A Study of Binary Codes Resistant to One and Two Bit Deletions

Hector Krentzel
h.krentzel@sacre-coeur.school

under the direction of
Aleksa Stanković
Department of Mathematics
KTH Royal Institute of Technology

Research Academy for Young Scientists
July 14, 2021

Abstract

When two parties communicate over a noisy channel, information can get lost. One way this can happen is that a certain amount of bits get deleted. For a message to remain legible, both parties have to agree to certain properties. The sender cannot send two strings with n bits that might end up becoming the same substring with $n-2$ bits. In this case, the receiver will not be able to reconstruct the message, because they might end up with the same substring with $n-2$ bits twice although they mean different things. This gives the sender a set C to choose the code words x from, where no strings share a common substring. The upper and a lower bound for the size of C was previously known to be $\frac{2^n}{n^4} \leq |C| \leq \frac{2^n}{n^2}$. The upper bound however is not optimal because it counts some strings multiple times. This can be avoided by implementing a new term f which represents the maximum amount of times that a string is counted. While this paper does not claim to prove a lower upper bound, the new approach might turn out to be useful in the future.

Acknowledgements

I would like to thank my mentor Aleksa Stanković for helping me with this paper and everything associated with it. Furthermore, I would like to thank my colleague Olle Lapidus for helping me with some of the ideas. I would also like to thank the organizers of Rays – for Excellence, Max Kenning, Miranda Carlsson, Markus Swift, and Ann-Kristin Malz for giving me this opportunity. Lastly, I would like to express my gratitude towards the collaborative partners of Rays – for Excellence which are the Kjell & Märta Beijers Stiftelse and AstraZeneca.

Contents

1 Introduction

In this paper the following scenario is considered: Two parties, from now on called person A and person B, are trying to communicate over a noisy channel in which a certain amount of information can get lost. In particular, suppose person A wants to send a n-bit string $(x_1, ..., x_n)$ to person B, but 2 bits get deleted. This leads to person B receiving a $n-2$-bit string $(y_1, ..., y_{n-3}, y_{n-2})$. Note that person B does not know the location of the bits that get deleted. Person B wants to recover the original string.

This paper studies a given bound for the minimal amount of necessary information needed for a message to be recoverable in spite of a one- or two-bit deletion.

1.1 Preliminaries

In this section concepts from coding theory and graph theory are introduced. A reader interested in more detailed introduction to coding theory is referred to [1], while for the introduction to graph theory [2] can be consulted.

1.1.1 Coding Theory

There are multiple steps involved in the communication between two parties. Person A wants to send $\ell \in \mathbb{N}$ different words to person B. Person A chooses a word $m \in M$ where M is the given vocabulary that person A can access. An example for a vocabulary could be M={"hello", "bye"}. Now person A can send a message that either says "hello" or "bye". Before sending this word, person A encodes the word m into a binary string $x \in C$ of length $n \in \mathbb{N}$ where C is a set of strings with n bits. Every word gets encoded into a unique binary string, where the number of bits n for every string remains constant. All the encoded words are contained within the code $C \subseteq \{0,1\}^n$ which is comparable to the vocabulary M for the original message m. $\{0,1\}^n$ is the set of all the binary strings of length n. Therefore, every string x that person A sends can be described as $x \in \{0,1\}^n$. The encoded message gets sent over a noisy channel by person A to person B. Due to the

channel being noisy a loss of information on the string x occurs. This information loss can happen in many different ways. This paper focuses on the setting where $k \in \mathbb{N}$ bits get deleted, and the location of the deletions are unknown. Person B receives a corrupted version of the original string $y \in \{0,1\}^{n-k}$.

