The Error Probability of Maximum-Likelihood Decoding over Two Deletion Channels

Omer Sabary  
Dept. of Computer Science  
Technion — Israel Inst. of Technology  
Haifa 3200003, Israel  
omersabary@cs.technion.ac.il

Eitan Yaakobi  
Dept. of Computer Science  
Technion — Israel Inst. of Technology  
Haifa 3200003, Israel  
yakobi@cs.technion.ac.il

Alexander Yucovich  
Dept. of Computer Science  
Technion — Israel Inst. of Technology  
Haifa 3200003, Israel  
yucovich@gmail.com

Abstract—This paper studies the problem of reconstructing a word given several of its noisy copies. This setup is motivated by several applications, among them is reconstructing strands in DNA-based storage systems. Under this paradigm, a word is transmitted over some fixed number of identical independent channels and the goal of the decoder is to output the transmitted word or some close approximation. The main focus of this paper is the case of two deletion channels and studying the error probability of the maximum-likelihood (ML) decoder under this setup. First, it is discussed how the ML decoder operates. Then, we observe that the dominant error patterns are deletions in the same run or errors resulting from alternating sequences. Based on these observations, it is derived that the error probability of the ML decoder is roughly $q^{-1}p^2$, when the transmitted word is any $q$-ary sequence and $p$ is the channel’s deletion probability. We also study the cases when the transmitted word belongs to the Varshamov Tenengolts (VT) code or the shifted VT code. Lastly, the insertion channel is studied as well. These theoretical results are verified by corresponding simulations.

I. INTRODUCTION

Reconstruction of sequences refers to a large class of problems in which there are several noisy copies of the information and the goal is to decode the information, either with high probability or in the worst case. One of the more relevant models to the study in this paper is the trace reconstruction problem [6, 17, 18, 29, 31], where it is assumed that a sequence is transmitted through multiple deletion channels, so each bit is deleted with some fixed probability $p$. Under this setup, the goal is to determine the minimum number of traces, i.e., channels, required to reconstruct the sequence with high probability. Other examples include the sequence reconstruction problem which was first studied by Levenshtein [13], [23]–[26], [32], [41], [42]. One of the dominant motivating applications of the sequence reconstruction problems is DNA storage [2], [9], [14], [29], [43], where every DNA strand has several noisy copies.

Many of the reconstruction problems are focused on studying the minimum number of channels required for successful decoding. However, in many cases, the number of channels is fixed and then the goal is to find the best code construction that is suitable for this channels setup. Motivated by this important observation, the goal of this paper is to study the error probability of maximum-likelihood decoding when a word is transmitted over two deletion or insertion channels. This study is also motivated by the recent works of Srinivasavaradhan et al. [35], [36], where reconstruction algorithms for the maximum-likelihood have been studied. Abroshan et al. presented in [1] a new coding scheme for sequence reconstruction which is based on the Varshamov Tenegolts (VT) code [40] and in a parallel work [22] it is studied how to design codes for the worst case, when the number of channels is given.

When a word is transmitted over the deletion channel, the channel output is necessary a subsequence of the transmitted word. Hence, when transmitting the same word over multiple deletion channels, the possible candidate words for decoding are the so-called common supersequences of all channels’ outputs. Hence, an important part of the decoding process is to find the set of all possible common supersequences and in particular the longest common supersequences (LCS) [21]. Even though this problem is in general NP hard [6] for an arbitrary number of sequences, for two words a dynamic programming algorithm exists with quadratic complexity; see [21] for more details and further improvements and approximations for two or more sequences [15], [20], [38], [39]. The case of finding the longest common subsequences (LCS) is no less interesting and has been extensively studied in several previous works; see e.g. [3], [8], [16], [19], [27], [33]. Most of these works focused on improving the complexity of the dynamic programming algorithm suggested in [3] and presented heuristics and approximations for the LCS.

