,t,0))$ instead of $M_q(n,t,0)$, because the bounds on $M_q(n,t,0)$ becomes large even for small n. By doing so, the upper bounds on $M_q(n,t,0)$ from this chapter, become lower bounds on $1 - \frac{1}{n}\log_q(M_q(n,t,0))$. The upper bounds on $M_q(n,t,0)$ have been rounded down to the nearest integer before computing the relative redundancy. Figure 5.1 displays the bounds of this chapter for $6 \leq n \leq 40$, $q \in \{2,4\}$ and $1 \leq t \leq 3$. For a given set of parameters n, q and t, a large value for $1 - \frac{1}{n}\log_q(M_q(n,t,0))$ corresponds to a strong bound. The exact values of $1 - \frac{1}{n}\log_q(M_q(n,t,0))$ lie in grey region of the figures. Several remarks and observations about this figure are worth pointing out.

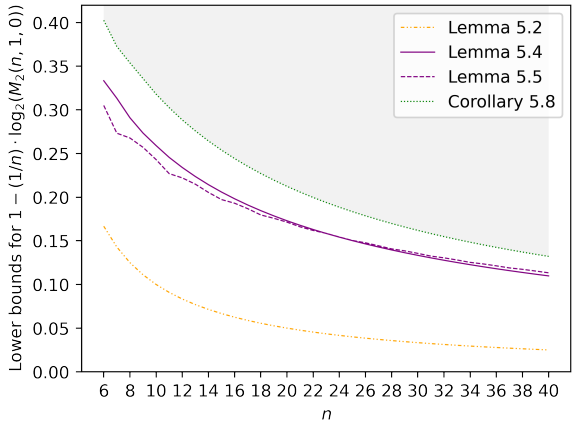
1. We remark that the upper bounds from Lemma 5.5 and Corollary 5.8 both use same lower bound; $|\mathcal{D}_t(\mathbf{x})| \geq \sum_{i=0}^{t} \binom{r(\mathbf{x})-t}{i}$. Therefore, the comparison is 'fair' in that sense. For Lemma 5.5 the bound has been optimized over r for each set of n, q and t separately. In this way, for each set of parameters the best possible bound that can be achieved using this lemma is shown. The optimization in terms of r means that for different sets of n, q and t also different values of r are used. This causes the seemingly oscillatory behavior in the graphs from Lemma 5.5.

2. The performance of the bounds is highly dependent on the values of the parameters n, q and t. It is striking that for each of the four bounds, there exists a set of parameters n, q and t in which it performs strictly better than the other bounds. Despite its simplicity, the Singleton bound from Lemma 5.2 outperforms the other bounds for $M_4(6,3,0)$. On the other hand, Levenshtein's upper bound from Lemma 5.5 yields the strongest bound for $n = 16$, $q = 2$ and $t = 3$, for example. This is especially surprising since the second term in this bound is crude. We conclude that Figure 5.1 shows that there is no single strongest bound on $M_q(n,t,0)$ amongst the bounds that we considered. Therefore, it makes sense to consider all these bounds in the generalization to t-indel s-substitution correcting codes in the next chapter.

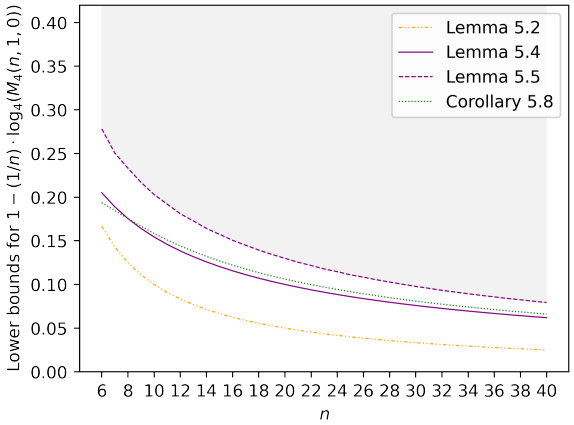
3. We observe in Figure 5.1 that the graphs of the Singleton bound seem not to differ for $q = 2$ and $q = 4$. This can be explained, because it holds that

$$1 - \frac{1}{n}\log_q(M_q(n,t,0)) \geq 1 - \frac{1}{n}\log_q(q^{n-t}) = 1 - \frac{n-t}{n} = \frac{t}{n}.$$

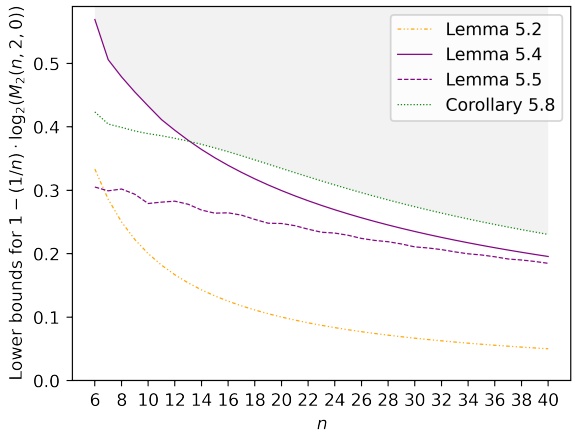
   Indeed, this does not depend on q. The other bounds show different behavior for $q = 2$ than for $q = 4$. For example, for $n = 12$ and $t = 1$, Figure 5.1 indicates that an optimal binary single-indel correcting code contains at least 28.8% redundant symbols. On the other hand, for an optimal quaternary ($q = 4$) code, this is at least 18.1%.



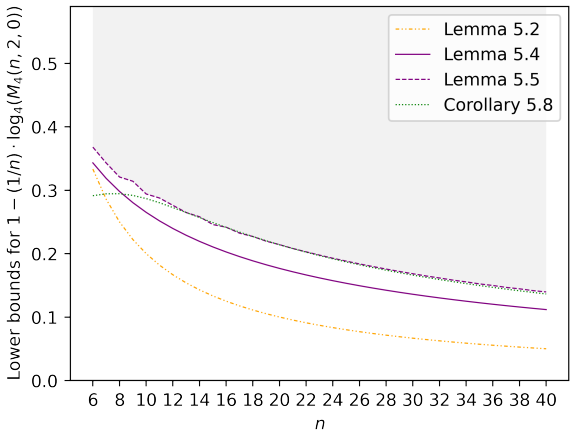
(a) $t = 1, q = 2$.



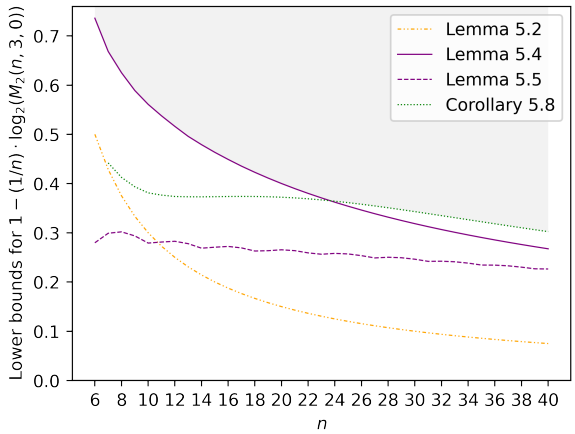
(b) $t = 1, q = 4$.



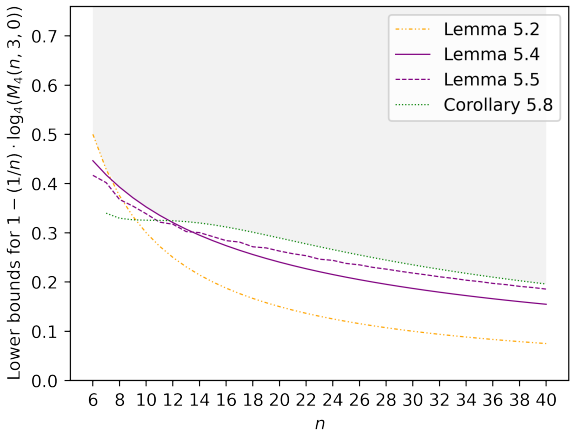
(c) $t = 2, q = 2$.



(d) $t = 2, q = 4$.



(e) $t = 3, q = 2$.



(f) $t = 3, q = 4$.

Figure 5.1: Lower bounds on the relative redundancy of an optimal code, i.e., $1 - \frac{1}{n}\log_q(M_q(n,t,0))$ based on the upper bounds in this chapter. Lemma 5.5 has been optimized over r, for each set of n, q and t separately. The feasible region for $1 - \frac{1}{n}\log_q(M_q(n,t,0))$ based on the considered bounds is indicated in grey.

6

Upper bounds for t-indel s-substitution correcting codes

In this chapter we focus on upper bounds for t-indel s-substitution correcting codes. In line with the research question, our main goal is to generalize the concepts from the previous chapter on upper bounds for t-indel correcting codes and s-substitution correcting codes. As a result, we will derive multiple upper bounds on $M_q(n, t, s)$. In particular, we first discuss existing results within literature on upper bounds for t-indel s-substitution correcting codes in Section 6.1. Subsequently, we generalize the Singleton bounds, sphere-packing bounds and the upper bound based on the hypergraph and matching strategy in Sections 6.2 – 6.5. In Section 6.6 the performance of the upper bounds on $M_q(n, t, s)$ is compared for $t = s = 1$ in a non-asymptotic setting. In Section 6.7, we briefly compare the upper bounds from this chapter with the lower bounds from Chapter 4 for $t = s = 1$. Lastly, we consider several asymptotic results based on the upper bounds on $M_q(n, t, s)$ from this chapter.

Based on the results in the previous chapters, it is not surprising that several upper bounds on $M_q(n, t, s)$ depend on the cardinality of the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$. However, within literature there is no expression for this cardinality that holds for general parameters, as far as the author is aware. For this reason, we choose to present the upper bounds on $M_q(n, t, s)$ in this chapter in the following way. First, we derive implicit upper bounds for general n, q, t and s. These bounds are implicit because they depend on an unknown expression for $|\mathcal{V}_{t',t'',s}(\mathbf{x})|$. Second, we derive explicit upper bounds on $M_q(n, 1, 1)$. This is possible using the known formulas for $|\mathcal{V}_{1,0,1}(\mathbf{x})|$ and $|\mathcal{V}_{0,1,1}(\mathbf{x})|$ from Chapter 3.

Remark that it is not possible to use existing upper bounds on $M_q(n, t+2s, 0)$ as upper bounds on $M_q(n, t, s)$, in a similar way as with lower bounds on $M_q(n, t + 2s, 0)$ from Chapter 4. Indeed, in Chapter 2 we established that $M_q(n, t+2s, 0) \leq M_q(n, t, s)$, which shows that upper bounds on $M_q(n, t+2s, 0)$ do not lead to upper bounds on $M_q(n, t, s)$. Nevertheless, we will show in this chapter that is possible to use the ideas from the upper bounds on $M_q(n, t, 0)$ and $M_q(n, 0, s)$ to prove similar bounds on $M_q(n, t, s)$. However, this will require separate proofs.

6.1 Existing upper bounds for t-indel s-substitution correcting codes

In literature, there are few contributions regarding upper bounds on $M_q(n,t,s)$ for $t,s \geq 1$. To the best of our knowledge, the following two upper bounds on $M_q(n,t,s)$ are the only non-asymptotic and non-trivial upper bounds for $t,s \geq 1$. Recently, Smagloy *et al.* showed in [11, Thrm. 4 & Thrm. 7, resp.] that

$$M_q(n,1,1) \leq \frac{3q^{n-1}}{(n-3)(n-5)(q-1)} + 5q, \tag{6.1}$$

$$M_2(n,1,s) \leq \frac{s!(2s+1)}{(n-2s)^s(n-1)} \cdot \left(2^n + \frac{2(n-1)^{2s+1}}{2s+1}\right). \tag{6.2}$$

The first bound holds for $n \geq 6$, $q \geq 2$ and $n \geq q$, whereas the second bound holds for[1] $n \geq 3$, $q = 2$, $s \geq 1$ and $n > 2s$. Both bounds are proven using an approach which resembles the hypergraph and matching strategy from Section 5.3. This approach will be detailed in Section 6.5 for t-indel s-substitution correcting codes. Hence, we omit a proof of these bounds here.

We note that Smagloy *et al.* simplified several expressions to arrive at the bounds (6.1) and (6.2). These simplifications yield easy-to-compute bounds, but they come at the cost of weaker bounds. The following example shows that (6.1) and (6.2) are indeed crude for particular sets of parameters.

Example 6.1. Let $n = 10$, $q = 3$ and $t = s = 1$. In this case, the bound (6.1) yields

$$M_3(10,1,1) \leq \frac{3 \cdot 3^9}{7 \cdot 5 \cdot 2} + 5 \cdot 3 \approx 858.56.$$

Hence, it holds that $M_3(10,1,1) \leq 858$. For comparison, we will show that this bound can be improved to $M_3(10,1,1) \leq 320$ in Example 6.8.

Furthermore, let $n = 15$, $q = 2$, $t = 1$ and $s = 3$. The upper bound in (6.2) gives

$$M_2(15,1,3) \leq \frac{3! \cdot 7}{(15-6)^3 \cdot 14} \cdot \left(2^{15} + \frac{2 \cdot 14^7}{7}\right) \approx 124077.8$$

To put this result into perspective, notice that there are only $2^{15} = 32768$ words in $\mathcal{B}_2(15)$. Hence, this existing upper bound exceeds even the trivial upper bound $M_q(n,t,s) \leq q^n$ for this particular set of parameters. In the next section, we will show that $M_q(n,t,s) \leq q^{n-t-2s}$ and thus $M_2(15,1,3) \leq 2^{15-1-2\cdot 3} = 2^8 = 256$.

All in all, this example shows that there is room for improvement with the existing upper bounds, at least for these particular sets of parameters. In what follows, we will show that these existing bounds can be improved.

[1] The requirement $n > 2s$ is not stated in [11] , but it is evident that it should be satisfied.

6.2 Singleton bound

In this section, we revisit the two Singleton bounds from Section 5.1 and derive a similar upper bound for t-indel s-substitution correcting codes. Based on the similarities between the Singleton bounds for $M_q(n, 0, s)$ and $M_q(n, t, 0)$, this generalization to an upper bound for $M_q(n, t, s)$ is rather straightforward. To the best of the author's knowledge, the following bound was not stated before in literature in its current form.

Theorem 6.1. *Let $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$ be integers such that $n - t - 2s \geq 0$. Then, the following holds,*

$$M_q(n, t, s) \leq q^{n-t-2s}. \tag{6.3}$$

Proof. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be a t-indel s-substitution correcting code of maximal size. Lemma 2.4 shows that $\mathcal{V}_{t,0,s}(\mathbf{c}_1) \cap \mathcal{V}_{t,0,s}(\mathbf{c}_2) = \emptyset$ for all pairs of distinct codewords $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$, since $\mathcal{C}$ is a t-indel s-substitution correcting code. Consider the shortened code $\mathcal{C}^- \subseteq \mathcal{B}_q(n - t - 2s)$ that is obtained from $\mathcal{C}$ by deleting the first $t + 2s$ symbols from all codewords in $\mathcal{C}$. This is possible because $t + 2s \leq n$.

We claim that two distinct codewords $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$ yield two distinct codewords $\mathbf{c}_1^-, \mathbf{c}_2^- \in \mathcal{C}^-$. For contradiction, suppose that there exist two codewords $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$ that agree on the last $n - t - 2s$ symbols, i.e., that they yield the same word after deleting the first $t+2s$ symbols. Then, we construct a word $\mathbf{z} \in \mathcal{V}_{t,0,s}(\mathbf{c}_1) \cap \mathcal{V}_{t,0,s}(\mathbf{c}_2)$. First, delete the first t symbols from both $\mathbf{c}_1$ and $\mathbf{c}_2$, to obtain $\mathbf{z}_1$ and $\mathbf{z}_2$, respectively. Notice that $\mathbf{z}_1$ and $\mathbf{z}_2$ agree on the last $n-2s$ symbols. Hence, they differ in at most $2s$ places. This means that there exists some $\mathbf{z} \in \mathcal{B}_q(n - t)$ that can be obtained from both $\mathbf{z}_1$ and $\mathbf{z}_2$ by at most s substitutions. It follows that $\mathbf{z} \in \mathcal{S}_s(\mathbf{c}_1) \cap \mathcal{S}_s(\mathbf{c}_2)$ and in turn that $\mathbf{z} \in \mathcal{V}_{t,0,s}(\mathbf{c}_1) \cap \mathcal{V}_{t,0,s}(\mathbf{c}_2)$. This forms a contradiction, because in the previous paragraph we established that this intersection is empty. Therefore, we conclude that the claim holds.

This claim implies that $\mathcal{C}$ and $\mathcal{C}^-$ have the same number of elements. Obviously, it holds that $|\mathcal{C}^-| \leq q^{n-t-2s}$, because $\mathcal{C}^- \subseteq \mathcal{B}_q(n - t - 2s)$ and there exist q^{n-t-2s} words in $\mathcal{B}_q(n - t - 2s)$. By combining the previous observations, it follows that

$$M_q(n, t, s) = |\mathcal{C}| = |\mathcal{C}^-| \leq q^{n-t-2s},$$

since $\mathcal{C}$ was chosen to be maximal in the set of t-indel s-substitution correcting codes. The last chain of (in)equalities concludes the proof. □

In the following example, we show that the repetition codes from Subsection 1.6.1 satisfy the aforementioned Singleton bound with equality, whenever $t + 2s = n - 1$.

Example 6.2. Recall the repetition code $\mathcal{C}_{5,4} = \{(a)^5 : a \in \mathcal{B}_4\} \subseteq \mathcal{B}_4(5)$ of size 4. In Subsection 1.6.1 we showed that $\mathcal{C}_{5,4}$ is a 2-indel 1-substitution correcting code. This implies that $M_4(5, 2, 1) \geq 4$. On the other hand, the previous Singleton bound gives that this code is optimal, because $M_4(5, 2, 1) \leq 4^{5-2-2\cdot 1} = 4$.

In general, let $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$ be integers such that $t + 2s = n - 1$, then we show that the repetition code $\mathcal{C}_{n,q} = \{(a)^n : a \in \mathcal{B}_q\}$ satisfies the Singleton bound with equality for these parameters. Notice that each word in $\mathcal{V}_{t,0,s}((a)^n)$ has length $n - t = 2s + 1$ and consists of at least $s + 1$ times the symbol a, where $a \in \mathcal{B}_q$.

Therefore, it holds that $\mathcal{V}_{t,0,s}(\mathbf{c}_1) \cap \mathcal{V}_{t,0,s}(\mathbf{c}_2) = \emptyset$ for all distinct $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}_{n,q}$. This implies that the repetition code $\mathcal{C}_{n,q}$ is a t-indel s-substitution correcting code by Lemma 2.4. Since it holds that $|\mathcal{C}_{n,q}| = q$, we find $M_q(n,t,s) \geq q$ for $t + 2s = n - 1$. On the other hand, we have $M_q(n,t,s) \leq q^{n-t-2s} = q$ by the Singleton bound, since we required that $t+2s = n-1$. All in all, we find that the Singleton bound is tight whenever $t+2s = n-1$ and that the class of repetition codes is optimal in this case.

6.3 Sphere-packing bounds using two clusters

In this section we revisit the sphere-packing strategy from Section 5.2 and adapt it to the setting of t-indel s-substitution correcting codes. In particular, we derive an implicit family of upper bounds on $M_q(n,t,s)$ in Subsection 6.3.1. Moreover, we will show that this result implies several bounds from Chapter 5. In Subsection 6.3.2 this family of implicit bounds is made explicit for $t = s = 1$.