The construction of optimal codes for various values of k has been an important open question which has attracted significant research interest in the past decades. In the single-deletion case ($k = 1$), an explicit construction of a code of asymptotic size $O(2^n/n)$ by Varshamov-Tenengolts [3] was shown to be optimal by Levenshtein [4]. With asymptotic size is meant that these bounds are only true for large values for n, which is implied by the O. Furthermore, these codes have very efficient encoding/decoding schemes and hence are suitable for practical applications. However, for $2 \leq k < n/2$ finding the optimal construction remains elusive. While it is known that the optimal codes with k-deletion and n bits have size

$$O(\frac{2^n}{n^{2k}}) \leq |C| \leq O(\frac{2^n}{n^k}), \tag{1}$$

the question of bridging this gap between both bounds remains open even after extensive effort by the research community. Recent progress has been made in construction of explicit codes for $k = 2$ deletions [5], i.e. codes which have poly(n)-time encoding and decoding schemes, and which match the existential bound $O(2^n/n^4)$ stated above. Finding explicit codes for general values of k has been studied by Brakensiek and others [6, 7, 8, 9]. While these works construct explicit codes, they still use hashing based recursive techniques and other approaches which makes the coding schemes have size asymptotically smaller than the one given by the existential bound $O(2^n/n^{2k})$. The case when k is very close to $n/2$ has been studied by a beautiful recent result [10] which improves on the upper bound stated above.

This work focuses on the case $k = 2$ and discusses a possible approach for bridging the gap between the upper and the lower bound stated in the previous paragraph. Therefore,

the string that person B receives can also be described as a sub-string with $y \in \{0,1\}^{n-2}$. Person B needs to recover the original string x from y, where these two could be very different due to the fact that a single deletions is capable of shifting the entire string. Note that there are strings, from now on called *special strings*, that are not affected by these shifts as they either consist of only ones or only zeros. This is where so called deletion-correction codes prove to be useful because they make it possible for person B to recreate the original string x. This process can be seen in Figure 1.

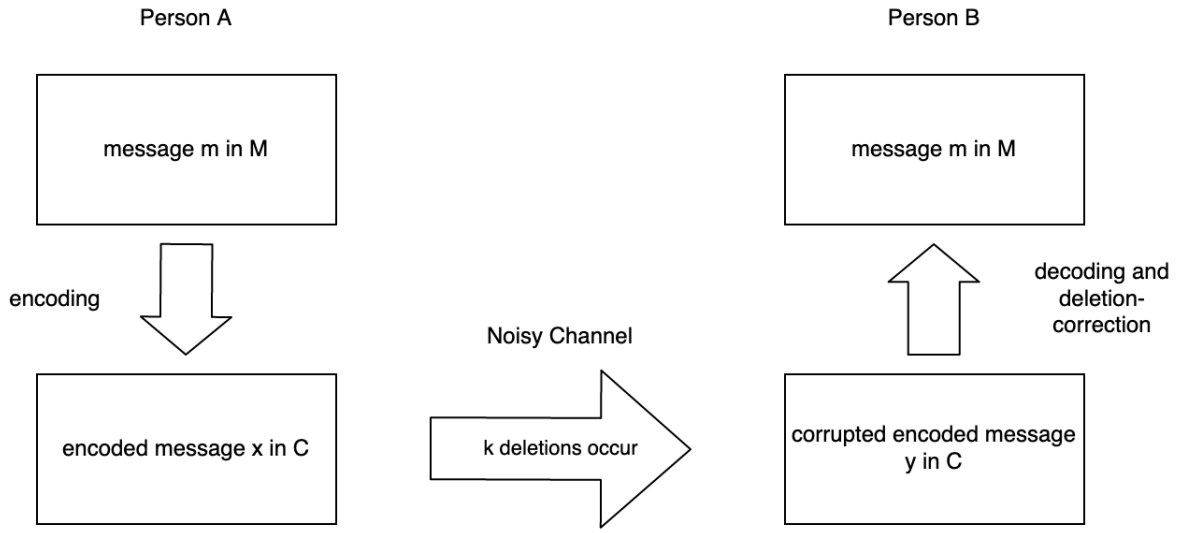

Figure 1: Process of message transmission in coding

An example of a so called deletion-correction code is the repetition code. A repetition code encodes m by first using the smallest number of bits to represent it as a binary string w, and then constructs x by repeating w r times. These repetition codes are very inefficient as the encoded message grows with a factor r. Therefore this paper looks into more efficient methods.