When a sequence is transmitted over two (or more) deletion channels, the first step in the ML decoding algorithm is to build upon the algorithm for finding the SCS to generate all possible candidate words, i.e., all shortest common supersequences. However, if there is more than one candidate, it is necessary to find the one that maximizes the probability that it was sent over all channels. It is shown that this problem is directly related to finding the embedding number between two sequences [4], [12]. This value is the number of times a sequence can be generated by deletions from its supersequence.

These two steps are in fact the process in which the ML decoder operates. Thus, the main goal of this paper is to determine the error probability of the ML decoder. Assume the deletion probability of every channel is $p$. If there are $t$ channels, then a lower bound on the error probability is at least $p^t$ since if a bit is deleted in all $t$ channels, then it will also be deleted at the ML decoder’s output. However, it will be observed that this lower bound is not tight. For example, if there are two channels and a bit is deleted in a run in both channels (i.e., not necessarily the same bit), then this run will have a deletion error as well. This indeed will be one of the main error patterns of the ML decoder. Furthermore, it will be observed that alternating subsequences in the word are also error prone and these two will be the only dominant error patterns. Thus, we will show that for arbitrary $q$-ary sequences, the error decoding probability for the runs is $q^{-1}p^2$, while for the alternating sequences is $2p^2$, independently of the field size.

The rest of the paper is organized as follows. Section II presents the notations and the formal definition of transmission over multiple channels which will be studied in the paper. Section III studies this problem for the insertion and deletion channels. In Section IV we present our main results for the case of two channels. We consider the average decoding failure probability of the ML decoder and its average decoding error probability when the code is the entire space, the VT code, and the shifted VT code. We then continue in Section V to study the equivalent problem for insertions. Section VI concludes the paper and discusses open problems. Due to the lack of space, some of the proofs in the paper are omitted.
II. DEFINITIONS AND PRELIMINARIES

We denote by $\Sigma_q = \{0, \ldots, q - 1\}$ the alphabet of size $q$ and $\Sigma_q^n = \bigcup_{i=0}^{n} \Sigma_q^i$. The length of $x \in \Sigma^n$ is denoted by $|x|$. The Levenshtein distance between two words $x, y \in \Sigma_q^*$, denoted by $d_{L}(x,y)$, is the minimum number of insertions and deletions required to transform $x$ into $y$, and $d_{H}(x,y)$ denotes the Hamming distance between $x$ and $y$, when $|x| = |y|$. A word $x \in \Sigma_q^n$ will be referred as an alternating sequence if it cyclically repeats all symbols in $\Sigma_q$ in the same order. For example, for $\Sigma_2 = \{0,1\}$, the two alternating sequences are 010101 and 101010, and in general there are $q!$ alternating sequences. For $n \geq 1$, the set $\{1, \ldots, n\}$ is abbreviated by $[n]$.

For a word $x \in \Sigma_q^n$ and a set of indices $I \subseteq [|x|]$, the word $x_I$ is the projection of $x$ on the indices of $I$ which is the subsequence of $x$ received by the symbols in the entries of $I$. A word $x \in \Sigma_q^*$ is called a supersequence of $y \in \Sigma_q^*$, if $y$ can be obtained by deleting symbols from $x$; that is, there exists a set of indices $I \subseteq [|x|]$ such that $y = x_I$. In this case, it is also said that $y$ is a subsequence of $x$. Furthermore, $x$ is called a common supersequence, subsequence of some words $y_1, \ldots, y_r$, if $x$ is a supersequence, subsequence of each one of these $r$ words. The set of all common supersequences of $y_1, \ldots, y_r \in \Sigma_q^n$ is denoted by $SCS(y_1, \ldots, y_r)$ and $SCS(y_1, y_r)$ is the length of the shortest common supersequence (SCS) of $y_1, \ldots, y_r$, that is, $SCS(y_1, \ldots, y_r) = \min_{x \in SCS(y_1, \ldots, y_r)} |x|$. Similarly, $LCS(y_1, \ldots, y_r)$ is the set of all subsequences of $y_1, \ldots, y_r$, and $LCS(y_1, y_r)$ is the length of the longest common subsequence (LCS) of $y_1, \ldots, y_r$, that is, $LCS(y_1, \ldots, y_r) = \max_{x \in LCS(y_1, \ldots, y_r)} |x|$.