6.3.1 Family of implicit sphere-packing upper bounds for $M_q(n,t,s)$

Recall Lemma 5.5 which uses a sphere-packing argument based on deletions in order to derive an upper bound on $M_q(n,t,0)$. The statement of the following theorem and its proof resemble this lemma, because this theorem offers a family of bounds as well. Namely, a different bound is obtained for each set of parameters r, and $0 \leq t', t'' \leq n$ such that $t'+t''$. This result has not been stated before in literature, up to the knowledge of the author.

Theorem 6.2. *Let $n \geq 2$, $q \geq 2$, $0 \leq t < n$ and $0 \leq s \leq n$ be integers. The following gives an upper bound on $M_q(n,t,s)$ for all integers $0 \leq r \leq n$ and $0 \leq t', t'' < n$ such that $t' + t'' = t$,*

$$M_q(n,t,s) \leq \frac{q^{n-t'+t''}}{\min\limits_{\substack{\mathbf{x} \in \mathcal{B}_q(n) \\ r(\mathbf{x}) > r}} |\mathcal{V}_{t',t'',s}(\mathbf{x})|} + q \sum_{i=1}^{r} \binom{n-1}{i-1} (q-1)^{i-1}.$$

For $r = 0$ the second addend denotes the empty sum and equals 0 and for $r = n$ the first addend is taken to be 0.

Proof. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be a maximum size t-indel s-substitution correcting code. The outline of the proof is similar to that of Lemma 5.5. The idea is to partition $\mathcal{B}_q(n)$ into two clusters $\mathcal{A}^-$ and $\mathcal{A}^+$ that contain the words with at most r runs and at least $r+1$ runs, respectively. Then, for both clusters we upper bound the number of codewords in $\mathcal{C}$ that they can contain, i.e., $|\mathcal{C} \cap \mathcal{A}^-|$ and $|\mathcal{C} \cap \mathcal{A}^+|$. Together, these bounds will form an upper bound on $M_q(n,t,s)$.

Trivially, $|\mathcal{A}^-|$ is an upper bound for the number of codewords from $\mathcal{C}$ in $\mathcal{A}^-$. Using Lemma 2.7 which counts the number of words with a given number of runs, the following bound is obtained

$$|\mathcal{C} \cap \mathcal{A}^-| \leq |\mathcal{A}^-| = |\{\mathbf{x} \in \mathcal{B}_q(n) : r(\mathbf{x}) \leq r\}| = q \sum_{i=1}^{r} \binom{n-1}{i-1} (q-1)^{i-1}.$$

For the codewords $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+$, we consider a sphere-packing argument for the sets $\mathcal{V}_{t',t'',s}(\mathbf{c})$. Since $\mathcal{C}$ is a t-indel s-substitution correcting code it follows from Lemma 2.4 that the sets $\mathcal{V}_{t',t'',s}(\mathbf{c})$ are disjoint for all $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+$. Note that the words in $\mathcal{V}_{t',t'',s}(\mathbf{c})$ have length $n - t' + t''$. Clearly, the size of the union of the sets $\mathcal{V}_{t',t'',s}(\mathbf{c})$ for $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+$ cannot exceed the total number of words in $\mathcal{B}_q(n - t' + t'')$. Therefore, we deduce the following bound,

$$\sum_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} |\mathcal{V}_{t',t'',s}(\mathbf{c})| = \left| \bigcup_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} \mathcal{V}_{t',t'',s}(\mathbf{c}) \right| \leq |\mathcal{B}_q(n - t' + t'')| = q^{n-t'+t''},$$

where we used in the first equality that the sets $\mathcal{V}_{t',t'',s}(\mathbf{c})$ are disjoint for $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+$. Moreover, it holds that

$$\sum_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} |\mathcal{V}_{t',t'',s}(\mathbf{c})| \;\geq\; |\mathcal{C} \cap \mathcal{A}^+| \cdot \min_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} |\mathcal{V}_{t',t'',s}(\mathbf{c})| \;\geq\; |\mathcal{C} \cap \mathcal{A}^+| \cdot \min_{\substack{\mathbf{x} \in \mathcal{B}_q(n) \\ r(\mathbf{x}) > r}} |\mathcal{V}_{t',t'',s}(\mathbf{x})|.$$

In the last inequality, we used that words in $\mathcal{A}^+$ have more than r runs. Next, recall that $\mathcal{C}$ was defined to be maximal, i.e., $|\mathcal{C}| = M_q(n, t, s)$. By combining the previous results, we obtain the desired result. Namely,

$$\begin{aligned} M_q(n, t, s) &= |\mathcal{C}| \\ &= |\mathcal{C} \cap \mathcal{A}^+| + |\mathcal{C} \cap \mathcal{A}^-| \\ &\leq \frac{q^{n-t'+t''}}{\min_{\substack{\mathbf{x} \in \mathcal{B}_q(n) \\ r(\mathbf{x}) > r}} |\mathcal{V}_{t',t'',s}(\mathbf{x})|} + q \sum_{i=1}^{r} \binom{n-1}{i-1} (q-1)^{i-1}, \end{aligned}$$

where we used that the sets $\mathcal{A}^+$ and $\mathcal{A}^-$ partition $\mathcal{B}_q(n)$. This concludes the proof. □

Theorem 6.2 provides a family of upper bounds on $M_q(n, t, s)$. In other words, for each set of parameters r, t' and t'' we obtain a different bound. In order to evaluate this family of upper bounds, an expression for the implicit term $\min_{\mathbf{x} \in \mathcal{B}_q(n), r(\mathbf{x}) > r} |\mathcal{V}_{t',t'',s}(\mathbf{x})|$ is needed. Unfortunately, to the best of our knowledge, no expression for $|\mathcal{V}_{t',t'',s}(\mathbf{x})|$ has been stated in literature for general parameters n, q, t', t'' and s. Therefore, we find that evaluating Theorem 6.2 in its most general form is complicated. We note the following about this problem.

1. For small parameters n and q, it is possible to compute the minimum using a brute approach. It is rather straightforward to compute $|\mathcal{V}_{t',t'',s}(\mathbf{x})|$ for all $\mathbf{x} \in \mathcal{B}_q(n)$, and minimize over all outcomes using a Python script such as in Appendix B. However, this becomes impractical for large n and q, given that the number of elements in $\mathcal{B}_q(n)$ grows exponentially in n. Moreover, using these numerical results it is not possible to derive asymptotic results.

2. In the special case of single-indel single-substitution correcting codes, i.e., $t'+t'' = 1$ and $s = 1$, analytic formulas for $|\mathcal{V}_{1,0,1}(\mathbf{x})|$ and $|\mathcal{V}_{0,1,1}(\mathbf{x})|$ are known for all $\mathbf{x} \in \mathcal{B}_q(n)$. These formulas are given in Lemmas 3.2 and 3.3, respectively. This enables us to compute an explicit expression for the minimum and to derive concrete upper bounds. This will be treated in more detail in the next subsection.

3. Given that the minimum is located in the denominator, a lower bound on this minimum can be used to make the result explicit at the cost of possibly obtaining a weaker bound. Note that Lemma 3.5 provides such a lower bound on $|\mathcal{V}_{t,0,s}(\mathbf{x})|$ which is also increasing in r. Hence, for each $\max\{2t+1, 2s+1\} \leq r \leq n$ we obtain

$$M_q(n,t,s) \leq \frac{q^{n-t}}{\sum_{i=0}^{s}\binom{\lfloor \frac{r}{2} \rfloor}{i}\sum_{j=0}^{t}\binom{\lceil \frac{r}{2} \rceil - t}{j}} + q\sum_{i=1}^{r}\binom{n-1}{i-1}(q-1)^{i-1},$$

However, we established in Subsection 3.4.2 that this lower bound on $|\mathcal{V}_{t,0,s}(\mathbf{x})|$ is weak for small n. Therefore, we will only consider this result in an asymptotic setting in Section 6.8. Instead, we focus on the stronger explicit bounds in case $t = s = 1$ for small n.

	t'			
r	0	1	2	3
0	<u>0.488</u>	0.477	0.461	0.439
1	0.495	<u>0.503</u>	0.489	0.466
2	0.501	<u>0.519</u>	0.504	0.483
3	0.506	<u>**0.529**</u>	0.522	0.501
4	0.4938	<u>0.5096</u>	0.5092	0.4977
5	0.4346	0.4383	<u>0.4386</u>	0.4370
6	0.3572	0.3578	<u>0.3579</u>	0.3577
⋮	⋮	⋮	⋮	⋮
19	<u>0.00002</u>	<u>0.00002</u>	<u>0.00002</u>	<u>0.00002</u>
20	<u>0.0</u>	<u>0.0</u>	<u>0.0</u>	<u>0.0</u>

$n = 20$, $q = 3$, $t = 3$, $s = 2$.

Table 6.1: Lower bounds on the optimal relative redundancy $1 - \frac{1}{20}\log_3(M_3(20,3,2))$ based on Theorem 6.2. The bound in each entry is computed using the indicated values of r (row) and t' (column) with $t'' = t - t' = 3 - t'$. The best bound in each row has been underlined, whereas the best bound over-all is indicated in bold. The unstated values for $7 \leq r \leq 18$ do not contain a better bound than that was found for $r = 3$ and $t' = 1$ and $t'' = 2$. All values have been rounded down to the respective number of decimals.

Example 6.3. In this example, we aim to upper bound $M_3(20,3,2)$ using Theorem 6.2. Since these upper bounds may become large, we compute the relative redundancy, i.e., $1 - \log_q(M_q(n,t,s))/n$ instead of $M_q(n,t,s)$.

For instance, let $r = 2$, $t' = 2$ and $t'' = 1$ in order to bound $M_3(20,3,2)$ from above. In this case, we find using the script in Appendix B that

$$\min_{\mathbf{x}\in\mathcal{B}_3(20), r(\mathbf{x})>2} |\mathcal{V}_{2,1,2}(\mathbf{x})| = 21979.$$

Next, we apply Theorem 6.2 and obtain,

$$M_3(20,3,2) \leq \frac{3^{20-2+1}}{21979} + 3\sum_{i=1}^{2}\binom{19}{i-1}2^i \approx 52997.5.$$

We conclude that $M_3(20,3,2) \leq 52997$ and in turn $1 - \frac{1}{20}\log_3(M_3(20,3,2)) \geq 0.504$. Recall that Theorem 6.2 offers a family of bounds, i.e., for each $0 \leq r \leq n$ and each pair $0 \leq t', t'' \leq n$ such that $t' + t'' = t$ a different bound is obtained. Obviously, we might get better bounds if we choose different r, t' and t''. Therefore, we performed this computation for all possible sets of parameters r, t' and t'' for $M_3(20,3,2)$. The results are listed in Table 6.1. The best over-all bound is indicated in bold. For instance, we find $1 - \frac{1}{20}\log_3(M_3(20,3,2)) \geq 0.529$ which corresponds to

$$M_3(20,3,2) \leq 3^{20\cdot(1-0.529)} \approx 31223.$$

Intuitively, this means that the codewords of an optimal 3-indel 2-substitution correcting code in $\mathcal{B}_3(20)$ contain at least 52.9% redundant symbols. This redundancy is the price for being able to correct three indels and two substitutions in codewords of length 20. Note that the improvement in relative redundancy from 0.529 to 0.504 seems small due to the logarithmic scale. However, on a linear scale this difference is significant: $M_3(20,3,2) \leq 31223$ versus $M_3(20,3,2) \leq 52997$, respectively.

Next, we show that several results from Chapter 5 are implied by Theorem 6.2. For instance, setting $t' = t$ and $r = t'' = s = 0$ in Theorem 6.2 results in the Singleton bound for t-indel correcting codes,

$$M_q(n,0,t) \leq \frac{q^{n-t}}{\min_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,0,0}(\mathbf{x})|} = q^{n-t},$$

where we used that $\min_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{t,0,0}(\mathbf{x})| = 1$. This holds because $|\mathcal{V}_{t,0,0}(\mathbf{x})| \geq 1$ for all $\mathbf{x} \in \mathcal{B}_q(n)$, while $|\mathcal{V}_{t,0,0}((0)^n)| = 1$. Indeed, deleting t symbols from the all-zero word always yields the all-zero word of length $n-t$, and thus $|\mathcal{V}_{t,0,0}((0)^n)| = 1$. Moreover, by setting $r = t = t' = t'' = 0$, the Hamming bound is obtained,

$$M_q(n,0,s) \leq \frac{q^n}{\min_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{0,0,s}(\mathbf{x})|} = \frac{q^n}{\sum_{i=0}^{s}\binom{n}{i}(q-1)^i}.$$

Similarly, setting $t'' = t$ and $t' = s = r = 0$ gives the bound from Lemma 5.4,

$$M_q(n,t,0) \leq \frac{q^{n+t}}{\min_{\mathbf{x}\in\mathcal{B}_q(n)} |\mathcal{V}_{0,t,0}(\mathbf{x})|} = \frac{q^{n+t}}{\sum_{i=0}^{t}\binom{n+t}{i}(q-1)^i}.$$

Lastly, by setting $t' = t$ and $t'' = s = 0$ Levenshtein's bound from Lemma 5.5 is retrieved. Suppose that the additional requirement from Lemma 5.5 that $\max\{1, t-1\} \leq r \leq n$ holds, then it follows that $|\mathcal{D}_t(\mathbf{x})| \leq \sum_{i=0}^{t}\binom{r+1-t}{i}$ whenever $r(\mathbf{x}) > r$ (cf. Section 3.2). In that case, it follows that

$$M_q(n,t,0) \leq \frac{q^{n-t}}{\min\limits_{\mathbf{x}\in\mathcal{B}_q(n),\, r(\mathbf{x})>r} |\mathcal{V}_{t,0,0}(\mathbf{x})|} + q\sum_{i=1}^{r}\binom{n-1}{i-1}(q-1)^i \leq \frac{q^{n-t}}{\sum\limits_{i=0}^{t}\binom{r+1-t}{i}} + q\sum_{i=1}^{r}\binom{n-1}{i-1}(q-1)^i,$$

which equals the statement of Lemma 5.5.

We conclude that Theorem 6.2 generalizes the sphere-packing results from Section 5.2 and even one Singleton bound to the setting of t-indel s-substitution correcting codes.

6.3.2 Explicit sphere-packing upper bounds for $M_q(n,1,1)$

In this subsection, we make Theorem 6.2 explicit in the case of a single indel and a single substitution. Let $t = s = 1$, then we find an expression for the minimum in the statement of Theorem 6.2. For $t = 1$, we have either $t' = 1$ and $t'' = 0$, or $t' = 0$ and $t'' = 1$. To this end, recall the following results from Lemmas 3.2 & 3.4,

$$|\mathcal{V}_{1,0,1}(\mathbf{x})| = L^{n,q}_{1,0,1}(r(\mathbf{x})) := \begin{cases} (n-1)(q-1)+1 & \text{if } r(\mathbf{x})=1, \\ r(\mathbf{x})\left((n-2)(q-1)-1\right)+q+2 & \text{if } r(\mathbf{x})\geq 2, \end{cases} \tag{6.4}$$

$$|\mathcal{V}_{0,1,1}(\mathbf{x})| \geq L^{n,2}_{0,1,1}(r(\mathbf{x})) := -\frac{1}{2}r(\mathbf{x})^2 + (n+\frac{1}{2})r(\mathbf{x}) + \frac{1}{2}(n^2+n+2). \tag{6.5}$$

Beware that the first result holds for q-ary words, whereas the second result holds for binary words only. In Section 3.3 we established that both $L^{n,q}_{1,0,1}$ and $L^{n,2}_{0,1,1}$ are increasing on $\{1,...,n\}$ as a function of r. Therefore, it follows that

$$\begin{aligned} \min_{\mathbf{x}\in\mathcal{B}_q(n),\, r(\mathbf{x})>r} |\mathcal{V}_{1,0,1}(\mathbf{x})| &= \min_{r'\in\{r+1,\ldots,n\}} L^{n,q}_{1,0,1}(r') = L^{n,q}_{1,0,1}(r+1), \\ \min_{\mathbf{x}\in\mathcal{B}_2(n),\, r(\mathbf{x})>r} |\mathcal{V}_{0,1,1}(\mathbf{x})| &\geq \min_{r'\in\{r+1,\ldots,n\}} L^{n,2}_{0,1,1}(r') = L^{n,2}_{0,1,1}(r+1). \end{aligned}$$

This allows us to apply Theorem 6.2 and derive the following family of upper bounds which holds for each $0 \leq r \leq n-1$,

$$M_q(n,1,1) \leq \frac{q^{n-1}}{L^{n,q}_{1,0,1}(r+1)} + q\sum_{i=1}^{r}\binom{n-1}{i-1}(q-1)^{i-1}, \tag{6.6}$$

$$M_2(n,1,1) \leq \frac{2^{n+1}}{L^{n,2}_{0,1,1}(r+1)} + 2\sum_{i=1}^{r}\binom{n-1}{i-1}. \tag{6.7}$$

Here, the first bound uses $t' = 1$ and $t'' = 0$, while the second bound uses $t' = 0$ and $t'' = 1$. Next, we consider a brief example in which we evaluate these bounds for a specific set of parameters.

Example 6.4. Let $n = 10$, $q = 2$, $t = 1$ and $s = 1$. For instance, for $r = 3$ we obtain

$$M_2(10,1,1) \leq \frac{2^9}{L^{n,2}_{1,0,1}(4)} + 2\sum_{i=1}^{3}\binom{9}{i-1} = \frac{512}{4\cdot 7+4} + 2\cdot(1+9+36) = 108,$$

$$M_2(10,1,1) \leq \frac{2^{11}}{L^{n,2}_{0,1,1}(4)} + 2\sum_{i=1}^{3}\binom{9}{i-1} = \frac{2048}{-8+42+56} + 2\cdot(1+9+36) = 114.75.$$

Hence, for these specific parameters the upper bound based on the single deletion and single substitution set, $\mathcal{V}_{1,0,1}(\mathbf{x})$, outperforms the bound that is based on the single insertion and single substitution set, $\mathcal{V}_{0,1,1}(\mathbf{x})$. Notice that both bounds improve upon the Singleton bound from Theorem 6.1 which yields $M_2(10,1,1) \leq 2^{10-1-2\cdot 1} = 128$.

A detailed comparison of these sphere-packing upper bounds and other upper bounds on $M_q(n,1,1)$ can be found in Section 6.6.

6.4 Sphere-packing bounds using multiple clusters

This section aims to improve the bounds from the previous section. Firstly, we make two observations about Theorem 6.2 that will enable us to accomplish this improvement. Then, we present a second family of sphere-packing bounds and show that it performs at least as well as Theorem 6.2. Analogous to the previous section, this result is first stated in an implicit way for general t and s, and then it is made explicit for $t = s = 1$.

6.4.1 Improving the implicit sphere-packing bounds

Consider the following to observations about the proof of Theorem 6.2.

Remark 6.5. Firstly, from Chapter 2 we know that the size of the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ depends strongly on $r(\mathbf{x})$ for many sets of parameters t', t'' and s. For this reason, in the proof of Theorem 6.2 the set $\mathcal{B}_q(n)$ was partitioned into two clusters $\mathcal{A}^-$ and $\mathcal{A}^+$ containing the words with 'few' and 'many' runs, respectively. The number of codewords in $\mathcal{A}^+$ was bounded using a sphere-packing argument, whereas the number of codewords in $\mathcal{A}^-$ was trivially bounded by $|\mathcal{A}^-|$. Here, we propose to partition $\mathcal{B}_q(n)$ into more than two clusters. Moreover, we do not prescribe for which cluster a sphere-packing argument or a trivial bound is used. Instead, for each cluster the best of both bounds is used.

