IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 7, 2018, accepted September 9, 2018, date of publication October 5, 2018, date of current version October 25, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2873604

Generalized Systematic Comma-Free Code

TIANBO XUE AND FRANCIS C. M. LAU, (Senior Member, IEEE)
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong

Corresponding author: Francis C. M. Lau (francis-cm.lau@polyu.edu.hk)

This work was supported by The Hong Kong Ph.D. Fellowship Scheme (HKPFS) provided by the RGC of Hong Kong.

ABSTRACT In channels that introduce substitution, insertion, and deletion errors, one challenging problem for a code designer to overcome is to avoid false code-synchronization. In other words, the probability of a false codeword occurring should be minimized with appropriate code designs. In this paper, we propose a new class of systematic comma-free code called generalized $F(n, s, t)$ code for channels that are impaired by substitution, insertion, and deletion errors. We first prove that this code is a synchronous code and then we derive the probabilities of false synchronization when a substitution, insertion, or deletion error occurs. We compare the theoretical and simulation results under different channel parameters. We also analyze the factors affecting the false synchronization for each case. The proposed code gives a higher code efficiency and better choice in terms of false synchronization compared with the classical F codes.

INDEX TERMS Comma free code, false codeword, false synchronization, SID channel.

I. INTRODUCTION

The transmitted signals are usually corrupted by noise in communication channels, which causes adjacent channel interference or introduces intersymbol interference, etc. The transmitted symbols are substituted with other symbols in some communication channels. For instance, substituting bit 1 with bit 0 or bit 0 with bit 1 for the binary case. When a fixed-length block code is used over such a channel, no synchronization problem needs to be considered. In other words, the decoder knows the boundaries of each codeword.

In contrast, Some channels not only substitute transmitted symbols, but also remove transmitted symbols and insert new symbols. When such insertion and deletion errors occur together with substitution errors, synchronization becomes a serious issue. It may not be possible for the decoder to decode one block after another because the boundaries of codewords are not easy to detect. Traditionally, deletion and insertion errors occur in optical media and reading magnetic [2], image watermarking [3], [4], satellite communication, mobile communication and data storage systems in addition to substitution errors. Deoxyribonucleic Acid (DNA)-based data storage system is a newly emerging field, in which data is stored as DNA nucleotides [5]–[8]. Such data storage systems can be viewed as digital communication systems. Several DNA-based storage systems have been illustrated in the past decade. For example, Church *et al.* [7] encoded an entire book in DNA sequences using new synthesis and sequencing technology. However, the nucleotides that store data may be substituted, removed or added during the sequencing and synthesis processes of DNA sequences, introducing substitution, deletion and insertion errors, respectively. Synchronization between codewords will get lost when deletion and/or insertion errors occur.

Synchronization is a challenging problem because a relatively small number of edits can cause obvious difference between the transmitted and received sequences in terms of Hamming distance. The maintenance of codeword synchronization is of importance in any communication and data storage systems. Many classes of codes with the capability to detect or correct synchronous errors have been constructed. In [9], Tenengol'ts has developed a block code that can correct one deletion error or two consecutive substitution errors for channels that are impaired by substitutions, insertions and deletions. However, Tenengol'ts has not discussed synchronization and has assumed that the boundaries of each codeword are known. The literature [10] introduce and studies variable-length self-synchronizing codes in details. Self-synchronization can be achieved by using variable-length binary codes [11]. Even though variable-length codes can keep a receiver synchronized, the delay to achieve synchronization relies on the transmitted message and may be very long. In addition, when a codeword is decoded using a wrong code length, all subsequent codewords will be decoded incorrectly until synchronization is regained.

Fixed-length encoding can be used to keep the synchronization delay below a limit. When a fixed-length block code is used, synchronization techniques must be taken to make sure that the receiver and transmitter stay synchronized. Fixed-length codes can be synchronized more easily. An effective approach to maintaining synchronization of codewords is to transmit "commas" between codewords in the presence of insertion or deletion errors [12]. This synchronization technique under various constraints has been studied. Another technique that does not rely on "commas" to maintain synchronization are referred to as "comma-free codes". The comma-free codes were first proposed by Golomb *et al.* [13]. The constructions of such codes have been discussed by Gilbert [12], Golomb *et al.* [13], [14] and Golomb [15], Jiggs [16] and Eastman [17].

Specifically, a comma-free code (CFC) is a set of codewords of length n over an alphabet such that given any two codewords $\mathbf{u} = u_1u_2\cdots u_n$ and $\mathbf{v} = v_1v_2\cdots v_n$ belonging to this set, the n letter concatenation $\mathbf{w} = u_k\cdots u_nv_1\cdots v_{k-1}$ $(k = 2, 3, \ldots, n)$ cannot construct a codeword. Examples of comma-free codes with a codeword length of 5 are {01000, 01100, 01010, 01110, 01011, 01111} and a codeword length of 7 are {0101000, 0101100, 0101110, 0101111, 0100010, 0110010, 0111010, 0111110, 0100011, 0110011, 0111011, 0111111, 0100000, 0110000, 0111000, 0111100, 0111110, 0111111}. From the view-point of mathematics, the upper bound of the maximal number of codewords in a CFC set given a codeword length n and an alphabet size q has been derived in [13] and [16] and is given by

$$W_n(q) \leq \frac{1}{n}\sum_{d|n}\phi(d)q^{n/d}, \tag{1}$$

where the summation is extended over all divisors d of n, and ϕ is the Mobius function defined as

$$\phi(d) = \begin{cases} 1 & \text{if } d = 1, \\ 0 & \text{if } d \text{ has any square factor,} \\ (-1)^r & \text{if } d = p_1p_2\cdots p_r, \text{ where} \\ & p_1, \cdots, p_r \text{ are distinct primes.} \end{cases} \tag{2}$$