We consider a channel $S$ that is characterized by a conditional probability $P_{R_S}$, which is defined by

$$P_{R_S}(y \text{ rec.} | x \text{ trans.}) = \prod_{i=1}^{n} P_{R_{S_i}}(y_i \text{ rec.} | x_i \text{ trans.}) = p_{tr}(y_i | x_i) (1 - p)^{|x| - d_{tr}(y_i,x_i)},$$

for all $(x,y) \in (\Sigma_q^*)^2$. Note that it is not assumed that the lengths of the input and output words are the same as we consider also deletions and insertions of symbols, which is the main topic of this work. As an example, it is well known that if $S$ is the binary symmetric channel (BSC) with crossover probability $0 \leq p \leq 1/2$, denoted by $BSC(p)$, it holds that

$$P_{BSC}(p) = \prod_{i=1}^{n} P_{R_{S_i}}(y_i \text{ rec.} | x_i \text{ trans.}) = p_{tr}(y_i | x_i) (1 - p)^{|x| - d_{tr}(y_i,x_i)},$$

for all $(x,y) \in (\Sigma_q^*)^2$, and otherwise (the lengths of $x$ and $y$ is not the same) this probability equals 0. Similarly, for the $Z$-channel, denoted by $Z(p)$, it is assumed that only a 0 can change to a 1 with probability $p$ and so

$$P_{Z(p)} = \prod_{i=1}^{n} P_{R_{S_i}}(y_i \text{ rec.} | x_i \text{ trans.}) = p_{tr}(y_i | x_i) (1 - p)^{|x| - d_{tr}(y_i,x_i)},$$

for all $(x,y) \in (\Sigma_q^*)^2$ such that $x \leq y$, and otherwise this probability equals 0.

In the deletion channel with deletion probability $p$, denoted by Del$(p)$, every symbol of the word $x$ is deleted with probability $p$. Similarly, in the insertion channel with insertion probability $p$, denoted by Ins$(p)$, a symbol is inserted in each of the possible $|x| + 1$ positions of the word $x$ with probability $p$, while the probability to insert each of the symbols in $\Sigma_q$ is the same and equals $\frac{p}{q}$. A decoder for a code $C$ with respect to the channel $S$ is a function $D : \Sigma_q^n \rightarrow C$. Its average decoding failure probability is defined by $P_{fail}(S, C, D) = \sum_{c \in C} P_{fail}(c, S, C, D)$, where

$$P_{fail}(c) = \sum_{y : D(y) \neq c} P_{R_S}(y \text{ rec.} | c \text{ trans.}).$$

We will also be interested in the average decoding error probability which is the average normalized distance between the transmitted word and the decoder’s output. The distance will depend upon the channel of interest. For example, for the BSC we will consider the Hamming distance, while for the deletion and insertion channels, the Levenshtein distance will be of interest. Hence, for a channel $S$, distance function $d$ and a decoder $D$, we let $P_{err}(S, C, D, d) = \sum_{c \in C} P_{fail}(c, S, C, D, d)$, where

$$P_{err}(c, d) = \sum_{y : D(y) \neq c} d(y, c) \cdot P_{R_S}(y \text{ rec.} | c \text{ trans.}).$$

The maximum-likelihood (ML) decoder for $C$ with respect to $S$, denoted by $D_{ML}$, outputs a codeword $c \in C$ that maximizes the probability $P_{R_S}(y \text{ rec.} | c \text{ trans.})$. That is, for $y \in \Sigma_q^n$,

$$D_{ML}(y) = \arg \max_{c \in C} \{P_{R_S}(y \text{ rec.} | c \text{ trans.})\}.$$
the code is the entire space, the Varshamov Tenengolts (VT) code \cite{40}, and the shifted VT (SVT) code \cite{34}.