Secondly, we observe that the sphere-packing argument can be strengthened. In the proof of Theorem 6.2 the quantity $|\mathcal{C} \cap \mathcal{A}^+|$ was bounded as follows. We used that the size of the union of the sets $\mathcal{V}_{t',t'',s}(\mathbf{c})$ for $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+$ is bounded by

$$\left| \bigcup_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} \mathcal{V}_{t',t'',s}(\mathbf{c}) \right| \leq |\mathcal{B}_q(n - t' + t'')| = q^{n-t'+t''}.$$

Note that in doing so, we ignore that $\mathbf{c} \in \mathcal{A}^+$, i.e., that $r(\mathbf{c}) > r$ for some $1 \leq r \leq n$. Using Lemma 2.6 we also know that $r(\mathbf{y}) \geq r(\mathbf{c}) - 2(t' + s)$ for all $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{c})$. Together these observations imply that $r(\mathbf{y}) > r - 2(t' + s)$ for all $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{c})$ whenever $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+$. It follows that

$$\left| \bigcup_{\mathbf{c} \in \mathcal{C} \cap \mathcal{A}^+} \mathcal{V}_{t',t'',s}(\mathbf{c}) \right| \leq |\{\mathbf{x} \in \mathcal{B}_q(n - t' + t'') : r(\mathbf{x}) > r - 2(t' + s)\}|,$$

which is clearly an improvement if and only if $r - 2(t' + s) > 0$.

Based on the aforementioned two observations, we present the following theorem. The idea is to partition $\mathcal{B}_q(n)$ into k clusters denoted by $\mathcal{A}_j$ for $1 \leq j \leq k$. Each cluster will contain the words $\mathbf{x} \in \mathcal{B}_q(n)$ for which $r(\mathbf{x})$ falls within a certain range. Loosely speaking, $\mathcal{A}_1$ contains the words in $\mathcal{B}_q(n)$ with very few runs, $\mathcal{A}_2$ with slightly more runs, and $\mathcal{A}_k$ contains the words in $\mathcal{B}_q(n)$ with (almost) n runs. As far as the author is aware, the following result has not been stated before in literature.

Theorem 6.3. *Let $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$, $0 \leq s \leq n$ and $1 \leq k \leq n$ be integers. For each sequence of integers $0 = r_0 < r_1 < \cdots < r_k = n$, and for each pair of integers $0 \leq t', t'' \leq n$ such that $t' + t'' = t$, the following gives an upper bound on $M_q(n, t, s)$,*

$$M_q(n,t,s) \leq \sum_{j=1}^{k} \min \left\{ \frac{\sum_{r=a_j}^{b_j} q\binom{n-t'+t''-1}{r-1}(q-1)^{r-1}}{\min\limits_{\mathbf{x}\in\mathcal{A}_j} |\mathcal{V}_{t',t'',s}(\mathbf{x})|}, \sum_{r=r_{j-1}+1}^{r_j} q\binom{n-1}{r-1}(q-1)^{r-1} \right\}$$

where for $1 \leq j \leq k$, $a_j := \max\{1, r_{j-1}+1-2(t'+s)\}$, $b_j := \min\{n-t'+t'', r_j+2(t''+s)\}$ and $\mathcal{A}_j := \{\mathbf{x} \in \mathcal{B}_q(n) : r_{j-1}+1 \leq r(\mathbf{x}) \leq r_j\}$.

Proof. Let $\mathcal{C} \subseteq \mathcal{B}_q(n)$ be a t-indel s-substitution correcting code of maximum size. The idea of this proof is to partition $\mathcal{B}_q(n)$ into k clusters based on the number of runs of the words in $\mathcal{B}_q(n)$. These clusters are given by $\mathcal{A}_j$ for $1 \leq j \leq k$. Since $\mathcal{C}$ is maximal and these clusters form a partition of $\mathcal{B}_q(n)$ it follows that $M_q(n,t,s) = \sum_{j=1}^{k} |\mathcal{C} \cap \mathcal{A}_j|$. Then, we bound $|\mathcal{C} \cap \mathcal{A}_j|$ from above in two different ways to arrive at the desired result.

Before doing so, we show that the clusters $\mathcal{A}_j$ indeed form a partition of $\mathcal{B}_q(n)$. Consider the partition of the set $\{1, ..., n\}$ given by the subsets $\{r_0+1, ..., r_1\}$, $\{r_1+1, ..., r_2\}$, $\ldots$, $\{r_{k-1}+1, ..., r_k\}$. This is a partition because each element $\{1, ..., n\}$ in contained in precisely one of these subsets of the form $\{r_{j-1}+1, ..., r_j\}$, since the sequence $0 = r_0 < r_1 < \cdots < r_k = n$ is strictly increasing. This partition of $\{1, ..., n\}$ into k sets implies that the clusters $\mathcal{A}_j$ form a partition of $\mathcal{B}_q(n)$ as well. Next, let $1 \leq j \leq k$ and consider only the cluster $\mathcal{A}_j$.

As a first upper bound on $|\mathcal{C} \cap \mathcal{A}_j|$, we use the trivial bound,

$$|\mathcal{C} \cap \mathcal{A}_j| \leq |\mathcal{A}_j| = \sum_{r=r_{j-1}+1}^{r_j} q\binom{n-1}{r-1}(q-1)^{r-1}.$$

This follows directly from the definition of $\mathcal{A}_j$ and Lemma 2.7 which counts the number of words with $r_{j-1}+1 \leq r \leq r_j$ runs.

For the second upper bound on $|\mathcal{C} \cap \mathcal{A}_j|$, Let $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}_j$ and consider a word $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{c})$. Then we claim that $a_j \leq r(\mathbf{y}) \leq b_j$ with a_j and b_j as given in the statement of this theorem. In order to show this claim note that $\mathbf{c}$ has length n and thus $\mathbf{y}$ has length $n - t' + t''$. It follows that $1 \leq r(\mathbf{y}) \leq n - t' + t''$. Moreover, note that by Lemma 2.6 it holds that $r(\mathbf{c}) - 2(t'+s) \leq r(\mathbf{y}) \leq r(\mathbf{c}) + 2(t''+s)$, since $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{c})$. Together with $r_{j-1}+1 \leq r(\mathbf{c}) \leq r_j$ which follows from the definition of $\mathcal{A}_j$, we find that $r_{j-1}+1-2(t'+s) \leq r(\mathbf{y}) \leq r_j + 2(t''+s)$. Hence, we have proven the claim and we continue with a sphere-packing argument.

Namely, note that the sets $\mathcal{V}_{t',t'',s}(\mathbf{c})$ with $\mathbf{c} \in \mathcal{C} \cap \mathcal{A}_j$ are disjoint. This follows from Lemma 2.4, because $\mathcal{C}$ is a t-indel s-substitution correcting code. For this reason, the combined size of all these spheres satisfies

$$\sum_{\mathbf{c}\in\mathcal{C}\cap\mathcal{A}_j} |\mathcal{V}_{t',t'',s}(\mathbf{c})| = \left| \bigcup_{\mathbf{c}\in\mathcal{C}\cap\mathcal{A}_j} \mathcal{V}_{t',t'',s}(\mathbf{c}) \right| \leq |\{\mathbf{y} \in \mathcal{B}_q(n-t'+t''), a_j \leq r(\mathbf{y}) \leq b_j\}|,$$

where we used the aforementioned claim to upper bound the size of the union. On the other hand, it also holds that

$$\sum_{\mathbf{c}\in\mathcal{C}\cap\mathcal{A}_j} |\mathcal{V}_{t',t'',s}(\mathbf{c})| \;\geq\; |\mathcal{C}\cap\mathcal{A}_j| \cdot \min_{\mathbf{c}\in\mathcal{C}\cap\mathcal{A}_j} |\mathcal{V}_{t',t'',s}(\mathbf{c})| \;\geq\; |\mathcal{C}\cap\mathcal{A}_j| \cdot \min_{\mathbf{x}\in\mathcal{A}_j} |\mathcal{V}_{t',t'',s}(\mathbf{x})|,$$

where we used that $\mathcal{C}\cap\mathcal{A}_j \subseteq \mathcal{A}_j$ in the last inequality. By combining the last two steps, it follows that

$$|\mathcal{C}\cap\mathcal{A}_j| \leq \frac{|\{\mathbf{y}\in\mathcal{B}_q(n-t'+t''), a_j \leq r(\mathbf{y}) \leq b_j\}|}{\min_{\mathbf{x}\in\mathcal{A}_j} |\mathcal{V}_{t',t'',s}(\mathbf{x})|} = \frac{\sum_{r=a_j}^{b_j} q\binom{n-t'+t''-1}{r-1}(q-1)^{r-1}}{\min_{\mathbf{x}\in\mathcal{A}_j} |\mathcal{V}_{t',t'',s}(\mathbf{x})|}.$$

The numerator in this last expression is evaluated using Lemma 2.7. All in all, the two upper bounds on $|\mathcal{C}\cap\mathcal{A}_j|$ for $1\leq j\leq k$ give the desired result

$$\begin{aligned} M_q(n,t,s) &= \sum_{j=1}^{k} |\mathcal{C}\cap\mathcal{A}_j| \\ &\leq \sum_{j=1}^{k} \min\left\{ \frac{\sum_{r=a_j}^{b_j} q\binom{n-t'+t''-1}{r-1}(q-1)^{r-1}}{\min_{\mathbf{x}\in\mathcal{A}_j} |\mathcal{V}_{t',t'',s}(\mathbf{x})|}, \sum_{r=r_{j-1}+1}^{r_j} q\binom{n-1}{r-1}(q-1)^{r-1} \right\}. \end{aligned}$$

The last line concludes the proof. □

In order to evaluate the upper bounds from the previous theorem, we need an expression or lower bound for $\min_{\mathbf{x}\in\mathcal{A}_j} |\mathcal{V}_{t',t'',s}(\mathbf{x})|$. Given that no general expression for $|\mathcal{V}_{t',t'',s}(\mathbf{x})|$ is known to us, evaluating this family of bounds is complicated. However, this problem is not more difficult than in the previous section. The comments from the previous section about computing this minimum apply here as well.

Theorem 6.3 offers a family of bounds with even more 'freedom' compared to Theorem 6.2. Indeed, for each sequence $0 = r_0 < r_1 < \cdots < r_k = n$ of arbitrary length $1 \leq k \leq n$, and for each pair of $0 \leq t', t'' \leq n$ such that $t' + t'' = t$ we obtain a different bound. In order to get the smallest upper bound, this family of bounds should be optimized over all these sets of parameters. It is a priori unclear which set(s) of parameters provide(s) the best upper bound. Nevertheless, even if we consider only three clusters, significant improvements can be made compared to Theorem 6.2. This is illustrated in the following example.

Example 6.6. In Example 6.3 we established that $M_3(20,3,2) \leq 31223$ using Theorem 6.2. This result was obtained after optimization over the parameters which means that it is the best achievable upper bound using Theorem 6.2. Here, we show that it can be improved using Theorem 6.3 with only three clusters.

To this end, we consider three clusters, i.e., we set $k = 3$. We set $r_0 = 0$, $r_1 = 2$, $r_2 = 5$ and $r_3 = 20$. Moreover, we set $t' = 2$ and $t'' = 1$ so that $t = t' + t'' = 3$. We note that these values chosen because they yield an improvement over the previous results. There is nothing special about this choice. We state that other sets of parameters give improvements as well, and yield possibly even better results.

In order to evaluate Theorem 6.3 we compute using the script in Appendix B that

$$\begin{aligned}
\min_{\mathbf{x}\in\mathcal{A}_1} |\mathcal{V}_{2,1,2}(\mathbf{x})| &= \min_{\mathbf{x}\in\mathcal{B}_3(20):\, 1\leq r(\mathbf{x})\leq 2} |\mathcal{V}_{2,1,2}(\mathbf{x})| \;=\; 8475,\\
\min_{\mathbf{x}\in\mathcal{A}_2} |\mathcal{V}_{2,1,2}(\mathbf{x})| &= \min_{\mathbf{x}\in\mathcal{B}_3(20):\, 3\leq r(\mathbf{x})\leq 5} |\mathcal{V}_{2,1,2}(\mathbf{x})| \;= 21979,\\
\min_{\mathbf{x}\in\mathcal{A}_3} |\mathcal{V}_{2,1,2}(\mathbf{x})| &= \min_{\mathbf{x}\in\mathcal{B}_3(20):\, 6\leq r(\mathbf{x})\leq 20} |\mathcal{V}_{2,1,2}(\mathbf{x})| = 73374.
\end{aligned}$$

Furthermore, we compute $a_1 = a_2 = a_3 = 1$ and $b_1 = 8$, $b_2 = 11$ and $b_3 = 19$. Using these values it is possible to evaluate the upper bound in Theorem 6.3 as follows,

$$\begin{aligned}
M_3(20,3,2) &\leq \sum_{j=1}^{3} \min\left\{ \frac{\sum_{r=a_j}^{b_j} 3\binom{20-2+1-1}{r-1}(3-1)^{r-1}}{\min\limits_{\mathbf{x}\in\mathcal{A}_j} |\mathcal{V}_{2,1,2}(\mathbf{x})|}, \sum_{r=r_{j-1}+1}^{r_j} 3\binom{20-1}{r-1}(3-1)^{r-1} \right\}\\
&= \min\{\frac{3\sum_{r=1}^{8}\binom{18}{r-1}2^{r-1}}{8475}, 3\sum_{r=1}^{2}\binom{19}{r-1}2^{r-1}\}\\
&\quad + \min\{\frac{3\sum_{r=1}^{11}\binom{18}{r-1}2^{r-1}}{21979}, 3\sum_{r=3}^{5}\binom{19}{r-1}2^{r-1}\}\\
&\quad + \min\{\frac{3\sum_{r=1}^{19}\binom{18}{r-1}2^{r-1}}{73374}, 3\sum_{r=6}^{20}\binom{19}{r-1}2^{r-1}\}\\
&\approx \min\{1974, 117\} + \min\{11806, 211356\} + \min\{15804, \sim 3^{20}\}\\
&= 27727.
\end{aligned}$$

In the step with the approximation we rounded down the fractions. This is allowed, since each term forms an upper bound on the cardinality of the set $|\mathcal{C}\cap\mathcal{A}_j|$ (in the context of the proof of Theorem 6.3). The cardinality of this set is clearly integer-valued.

All in all, we conclude that $M_3(20,3,2) \leq 27727$. This forms an improvement on the best achievable results using Theorem 6.2, see Example 6.3. Furthermore, the Singleton bound from Theorem 6.1 gives $M_3(20,3,2) \leq 3^{20-3-2\cdot 2} = 3^{13} \approx 1.59\cdot 10^6$ which is significantly weaker as well. Note that we could possibly obtain an even better bound by choosing other parameters for t', t'', k and $r_1 < r_2 < \cdots < r_{k-1}$.

The previous example shows that there are instances in which Theorem 6.3 gives a strict improvement over Theorem 6.2. By the way in which both theorems are constructed, it is not surprising that Theorem 6.3 implies Theorem 6.2. In order to show this, we take $k = 2$ clusters and the strictly increasing sequence $0 = r_0 < r_1 < r_2 = n$. Hence, this results in a summation of two terms in Theorem 6.3. By choosing the second term in the minimum for $j = 1$ and the first term for $j = 2$, we get

$$\begin{aligned}
M_q(n,t,s) &\leq q\sum_{i=r_0+1}^{r_1}\binom{n-1}{i-1}(q-1)^{i-1} + \frac{\sum_{r=a_2}^{b_2} q\binom{n-t'+t''-1}{r-1}(q-1)^{r-1}}{\min_{\mathbf{x}\in\mathcal{A}_2(n)}|\mathcal{V}_{t',t'',s}(\mathbf{x})|}\\
&\leq q\sum_{i=1}^{r_1}\binom{n-1}{i-1}(q-1)^{i-1} + \frac{q^{n-t'+t''}}{\min\limits_{\substack{\mathbf{x}\in\mathcal{B}_q(n)\\ r(\mathbf{x})>r_1}}|\mathcal{V}_{t',t'',s}(\mathbf{x})|},
\end{aligned}$$

where the first inequality follows from the application of Theorem 6.3. The second inequality uses that $\mathcal{A}_2(n) = \{\mathbf{x} \in \mathcal{B}_q(n) : r(\mathbf{x}) > r_1\}$, and the observation that $\sum_{r=a_2}^{b_2} q\binom{n-t'+t''-1}{r-1}(q-1)^{r-1}$ counts the number of words with length $n-t'+t''$ and $a_2 \leq r \leq b_2$ runs. Clearly, this quantity is at most $q^{n-t'+t''}$, i.e., the total number of words with length $n-t'+t''$. This proves that Theorem 6.3 implies Theorem 6.2.

6.4.2 Explicit sphere-packing upper bounds for $M_q(n,1,1)$

In this subsection, the previous family of upper bounds is made explicit for $t = s = 1$. In order to make Theorem 6.3 explicit, we use the existing formulas for $|\mathcal{V}_{1,0,1}(\mathbf{x})|$ and $|\mathcal{V}_{0,1,1}(\mathbf{x})|$ again. Recall the expressions $L^{n,q}_{1,0,1}(r(\mathbf{x}))$ and $L^{n,q}_{1,0,1}(r(\mathbf{x}))$ from (6.4) & (6.5) for the cardinalities of $\mathcal{V}_{1,0,1}(\mathbf{x})$ and $\mathcal{V}_{0,1,1}(\mathbf{x})$, respectively. Given that these formulas are increasing on $\{1,...,n\}$ as a function of $r = r(\mathbf{x})$, it follows that

$$\begin{aligned}
\min_{\mathbf{x}\in\mathcal{A}_j} |\mathcal{V}_{1,0,1}(\mathbf{x})| &= \min_{r'\in\{r_{j-1}+1,\ldots,r_j\}} L^{n,q}_{1,0,1}(r') = L^{n,q}_{1,0,1}(r_{j-1}+1), \\
\min_{\mathbf{x}\in\mathcal{A}_j} |\mathcal{V}_{0,1,1}(\mathbf{x})| &\geq \min_{r'\in\{r_{j-1}+1,\ldots,r_j\}} L^{n,2}_{0,1,1}(r') = L^{n,2}_{0,1,1}(r_{j-1}+1).
\end{aligned}$$

Then, we can apply Theorem 6.3. For each sequence of integers $0 = r_0 < r_1 < ... < r_k = n$ we obtain the following two bounds,

$$M_q(n,1,1) \leq \sum_{j=1}^{k} \min\left\{ \frac{\sum_{r=a_j}^{b_j} q\binom{n-2}{r-1}(q-1)^{r-1}}{L^{n,q}_{1,0,1}(r_{j-1}+1)}, \sum_{r=r_{j-1}+1}^{r_j} q\binom{n-1}{r-1}(q-1)^{r-1} \right\}, \quad (6.8)$$

$$M_2(n,1,1) \leq \sum_{j=1}^{k} \min\left\{ \frac{\sum_{r=a_j}^{b_j} 2\binom{n}{r-1}}{L^{n,2}_{0,1,1}(r_{j-1}+1)}, \sum_{r=r_{j-1}+1}^{r_j} 2\binom{n-1}{r-1} \right\}. \quad (6.9)$$

where the first bound uses $t' = 1$ and $t'' = 0$, while the second bound uses $t' = 0$ and $t'' = 1$. Note that both bounds consist only of known quantities. Namely, by specifying a value for $1 \leq k \leq n$ and a sequence of integers $0 = r_0 < r_1 < ... < r_k = n$, the values for a_j and b_j can be determined using $a_j := \max\{1, r_{j-1}+1-2(t'+s)\}$ and $b_j := \min\{n-t'+t'', r_j+2(t''+s)\}$.

In Section 6.6, these two families of upper bounds will be compared against other upper bounds in this chapter.

6.5 Hypergraph and matching upper bound

In this section, we derive another upper bound on t-indel s-substitution correcting codes. Similar to the previous sections, we first derive an implicit result for general t and s. Thereafter, we make this result explicit for $t = s = 1$ using the known formulas for the cardinalities of $\mathcal{V}_{1,0,1}(\mathbf{x})$ and $\mathcal{V}_{0,1,1}(\mathbf{x})$.