1.1.2 Graph Theory

Now useful concepts in graph theory will get introduced. A graph $G = (V, E)$ consists of a set of vertices V and a set of edges E, where each edge $e = (v, u) \in E$ has a pair of vertices $u, v \in V$ as its endpoints. Vertices that are connected through an edge are *neighbours*. For a vertex $v \in V$ we define the *neighbourhood* $N(v)$ as the set of all neighbours of v,

i.e. the set of all $u \in V$ such that $e = (u, v) \in E$. We use $d_v := |N(v)|$ for the number of neighbours of v, which we also call the degree of v.

Next, we define an *independent set* $I \subseteq V$ as a set of vertices in which no two vertices $v, u \in I$ are neighbours of each other. An example of a graph and an independent set is given in Figure 2. Observe that a graph can have many different independent sets.

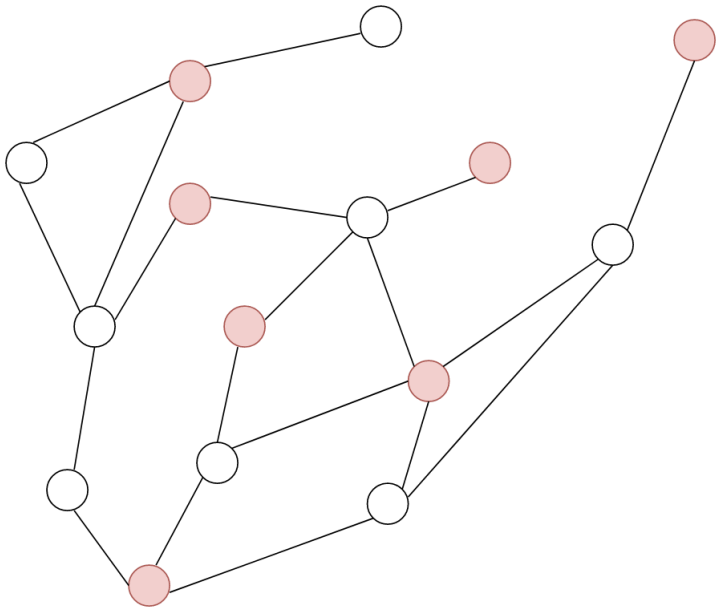

Figure 2: The coloured nodes are an example for an independent set

1.2 Coding Theory Meets Graph Theory

Coding theory and graph theory can be combined in order to create a new deletion-correction code that is more efficient than the repetition code that got mentioned earlier. Each string $x \in \{0, 1\}^n$ gets assigned to a vertex $v \in V$. These vertices share an edge $e \in E$ if their respective strings share a sub-string $y \in \{0, 1\}^{n-k}$. If this is the case, the two nodes representing the strings are *connected*. If these connected strings x_1 and x_2 get sent through the noisy channel they could end up becoming identical after the deletion process due to the shared substring y. An independent set $I \subseteq V$ therefore consists of strings that are not neighbours of each-other. An example of a graph constructed as explained above can be seen in Figure 3 where $n = 3$ and $k = 2$. Furthermore, the coloured

sequences are part of an independent set.