This model is closely connected to several related problems. In the reconstruction problem studied by Levenshtein \cite{24, 25}, it was assumed that the word is transmitted over several noisy channels and the goal of the decoder is to decode the transmitted word in the worst case, assuming that all channels’ outputs are different from each other. Several extensions of these problems have been studied; see e.g. \cite{13, 23, 26, 32, 41, 42}, however in all of them the goal is to find the number of channel that guarantees unique decoding in the worst case. The most relevant works to our study are the recent one \cite{19, 21}, where decoding algorithms for maximum likelihood are presented for a fixed number of channels.

The most relevant case of the reconstruction problem to our work is the one studied in \cite{7}, where it was shown how the shifted VT codes can be used for the trace reconstruction problem. Yet another a highly related problem is the one of noisy channels and the goal of the decoder is to decode the word, for many of the previous works see e.g. \cite{13, 23, 26, 29, 31}. The most relevant works to our study are the recent ones \cite{35, 36}, where decoding algorithms for maximum likelihood are presented for a fixed number of channels.

III. THE DELETION AND INSERTION CHANNELS

In this section we establish several basic results for the deletion channel with multiple instances. We start with several useful definitions. For two words $x, y \in \Sigma_x^n$, the number of times that $y$ can be received as a subsequence of $x$ is called the embedding number of $y$ in $x$ and is defined by

$\text{Emb}(x; y) = |\{ I \subseteq [|x|] \mid x_{|I|} = y \}|.$

Note that if $y$ is not a subsequence of $x$ then $\text{Emb}(x; y) = 0$. The embedding number has been studied in several previous works; see e.g. \cite{4, 12} and in \cite{35} it was referred as the binomial coefficient. In particular, this value can be computed with quadratic complexity \cite{12}.

While the calculation of the conditional probability $P_{\text{Del}}(y \mid x)$ is a rather simple task for many of the known channels, it is not straightforward for channels which introduce insertions and deletions. The following basic lemma is well known and was also stated in \cite{35}, however it is presented here for the completeness of the results in the paper and since it will be used in our derivations to follow.

**Claim 1.** For all $(x, y) \in (\Sigma_x^n)^2$, it holds that

$P_{\text{Del}}(y \mid x) = [x]|y| \cdot \text{Emb}(x; y),$

$P_{\text{Ins}}(y \mid x) = [y]|x| \cdot \text{Emb}(y; x).$

According to Claim 1 it is possible to explicitly characterize the ML decoder for the deletion and insertion channels as described also in \cite{35}.

**Claim 2.** Assume $c \in C \subseteq (\Sigma_x^n)^n$ is the transmitted word and $y \in (\Sigma_x)^\leq n$ is the output of the deletion channel $\text{Del}(p)$, then

$D_{\text{ML}}(y) = \arg \max_{c \in C} \{ \text{Emb}(c; y) \}.$

Similarly, for the insertion channel $\text{Ins}(p)$, for $y \in (\Sigma_x)^\geq n$,

$D_{\text{ML}}(y) = \arg \max_{c \in C} \{ \text{Emb}(y; c) \}.$

In case there is more than a single instance of the deletion channel, the following claim follows.

**Claim 3.** Assume $c \in C \subseteq (\Sigma_x^n)^n$ is the transmitted word and $y_1, \ldots, y_t \in (\Sigma_x)^\leq n$ are the output words from $\text{Del}(p)$, then

$D_{\text{ML}}(y_1, \ldots, y_t) = \arg \max_{c \in C \cap \Sigma_x y_i} \{ \prod_{i=1}^t \text{Emb}(c; y_i) \},$

and for the insertion channel $\text{Ins}(p)$, for $y_1, \ldots, y_t \in (\Sigma_x)^\geq n$,