The strategy to obtain these bounds has been detailed in Section 5.3 for t-indel correcting codes, and uses matchings in hypergraphs. These ideas are based on the work by Kulkarni and Kiyavash [8]. Recently, Smagloy *et al.* [11] used the same strategy to derive upper bounds on $M_q(n,1,1)$ and $M_2(n,1,s)$, see Section 6.1. Here, we aim to generalize the results from Section 5.3 and improve the existing bound on $M_q(n,1,1)$.

6.5.1 Implicit hypergraph and matching upper bound on $M_q(n,t,s)$

The hypergraph and matching strategy from Section 5.3 consists of three steps. We briefly repeat these steps below and explain how this strategy can be adapted to t-indel s-substitution correcting codes.

1. An upper bound on the maximum size of matching in a hypergraph is found using an integer linear programming approach. More precisely, an upper bound on the matching number $\nu(\mathcal{H})$ of a general hypergraph $\mathcal{H}$ was found in Lemma 5.6. This first step does not need to be adapted, since it does not involve error correcting codes.

2. The maximum size of a matching in a hypergraph was related to the maximum size of a t-indel correcting code in $\mathcal{B}_q(n)$. Then, the bound from the first step was used to obtain an implicit upper bound on $M_q(n,t,0)$ in Lemma 5.7. Analogously, we relate a maximum size matching to a maximum size t-indel s-substitution correcting code. In order to apply the result from the first step and derive an implicit upper bound for $M_q(n,t,s)$, we will need to alter the argument slightly and introduce some additional requirements. This second step will be covered in the remainder of this subsection.

3. The implicit upper bounds on $M_q(n,t,0)$ and $M_q(n,t,s)$ are made explicit using known results about $|\mathcal{D}_t(\mathbf{x})|$ and $|\mathcal{V}_{t',t'',s}(\mathbf{x})|$, respectively. This third step will be addressed in the next subsection.

Before stating the main result of this section, we make an observation about the second step. This observation will show that we cannot simply repeat the proof of Lemma 5.7 to derive an upper bound on $M_q(n,t,s)$. We explain why a generalization to t-indel s-substitution correcting codes makes the proof more difficult, and we argue how this difficulty can be solved.

Remark 6.7. The upper bound on $M_q(n,t,0)$ from Lemma 5.7 was found by applying Lemma 5.6. A vector $\mathbf{w} = (w(\mathbf{y}))_{\mathbf{y}\in\mathcal{B}_q(n-t)}$ was constructed that satisfies the two conditions from this lemma: 1) $\mathbf{w} \geq \mathbf{0}$ and 2) $\sum_{\mathbf{y}\in\mathcal{D}_t(\mathbf{x})} w(\mathbf{y}) \geq 1$ for all $\mathbf{x} \in \mathcal{B}_q(n)$. The vector $\mathbf{w}$ with $w(\mathbf{y}) = \frac{1}{|\mathcal{V}_{t,0,0}(\mathbf{y})|} = \frac{1}{|\mathcal{D}_t(\mathbf{y})|}$ and $\mathbf{y} \in \mathcal{B}_q(n-t)$ was shown to satisfy both conditions. A crucial step in proving that it satisfies the second condition is the following property, which was proven in [8, Lem. 4.1]. Namely, it holds that $|\mathcal{V}_{t,0,0}(\mathbf{y})| \leq |\mathcal{V}_{t,0,0}(\mathbf{x})|$, whenever $\mathbf{y} \in \mathcal{V}_{t,0,0}(\mathbf{x})$. This property implies the second condition, i.e., for any $\mathbf{x} \in \mathcal{B}_q(n)$,

$$\sum_{\mathbf{y}\in\mathcal{V}_{t,0,0}(\mathbf{x})} \frac{1}{|\mathcal{V}_{t,0,0}(\mathbf{y})|} \geq \sum_{\mathbf{y}\in\mathcal{V}_{t,0,0}(\mathbf{x})} \frac{1}{|\mathcal{V}_{t,0,0}(\mathbf{x})|} = 1.$$

In order to derive an upper bound for t-indel s-substitution correcting codes we may hope to use the same reasoning. In other words, we define a vector $\mathbf{w}$ and aim to show that it satisfies the two conditions of Lemma 5.6. As we will show in the next proof, in the setting of t-indel s-substitution correcting codes these two conditions are given by 1) $\mathbf{w} \geq \mathbf{0}$ and 2) $\sum_{\mathbf{y}\in\mathcal{V}_{t,0,s}(\mathbf{x})} w(\mathbf{y}) \geq 1$ for all $\mathbf{x} \in \mathcal{B}_q(n)$.

In line with the case of solely deletions, we might define the vector $\mathbf{w}$ with $w(\mathbf{y}) = \frac{1}{|\mathcal{V}_{t,0,s}(\mathbf{y})|}$ and $\mathbf{y} \in \mathcal{B}_q(n-t)$. The first condition can easily be proven, so the main effort

lies in showing the second condition. Again, the property $|\mathcal{V}_{t,0,s}(\mathbf{x})| \leq |\mathcal{V}_{t,0,s}(\mathbf{y})|$ for all $\mathbf{x} \in \mathcal{V}_{t,0,s}(\mathbf{y})$ would be sufficient to prove this second condition. Unfortunately, this property does not hold in general. For instance, let $q = 4$, $\mathbf{x}_1 = 010$, $\mathbf{x}_2 = 000$ and $\mathbf{y} = 0000$, then surely $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{V}_{1,0,1}(\mathbf{y})$. Furthermore, it can be easily verified that

$$\begin{aligned}
\mathcal{V}_{1,0,1}(\mathbf{y}) &= \{000, 001, 010, 100, 002, 020, 200, 003, 030, 300\}, \\
\mathcal{V}_{1,0,1}(\mathbf{x}_1) &= \{00, 01, 10, 11, 02, 20, 03, 30, 12, 21, 13, 31\}, \\
\mathcal{V}_{1,0,1}(\mathbf{x}_2) &= \{00, 01, 10, 02, 20, 03, 30\}.
\end{aligned}$$

It follows that $12 = |\mathcal{V}_{1,0,1}(\mathbf{x}_1)| > |\mathcal{V}_{1,0,1}(\mathbf{y})| = 10$, while $7 = |\mathcal{V}_{1,0,1}(\mathbf{x}_2)| < |\mathcal{V}_{1,0,1}(\mathbf{y})| = 10$. Consequently, the statement $|\mathcal{V}_{t,0,s}(\mathbf{x})| \leq |\mathcal{V}_{t,0,s}(\mathbf{y})|$ for all $\mathbf{x} \in \mathcal{V}_{t,0,s}(\mathbf{y})$ is not true in general. This shows that we cannot simply replace $w(\mathbf{x}) = \frac{1}{|\mathcal{V}_{t,0,0}(\mathbf{x})|}$ by $w(\mathbf{x}) = \frac{1}{|\mathcal{V}_{t,0,s}(\mathbf{x})|}$ and expect to derive an upper bound on $M_q(n,t,s)$ in a similar way as was done for the upper bound on $M_q(n,t,0)$.

The previous remark shows that we need to revise the strategy for the second step. In this thesis, we show that this can be done as follows. Although there exist examples of $\mathbf{x}$ and $\mathbf{y}$ such that $|\mathcal{V}_{t,0,s}(\mathbf{x})| < |\mathcal{V}_{t,0,s}(\mathbf{y})|$ for $\mathbf{x} \in \mathcal{V}_{t,0,s}(\mathbf{y})$, it is possible to show that, loosely speaking, it cannot happen that $|\mathcal{V}_{t,0,s}(\mathbf{x})| \ll |\mathcal{V}_{t,0,s}(\mathbf{y})|$. By exploiting this idea, we manage to construct a different vector $\mathbf{w}$ which does satisfy both conditions from Lemma 5.6. This leads to the upper bound that we present in the following theorem. This theorem provides an upper bound on $M_q(n,t,s)$ in terms of a lower bound for $|\mathcal{V}_{t',t'',s}(\mathbf{x})|$. To the best of our knowledge, the following result was not yet stated before in literature for general $t \geq 0$ and $s \geq 0$.

Theorem 6.4. *Let $n \geq 1$, $q \geq 2$, $0 \leq t \leq n$ and $0 \leq s \leq n$ be integers. For each pair of integers $t', t'' \geq 0$ such that $t + t'' = t$, and for each non-decreasing function $L : \{1, ..., n\} \to \mathbb{R}_{\geq 1}$ that satisfies $L(r(\mathbf{x})) \leq |\mathcal{V}_{t',t'',s}(\mathbf{x})|$ for all $\mathbf{x} \in \mathcal{B}_q(n)$, the following gives an upper bound on $M_q(n,t,s)$,*

$$M_q(n,t,s) \leq \sum_{r=1}^{n-t'+t''} \frac{q\binom{n-t'+t''-1}{r-1}(q-1)^{r-1}}{L(c(r))},$$

where $c(r) = \max\{1, r - 2(t'' + s)\}$.

Proof. The reasoning of this proof will be similar to that of Lemma 5.7. Namely, the problem of bounding the maximum size of a t-indel s-substitution correcting code will be translated to the equivalent problem of bounding the size of a maximum sized matching in a hypergraph. Subsequently, a vector $\mathbf{w}^*$ is constructed that satisfies the two conditions of Lemma 5.6. The conclusion of Lemma 5.6 leads to an upper bound of $M_q(n,t,s)$. Throughout the remainder of this proof $\mathbf{x}$ will denote a word of length n and $\mathbf{y}$ of length $n - t' + t''$.

Define the following hypergraph $\mathcal{H} := (\mathcal{B}_q(n - t' + t''), \{\mathcal{V}_{t',t'',s}(\mathbf{x}) : \mathbf{x} \in \mathcal{B}_q(n)\})$. This hypergraph is well-defined because the words in $\mathcal{V}_{t',t'',s}(\mathbf{x})$ have length $n - t' + t''$. First, we show that $M_q(n,t,s) = \nu(\mathcal{H})$. To this end, let $\mathcal{C}$ be a t-indel s-substitution correcting code. By Lemma 2.4 this is equivalent to stating that each pair of hyperedges in the set $\mathcal{M} := \{\mathcal{V}_{t',t'',s}(\mathbf{c}) : \mathbf{c} \in \mathcal{C}\}$ is disjoint. In turn, this is the same as $\mathcal{M}$ being a matching

of $\mathcal{H}$. Therefore, it follows that the code $\mathcal{C}$ and matching $\mathcal{M}$ have equal size. Hence, a maximum size t-indel s-substitution correcting code induces a matching of the same size and vice versa. In short, this means that $M_q(n,t,s) = \nu(\mathcal{H})$.

Next, we apply Lemma 5.6. This means that we construct a vector $\mathbf{w} = (w(\mathbf{y}))_{\mathbf{y}\in\mathcal{B}_q(n-t'+t'')}$ which satisfies the conditions: 1) $\mathbf{w} \geq \mathbf{0}$ and 2) $\sum_{\mathbf{y}\in\mathcal{V}_{t',t'',s}(\mathbf{x})} w(\mathbf{y}) \geq 1$ for all $\mathbf{x} \in \mathcal{B}_q(n)$. To this end, define the vector $\mathbf{w}^*$ as follows

$$w^*(\mathbf{y}) = \frac{1}{L(c(r(\mathbf{y})))} = \begin{cases} \frac{1}{L(r(\mathbf{y})-2(t''+s))} & \text{if } r(\mathbf{y}) > 2(t''+s), \\ \frac{1}{L(1)} & \text{if } r(\mathbf{y}) \leq 2(t''+s), \end{cases}$$

for all $\mathbf{y} \in \mathcal{B}_q(n-t'+t'')$ and $c(r) = \max\{1, r-2(t''+s)\}$. We show that $\mathbf{w}^*$ is well-defined and that $\mathbf{w}^*$ satisfies the two conditions.

The vector $\mathbf{w}^*$ is well-defined if $c(r(\mathbf{y})) \in \{1,...,n\}$ for each $\mathbf{y} \in \mathcal{B}(n-t'+t'')$, because in that case $c(r(\mathbf{y}))$ is an element of the domain of L. Let $\mathbf{y} \in \mathcal{B}(n-t'+t'')$, then it holds that $c(r(\mathbf{y}))$ is integer-valued. Furthermore, we have $c(r(\mathbf{y})) \geq 1$ and

$$c(r(\mathbf{y})) = \max\{1, r(\mathbf{y}) - 2(t''+s)\} \leq \max\{1, n-t'+t''-2(t''+s)\} \leq n.$$

Moreover, the L is strictly positive function, so division by zero does not occur. Hence, we conclude that $\mathbf{w}^*$ is well-defined.

The first condition $\mathbf{w}^* \geq \mathbf{0}$ is satisfied, because L is a strictly positive function. For the second condition, let $\mathbf{x} \in \mathcal{B}_q(n)$ and $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{x})$. Recall Lemma 2.6 which relates $r(\mathbf{x})$ and $r(\mathbf{y})$. This lemma states that $r(\mathbf{y}) - 2(t''+s) \leq r(\mathbf{x})$, which gives

$$c(r(\mathbf{y})) = \max\{1, r(\mathbf{y}) - 2(t''+s)\} \leq r(\mathbf{x}).$$

This implies that $L(c(r(\mathbf{y}))) \leq L(r(\mathbf{x}))$, since L is a non-decreasing function by definition. All in all, we obtain the second condition for $\mathbf{w}^*$,

$$\sum_{\mathbf{y}\in\mathcal{V}_{t',t'',s}(\mathbf{x})} w^*(\mathbf{y}) = \sum_{\mathbf{y}\in\mathcal{V}_{t',t'',s}(\mathbf{x})} \frac{1}{L(c(r(\mathbf{y})))} \geq \sum_{\mathbf{y}\in\mathcal{V}_{t',t'',s}(\mathbf{x})} \frac{1}{L(r(\mathbf{x}))} \overset{(*)}{=} \frac{|\mathcal{V}_{t',t'',s}(\mathbf{x})|}{L(r(\mathbf{x}))} \overset{(**)}{\geq} 1,$$

where we used in $(*)$ that the summands do not depend on $\mathbf{y}$, and in $(**)$ that $L(r(\mathbf{x}))$ bounds $|\mathcal{V}_{t',t'',s}(\mathbf{x})|$ from below. We conclude that $\mathbf{w}^*$ satisfies the two aforementioned conditions. Therefore, we can apply Lemma 5.6. This lemma states $\nu(\mathcal{H}) \leq \mathbf{1}^\top \mathbf{w}^*$. By combining the previous results from this proof the desired upper bound is obtained,

$$\begin{aligned} M_q(n,t,s) &= \nu(\mathcal{H}) \\ &\leq \mathbf{1}^\top \mathbf{w}^* \\ &= \sum_{\mathbf{y}\in\mathcal{B}_q(n-t'+t')} w^*(\mathbf{y}) \\ &= \sum_{\mathbf{y}\in\mathcal{B}_q(n-t'+t')} \frac{1}{L(c(r(\mathbf{y})))} \\ &= \sum_{r=1}^{n-t'+t''} \sum_{\substack{\mathbf{y}\in\mathcal{B}_q(n-t'+t') \\ r(\mathbf{y})=r}} \frac{1}{L(c(r))} \\ &= \sum_{r=1}^{n-t'+t''} \frac{q\binom{n-t'+t''-1}{r-1}(q-1)^{r-1}}{L(c(r))}. \end{aligned}$$

In the last equality we used Lemma 2.7 which counts the number of words in $\mathcal{B}_q(n-t'+t')$ with a given number of runs. The last chain of (in)equalities concludes the proof. □

Next, we make two remarks about this theorem.

1. In its current form Theorem 6.4 is implicit because it depends on a non-decreasing lower bound L for $|\mathcal{V}_{t',t'',s}(\mathbf{x})|$ with $\mathbf{x} \in \mathcal{B}_q(n)$. Observe that such a non-decreasing lower bound is given in Lemma 3.5 and can thus be used to make this theorem explicit. Since this lower bound for general t and s was shown to be weak for small n, we do not continue with this general bound here. Instead, we focus on the case $t = s = 1$ where stronger lower bounds are known. This will be done in more detail in Subsection 6.5.2.

2. The requirement that L is a non-decreasing lower bound might seem restrictive, but this is not necessarily the case. Observe that the lower bounds for $|\mathcal{V}_{t',t'',s}(\mathbf{x})|$ from Chapter 3 all satisfy this property. Indeed, it holds for the lower bounds on $|\mathcal{V}_{t,0,s}(\mathbf{x})|$ and $|\mathcal{V}_{0,1,1}(\mathbf{x})|$ of Lemmas 3.5 & 3.3, as well as the expression for $|\mathcal{V}_{1,0,1}(\mathbf{x})|$ in Lemma 3.2.

6.5.2 Explicit hypergraph and matching upper bounds on $M_q(n,1,1)$

Theorem 6.4 can be made explicit for $t = s = 1$, because we have a known expression for $|\mathcal{V}_{1,0,1}(\mathbf{x})|$ in terms of $r(\mathbf{x})$ and a lower bound for $|\mathcal{V}_{0,1,1}(\mathbf{x})|$ in terms of $r(\mathbf{x})$ as well. Namely, using Lemmas 3.2 & 3.4, we obtain

$$|\mathcal{V}_{1,0,1}(\mathbf{x})| = L^{n,q}_{1,0,1}(r(\mathbf{x})) := \begin{cases} (n-1)(q-1)+1 & \text{if } r(\mathbf{x})=1, \\ r(\mathbf{x})\left((n-2)(q-1)-1\right)+q+2 & \text{if } r(\mathbf{x})\geq 2, \end{cases} \tag{6.10}$$

$$|\mathcal{V}_{0,1,1}(\mathbf{x})| \geq L^{n,2}_{0,1,1}(r(\mathbf{x})) := -\frac{1}{2}r(\mathbf{x})^2 + (n+\frac{1}{2})r(\mathbf{x}) + \frac{1}{2}(n^2+n+2). \tag{6.11}$$

Here, (6.10) holds for $\mathbf{x} \in \mathcal{B}_q(n)$, whereas (6.11) holds only for $\mathbf{x} \in \mathcal{B}_2(n)$. In Section 3.3 it was established that both $L^{n,q}_{1,0,1}$ and $L^{n,2}_{0,1,1}$ are non-decreasing as a function of $r = r(\mathbf{x})$. Therefore, the conditions of Theorem 6.4 are satisfied.

Using $t' = 1$, $t'' = 0$ and $L^{n,q}_{1,0,1}$, we obtain for $n \geq 4$ and $q \geq 2$,

$$\begin{aligned} M_q(n,1,1) &\leq \sum_{r=1}^{n-1} \frac{q\binom{n-2}{r-1}(q-1)^{r-1}}{L^{n,q}_{1,0,1}(\max\{1,r-2\})} \\ &= \sum_{r=1}^{3} \frac{q\binom{n-2}{r-1}(q-1)^{r-1}}{L^{n,q}_{1,0,1}(1)} + \sum_{r=4}^{n-1} \frac{q\binom{n-2}{r-1}(q-1)^{r-1}}{L^{n,q}_{1,0,1}(r-2)} \\ &= \frac{q+q(q-1)(n-2)+\frac{1}{2}q(n-2)(n-1)(q-1)^2}{(n-1)(q-1)+1} \\ &\quad + \sum_{r=4}^{n-1} \frac{q\binom{n-2}{r-1}(q-1)^{r-1}}{(r-2)\left((n-2)(q-1)-1\right)+q+2}. \end{aligned} \tag{6.12}$$

Note that this bound cannot be improved by choosing a different lower bound on $|\mathcal{V}_{1,0,1}(\mathbf{x})|$, because $L^{n,q}_{1,0,1}$ provides the exact expression for $|\mathcal{V}_{1,0,1}(\mathbf{x})|$. Similarly, using $t' = 0$, $t'' = 1$ and $L^{n,2}_{0,1,1}$ we get the following bound for $n \geq 4$ and $q = 2$,

$$\begin{aligned} M_2(n,1,1) &\leq \sum_{r=1}^{n+1} \frac{2\binom{n}{r-1}}{L^{n,2}_{0,1,1}(\max\{1,r-4\})} \\ &= \frac{2+2n+n(n+1)+\frac{1}{3}n(n-1)(n-2)}{\frac{1}{2}n^2+n+2} + \\ &\quad \sum_{r=5}^{n+1} \frac{2\binom{n}{r-1}}{-\frac{1}{2}(r-4)^2+(n+\frac{1}{2})(r-4)+\frac{1}{2}(n^2+n+2)} \\ &= \frac{2(n^3+11n+6)}{3(n^2+2n+4)} + \sum_{r=1}^{n-3} \frac{4\binom{n}{r+3}}{-r^2+(2n+1)r+n^2+n+2}. \end{aligned} \tag{6.13}$$

This bound can also not be improved by choosing a different lower bound on $|\mathcal{V}_{0,1,1}(\mathbf{x})|$. This follows from the proof of Lemma 3.3 in which we showed that $L^{n,2}_{0,1,1}(r)$ is tight for all $1 \leq r \leq n$. We remark that (6.12) & (6.13) do not give neat expressions, but both bounds are explicit. Therefore, they can be easily evaluated for small n and q.