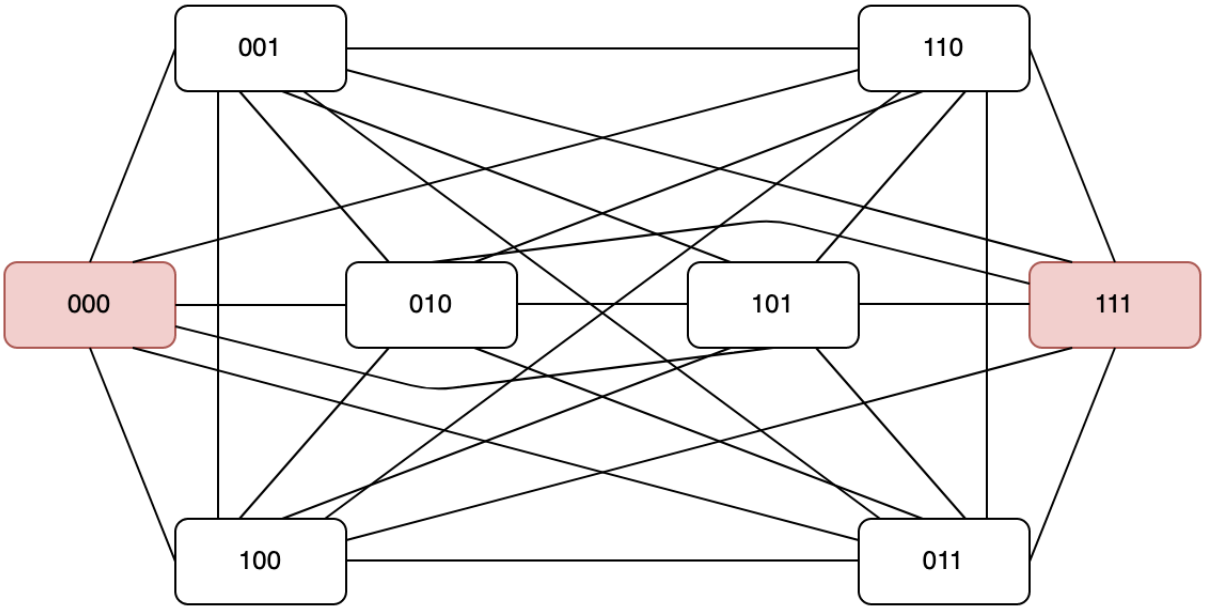

Figure 3: The coloured strings are the only independent set since any other node shares an edge with every node. In this example, $n = 3$ and $k = 2$

1.2.1 Greedy Deletion-Correction

Person A and person B have to agree on an independent set. Otherwise, person B will not be able to reconstruct the message correctly, because person B might receive the same substring y twice although they originate from different strings x_1 and x_2. The code corresponding to the largest independent set is the most effective among all 2-bit binary deletion codes on n-bits.

Lemma 1: *Given a vocabulary M that corresponds to a code $C \subseteq \{0,1\}^n$, where C is an independent set. The set C has a maximum asymptotic size of $O(\frac{2^n}{n^4}) \leq |C| \leq O(\frac{2^n}{n^2})$.*

Proof: This proof uses the degree of a node, where there are two values for the degree. The maximum degree $D = n^4$ due to there being n^2 ways of deleting two bits and then again n^2 ways of inserting two bits. The minimum degree $d = n^2$ due to there being

certain special strings with only one way of deleting two bits but still n^2 ways of inserting two bits.

The lower bound for the size of code C can be calculated by dividing the set of all the vertices V by the maximum degree D of the nodes, resulting in $|c| \geq \frac{2^n}{n^4}$.

The upper bound however originates from the following approach. The size of the set of all the vertices $v \in V$ is equal to the union of the neighbourhoods N of the vertices $v \in C$. This can be seen in equation(2).

$$|V| = |\bigcup_{v \in C} N(v)| \tag{2}$$

Furthermore, it can be stated that the union of the neighbourhoods N of the vertices $v \in C$ is approximately equal to the sum of all the neighbourhoods N of the vertices $v \in C$ [5]. See equation (3).

$$|\bigcup_{v \in C} N(v)| \approx \sum_{v \in C} |N(v)| \tag{3}$$

The sum of all the neighbourhoods N of the vertices $v \in C$ can also be calculated by multiplying the size of the code $|C|$ with the degree. In this case the minimum degree d gets used due to this being the upper bound. This can be seen in equation (4).

$$\sum_{v \in C} |N(v)| = |C| \cdot n^2 \tag{4}$$

By combining the equations (2), (3) and (4) the upper bound for $|C|$ can be created, resulting in $|C| \leq \frac{2^n}{n^2}$.

The upper and lower bound combined result in the bound seen in equation (5) with asymptotic growth due to these bounds only being true for large n's.

$$O(\frac{2^n}{n^4}) \leq |C| \leq O(\frac{2^n}{n^2}) \tag{5}$$

□

1.2.2 Reason for Suspecting that the Upper Bound is not Optimal

As can be seen in equation(3) it gets stated that the union of the neighbourhoods N of the vertices $v \in C$ is approximately equal to the sum of all the neighbourhoods N of the vertices $v \in C$. However, this is not true due to the fact that the neighbourhoods N of the vertices $v \in C$ are overlapping. An example for this would be the graph seen in Figure 3 where the maximum independent set consists of the strings $x_1 = 000$ and $x_2 = 111$. The two nodes that represent the strings share all their neighbours, which supports the argument of the upper bound not being optimal.