$D_{\text{ML}}(y_1, \ldots, y_t) = \arg \max_{c \in C \cap \Sigma_x y_i} \{ \prod_{i=1}^t \text{Emb}(y_i; c) \}.$

**Proof:** Every candidate to be considered in the ML decoder is a common subsequence of $y_1, \ldots, y_t$. Hence, $D_{\text{ML}}(y) = c$, where $c \in C \cap \Sigma_x y_i$ maximizes

$P_{\text{Del}}(y_1, \ldots, y_t, \mid \text{trans.}) = \prod_{i=1}^t p(y_i | \text{del.}) \cdot \text{Emb}(c; y_i).$

Since $\prod_{i=1}^t p(y_i | \text{del.})$ is the same for all candidates $c$, the statement holds. A similar proof holds for the insertion channel.

Note that since there is more a single channel, when the goal is to minimize the average decoding error probability, the ML decoder does not necessarily have to output a codeword but any word that minimizes the average decoding error probability. Thus, for the rest of the paper, when discussing the average decoding error probability it is assumed that the ML decoder can output any word and not necessarily a codeword from $C$. Thus, we get the following claim.

**Claim 4.** Assume $c \in C \subseteq (\Sigma_x^n)^n$ is the transmitted word and $y_1, \ldots, y_t \in (\Sigma_x)^\leq n$ are the output words from $\text{Del}(p)$, then

$D_{\text{ML}}(y_1, \ldots, y_t) = \arg \max_{x \in \Sigma_x y_i} \{ \prod_{i=1}^t \text{Emb}(x; y_i) \},$

and for the insertion channel $\text{Ins}(p)$, for $y_1, \ldots, y_t \in (\Sigma_x)^\geq n$,

$D_{\text{ML}}(y_1, \ldots, y_t) = \arg \max_{x \in \Sigma_x y_i} \{ \prod_{i=1}^t \text{Emb}(y_i; x) \}.$

Assume $C = \Sigma_x^n$. The average decoding failure probability of the ML decoder over the deletion channel $\text{Del}(p)$ with $t$ instances is denoted by $P_{\text{fail}}(\text{Del}(p), t, \Sigma_x^n, D_{\text{ML}})$ and shortly $P_{\text{fail}}(q, p, t)$. Similarly, the average decoding error probability is $P_{\text{err}}(\text{Del}(p), t, \Sigma_x^n, D_{\text{ML}}, d_1)$ and shortly $P_{\text{err}}(q, p, t)$.

If $t = 2$ it will be removed from the notations.

Our main goal in the rest of the paper is to calculate close approximations for $P_{\text{fail}}(q, p, t)$ and $P_{\text{err}}(q, p, t)$ when $t = 2$. Note that a lower bound on these probabilities is $p^t$ since if the same symbol is deleted in all of the channels, then it is not possible to recover its value and thus it will be deleted also in the output of the ML decoder. This was already observed in \cite{35} and in their simulation results. In the next section, we will analyze these probabilities for the special case of $t = 2$, when the code is $\Sigma_x^n$, the VT code \cite{40}, and the SVT code \cite{34}.

Complexity wise, it is well known that the time complexity to calculate the list of SCSSs and LCSCs is quadratic \cite{21}. This will also be the complexity to calculate the embedding numbers \cite{12} and thus the complexity of the ML decoder will be quadratic for $t = 2$. The main idea behind these algorithms uses dynamic programming in order to calculate the list of SCSSs, LCSCs, and embedding numbers for every prefixes of the given words. However, when calculating for example the SCSSs for $y_1$ and $y_2$, it is already known that $\text{SCS}(y_1, y_2) \leq n$. Hence, it is not hard to observe that (see e.g. \cite{3}) many paths corresponding to prefixes which their length difference is greater than $d_1 + d_2$ can be eliminated, when $d_1, d_2$ is the number of deletions in $y_1, y_2$, respectively. In particular, when $d_1$ and $d_2$ are fixed then the time complexity is linear. In our simulations we used this improvement when implementing the ML decoder. Other improvements and algorithms of the ML decoder are discussed in \cite{35, 36}.
IV. TWO DELETION CHANNELS