In the next example, we show that (6.12) gives a significant improvement over the existing bound from (6.1) for a particular set of parameters.

Example 6.8. In Example 6.1 we found that $M_3(10,1,1) \leq 858$ according to the existing bound (6.1). Using the bound (6.12) we obtain

$$M_3(10,1,1) \leq \frac{3+48+432}{19} + \sum_{r=4}^{9} \frac{3\binom{8}{r-1}\cdot 2^{r-1}}{17(r-2)+5} \approx 25.42 + 294.82 = 320.24$$

Therefore, we conclude that $M_3(10,1,1) \leq 320$. Indeed, our result is significantly better than the existing bound $M_3(10,1,1) \leq 858$. This could be expected, because (6.1) was derived in [11] using the same technique as (6.12). However, we state that for (6.1) several terms were bounded to arrive at a neater formula, at the cost of a weaker result.

6.6 Non-asymptotic comparison of upper bounds on $M_q(n,1,1)$

In this chapter we reviewed and derived several upper bounds on $M_q(n,t,s)$. Next, we restrict our attention to single-indel single-substitution correcting codes and compare these bounds non-asymptotically.

To be precise, we compare nine different bounds. First of all, the two existing bounds (6.1) and (6.2) from Section 6.1 are considered. Moreover, we take the the Singleton bound from Section 6.2 into account. For the remaining six bounds, we distinguish between the bounds (6.6), (6.8) and (6.12) that were derived using the cardinality of the set $\mathcal{V}_{1,0,1}(\mathbf{x})$ and (6.7), (6.9) and (6.13) that used (a lower bound on) $\mathcal{V}_{0,1,1}(\mathbf{x})$. In other words, we distinguish between a single deletion and a single substitution versus a single insertion and a single substitution. Recall that these bounds have been derived using the sphere-packing approach with two clusters, sphere-packing with k clusters, and the hypergraph and matching approach, respectively. In Figure 6.1, these upper bounds are compared for $q = 2$. Next, we list several remarks and observations about this figure.

1. Figure 6.1 compares the bounds in terms of the relative redundancy. This means that the upper bounds of this chapter induce lower bounds on $1-\frac{1}{n}\log_2(M_2(n,1,1))$. The feasible region for $1-\frac{1}{n}\log_2(M_2(n,1,1))$ based on these bounds is indicated in grey. As stated before, the differences on a logarithmic scale seems smaller than on a linear scale.
2. Notice that the best upper bound on $M_2(n,1,1)$ is not given by a single bound. For instance, the bound (6.9) performs best when $n=6$, while (6.12) performs best for $n=25$. Recall that (6.9) was derived using the set $\mathcal{V}_{0,1,1}(\mathbf{x})$, while (6.9) was derived using $\mathcal{V}_{1,0,1}(\mathbf{x})$. The fact that the bounds from multiple approaches yield good results for different parameters, indicates that it has been worth the effort to consider multiple strategies for upper bounds on $M_q(n,t,s)$.
3. We observe that the bounds (6.9) and (6.12) outperform the existing bounds (6.1) and (6.2), as well as the Singleton bound (6.3) for $6 \leq n \leq 25$. This shows that we constructed multiple bounds that improve upon existing results in literature, at least for this specific sets of parameters.
4. It is striking that (6.9) with the k cluster approach does not seem to improve much on the 2 cluster approach from (6.7) for $8 \leq n \leq 22$. For $6 \leq n \leq 7$ and $23 \leq n \leq 25$ improvements can be observed, although they are minor. For instance, we found that $M_2(23,1,1) \leq 46243$ with (6.9) and $M_2(23,1,1) \leq 46726$ with (6.7). Note that these results use (lower bounds on) the cardinality of the single insertion and single substitution set $\mathcal{V}_{0,1,1}(\mathbf{x})$. On the other hand, we observe that the k cluster approach of (6.8) strongly outperforms the 2 cluster approach of (6.6) for $11 \leq n \leq 25$. These results were obtained using the single deletion and single substitution set $\mathcal{V}_{1,0,1}(\mathbf{x})$. A potential explanation for the differences between the results with either $\mathcal{V}_{1,0,1}(\mathbf{x})$ or $\mathcal{V}_{0,1,1}(\mathbf{x})$ is given below.

   Notice that the clustering approach was originally introduced in Chapter 5 for deletions only, in order to improve the sphere-packing argument in Lemma 5.5. For insertions only, it was unnecessary to consider multiple clusters. With this in mind, it is not surprising that introducing more clusters is beneficial when using $\mathcal{V}_{1,0,1}(\mathbf{x})$, and to a lesser extent when using $\mathcal{V}_{0,1,1}(\mathbf{x})$.

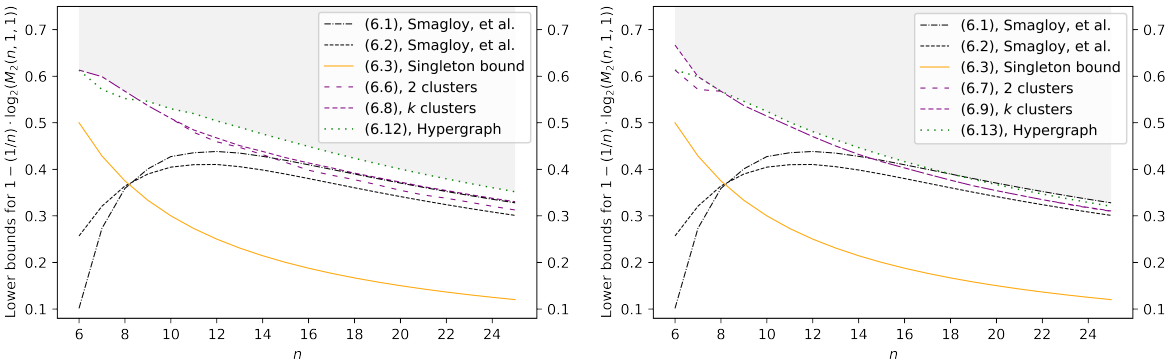


Figure 6.1: Comparison of nine upper bounds for $M_2(n,1,1)$ based on $|\mathcal{V}_{1,0,1}(\mathbf{x})|$ (left) and $|\mathcal{V}_{0,1,1}(\mathbf{x})|$ (right). The first three bounds are not based on either of these sets, so they are equal in both figures. The bounds (6.6) and (6.7) have been optimized in terms of $0 \leq r \leq n-1$. The bounds (6.8) and (6.9) have been optimized over all sequences $0=r_0<r_1<\cdots<r_k=n$ with $k \leq 5$.

6.7 Non-asymptotic comparison of lower and upper bounds on $M_2(n, 1, 1)$

In this thesis, we constructed multiple lower and upper bounds on $M_q(n, t, s)$. Here, we briefly compare the best lower and upper bounds on $M_2(n, 1, 1)$ for small n.

On the lower bound side, we consider Lemma 4.7 based on the Caro-Wei theorem. This implicit bound was shown to perform at least as well as all other lower bounds from Chapter 4. On the upper bound side, we consider the sphere-packing bound with k-clusters from (6.9) and the hypergraph and matching bound from (6.12). In the previous section, we established that these two bounds on $M_2(n, 1, 1)$ perform well in comparison to the other upper bounds. In Figure 6.2, these three bounds on $M_2(n, 1, 1)$ are considered for $4 \leq n \leq 18$. We make several observations about this figure.

1. First of all, the lower bound from Lemma 4.7 and upper bound (6.9) agree only for $n = 4$. Therefore, it holds in this case that $1 - \frac{1}{4}\log_2(M_2(4, 1, 1)) = 0.75$, which is equivalent to stating that $M_2(4, 1, 1) = 4$. In Subsection 1.6.1, we showed that the repetition code $\mathcal{C}_{4,2}$ is a single-indel single-substitution correcting code that attains this size. For $5 \leq n \leq 18$, there is a strict different between the lower bound and the (best) upper bound. In these cases, the exact value of $M_2(n, 1, 1)$ is not known and thus we can only rely on bounds.
2. Given the logarithmic scale in Figure 6.2 the differences between the best lower and upper bounds are significant for $n \geq 5$. For instance, we find that
$$0.426 \leq 1 - \frac{1}{18}\log_2(M_2(18, 1, 1)) \leq 0.739.$$
This corresponds to the bounds $26 \leq M_2(18, 1, 1) \leq 1328$. Despite, this major gap, we have still shown improvements upon bounds in literature. However, this gap also suggests there is still room for improving the lower and/or upper bounds. We note that this is likely the case for the lower bound from Lemma 4.7. This lower bound was based on the Gilbert-Varshamov bound for $M_q(n, 0, s)$ and Levenshtein's bound for $M_q(n, t, 0)$. However, we have seen in Section 4.1 that these two bounds are far from optimal in some instances (cf. Examples 4.1 & 4.2). Hence, it is plausible that Lemma 4.7 is also non-optimal for some sets of parameters.

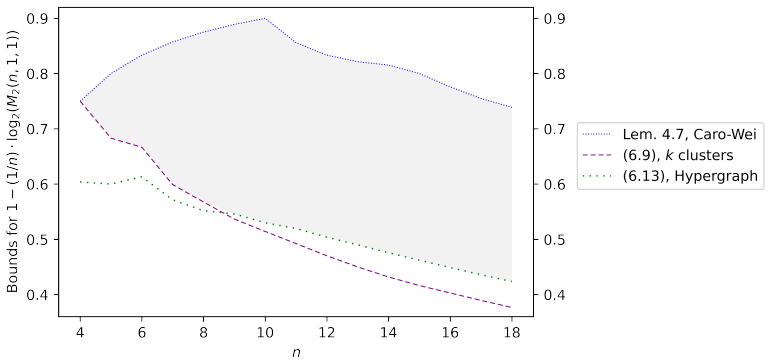


Figure 6.2: Comparison of three lower and upper bounds on $M_2(n, 1, 1)$, in terms of the relative redundancy $1 - \frac{1}{n}\log_q(M_2(n, 1, 1))$. The feasible region for $1 - \frac{1}{n}\log_q(M_2(n, 1, 1))$ is indicated in grey.

6.8 Asymptotic implications

At last, we consider several asymptotic implications of the upper bounds on $M_q(n,t,s)$ from this chapter. Firstly, we show that the asymptotic redundancy of an optimal t-indel s-substitution correcting code is at least logarithmic in n. Secondly, we consider briefly the asymptotic behavior of Theorem 6.1 when t and s grow linearly with n.

In particular, for fixed q, t and s and $n \to \infty$, we show that a t-indel s-substitution correcting code of maximal size has an asymptotic redundancy of at least $(t+s)\log_q(n) + o(\log_q(n))$. This result is obtained by combining the upper bound on $M_q(n,t,s)$ from Theorem 6.2 with the lower bound on the cardinality of $\mathcal{V}_{t,0,s}(\mathbf{x})$ from Lemma 3.5.

Theorem 6.5. *Let $q \geq 2$ be an integer. For non-negative integers t and s such that $t+s \geq 1$, the following holds*

$$\liminf_{n\to\infty} \frac{n - \log_q(M_q(n,t,s))}{(t+s)\log_q(n)} \geq 1.$$

The proof of this result is rather lengthy and not insightful for the remainder of this section. Hence, we have relegated the proof to Appendix A.8. As a direct consequence of the previous theorem we find the following asymptotic result.

Corollary 6.6. *A maximal size t-indel s-substitution correcting code has an asymptotic redundancy of at least $(t+s)\log_q(n) + o(\log_q(n))$.*

Let us put this result into perspective. Recall that we have already shown that a maximal size t-indel s-substitution correcting code has an asymptotic redundancy of at most $2(t+s)\log_q(n) + o(\log_q(n))$ in Corollary 4.11. In other words, we have thus established that the asymptotic redundancy of a maximal size t-indel s-substitution correcting code falls between $(t+s)\log_q(n)$ and $2(t+s)\log_q(n) + o(\log_q(n))$. Intuitively, this means that the number of redundant symbols needed to correct a fixed number of t indels and s substitutions grows logarithmically in n.

Our result extends a result from Levenshtein [7] from binary to q-ary codes. Namely, Levenshtein showed that a binary t-indel s-substitution correcting code of maximal size has an asymptotic redundancy between $(t+s)\log_2(n)$ and $2(t+s)\log_2(n) + o(\log_2(n))$. The results by Levenshtein are purely asymptotic, which means that they have been derived from asymptotic bounds on $M_2(n,t,s)$. In contrast, Corollaries 4.11 & 6.6 are a consequence of non-asymptotic bounds on $M_q(n,t,s)$.

Next, we consider briefly the asymptotic setting where $t = \tau n$ and $s = \sigma n$ for $\tau, \sigma \in [0,1]$, and n tends to infinity. In this setting, we define the (inferior) asymptotic relative redundancy function

$$R_q^-(\tau,\sigma) := \liminf_{n\to\infty} \left(1 - \frac{1}{n}\log_q(M_q(n, \lfloor \tau n \rfloor, \lfloor \sigma n \rfloor))\right) \tag{6.14}$$

The reason that this function is defined using a limit inferior instead of a limit is to ensure that (6.14) is well-defined. From (4.19) and (6.14) it is immediate that $R_q^-(\tau,\sigma) \leq$

$R_q^+(\tau,\sigma)$. The Singleton bound on $M_q(n,t,s)$ from Theorem 6.1 gives,

$$\begin{aligned} R_q^-(\tau,\sigma) &\geq \liminf_{n\to\infty} \left(1 - \frac{1}{n}\log_q(q^{n-\lfloor \tau n\rfloor - 2\lfloor \sigma n\rfloor})\right) \\ &= \liminf_{n\to\infty} \left(1 - \frac{n-\lfloor \tau n\rfloor - 2\lfloor \sigma n\rfloor}{n}\right) \\ &= \tau + 2\sigma, \end{aligned} \tag{6.15}$$

where we used that $\frac{\lfloor \tau n\rfloor}{n} \to \tau$ and $\frac{\lfloor \sigma n\rfloor}{n} \to \sigma$ for $n \to \infty$.

Example 6.9. Let $\tau = \frac{1}{25}$ and $\sigma = \frac{1}{50}$, then it follows from the Singleton bound that $R_2^-(\frac{1}{25}, \frac{1}{50}) \geq \frac{1}{25} + 2\cdot\frac{1}{50} = 0.08$. Intuitively, this means that asymptotically at least 8% of the symbols have to be redundant when correcting up to three errors (one of each type) per 50 symbols.

For comparison, recall from Example 4.10 that $R_2^+\left(\frac{1}{25}, \frac{1}{50}\right) \leq 0.7645$. Hence, we find that $0.08 \leq R_2^-(\frac{1}{25}, \frac{1}{50}) \leq R_2^-(\frac{1}{25}, \frac{1}{50}) \leq 0.7645$, which shows that there is quite a large gap between the asymptotic lower bounds. This is not surprising, since it is known in literature that the Singleton bound gives rather weak results in this asymptotic setting when $\tau = 0$ [6, Sec. 4.5].

It would be interesting to investigate what can be inferred about $R_q^-(\tau,\sigma)$ based on the other upper bounds on $M_q(n,t,s)$ from this chapter. However, we state that for these bounds it is rather complicated to evaluate the limit inferior from (6.14). For this reason, we have only been able to derive the bound from (6.15).

Based on discussions in literature about the bounds in Chapter 5, it seems plausible that improvements upon (6.15) can be found. Namely, in [6, Sec. 4.5] it was found that the sphere-packing Hamming bound improves upon the singleton bound for s-substitution correcting codes in this asymptotic regime. Moreover, in [8, 9, 38] it was shown that the sphere-packing upper bounds as well as the matching in hypergraph upper bounds on $M_q(n,t,0)$ both outperform the Singleton bound for t-indel correcting codes, in this asymptotic regime.

7
Conclusions and discussions

In this thesis, we aimed to answer how existing bounds on the maximum size of t-indel correcting codes and s-substitution correcting codes can be generalized to construct bounds on the maximum size of t-indel s-substitution correcting codes. In our research we treated lower bounds and upper bounds separately, because their derivations were based on different concepts. The main results and contributions with respect to the research question are discussed below. Moreover, we identify three directions for further research.

On the lower bound side, we conclude the following about the research question.

1. It was established that each $(t + 2s)$-indel correcting code is also a t-indel s-substitution correcting code in Lemma 2.5. Therefore, any existing lower bound on $M_q(n, t + 2s, 0)$ also forms a lower bound on $M_q(n, t, s)$ in a trivial way. Note that such an implication does not hold for upper bounds.

2. In Section 4.2 it was found that the Gilbert-Varshamov bound for s-substitution correcting codes and a lower bound for t-indel correcting codes by Levenshtein can be generalized to a lower bound on $M_q(n, t, s)$. Namely, it was recognized that both bounds can be proven using a similar approach which enables the generalization. By translating the problem of finding a large code in $\mathcal{B}_q(n)$ to the finding a large clique in a graph, existing bounds on the maximal size of cliques could be used to formulate bounds on the maximum size of a code. As a result, we established an implicit lower bound on $M_q(n, t, s)$ in Lemma 4.3 and made this bound explicit in Theorem 4.4 at the cost of obtaining a weaker bound.

3. In Section 4.3, we identified an improvement for the previous approach by using a different lower bound on the maximal size of a clique. This led to a second implicit lower bound on $M_q(n, t, s)$ in Lemma 4.7, which was also made explicit in Theorem 4.9. The second implicit bound was shown to improve upon the first implicit bound. In a non-asymptotic comparison it was found that none of the two explicit bound outperforms the other explicit bounds for general parameters n, q, t and s. Both explicit bounds improved upon the Levenshtein's lower bound on $M_q(n, t + 2s)$ in most considered instances.

On the upper bound side, we identified three types of bounds on $M_q(n,t,0)$ and $M_q(n,0,s)$ that can generalized to construct bounds on $M_q(n,t,s)$.

1. Arguably the most straightforward generalization was found by combining the two existing Singleton upper bounds for $M_q(n,t,0)$ and $M_q(n,0,s)$ into an upper bound for $M_q(n,t,s)$. This Singleton bound on $M_q(n,t,s)$ has the advantage of being explicit and easy-to-compute for all parameters n, q, t and s. However, in a non-asymptotic comparison for single-indel single-substitution correcting codes this bound was shown to be weaker than most other upper bounds.