2 Method

In order to improve the given bounds, a new interpretation combining graph theory and coding theory is proposed. This graph sections the type of neighbours into different tiers, where the strings within the first tier only differ from the original string by one shift, the second tier neighbours by two, the third by three, and so on. However, a neighbour can at most only differ from the original string by four shifts (a shift occurs when a bit deletion results in the remaining bits shifting to the left in order to fill the gap created by the deletion) due to the fact that all four shifts can get deleted because both strings can delete two. If this happens they share the same substring with $n - 4$ bits. Note that there are approximately n first tier neighbours (because there are $n - 1$ different ways of deleting one bit and therefore shifting the string by one). There are approximately n^2 amount of second tier neighbours (because there are $\binom{n}{2}$ different ways in which two bits can get deleted causing two shifts). For the same reasons there are approximately n^3 amount of third tier neighbours and n^4 amount of fourth tier neighbours. These approximations are asymptotically true. Furthermore, the first tier neighbours are included into the second tier neighbours and so on. Thus meaning that the fourth tier neighbours include all the others. A representation of this graph can be seen in Figure 4.

2.1 Improving the Upper Bound

To improve the upper bound the previously mentioned overlaps within the neighbourhoods N of the nodes of the independent set $v \in C$ have to be considered. These overlaps can be taken into account by introducing a new set F, with size $f = |F|$. F is defined as the maximum independent set of a neighbourhood of a node. By doing this, the amount of nodes $v \in C$ sharing a neighbour can be calculated. This new term alongside a constant j can be inserted into equation (3) resulting in equation (6). The constant j is added due to the fact that the equation is not optimal, but not much is known about this constant.

$$|\bigcup_{v \in C} N(v)| \approx \sum_{v \in C} |N(v)| \cdot \frac{1}{f} \cdot j \tag{6}$$

With this improved interpretation a new upper bound can be created. However, n^4 is used as the degree because this will be true for most strings. Therefore the new upper bound seen in equation (7) can be acquired.

$$\frac{2^n}{n^4} \leq |C| \leq \frac{2^n \cdot f}{n^4 \cdot j} \tag{7}$$

Lemma 2: *f is bounded by n^4 asymptotically.*

Proof: Neighbours of first and second tier will always share and edge whereas neighbours of third and fourth tier will not necessarily. This is because neighbours of first and second tier are separated by four shifts or less. Therefore only neighbours of third degree and fourth tier can be a part of F which can be seen in Figure (4) as the coloured sections. As the n^3 third tier neighbours are included into the n^4 fourth tier neighbours f results in being having at most a value of n^4. However, the neighbours of first and second tier should be excluded as they all share an edge with each-other. As the n first tier neighbours are included into the n^2 second tier neighbours $f = n^4 - n^2$.