In this section we consider only the case of two deletion channels and prove in Theorem 8 an approximation for the average decoding error probability in the form of

\[ P_{\text{err}}(q,p) \approx \frac{3q-1}{q-1} p^2 + O(p^3). \]

As mentioned in Section III a lower bound on the value of \( P_{\text{err}}(q,p,t) \) is \( p^t \). This lower bound is indeed not tight since if symbols from the same run are deleted then the outputs of the two channels of this run are the same and it is impossible to recover that this run experienced a deletion in both of its copies. The probability of deletions due to runs is denoted by \( P_{\text{run}}(q,p) \) and the next lemma approximates this probability.

**Lemma 5.** For the deletion channel Del\((p)\), it holds that

\[ P_{\text{run}}(q,p) \approx \frac{q+1}{q-1} p^2. \]

**Proof:** Given a run of length \( r \), the probability that both of its copies have experienced a deletion is at roughly \((rp)^2\). Furthermore, the occurrence probability of a run of length exactly \( r \) is \( \left( \frac{1}{q} \right)^{r-1} \cdot \frac{q-1}{q} \). Thus, for \( n \) large enough, the error probability is approximated by

\[
\sum_{r=1}^{\infty} (rp)^2 \left( \frac{1}{q} \right)^{r-1} \cdot \frac{q-1}{q} = p^2 \cdot \sum_{r=1}^{\infty} \left( \frac{1}{q} \right)^{r-1} \cdot \frac{q-1}{q} = p^2 \cdot \frac{q+1}{q-1}.
\]

The expected length of a run is given by

\[
\frac{\sum_{r=1}^{\infty} r \cdot \left( \frac{1}{q} \right)^{r-1} \cdot \frac{q-1}{q} \cdot \sum_{r=1}^{\infty} r \cdot \left( \frac{1}{q} \right)^{r-1}}{\sum_{r=1}^{\infty} \left( \frac{1}{q} \right)^{r-1} \cdot \frac{q-1}{q}} = \frac{q}{q-1}.
\]

Hence, the expected number of runs in a vector of length \( n \) is \( n \cdot \frac{q-1}{q} \) and thus the approximated number of deletions due to runs is

\[
n \cdot \frac{q-1}{q} \cdot p^2 \cdot \frac{q+1}{q-1} = np^2 \cdot \frac{q+1}{q-1},
\]

which verifies the statement in the lemma.

However, runs are not the only source of errors in the output of the ML decoder. For example, assume the \( i \)-th, \((i+1)\)-st symbols are deleted from the two channels. If the transmitted word \( x \) is of the form \( x = (x_1, x_i-1, 0, 1, x_i+2, \ldots, x_n) \), then the two channels’ outputs are \( y_1 = (x_1, x_i-1, 0, x_i+2, \ldots, x_n) \) and \( y_2 = (x_1, x_i-1, 1, x_i+2, \ldots, x_n) \). However, these two outputs could also be received upon deletions exactly in the same positions if the transmitted word is \( x' = (x_1, \ldots, x_{i-1}, 1, 0, x_{i+2}, \ldots, x_n) \). Hence, the ML decoder can output the correct word only in one of these two cases. Longer alternating sequences cause the same problem as well and the occurrence probability of this event, denoted by \( P_{\text{alt}}(q,p) \), will be estimated in the next lemma.