2. A second generalization was found with the sphere-packing approach in Section 6.3. In literature, this approach has been used extensively to derive upper bounds on $M_q(n,t,0)$ and $M_q(n,0,s)$. We constructed an implicit family of sphere-packing upper bounds for $M_q(n,t,s)$ in Theorem 6.2. This generalization led to implicit bounds, since the sets $\mathcal{V}_{t',t'',s}(\mathbf{x})$ are not of equal size for all $\mathbf{x} \in \mathcal{B}_q(n)$ and because no expression for the cardinality of $\mathcal{V}_{t',t'',s}(\mathbf{x})$ is known in general, to the best of our knowledge. On the other hand, the cardinalities of $\mathcal{V}_{1,0,1}(\mathbf{x})$ and $\mathcal{V}_{0,1,1}(\mathbf{x})$ are known for all $\mathbf{x} \in \mathcal{B}_q(n)$. This enabled us to make Theorem 6.2 explicit and derive two families of explicit upper bounds on $M_q(n,1,1)$. Furthermore, two improvements for Theorem 6.2 were identified in Section 6.4. This led to a second family of implicit sphere-packing upper bounds on $M_q(n,t,s)$, and explicit bounds on $M_q(n,1,1)$. These explicit bounds outperformed existing bounds for several sets of parameters.

3. We considered an approach based on hypergraphs and matchings that was used in literature to derive an upper bound on $M_q(n,t,0)$ in section 5.3. Using an existing upper bound on the maximum size of a matching in a hypergraph an upper bound for $M_q(n,t,0)$ was constructed. We have shown how to generalize this approach to the setting of t-indel s-substitution correcting codes using additional requirements. As a result, we derived an implicit upper bound on $M_q(n,t,s)$ in Theorem 6.4 in terms of the cardinality of $\mathcal{V}_{t',t'',s}(\mathbf{x})$. The expressions for $|\mathcal{V}_{1,0,1}(\mathbf{x})|$ and $|\mathcal{V}_{0,1,1}(\mathbf{x})|$ were again used to obtain explicit bounds on $M_q(n,1,1)$. These bounds provided improvements upon existing bounds in literature.

Furthermore, we established an asymptotic implication from the bounds derived in this thesis. Namely, the asymptotic redundancy of a maximal size t-indel s-substitution correcting code falls between $(t+s)\log_q(n)$ and $2(t+s)\log_q(n) + o(\log_q(n))$.

In short, we conclude that the Gilbert-Varshamov type lower bounds, the Singleton upper bounds, sphere-packing upper bounds and upper bound using the hypergraph and matching approach can all be generalized to the setting of t-indel s-substitution correcting codes. For all these approaches we have shown how to construct bounds on $M_q(n,t,s)$. However, we find that deriving general and explicit bounds is often complicated due to the limited knowledge on the cardinality of the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$.

Next, three directions for further research are considered.

1. In this thesis, we discussed reviewed existing lower and upper bounds on $M_q(n, t, 0)$ and $M_q(n, 0, s)$ in detail. Nonetheless, within literature there are numerous other lower and upper bounds that have not been investigated in this thesis. For instance, we refer to [58–66] for bounds on $M_q(n, 0, s)$ and to [9, 48, 67] for bounds on $M_q(n, t, 0)$. We recognize that these bounds can potentially be generalized to the setting of t-indel s-substitution correcting codes as well.

2. Furthermore, we observed that the cardinality of the set $\mathcal{V}_{t',t'',s}(\mathbf{x})$ plays an important role in deriving both lower and upper bounds on $M_q(n, t, s)$. As discussed in Chapter 3 determining the cardinality of this set for general parameters is a difficult task. The implicit bounds on $M_q(n, t, s)$ derived in this thesis and future work on these bounds would benefit from more general expressions or tight bounds on the cardinality of $\mathcal{V}_{t',t'',s}(\mathbf{x})$. Therefore, deriving such expressions or bounds would be a challenging and useful problem.

3. A third direction for future research would be to construct t-indel s-substitution correcting codes that can be used in DNA data storage or racetrack memory systems. Apart from being able to correct combinations of deletions, insertions and substitutions with low redundancy, these codes have more requirements. For instance, efficient encoding and decoding algorithms are needed for practical use. Moreover, the codewords should satisfy additional constraints in order to guarantee reliable storage. Designing a system that encompasses all these requirements is challenging.

Bibliography

[1] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 3, p. 379–423, Jul. 1948.

[2] C. E. Mackenzie, *Coded character sets, history and development*, 1st ed. Addison-Wesley Publishing company, 1980.

[3] R. W. Hamming, "Error detecting and error correcting codes," *Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, Apr. 1950.

[4] E. N. Gilbert, "A comparison of signalling alphabets," *The Bell System Technical Journal*, vol. 31, no. 3, pp. 504–522, May 1952.

[5] R. R. Varshamov, "Estimate of the number of signals in error correcting codes," *Doklady Akademii Nauk SSSR*, vol. 117, no. 5, pp. 739–741, Jun. 1957.

[6] R. Roth, *Introduction to Coding Theory*. Cambridge: Cambridge University Press, 2006.

[7] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Soviet Physics Doklady*, vol. 10, no. 8, pp. 707–710, Feb. 1966.

[8] A. A. Kulkarni and N. Kiyavash, "Non-asymptotic upper bounds for deletion correcting codes," *IEEE Transactions on Information Theory*, vol. 59, no. 8, pp. 5115–5130, Apr. 2013.

[9] L. Tolhuizen, "Upper bounds on the size of insertion/deletion correcting codes," *Proceedings 8-th International Workshop on Algebraic and Combinatorial Coding Theory, Russia*, pp. 242–246, Sept. 2002.

[10] A. Fazeli, A. Vardy, and E. Yaakobi, "Generalized sphere packing bound," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2313–2334, 2015.

[11] I. Smagloy, L. Welter, A. Wachter-Zeh, and E. Yaakobi, "Single-deletion single-substitution correcting codes," *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 775–780, Aug. 2020.

[12] J. Sima, R. Gabrys, and J. Bruck, "Optimal codes for the q-ary deletion channel," *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 740–745, Aug. 2020.

[13] W. Song, N. Polyanskii, K. Cai, and X. v. He, "On multiple-deletion multiple-substitution correcting codes," *2021 IEEE International Symposium on Information Theory (ISIT)*, pp. 2655–2660, Sep. 2021.

[14] W. Song, N. Polyanskii, K. Cai, and X. He, "Systematic codes correcting multiple-deletion and multiple-substitution errors," *IEEE Transactions on Information Theory*, vol. 68, no. 10, pp. 6402–6416, 2022.

[15] R. R. Varshamov and G. M. Tenengolts, "Codes which correct single asymmetric errors (in Russian)," *Automatika i Telemkhanika*, vol. 161, no. 3, pp. 288–292, 1965.

[16] R. Gabrys, V. Guruswami, J. Ribeiro, and K. Wu, "Beyond single-deletion correcting codes: substitutions and transpositions," *IEEE Transactions on Information Theory*, vol. 69, no. 1, pp. 169–186, 2023.

[17] G. Church, Y. Gao, and S. Kosuri, "Next-generation digital information storage in DNA," *Science (New York, N.Y.)*, vol. 337, p. 1628, Sep. 2012.

[18] S. M. H. T. Yazdi, H. M. Kiah, E. Garcia-Ruiz, J. Ma, H. Zhao, and O. Milenkovic, "DNA-based storage: trends and methods," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 1, no. 3, pp. 230–248, Sep. 2015.

[19] N. Goldman, P. Bertone, S. Chen, C. Dessimoz, E. M. LeProust, B. Sipos, and E. Birney, "Towards practical, high-capacity, low-maintenance information storage in synthesized DNA," *Nature*, vol. 494, pp. 797–800, Jan. 2013.

[20] R. Grass, R. Heckel, M. Puddu, D. Paunescu, and W. J. Stark, "Robust chemical preservation of digital information on DNA in silica with error-correcting codes," *Angewandte Chemie Int.*, vol. 54, pp. 2552–2555, Feb. 2015.

[21] S. M. H. T. Y. Yazdi, Y. Yuan, J. Ma, H. Zhao, and O. Milenkovic, "A rewritable random-access DNA-based storage system," *Scientific Reports*, vol. 5, Sep. 2015.

[22] S. Yazdi, R. Gabrys, and O. Milenkovic, "Portable and error-free DNA-based data storage," *Scientific reports*, vol. 7, no. 5011, Jul. 2017.

[23] P. Antkowiak, J. Lietard, M. Darestani, and et al., "Low cost DNA data storage using photolithographic synthesis and advanced information reconstruction and error correction," *Nature Communications*, vol. 11, no. 5345, Oct. 2020.

[24] Y. Erlich and D. Zielinski, "DNA fountain enables a robust and efficient storage architecture," *Science*, vol. 355, no. 6328, pp. 950–954, Mar. 2017.

[25] BCC publishing staff, "DNA data storage: global markets and technologies," BCC Research LLC, Tech. Rep., Mar. 2023.

[26] R. Heckel, G. Mikutis, and R. N. Grass, "A characterization of the DNA data storage channel," *Scientific Reports*, vol. 9, Jul. 2019.

[27] E. Brill and R. C. Moore, "An improved error model for noisy channel spelling correction," *Proceedings of the 38st Annual Meeting of the Association for Computational Linguistics*, vol. 1, Oct. 2000.

[28] F. J. Och, "Minimum error rate training in statistical machine translation," *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, vol. 1, Jul. 2003.

[29] A. Jiang, Y. Li, and J. Bruck, "Error correction through language processing," *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5, 2015.

[30] S. S. Parkin, M. Hayashi, and L. Thomas, "Magnetic domain-wall racetrack memory," *Science*, vol. 320, no. 5873, pp. 190–194, Apr. 2008.

[31] M. Hayashi, L. Thomas, R. Moriya, C. Rettner, and S. S. Parkin, "Current-controlled magnetic domain-wall nanowire shift register," *Science*, vol. 320, no. 5873, p. 209–211, Apr. 2008.

[32] Y. M. Chee, H. M. Kiah, A. Vardy, V. K. Vu, and E. Yaakobi, "Coding for racetrack memories," *IEEE Transactions on Information Theory*, vol. 64, no. 11, pp. 7094–7112, 2018.

[33] J. B. Kruskal, "An overview of sequence comparison: time wraps, string edits, and macromolecules," *SIAM Review*, vol. 25, no. 2, pp. 201–237, Apr. 1983.

[34] G. Tenengolts, "Non-binary codes, correcting single deletion or insertion (Corresp.)," *IEEE Transactions on Information Theory*, vol. 30, no. 5, pp. 766–769, Sep. 1984.

[35] D. Cullina and N. Kiyavash, "An improvement to Levenshtein's upper bound on the cardinality of deletion correcting codes," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 3862–3870, 2014.

[36] M. Abu-Sini and E. Yaakobi, "On Levenshtein's reconstruction problem under insertions, deletions, and substitutions," *IEEE Transactions on Information Theory*, vol. 67, no. 11, pp. 7132–7158, Sep. 2021.

[37] D. Bar-Lev, T. Etzion, and E. Yaakobi, "On Levenshtein balls with radius one," *2021 IEEE International Symposium on Information Theory (ISIT)*, pp. 1979–1984, Jul. 2021.

[38] V. I. Levenshtein, "Bounds for deletion/insertion correcting codes," *Proceedings IEEE International Symposium on Information Theory*, Jul. 2002.

[39] M. Abu-Sini and E. Yaakobi, "Reconstruction of sequences in DNA storage," *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 290–294, Sep. 2019.

[40] V. I. Levenshtein, "Elements of the coding theory (in Russian)," *Discrete mathematics and mathematics problems of cybernetics Nauka, Moscow*, pp. 207–235, 1974.

[41] R. Ahlswede, L. Bäumer, N. Cai, H. Aydinian, V. Blinovsky, C. Deppe, and H. Mashurian, *General theory of information transfer and combinatorics*. Cambridge: Springer, 2006.

[42] H. Mercier, M. Khabbazian, and V. K. Bhargava, "On the number of subsequences when deleting symbols from a string," *IEEE Transactions on Information Theory*, vol. 54, no. 7, pp. 3279–3285, Jun. 2008.

[43] D. S. Hirschberg and M. Regnier, "Tight bounds on the number of string subsequences," *Journal of Discrete Algorithms*, vol. 1, Jun. 2001.

[44] Y. Liron and M. Langberg, "A characterization of the number of subsequences obtained via the deletion channel," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2300–2312, Mar. 2015.

[45] F. Sala and L. Dolecek, "Counting sequences obtained from the synchronization channel," *2013 IEEE International Symposium on Information Theory*, pp. 2925–2929, Jul. 2013.

[46] L. Tolhuizen, "The generalized Gilbert-Varshamov bound is implied by Turan's theorem," *IEEE Transactions on Information Theory*, vol. 43, no. 5, pp. 1605–1606, Sep. 1997.

[47] J. H. van Lint and R. M. Wilson, *A course in combinatorics*, 2nd ed. Cambridge university press, 2001.

[48] F. Sala, R. Gabrys, and L. Dolecek, "Gilbert-varshamov-like lower bounds for deletion-correcting codes," *2014 IEEE Information Theory Workshop (ITW 2014)*, pp. 147–151, 2014.

[49] Y. Caro, "New results on the independence number," *Technical Report, Tel-Aviv University*, pp. 75–79, 1979.

[50] V. Wei, "A lower bound on the stability number of a simple graph," *Bell Laboratories Technical Memorandum*, no. 81–11217, 1981.

[51] S. Khogan, "A note on a caro–wei bound for the bipartite independence number in graphs," *Discrete Mathematics*, vol. 344, no. 4, Apr. 2021.

[52] T. Jiang and A. Vardy, "Asymptotic improvement of the gilbert- varshamov bound on the size of binary codes," *IEEE Transactions Information Theory*, vol. 50, no. 8, pp. 1655–1664, Aug. 2004.

[53] K. Dutta, D. Mubayi, and C. Subramanian, "New lower bounds for the independence number of sparse graphs and hypergraphs," *SIAM Journal Discrete Mathematics*, vol. 26, no. 3, pp. 1134–1147, Aug. 2012.

[54] J. Sima, R. Gabrys, and J. Bruck, "Optimal systematic t-deletion correcting codes," *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 769–774, Aug. 2020.

[55] R. Singleton, "Maximum distance q-n-ary codes," *IEEE Transactions on Information Theory*, vol. 10, no. 2, pp. 116–118, 1964.

[56] S. Liu and C. Xing, "Bounds and constructions for insertion and deletion codes," *IEEE Transactions on Information Theory*, vol. 69, no. 2, pp. 928–940, Feb. 2023.

[57] I. S. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, June 1960.

[58] S. Johnson, "A new upper bound for error-correcting codes," *IRE Transactions on Information Theory*, vol. 8, no. 3, pp. 203–207, Apr. 1962.

[59] B. Mounits, T. Etzion, and S. Litsyn, "Improved upper bounds on sizes of codes," *IEEE Transactions on Information Theory*, vol. 48, no. 4, pp. 880–886, Apr. 2002.

[60] M. Plotkin, "Binary codes with specified minimum distance," *IRE Transactions on Information Theory*, vol. 6, no. 4, pp. 445–450, Sept. 1960.

[61] P. Delsarte, "Bounds for unrestricted codes, by linear programming," *Philips Research Reports*, vol. 27, p. 272–289, 1972.

[62] P. R. J. Ostergard, "The sextuply shortened binary golay code is optimal," *Designs, Codes and Cryptography*, vol. 87, no. 2-3, pp. 341–347, 2019.

[63] M. R. Best, A. E. Brouwer, F. J. MacWilliams, A. M. Odlyzko, and N. J. A. Sloane, "Bounds for binary codes of length less than 25," *IEEE Transactions on Information Theory*, vol. 87, p. 81–93, 1978.

[64] M. Laurant, "Strengthened semidefinite programming bounds for codes," *Mathematical Programming*, vol. 109, no. 4, pp. 239–261, Oct. 2006.

[65] D. C. Gijswijt, H. D. Mittelmann, and A. Schrijver, "Semidefinite code bounds based on quadruple distances," *IEEE Transactions on Information Theory*, vol. 58, no. 5, pp. 2697–2705, Jan. 2012.

[66] H. K. Kim and P. T. Toan, "Improved semidefinite programming bound on sizes of codes," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7337–7345, 2013.

[67] V. I. Levenshtein, "On perfect codes in deletion and insertion metric," *Discrete Mathematics and Applications*, vol. 2, no. 3, p. 241–258, Oct. 1992.

[68] A. Schrijver, *Theory of linear and integer programming.* John Wiley & Sons, Jun. 1998.

A
Proofs of several results

A.1 Proof of Lemma 2.6

Lemma 2.6 states the following. Let $n \geq 1$, $q \geq 2$, $t' \geq 0$, $t'' \geq 0$ and $s \geq 0$ be integers such that $n - t' + t'' \geq 1$. Let $\mathbf{x} \in \mathcal{B}_q(n)$ and $\mathbf{y} \in \mathcal{V}_{t',t'',s}(\mathbf{x})$, then the following holds,

$$r(\mathbf{x}) - 2(t' + s) \leq r(\mathbf{y}) \leq r(\mathbf{x}) + 2(t'' + s).$$

Proof. Consider an arbitrary word $\mathbf{x} \in \mathcal{B}_q(n)$ with $r = r(\mathbf{x})$ runs, then we argue how a single deletion, insertion or substitution in $\mathbf{x}$ can affect the number of runs of $\mathbf{x}$. Let l_j denote the length of the j-th run in $\mathbf{x}$.

After a single deletion in the i-th run of $\mathbf{x}$, we claim that the number of runs in $\mathbf{x}$ decreases by at most two, but does not increase. This claim can be shown by considering four cases. First, in case $l_i > 1$, then the number of runs is unaffected because the i-th run is shortened by one, and no runs are deleted or created. In case $l_i = 1$ and the i-th run is the first or last run in $\mathbf{x}$, then removing this run doe not affect the other runs and reduces the number of runs by one. In case, $l_i = 1$ and the $(i-1)$-th and $(i+1)$-th run contain the same symbol values, then the number of runs decreases by two. This follows because $(i-1)$-th and $(i+1)$-th join to form a single run after the i-th run is removed by the deletion. Lastly, in case $l_i = 1$, and the $(i-1)$-th and $(i+1)$-th run contains different symbols only the i-th run is removed and the number of runs decreased by one. In each of the four cases, the claim holds which proves the claim.

As a result of a single insertion into $\mathbf{x}$, we claim that the number of runs in $\mathbf{x}$ does not decrease and increases by at most two. We distinguish three cases. In case the insertion is carried out before the first symbol or after the last symbol of $\mathbf{x}$, then either an additional unit run is formed, or the inserted symbol joins its neighbouring run. In case, the symbol is inserted in the middle of a run, then the number of runs in $\mathbf{x}$ is unaffected if the inserted symbol value equals the value of the symbols in that run. Otherwise, the number of runs increases by two. Lastly, in case the insertion is carried out in between two runs, then either the inserted symbol joins one (and not both) neighbouring run(s), or it forms a unit run. In all cases, the number of runs in $\mathbf{x}$ does not decrease and increases by at most two, proving the claim.

For a single substitution we claim that the number of runs in $\mathbf{x}$ can both be increased and decreased by at most two. This follows by considering three cases. First, suppose that a substitution is carried out in a unit run, then the number of runs can decrease

by at most two, but not increase. Namely, after the substitution it can either remain a unit run in which case the number of runs in unaffected, or it joins with one or two neighbouring runs in which case the number of runs decreases by one or two, respectively. Secondly, assume the substitution is performed in the first or last element of a run of length at least of two, then either a unit run is formed in the position of the substitution or the symbol is changed such that it matches the symbols in the neighbouring run. Hence, the number of runs increases by one or does not change. Lastly, suppose that the substitution is carried out in a run of length at least three and neither in the first nor last element of this run. In this case we create always a unit run at the position of the substitution and two runs on either side of it. Therefore, the number of runs increases by precisely two. Combining the results from these three cases yields the claim.