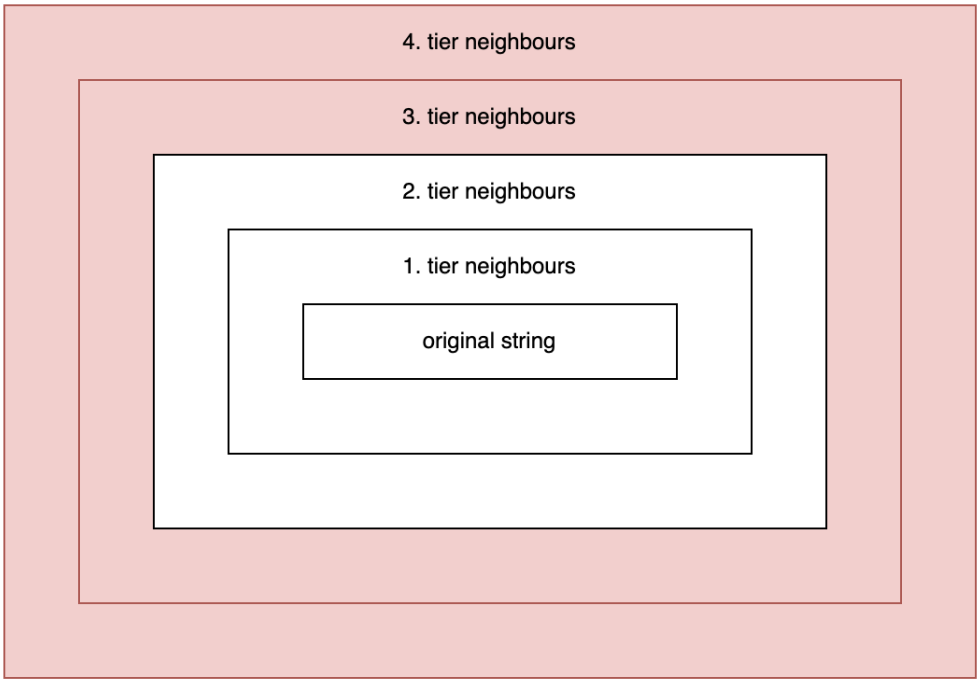

Figure 4: The different tiers within the neighbourhood of a string where the coloured tiers represent the neighbours that are part of the set F

□

3 Results and Discussion

As the previous upper bound was $\frac{2^n}{n^2}$, f needs to be below n^2 in order for there to be an improvement. For the case of the two bit deletion f proved to be greater than n^2. To be precise it resulted in being $f = n^4 - n^2$ which is asymptotically equivalent to n^4. This can be seen as an upper bound for f due to the fact that neighbours of third and fourth tier will often be connected. This upper bound is trivial because f can never be greater than n^4 as this also is the maximum degree. Another factor that limits this approach is that it only works for regular strings that are not special strings. With a special string the shifting effect of the deletions would be minimal because no position changes. It also does not work because strings of third and fourth tier will also be neighbours of each-other in many cases.

Although this paper did not manage to improve the previously set bounds it still

managed to find a new approach. This new approach included a new interpretation of the combination of the graph and coding theory. If the limiting factors mentioned above could be mathematically incorporated, a better result may be achieved. Additionally, this new way of viewing the problem might prove to be useful in the future, where a value of f below n^2 is desired. As not much is known about the constant j, new information about this constant would improve the bounds and would also be essential for optimizing the bounds.

References

[1] Venkatesan Guruswami AR, Sudan M. Essential Coding Theory;.

[2] Bollobás B. Modern Graph Theory. 1st ed. Graduate Texts in Mathematics 184. Springer-Verlag New York; 1998.

[3] Varshamov RR, Tenengolts GM. Codes which correct single asymmetric errors. Autom Remote Control. 1965.

[4] Levenshtein VI. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics Doklady. 1966.

[5] Guruswami V, Håstad J. Explicit two-deletion codes with redundancy matching the existential bound. In: Marx D, editor. Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021. SIAM; 2021. p. 21–32. Available from: https://doi.org/10.1137/1.9781611976465.2.

[6] Brakensiek J, Guruswami V, Zbarsky S. Efficient Low-Redundancy Codes for Correcting Multiple Deletions. IEEE Trans Inf Theory. 2018;64(5):3403–3410. Available from: https://doi.org/10.1109/TIT.2017.2746566.

[7] Belazzougui D. Efficient Deterministic Single Round Document Exchange for Edit Distance. CoRR. 2015;abs/1511.09229. Available from: http://arxiv.org/abs/1511.09229.

[8] Cheng K, Jin Z, Li X, Wu K. Deterministic Document Exchange Protocols, and Almost Optimal Binary Codes for Edit Errors. In: Thorup M, editor. 59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018. IEEE Computer Society; 2018. p. 200–211. Available from: https://doi.org/10.1109/FOCS.2018.00028.

[9] Haeupler B. Optimal Document Exchange and New Codes for Insertions and Deletions. In: Zuckerman D, editor. 60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019. IEEE Computer Society; 2019. p. 334–347. Available from: https://doi.org/10.1109/FOCS.2019.00029.

[10] Guruswami V, He X, Li R. The zero-rate threshold for adversarial bit-deletions is less than 1/2. Electron Colloquium Comput Complex. 2021;28:79. Available from: https://eccc.weizmann.ac.il/report/2021/079.