**Lemma 6.** For the deletion channel Del\((p)\), it holds that

\[ P_{\text{alt}}(q,p) \approx 2p^2. \]

**Proof:** Assume there is a deletion in the first channel in the \( i \)-th position and the closest deletion in the second channel is \( j > 0 \) positions apart, i.e., either in position \( i - j \) or \( i + j \). For simplicity assume it is in \((i+j)\)-th position and \( x_{i+j} \) is an alternating sequence \( ABAB \cdots \). Then, the same outputs from the two channels could be received if the transmitted word is the same as \( x \) but with changing the order of the alternating sequence, that is, the symbols of the word in the positions of \([i : j]\) are \( BABA \cdots \). Therefore, the occurrence probability of this event can be approximated by

\[
2p^2 \cdot \sum_{j=1}^{\infty} \frac{q-1}{q} \cdot \frac{1}{q^{j-1}} = 2p^2,
\]

where \( \frac{q-1}{q} \cdot \frac{1}{q^{j-1}} \) is the probability that \( x_{[i:j]} \) is any alternating sequence and the multiplication by 2 takes into account the cases of deletion in either position \( i - j \) or \( i + j \).

At this point one may ask whether these two error events are the only dominant ones and indeed this question is answered in the affirmative, as stated in the following lemma.

**Lemma 7.** If there is a deletion in the first, second channel in position \( i, i+j \), where \( j > 0 \), respectively, and the sequence \( x_{[i:j]} \) is neither a run nor an alternating sequence, then these deletions are corrected successfully by the ML decoder.

We are now ready to show the following theorem on the Levenshtein error rate for the case of two channels.

**Theorem 8.** The Levenshtein error rate for two deletion channels is approximated by

\[
P_{\text{err}}(q,p) \approx P_{\text{run}}(q,p) + P_{\text{alt}}(q,p) + O(p^3) = \frac{3q-1}{q-1} p^2 + O(p^3).
\]

**Proof:** The proof follows from the above few lemmas. Note that each run error translates to an increase of the Levenshtein distance by one. On each occurrence of the alternating event the decoder chooses the correct subsequence with probability 0.5 and every error increases the Levenshtein distance by two since it translates to one insertion and one deletion. Lastly, the \( O(p^3) \) expression compensates for all other less dominant error events which introduce more than two deletions at that are close to each other at least in one of the channels.

Using these observations, we are also able to approximate the average decoding failure probability.

**Theorem 9.** The average decoding failure probability is

\[
P_{\text{fail}}(q,p) \approx e^{-\frac{3q-1}{q-1} p^2 n}.
\]

**Proof:** This is the probability that there was neither a deletion error because of the runs nor errors because of the alternating sequences. Hence, this probability becomes

\[
(1 - P_{\text{run}}(q,p))^n \cdot (1 - P_{\text{alt}}(q,p))^n 
\approx (1 - (P_{\text{run}}(q,p) + P_{\text{alt}}(q,p)))^n 
= \left( 1 - \frac{3q-1}{q-1} p^2 \right)^n 
\approx e^{-\frac{3q-1}{q-1} p^2 n}.
\]

So far we have discussed only the case in which the code \( C \) is the entire space. However, the most popular deletion-correcting code is the VT code \cite{VT}. Recently, SVT, an extension of the VT code has been proposed in \cite{SVT} for the correction of burst deletions, where the goal of this code construction was to correct a deletion error that its location is known up to some roughly \( \log(n) \) consecutive locations. However, this construction has been recently used in \cite{ML} to build a code that is specifically targeted for the reconstruction of a word that is transmitted through two single-deletion channels. Due to the relevance of correcting deletion and alternating errors, the decoding failure probabilities of these two codes is investigated in this work. We abbreviate the notation of \( P_{\text{fail}}(\text{Del}(p), 2, VT_n, DML), P_{\text{fail}}(\text{Del}(p), 2, SVT_n, DML) \) by \( P_{\text{fail}}(VT_n, q, p), P_{\text{fail}}(SVT_n, q, p) \), respectively. The following theorem summarizes these results.
Theorem 10. The average decoding failure probabilities for the VT and SVT codes are given by

\[
P_{\text{fail}}(VTn, q, p) \approx (1 - P_{\text{run}}(q, p))^n \cdot (1 - P_{\text{alt}}(q, p))^n + (1 - P_{\text{run}}(q, p))^n \cdot nP_{\text{alt}}(q, p) (1 - P_{\text{alt}}(q, p))^{n-1} + nP_{\text{run}}(q, p) (1 - P_{\text{run}}(q, p))^{n-1} \cdot (1 - P_{\text{alt}}(q, p))^n
\]