By repeatedly applying single edits, the previous reasoning shows that after t' deletions, t'' insertions and s substitutions the number of runs in $\mathbf{x}$ decreases by at most $2(t' + s)$ and increases by at most $2(t'' + s)$. This proves the desired result. □

A.2 Proof of Lemma 3.1

Lemma 3.1 states the following. For integers $n \geq 1$, $q \geq 2$ and $t \geq 0$ the cardinality of the set $\mathcal{I}_t(\mathbf{x})$ is given by

$$|\mathcal{I}_t(\mathbf{x})| = \sum_{i=0}^{t} \binom{n+t}{i} (q-1)^i,$$

for each $\mathbf{x} \in \mathcal{B}_q(n)$.

Proof. The idea of the proof is to first show that $|\mathcal{I}_t(\mathbf{x})|$ depends only on n, q and t. Consequently, $|\mathcal{I}_t(\mathbf{x})|$ is equal for each $\mathbf{x} \in \mathcal{B}_q(n)$. As a result, it suffices to prove that the formula holds for one $\mathbf{x}$ in particular. For this proof, recall the notation $(a)^n \in \mathcal{B}_q(n)$ which denotes the word that only consists of the symbol $a \in \mathcal{B}_q$ repeated n times.

We prove the first statement by induction on n. For $n = 1$, let $\mathbf{x} = (k) \in \mathcal{B}_q(1)$, where $k \in \mathcal{B}_q$ is a symbol. Notice that $\mathbf{y} \in \mathcal{B}_q(1 + t)$ can be reached from $\mathbf{x}$ by precisely t insertions if and only if $\mathbf{y}$ contains at least one symbol that is equal to k. The latter observation is in turn equivalent to stating that $\mathbf{y} \in \mathcal{S}_t((k)^{1+t})$. Therefore, it holds that $|\mathcal{I}_t(\mathbf{x})| = |\mathcal{S}_t((k)^{1+t})| = \sum_{i=0}^{t} \binom{1+t}{i}(q-1)^i$, which shows the statement for $n = 1$.

For the induction hypothesis, we assume that there exists some n for which $|\mathcal{I}_t(\mathbf{x})|$ depends on n, q and t only. Let $\mathbf{y} \in \mathcal{B}_q(n + 1)$, then we partition $\mathcal{I}_t(\mathbf{y})$ into the classes C_m for $1 \leq m \leq t + 1$, where C_m contains as all the words $\mathbf{z} \in \mathcal{I}_t(\mathbf{y})$ such that $z_m = y_1$ and $z_i \neq y_1$ for $i < m$. Intuitively, $\mathbf{z}$ belongs to the class C_m if m is the leftmost position in $\mathbf{z}$ where y_1 could end up. The following equivalence holds: $\mathbf{z}$ is an element of C_m if and only if the sequence $(z_{m+1}, \ldots, z_{n+1+t})$ can be obtained from the sequence $(y_2, \ldots y_{n+1})$ by $t - m$ insertions. This follows because y_1 is placed at position m in $\mathbf{z}$, so all y_j for $2 \leq j \leq n+1$ should be placed in $\mathbf{z}$ at positions to the right of m. Hence, the cardinality of C_m is equal to the cardinality of $\mathcal{I}_{t-m}((y_2, \ldots, y_{n+1}))$, which does only depend on n, q and t by the induction hypothesis. Since the classes C_m form a partition of $\mathcal{I}_t(\mathbf{y})$, the sum of their cardinalities equals $|\mathcal{I}_t(\mathbf{y})|$. Therefore, we conclude that $|\mathcal{I}_t(\mathbf{y})|$ depends only on n, q and t as well. By the principle of induction, this proves the first statement of this proof.

In the remainder of this proof we will show that the formula for the cardinality of $|\mathcal{I}_t(\mathbf{x})|$ holds for $\mathbf{x} = (0)^n$. Notice that a word $\mathbf{y} \in \mathcal{B}_q(n+t)$ can be obtained from $(0)^n$ by precisely t insertions if and only if $\mathbf{y} \in \mathcal{S}_t((0)^{n+t})$. Indeed, if a word $\mathbf{y} \in \mathcal{B}_q(n+t)$ can be obtained from $(0)^n$ by precisely t insertions, then $\mathbf{y}$ contains at most t non-zero symbols. Hence, it can also be obtained from the all-zero word of length $n+t$ by t substitutions, i.e., $\mathbf{y} \in \mathcal{S}_t((0)^{n+t})$. Conversely, a word $\mathbf{z} \in \mathcal{S}_t((0)^{n+t})$ contains at most t non-zero symbols due to the at most t substitutions and consequently can be obtained from $(0)^n$ by inserting these non-zero (and possibly additional zero) symbols. Therefore, the cardinalities of the sets $\mathcal{I}_t((0)^n)$ and $\mathcal{S}_t((0)^{n+t})$ agree.

By combining all previous observations and Equation (3.1) it can be concluded that

$$|\mathcal{I}_t(\mathbf{x})| = |\mathcal{I}_t((0)^n)| = |\mathcal{S}_t((0)^{n+t})| = \sum_{i=0}^{t} \binom{n+t}{i} (q-1)^i,$$

for each $\mathbf{x} \in \mathcal{B}_q(n)$, which finalizes the proof. □

A.3 Proof of Lemma 3.2

Lemma 3.2 states the following. Let $n \geq 1$ and $q \geq 2$ be integers and $\mathbf{x} \in \mathcal{B}_q(n)$. Then, the following holds,

$$|\mathcal{V}_{1,0,1}(\mathbf{x})| = L^{n,q}_{1,0,1}(r(\mathbf{x})) := \begin{cases} (n-1)(q-1)+1 & \text{if } r(\mathbf{x}) = 1, \\ r(\mathbf{x})\left((n-2)(q-1)-1\right)+q+2 & \text{if } r(\mathbf{x}) \geq 2. \end{cases}$$

Proof. Let $r = r(\mathbf{x})$ be the number of runs in $\mathbf{x}$. By Lemma 2.1, each element in the set $\mathcal{V}_{1,0,1}(\mathbf{x})$ can be obtained by deleting a single symbol followed by substituting a symbol. Therefore it holds that $\mathcal{V}_{1,0,1}(\mathbf{x}) = \cup_{\mathbf{y} \in \mathcal{D}_1(\mathbf{x})} \mathcal{S}_1(\mathbf{y})$. Clearly, it does not matter which symbol is deleted from a particular run of $\mathbf{x}$, because always the same word is obtained. Hence, $\mathcal{D}_1(\mathbf{x}) = \{\mathbf{x}^1, ..., \mathbf{x}^r\}$ where $\mathbf{x}^i \in \mathcal{B}_q(n)$ for $1 \leq i \leq r$ denotes the word that is obtained by deleting the last symbol from the i-th run in $\mathbf{x}$. Next, we distinguish between the cases $r = 1$ and $r \geq 2$.

First, suppose that $r = 1$, which gives $\mathcal{D}_1(\mathbf{x}) = \{\mathbf{x}^1\}$ and in turn

$$\mathcal{V}_{1,0,1}(\mathbf{x}) = \bigcup_{\mathbf{y} \in \mathcal{D}_1(\mathbf{x})} \mathcal{S}_1(\mathbf{y}) = \mathcal{S}_1(\mathbf{x}^1).$$

The word $\mathbf{x}^1$ has length $n-1$ after one deletion from $\mathbf{x}$. It follows from (3.1) that

$$|\mathcal{V}_{1,0,1}(\mathbf{x})| = |\mathcal{S}_1(\mathbf{x}^1)| = \sum_{i=1}^{1} \binom{n-1}{i} (q-1)^i = 1 + (n-1)(q-1).$$

Second, suppose that $r \geq 2$. In what follows, we apply the inclusion-exclusion prin-

ciple. That means,

$$\begin{aligned}|\mathcal{V}_{1,0,1}(\mathbf{x})| &= \left|\bigcup_{\mathbf{x}^*\in\{\mathbf{x}^1,\dots,\mathbf{x}^r\}} \mathcal{S}_1(\mathbf{x}^*)\right| \\ &= \sum_{i\in\{1,\dots,r\}} |\mathcal{S}_1(\mathbf{x}^i)| - \sum_{\substack{i,j\in\{1,\dots,r\}\\ i\neq j}} |\mathcal{S}_1(\mathbf{x}^i)\cap\mathcal{S}_1(\mathbf{x}^j)| \\ &\quad + \sum_{\substack{i,j,k\in\{1,\dots,r\}\\ i\neq j,\, i\neq k,\, j\neq k}} |\mathcal{S}_1(\mathbf{x}^i)\cap\mathcal{S}_1(\mathbf{x}^j)\cap\mathcal{S}_1(\mathbf{x}^k)| \pm \cdots . \end{aligned} \tag{A.1}$$

This alternating sum of positive and negative terms continues with the intersections over $4, 5, ..., r$ sets $\mathcal{S}_1(\mathbf{x}^j)$. Hence, in order to evaluate the expressions in (A.1), we aim to compute the size of the intersection

$$\left|\bigcap_{k\in K} \mathcal{S}(\mathbf{x}^k)\right|, \tag{A.2}$$

for any non-empty set $K \subseteq \{1, ..., r\}$. In order to accomplish this, we first prove the claim that $d(\mathbf{x}^i, \mathbf{x}^j) = |i - j|$, where d denotes the Hamming distance function. Without loss of generality, we can assume that $i < j$. Let $l_k \in \{1, ..., n\}$ denote the position of the last symbol from the k-th run in $\mathbf{x}$. The claim follows from the observation that the words $\mathbf{x}^i$ and $\mathbf{x}^j$ differ only in the positions l_k for $i \leq k < j$. A direct consequence of this claim is the following useful result. For a pair of integers $i, j \in \{1, 2, ..., r\}$ such that $|i - j| \geq 3$ it holds that $d(\mathbf{x}^i, \mathbf{x}^j) \geq 3$ and thus $\mathcal{S}_1(\mathbf{x}^i) \cap \mathcal{S}_1(\mathbf{x}^j) = \emptyset$.

This claim allows us to evaluate (A.2) and to simplify the expressions in (A.1). Note that for $K \subseteq \{1, ..., r\}$ with $|K| \geq 4$ it necessarily holds that there exist such $i, j \in K$ with $|i - j| \geq 3$. In other words, we conclude that (A.2) is zero for all sets K such that $|K| \geq 4$. For sets $K \subseteq \{1, ..., r\}$ with $|K| \leq 3$ the summations in (A.2) can be simplified as follows

$$\begin{aligned}\sum_{\substack{i,j\in\{1,\dots,r\}\\ i\neq j}} |\mathcal{S}_1(\mathbf{x}^i)\cap\mathcal{S}_1(\mathbf{x}^j)| &= \sum_{i\in\{1,\dots,r-1\}} |\mathcal{S}_1(\mathbf{x}^i)\cap\mathcal{S}_1(\mathbf{x}^{i+1})| \\ &\quad + \sum_{i\in\{1,\dots,r-2\}} |\mathcal{S}_1(\mathbf{x}^i)\cap\mathcal{S}_1(\mathbf{x}^{i+2})|, \\ \sum_{\substack{i,j,k\in\{1,\dots,r\}\\ i\neq j,\, i\neq k,\, j\neq k}} |\mathcal{S}_1(\mathbf{x}^i)\cap\mathcal{S}_1(\mathbf{x}^j)\cap\mathcal{S}_1(\mathbf{x}^k)| &= \sum_{i\in\{1,\dots,r-2\}} |\mathcal{S}_1(\mathbf{x}^i)\cap\mathcal{S}_1(\mathbf{x}^{i+1})\cap\mathcal{S}_1(\mathbf{x}^{i+2})|. \end{aligned}$$

In order to evaluate these expressions, observe the following. For $i \in \{1, ..., r - 1\}$ it holds that $d(\mathbf{x}^i, \mathbf{x}^{i+1}) = |i - (i - 1)| = 1$, and thus $\mathbf{x}^i$ and $\mathbf{x}^{i+1}$ differ in one position. It follows that $|\mathcal{S}_1(\mathbf{x}^i) \cap \mathcal{S}_1(\mathbf{x}^{i+1})| = q$. Similarly, for $i \in \{1, ..., r - 2\}$ it holds that $d(\mathbf{x}^i, \mathbf{x}^{i+1}) = |i - (i - 2)| = 2$. Then it follows that $|\mathcal{S}_1(\mathbf{x}^i) \cap \mathcal{S}_1(\mathbf{x}^{i+2})| = 2$. Moreover, for $i \in \{1, ..., r - 3\}$ it holds that $|\mathcal{S}_1(\mathbf{x}^i) \cap \mathcal{S}_1(\mathbf{x}^{i+1}) \cap \mathcal{S}_1(\mathbf{x}^{i+2})| = 1$. Lastly, recall from (3.1) that $|\mathcal{S}_1(\mathbf{x}^i)| = \sum_{i=0}^{1} \binom{n-1}{i}(q-1)^i = 1 + (n-1)(q-1)$ for all $i \in \{1, ..., r\}$.

Now, we are ready to prove the main result:

$$\begin{aligned}
|\mathcal{V}_{1,0,1}(\mathbf{x})| &= \sum_{i=1}^{r} |\mathcal{S}_1(\mathbf{x}^i)| \\
&\quad - \sum_{i\in\{1,\dots,r-1\}} |\mathcal{S}_1(\mathbf{x}^i) \cap \mathcal{S}_1(\mathbf{x}^{i+1})| \\
&\quad - \sum_{i\in\{1,\dots,r-2\}} |\mathcal{S}_1(\mathbf{x}^i) \cap \mathcal{S}_1(\mathbf{x}^{i+2})| \\
&\quad + \sum_{i\in\{1,\dots,r-2\}} |\mathcal{S}_1(\mathbf{x}^i) \cap \mathcal{S}_1(\mathbf{x}^{i+1}) \cap \mathcal{S}_1(\mathbf{x}^{i+2})| \\
&= r(\mathbf{x}) \cdot ((n-1)(q-1)+1) \\
&\quad - (r(\mathbf{x})-1) \cdot q \\
&\quad - (r(\mathbf{x})-2) \cdot 2 \\
&\quad + (r(\mathbf{x})-2) \cdot 1 \\
&= r(\mathbf{x})((n-1)(q-1)+1) - r(\mathbf{x})(q+1) + q + 2 \\
&= r(\mathbf{x})\left((n-2)(q-1)-1\right) + q + 2.
\end{aligned}$$

The result now follows from combining the two cases $r(\mathbf{x}) = 1$ and $r(\mathbf{x}) \geq 2$. □

A.4 Proof of Theorem 4.5

Theorem 4.5 states the following. Let $G = (V, E)$ be a simple graph, then G contains a clique of size

$$\sum_{v\in V} \frac{1}{|V| - deg(v)}.$$

Proof. The idea of the proof is to first translate the statement of this theorem to an equivalent statement about independent sets. Then we are able to apply the original argument by Caro [49] using induction on $|V|$.

An independent set of $G = (V, E)$ is a subset of the vertices of G so that no pair of these vertices is joined by an edge. Let $G' = (V', E')$ be the complement graph of G, which means the vertex set $V' = V$ is unchanged and $E' = V \times V \setminus E$. In other words, the edges in G' are non-edges in G and vice versa. It follows that an independent set in G forms a clique in G' of equal size, and vice versa. Note that for any vertex $v \in V$ it holds that $deg_G(v) + deg_{G'}(v) + 1 = |V|$. The statement of this theorem is thus equivalent to the following statement. Let $G = (V, E)$ be a simple graph, then G contains an independent set of size

$$\sum_{v\in V} \frac{1}{|V| - deg_{G'}(v)} = \sum_{v\in V} \frac{1}{|V| - (|V| - deg_G(v) - 1)} = \sum_{v\in V} \frac{1}{deg_G(v) + 1}. \tag{A.3}$$

Below, we focus only on the statement about independent sets. In order to show that $G = (V, E)$ contains an independent set of size $\sum_{v\in V} \frac{1}{deg_G(v)+1}$ we perform induction on $|V|$. Let $n = |V|$ for ease of notation. For $n = 1$, the graph G contains a single vertex

which forms an independent set of size one. This single vertex must have degree zero since there are no other vertices and thus (A.3) also gives one. Hence, for $n = 1$ the statement holds.

Suppose that there exists some integer n so that all graphs with n or fewer vertices contain an independent of size (A.3). Let $G = (V_G, E_G)$ be a graph with $|V_G| = n + 1$ vertices. Let $v^* \in V_G$ be a vertex of minimal degree. Let $N[v^*] \subseteq V_G$ denote the set of vertices that are incident to v^*, and v^* itself. Therefore, it holds that $|N(v^*)| = deg_G(v^*) + 1$. Let $H = (V_H, E_H) = G - N[v^*]$ denote the graph that is obtained from G by deleting the vertices in $N[v^*]$ and deleting the edges that are incident to vertices in $N[v^*]$. Let $\alpha(G)$ and $\alpha(H)$ denote the sizes of a largest independent set in G and H, respectively. Then it holds that $\alpha(G) \geq 1 + \alpha(H)$. This follows because a maximal independent set in H of size $\alpha(H)$ can be turned into an independent set in G of size $\alpha(H) + 1$ by including the vertex v^* into this independent set. This is possible since all vertices incident to v^* have been removed in H.

By the induction hypothesis, there exists an independent set in H of size $\sum_{v \in V_H} \frac{1}{deg_H(v)+1}$. Therefore, it holds that

$$
\begin{aligned}
\alpha(G) \geq \quad & 1 + \alpha(H) \\
\overset{\text{Ind. hyp.}}{\geq} \quad & 1 + \sum_{v \in V_H} \frac{1}{deg_H(v) + 1} \\
\geq \quad & 1 + \sum_{v \in V_G \setminus N[v^*]} \frac{1}{deg_H(v) + 1} \\
\overset{\dagger}{\geq} \quad & 1 + \sum_{v \in V_G \setminus N[v^*]} \frac{1}{deg_G(v) + 1} \\
= \quad & \sum_{v \in N[v^*]} \frac{1}{|N[v^*]|} + \sum_{v \in V_G \setminus N[v^*]} \frac{1}{deg_G(v) + 1} \\
= \quad & \sum_{v \in N[v^*]} \frac{1}{deg_G(v^*) + 1} + \sum_{v \in V_G \setminus N[v^*]} \frac{1}{deg_G(v) + 1} \\
\overset{\dagger\dagger}{\geq} \quad & \sum_{v \in N[v^*]} \frac{1}{deg_G(v) + 1} + \sum_{v \in V_G \setminus N[v^*]} \frac{1}{deg_G(v) + 1} \\
= \quad & \sum_{v \in V_G} \frac{1}{deg_G(v) + 1}.
\end{aligned}
$$

In (†) we used that $deg_H(v) \leq deg_G(v)$ for all $v \in V_G \setminus N[v^*]$, because edges are only removed when turning G into H. We used in (††) that v^* is of minimal degree. The last chain of (in)equalities shows that G has an independent set of the desired size. Therefore, the statement about independent sets holds for all simple graphs by the induction principle. As a result, this proves the statement of this theorem. □

A.5 Proof of Lemma 4.6

Lemma 4.6 states the following. Let $G = (V, E)$ be a simple graph, then it holds that

$$\frac{|V|^2}{|V|^2 - 2|E|} \leq \sum_{v \in V} \frac{1}{|V| - deg(v)}. \tag{A.4}$$

Consequently, the Caro-Wei theorem implies Turán's theorem of (4.13).

Proof. For ease of notation, let $n = |V|$ and $m = |E|$. First, notice that the average degree of all nodes is given by

$$\frac{1}{n} \sum_{v \in V} deg(v) = \frac{2m}{n},$$

because each edge has two incident vertices.