\[
P_{\text{fail}}(SVTn, q, p) \approx (1 - P_{\text{run}}(q, p))^n \cdot (1 - P_{\text{alt}}(q, p))^n + (1 - P_{\text{run}}(q, p))^n \cdot nP_{\text{alt}}(q, p) (1 - P_{\text{alt}}(q, p))^{n-1}.
\]

Proof: The proof follows from the observation that the VT code is capable of decoding either a single run error or a single alternating error. On the other hand, the shifted VT code is capable of decoding either a single run error or a single alternating error that results from the occurrence of an alternating sequence. We denote by \( P_{\text{run}} \) and \( P_{\text{alt}} \) the error rate of the ML decoder upon two instances of the insertion channel \( \text{Ins}(p) \). Similarly, \( P_{\text{fail}} \) is the average decoding failure probability and lastly \( P_{\text{run}}(q, p), P_{\text{alt}}(q, p) \) is the insertion probability due to runs, occurrence probability due to alternating sequences, respectively. The following theorem summarizes the results of this section.

Theorem 11. For the insertion channel \( \text{Ins}(p) \), it holds that

\[
P_{\text{run}}(q, p) \approx \frac{q + 1}{q(q - 1)} p^2, \quad P_{\text{alt}}(q, p) \approx \frac{2}{q} p^2,
\]

\[
P_{\text{err}}(q, p) \approx \frac{3q - 1}{q(q - 1)} p^2 + O(p^3), \quad P_{\text{fail}}(q, p) \approx e^{-\frac{2}{\sqrt{\pi}} p^2 n}.
\]

Since the proof of this theorem repeats the same ideas as the ones for the deletion case we omit its details here.

The theoretical results of Theorem 11 have also been verified by simulation results over words of length \( n = 500 \), which were used to create two noisy copies with a given fixed insertion probability \( p \in [0.005, 0.05] \). Then, the two copies were decoded by the ML decoder as described in Claim 4. Finally, we calculated and plotted the Levenshtein error rate of the decoded word as well as the average decoding failure probability (referred here as failure rate). Fig. 1 plots the results of the Levenshtein error rate which confirms the probability expression \( P_{\text{err}}(q, p) \) from Theorem 11. Similarly, in Fig. 2 we present separately the error probability \( P_{\text{alt}}(q, p), P_{\text{run}}(q, p) \), along with the corresponding value as calculated in Lemma 5 and Lemma 6, respectively. Lastly, in Fig. 3, we simulated the ML decoder for the VT codes and the SVT codes and calculated the failure rate \( P_{\text{fail}}(q, p) \). The implementation of the VT code was taken from [47], and we modified this implementation for SVT codes.

V. TWO INSERTION CHANNELS

This section continues the two-channel study but for for insertion case. In a similar manner to the deletion case, also here the dominant errors result from increasing the length of a run and error that results from the occurrence of an alternating sequence. We denote by \( P_{\text{err}}(q, p) \) the Levenshtein error rate of the ML decoder, stratified by the coding scheme. These simulations verify Theorem 7 and Theorem 8.

VI. CONCLUSION

The main contribution of this paper is the study of the decoding error probability of the ML decoder for two deletion or insertion channels. While the results in the paper provide a significant contribution in the area of codes for insertions and deletions and sequence reconstruction, there are still several interesting problems which are left open. Some of them are summarized as follows:

1. Study the non-identical channels case. For example two deletion channels with different probabilities \( p_1 \) and \( p_2 \).
2. Study the decoding error probability for more than two channels, both for insertions and deletions.
3. Study channels which introduce insertions, deletions, and substitutions.
4. Design coding schemes as well as complexity-efficient algorithms for the ML decoder in each case.