Next, consider the function $f_n : [0, n) \to \mathbb{R}$ given by $f_n(x) = \frac{1}{n-x}$. Note that f_n is convex on $[0, n)$. Moreover, note that $0 \leq deg(v) \leq n - 1$ for all $v \in V$ and also that $0 \leq \frac{1}{n} \sum_{v \in V} deg(v) \leq n - 1$. Therefore, it follows that

$$\begin{aligned}
\frac{1}{n} \sum_{v \in V} \frac{1}{n - deg(v)} &= \frac{1}{n} \sum_{v \in V} f_n \left(deg(v) \right) \\
&\geq f_n \left(\frac{1}{n} \sum_{v \in V} deg(v) \right) \\
&= f_n \left(\frac{2m}{n} \right) \\
&= \frac{1}{n - \frac{2m}{n}} \\
&= \frac{n}{n^2 - 2m},
\end{aligned}$$

where we used the convexity of f_n on $[0, n)$. This shows that Inequality (A.4) holds. Let k be the largest integer so that the condition (4.13) in Turán's theorem is satisfied. In other words, k is given by

$$k = \left\lceil \frac{|V|^2}{|V|^2 - 2|E|} \right\rceil = \left\lceil \frac{n^2}{n^2 - 2m} \right\rceil.$$

Given that the size of a clique is integer-valued, the Caro-Wei theorem shows the existence of a clique of size

$$\left\lceil \sum_{v \in V} \frac{1}{n - deg(v)} \right\rceil.$$

Using (A.4) it follows that

$$\left\lceil \sum_{v \in V} \frac{1}{n - deg(v)} \right\rceil = \left\lceil n \cdot \frac{1}{n} \sum_{v \in V} \frac{1}{n - deg(v)} \right\rceil \geq \left\lceil \frac{n^2}{n^2 - 2m} \right\rceil = k.$$

This shows that Turán's theorem is implied by the Caro-Wei theorem. □

A.6 Proof of Theorem 4.10

Theorem 4.10 states the following. Let $q \geq 2$ be an integer. For non-negative integers s and t such that $s + t \geq 1$, the following holds

$$\limsup_{n\to\infty} \frac{n - \log_q(M_q(n,t,s))}{(2t+2s)\log_q(n)} \leq 1.$$

Proof. Let $s, t \geq 0$ be fixed integers for which $s + t \geq 1$ and let $n \geq 1$ be an integer such that $n \geq \max\{s,t\}$ and $0 \leq \max\{\frac{t}{n}, \frac{2s}{n-t}\} \leq 1 - \frac{1}{q}$. Clearly, these conditions are satisfied when n is sufficiently large. Using Theorem 4.4 and the upper bound from Lemma 3.6 the redundancy of an optimal size t-indel s-substitution correcting code can be upper bounded by

$$\begin{aligned}
n-\log_q(M_q(n,t,s)) \leq\ & n - (n+t) + 2\log_q\left(\sum_{i=0}^{t}\binom{n}{i}(q-1)^i\right) + \log_q\left(\sum_{i=0}^{2s}\binom{n-t}{i}(q-1)^i\right) \\
\leq\ & -t + 2nH_q\left(\frac{t}{n}\right) + (n-t)H_q\left(\frac{2s}{n-t}\right) \\
=\ & -t + 2t\log_q(q-1) - 2t\log_q(\tfrac{t}{n}) - 2n(1-\tfrac{t}{n})\log_q(1-\tfrac{t}{n}) + \\
& 2s\log_q(q-1) - 2s\log_q(\tfrac{2s}{n-t}) - (n-t)(1-\tfrac{2s}{n-t})\log_q(1-\tfrac{2s}{n-t}) \\
=\ & -t + 2t\log_q(q-1) - 2t\log_q(t) + 2t\log_q(n) - 2(n-t)\log_q(\tfrac{n-t}{n}) + \\
& 2s\log_q(q-1) - 2s\log_q(2s) + 2s\log_q(n-t) - (n-t-2s)\log_q(\tfrac{n-t-2s}{n-t}) \\
=\ & 2t\log_q(n) - 2(n-t)\log_q(\tfrac{n-t}{n}) + \\
& 2s\log_q(n-t) - (n-t-2s)\log_q(\tfrac{n-t-2s}{n-t}) - F(q,t,s),
\end{aligned}$$

where $F(q,t,s)$ is some function depending on variables q, t and s only. Let C_1, C_2 be fixed constants, then the following holds

$$\begin{aligned}
\lim_{n\to\infty}(n - C_1 - C_2)\log_q(\frac{n-C_1-C_2}{n-C_2}) &= \lim_{n'\to\infty}(n'-C_1)\log_q(\frac{n'-C_1}{n'}) \\
&= \lim_{n'\to\infty}\frac{n'}{\ln(q)}\ln(1-\frac{C_1}{n'}) - \lim_{n\to\infty}\frac{C_1}{\ln(q)}\ln(1-\frac{C_1}{n'}) \\
&= \lim_{n'\to\infty}\frac{n'}{\ln(q)}\left(-\frac{C_1}{n'} - \mathcal{O}\left(\left(\frac{C_1}{n'}\right)^2\right)\right) - 0 \\
&= \frac{-C_1}{\ln(q)},
\end{aligned}$$

where we applied the change of variables $n' = n - C_2$. Furthermore, we notice that separating the limit into two limits in the second equality is allowed, because all limits exist and are finite. The upper bound on the redundancy combined with the previous

limit enable us to derive the bound on the main limit of this theorem, i.e.,

$$\begin{aligned}\limsup_{n\to\infty} \frac{n-\log_q(M_q(n,t,s))}{(2t+2s)\log_q(n)} &\leq \lim_{n\to\infty} \frac{2t\log_q(n)+2s\log_q(n-t)}{(2t+2s)\log_q(n)} - \\ &\quad \lim_{n\to\infty} \frac{2(n-t)\log_q(\frac{n-t}{n})+(n-t-2s)\log_q(\frac{n-t-2s}{n-t})}{(2t+2s)\log_q(n)} - \\ &\quad \lim_{n\to\infty} \frac{F(q,t,s)}{(2t+2s)\log_q(n)} \\ &= \lim_{n\to\infty} \frac{2t\log_q(n)+2s\log_q(n)-2s\log_q(n)+2s\log_q(n-t)}{(2t+2s)\log_q(n)} \\ &= \lim_{n\to\infty} \frac{(2t+2s)\log_q(n)+2s\log_q(\frac{n-t}{n})}{(2t+2s)\log_q(n)} = 1,\end{aligned}$$

where we applied basic rules for limits, used that all limits on the right-hand side exist, and used that the two indented limits on the right-hand side tend to zero for sufficiently large n, since their numerators go to some finite constant independent of n. □

A.7 Proof of Lemma 5.6

Lemma 5.6 states the following. Let $\mathcal{H} = (V, \mathcal{E})$ be a hypergraph. Let $\mathbf{w}^* = (w^*(x))_{x\in V}$ be a real-valued vector that satisfies the following two conditions,

1. $w^*(x) \geq 0$ for all $x \in V$,
2. $\sum_{x\in E} w^*(x) \geq 1$ for all $E \in \mathcal{E}$.

Then, it holds that $\nu(\mathcal{H}) \leq \mathbf{1}^\top \mathbf{w}^* = \sum_{x\in V} w^*(x)$. Here, $\mathbf{1}$ denotes the all-one column-vector of the appropriate length.

Proof. The idea of the proof is to write $\nu(\mathcal{H})$ in terms of the optimal value of an integer linear program. Then, a feasible point in the dual of a linear programming relaxation of this integer linear program is constructed. The objective value of this dual feasible point gives an upper bound on the optimal value of the primal program.

Let A denote a $|V| \times |\mathcal{E}|$ adjacency matrix of $\mathcal{H}$. The problem of finding a maximum size matching in $\mathcal{H}$ can be formulated in terms of the following integer linear program [8, Lem. 2.3]:

$$\nu(\mathcal{H}) = \max\{\mathbf{1}^\top \mathbf{z} | A\mathbf{z} \leq \mathbf{1}, \mathbf{z} \in \{0,1\}^{|\mathcal{E}|}\}. \tag{A.5}$$

In order to show this equality, let $\mathcal{M} := \{E_1, ..., E_j\} \subseteq \mathcal{E}$ be a matching of maximum size, i.e., $|\mathcal{M}| = j = \nu(\mathcal{H})$. Let $\mathbb{1}_\mathcal{M} \in \{0,1\}^{|\mathcal{E}|}$ be the indicator vector for which the l-th element equals one if the hyperedge corresponding to the l-th column of A is an element of $\mathcal{M}$, and equals zero otherwise. This implies that the indicator vector $\mathbb{1}_\mathcal{M}$ is feasible for the program on the right. Indeed, $\mathcal{M}$ is a matching and thus each vertex is contained in at most one hyperedge. This gives that $A\mathbb{1}_\mathcal{M} \leq \mathbf{1}$, while we have $\mathbb{1}_\mathcal{M} \in \{0,1\}^{|\mathcal{E}|}$ by the definition of the indicator vector. Hence, it follows that

$$\nu(\mathcal{H}) = |\mathcal{M}| = \mathbf{1}^\top \mathbb{1}_\mathcal{M} \leq \max\{\mathbf{1}^\top \mathbf{z} | A\mathbf{z} \leq \mathbf{1}, \mathbf{z} \in \{0,1\}^{|\mathcal{E}|}\}.$$

Conversely, let $\mathbf{z}'$ be a feasible vector for the program on the right of (A.5) that attains the maximum, and let K be the set of indices where $\mathbf{z}'$ equals 1. Consider the columns of A of which the indices belong to K. These columns correspond to hyperedges in $\mathcal{E}$ which we collect in the set $\mathcal{M}'$. Then, the conditions $A\mathbf{z}' \leq \mathbf{1}$ and $\mathbf{z}' \in \{0,1\}^{|\mathcal{E}|}$ jointly imply that each vertex can be contained in at most one of the hyperedges in $\mathcal{M}'$. Hence, $\mathcal{M}'$ is a matching, and it has a size $|\mathcal{M}'| = |K| = \mathbf{1}^\top \mathbf{z}'$. Since $\mathbf{z}'$ is maximal, we find

$$\max\{\mathbf{1}^\top \mathbf{z} | A\mathbf{z} \leq \mathbf{1}, \mathbf{z} \in \{0,1\}^{|\mathcal{E}|}\} = \mathbf{1}^\top \mathbf{z}' = |K| = |\mathcal{M}'| \leq \nu(\mathcal{H}).$$

Hence, the equality in (A.5) holds.

By relaxing the constraint $\mathbf{z} \in \{0,1\}^{|\mathcal{E}|}$ to $\mathbf{z} \geq \mathbf{0}$ in (A.5) the following linear programming relaxation is obtained

$$\nu^*(\mathcal{H}) = \max\{\mathbf{1}^\top \mathbf{z} | A\mathbf{z} \leq \mathbf{1}, \mathbf{z} \geq \mathbf{0}, \mathbf{z} \in \mathbb{R}^{|\mathcal{E}|}\}. \tag{A.6}$$

Since this is a maximization program, it follows that $\nu(\mathcal{H}) \leq \nu^*(\mathcal{H})$. The dual of (A.6) is given by,

$$\nu_d^*(\mathcal{H}) = \min\{\mathbf{1}^\top \mathbf{w} | A^\top \mathbf{w} \geq \mathbf{1}, \mathbf{w} \geq \mathbf{0}, \mathbf{w} \in \mathbb{R}^{|V|}\}, \tag{A.7}$$

and by the *strong duality* property of linear programs it holds that $\nu^*(\mathcal{H}) = \nu_d^*(\mathcal{H})$ [68, Cor. 7.1g].

Let $\mathbf{w} = (w(x))_{x \in V}$ be a vector, then we will show that $\mathbf{w}$ is feasible for (A.7) if and only if $\mathbf{w}$ satisfies the two conditions in the statement of this lemma. Clearly, $\mathbf{w} \geq 0$ is equivalent to the first condition. The second condition is equivalent to $\mathbb{1}_E^\top \mathbf{w} \geq 1$ for each $E \in \mathcal{E}$, where $\mathbb{1}_E = (\mathbb{1}_E(x))_{x \in V}$ denotes the indicator vector that equals one when a vertex x belongs to hyperedge E and zero otherwise. We notice that the rows of A are precisely the indicator vectors $\mathbb{1}_E$ for $E \in \mathcal{E}$. Therefore, it holds that $A^\top \mathbf{w} \geq \mathbf{1}$ if and only if $\sum_{x \in E} w(x) \geq 1$ for all $E \in \mathcal{E}$.

Recall that $\mathbf{w}^*$ is a real-valued vector that satisfies the two conditions from this lemma. Using the previous observations it is also feasible for (A.7), and we obtain the following chain of (in)equalities

$$\nu(\mathcal{H}) \leq \nu^*(\mathcal{H}) = \nu_d^*(\mathcal{H}) \leq \mathbf{1}^\top \mathbf{w}^* = \sum_{x \in V} w^*(x),$$

where we used the fact that the objective value of a feasible point for a minimization problem upper bounds the optimal value of that problem. This concludes the proof. □

A.8 Proof of Theorem 6.5

Theorem 6.5 states the following. Let $q \geq 2$ be an integer. For non-negative integers t and s such that $t + s \geq 1$, it holds that

$$\liminf_{n \to \infty} \frac{n - \log_q(M_q(n,t,s))}{(t+s)\log_q(n)} \geq 1.$$

Proof. Let $t, s \geq 0$ be fixed integers such that $t + s \geq 1$. Let $n \gg 1$ be an integer. Moreover, let $\rho \in (0, 1 - \frac{1}{q})$ be a fixed real number and set $r = \lfloor \rho n \rfloor$. In this case, Theorem 6.2 yields,

$$M_q(n,t,s) \leq \frac{q^{n-t}}{\min\limits_{\substack{\mathbf{x} \in \mathcal{B}_q(n) \\ r(\mathbf{x}) > \lfloor \rho n \rfloor}} |\mathcal{V}_{t,0,s}(\mathbf{x})|} + q \sum_{i=1}^{\lfloor \rho n \rfloor} \binom{n-1}{i-1} (q-1)^{i-1}.$$

The idea of the proof is to make the first term explicit using an appropriate lower bound for the minimum. Then, we show that the first term dominates the second term asymptotically and leads to the desired result.

For the first term, we make the denominator explicit using the lower bound on $|\mathcal{V}_{t,0,s}(\mathbf{x})|$ from Lemma 3.5. This bound is increasing in r and thus we find,

$$\min_{\substack{\mathbf{x} \in \mathcal{B}_q(n) \\ r(\mathbf{x}) > r}} |\mathcal{V}_{t,0,s}(\mathbf{x})| \geq \sum_{i=0}^{s} \binom{\lfloor \frac{r}{2} \rfloor}{i} (q-1)^i \cdot \sum_{j=0}^{t} \binom{\lceil \frac{r}{2} \rceil - t}{j} \geq \sum_{i=0}^{s} \binom{\lfloor \frac{\rho n}{2} \rfloor}{i} (q-1)^i \cdot \sum_{j=0}^{t} \binom{\lfloor \frac{\rho n}{2} \rfloor - t}{j}.$$

In the asymptotic regime where $n \to \infty$, with fixed $k \in \mathbb{Z}_{\geq 1}$ and $\rho' \in (0,1)$, it holds that $\binom{\lfloor \rho' n \rfloor}{k} = \mathcal{O}(n^k)$. Hence, there exist a constant $C > 0$ (with respect to n, but possibly dependent on q, s, t and ρ) and an $n' \in \mathbb{Z}_{\geq 0}$ such that

$$\sum_{i=0}^{s} \binom{\lfloor \frac{\rho n}{2} \rfloor}{i} (q-1)^i \cdot \sum_{j=0}^{t} \binom{\lfloor \frac{\rho n}{2} \rfloor - t}{j} \geq C \cdot n^s \cdot n^t,$$

for all $n \geq n'$. Next, we bound the second term from above in terms of the q-ary entropy function H_q. Note that $\frac{r}{n} \leq \rho < 1 - \frac{1}{q}$, and thus we can apply Lemma 3.6 and as a result we obtain,

$$q \sum_{i=1}^{\lfloor \rho n \rfloor} \binom{n-1}{i-1} (q-1)^{i-1} < q \sum_{i=0}^{\lfloor \rho n \rfloor - 1} \binom{n}{i} (q-1)^i \leq q \cdot q^{n H_q(\frac{\lfloor \rho n \rfloor}{n})} \leq q^{1 + n H_q(\rho)}.$$

Here we used that $\binom{n-1}{i} < \binom{n}{i}$ and that H_q is an increasing function. All in all, for sufficiently large n it holds that

$$M_q(n,t,s) \leq \frac{1}{C q^t} \cdot \frac{q^n}{n^{t+s}} + q^{1 + n H_q(\rho)}.$$

Observe that the first term dominates the second term, i.e., $q^{1+nH_q(\rho)} = o(\frac{q^n}{n^{t+s}})$. This holds because $\rho < 1 - \frac{1}{q}$ and thus $H_q(\rho) < 1$. Therefore, we conclude that there exists some $C' > 0$ such that for sufficiently large n it holds that

$$M_q(n,t,s) \leq C' \cdot \frac{q^n}{n^{t+s}}.$$

Now, we are able to prove the desired result,

$$\begin{aligned} \liminf_{n \to \infty} \frac{n - \log_q(M_q(n,t,s))}{(t+s)\log_q(n)} &\geq \lim_{n \to \infty} \frac{n - \log_q(C' \frac{q^n}{n^{t+s}})}{(t+s)\log_q(n)} \\ &= \lim_{n \to \infty} \frac{-\log_q(C') + (t+s)\log_q(n)}{(t+s)\log_q(n)} \\ &= 1, \end{aligned}$$

where we used that C' is a constant with respect to n. This concludes the proof. □

B

Python code

```
### Creating the set V_{t', t'', s}(x) for x in B_q(n). x is a string.
import math

def insert(x, k, i):
    # insert symbol k on position i into x where 0 <= i <= n
    return x[0:max(0,i)] + str(k) + x[i:len(x)]
def substitute(x, k, i):
    # substitute symbol k on position i into x where 1 <= i <= n
    return x[0:max(0,i-1)] + str(k) + x[i:len(x)]
def delete(x,i):
    # delete position i from x
    return x[0:max(0,i-1)] + x[i:len(x)]

def insertion_set(x, q):
    # returns the set of words that can be reached
    # from x by 1 insertion
    n = len(x)
    symbols = range(0,q)
    x_ins = []
    for i in range(0, n+1):
        for k in symbols:
            y = insert(x, k, i)
            x_ins.append(y)
    return set(x_ins)

def substitution_set(x, q):
    # returns the set of words that can be reached
    # from x by at most 1 substitution
    n = len(x)
    symbols = range(0,q)
    x_sub = []
    for i in range(1, n+1):
        for k in symbols:
            y = substitute(x, k, i)
            x_sub.append(y)
    return set(x_sub)
```

```
def deletion_set(x):
    # returns the set of words that can be reached from x by 1 deletion
    n = len(x)
    x_del = []
    for i in range(1, n+1):
        y = delete(x, i)
        x_del.append(y)
    return set(x_del)

def V(x, q, td, ti, ss):
    # returns the set of words that can be reached from x by exactly
    # td deletions, ti insertions and at most ss substitutions.
    # The 'set' function ensures that no words are counted double.
    V_list = [x]
    while td > 0:
        y_list = []
        for x in V_list:
            y_list += list(deletion_set(x))
        V_list = list(set(y_list))
        td -= 1

    while ss > 0:
        y_list = []
        for x in V_list:
            y_list += list(substitution_set(x,q))
        V_list = list(set(y_list))
        ss -= 1

    while ti > 0:
        y_list = []
        for x in V_list:
            y_list += list(insertion_set(x,q))
        V_list = list(set(y_list))
        ti -= 1
    return V_list
```