Golomb *et al.* [13] first show how to construct the maximal cardinality of CFC for codeword lengths of 3, 5, 7 and 9. They further prove that the construction can be extended to codeword lengths of 11, 13 and 15. In [17], a strategy is given for constructing maximal comma-free codes in terms of number of words for any odd codeword length n. However, the upper bound cannot always be achieved. This type of code allows the receiver to re-establish synchronization after an insertion or deletion error with a delay of at most two blocks. Eastman and Even [18] have constructed a block code, which is able to detect synchronization errors after receiving some finite number of codewords. Comma-free codes have relatively low efficiency because of a strong synchronizing condition. The error correcting properties of these codes are not known. In addition, CFC proposed by Golomb *et al.* leads to difficult decoding unless small length of codes are used.

The aforementioned codes do not provide specific fixed positions for transmitting data. The systematic fixed-length block comma-free codes are investigated in [19]. A systematic comma-free code is a subclass of comma-free codes where fixed locations are used to maintain synchronization in each codeword and remaining locations are used to carry information. An effective method is to fix the first few bits of a codeword to a sequence of bit 1 or bit 0. The fixed sequence is referred to as the primer drive of CFC and it appears at the beginning of each codeword. When messages of binary digits are transmitted as blocks, identifying the boundary between consecutive blocks are of much significance to maintain the receiver in synchronization with the transmitter. If the same primer drive and fixed positions appear when two codewords are concatenated due to errors, the decoder would give a false synchronization and such a sequence is called a false codeword. Two systematic CFCs are given in the following.

A. GILBERT CODE

The primer drive of a Gilbert code consists of bit zeros and is placed as a prefix to each block. In order to ensure the primer drive not to appear in the body of the codeword, the other digits should be constrained. Suppose that the primer drive sequence $\boldsymbol{p}$ consists of s binary digits $p_1, p_2, \cdots, p_s$. Each codeword of length n $(p_1, p_2, \cdots, p_s, x_1, x_2, \cdots, x_{n-s})$ is constructed by carefully choosing $(n-s)$ binary digits $x_1, x_2, \cdots, x_{n-s}$. Gilbert codes ensure that the primer drive sequence $\boldsymbol{p}$ of length s will not appear in $(p_2, \cdots, p_s, x_1, x_2, \cdots, x_{n-s}, p_1, \cdots, p_{s-1})$. Each subsequent sequence of s bits begins with a bit one and the last digit is also a bit one. For example, Gilbert code with parameters $n = 16$ and $s = 4$ encodes the information sequence "10110110" into the codeword "**00001**101**1**101**1**10**1**", where the bold bits represent the fixed bits used to keep synchronization. Obviously, the primer drive sequence of s zeros can never appear in the body of a codeword even if information bits are all zeros. Gilbert codes have a relatively strong condition to maintain synchronization because it is required that no primer drive sequence appears in the body of a codeword and the efficiency of the code is reduced.

B. F CODE

A more efficient classic systematic comma-free code in terms of transmitting information called "F code" is introduced in [19]. "F code" allows a primer drive to exist in the body of a codeword, however, it precludes the possibility that the bits between concatenated codewords form a codeword. Indicate the codeword length by n and the parameter r is the minimum integer greater than or equal to $\sqrt{2(n-1)}$, the parameter s is the minimum integer no less than $\frac{r}{2}$ and $t = r - s$. An F code with codeword length n has $s + t$ fixed positions. 0s are located at the positions $1, 2, \ldots, s$ and 1s are located at the positions $n, n-s, n-2s, \ldots, n-(t-1)s$ while all remaining locations are arbitrary for transmiting information symbols. For instance, the information sequence "010010011" is encoded into an F code with a codeword length $n = 15$ "**000**01001**1**00**1**11**1**", where the bold bits are the fixed bits which are used to maintain synchronization. The class of F codes proposed by Clague are similar to Gilbert codes but have a relatively weaker condition to maintain synchronization, which results in a higher efficiency. Some fixed positions of a Gilbert code become arbitrary bits in an F code of the same length. The difference becomes more obvious as the length of the code increases. Therefore, F codes have fewer fixed places and are more efficient in terms of carrying data comparing with Gilbert codes. In fact, the class of F codes have been demonstrated as the most efficient systematic comma-free code that maintains synchronization using fixed places [19].

A new systematic CFC referred to as "generalized F code" is proposed in this paper and it includes "F code" as a special case. The probabilities of a false synchronization due to substitution/insertion/deletion errors have been derived. This paper begins with a definition of the generalized $F(n, s, t)$ codes in Sect. II. This is followed by the construction of the codes, the proof that it is a systematic comma-free code and the encoding and decoding of the proposed generalized F code. Sect. III shows the error types and channel model. The false synchronization probabilities caused by a single substitution, deletion or insertion error are derived, and performance results are shown in Sect. IV. Concluding remarks are given in Sect. VI.

II. PROPOSED $F(n, s, t)$ CODES AND ITS ENCODING AND DECODING

A. PROPOSED $F(n, s, t)$ CODES

A "generalized $F(n, s, t)$ code" of codeword length n includes s fixed 0s and t fixed 1s, where $st \geq \frac{n-1}{2}$. In addition, the fixed 0s are at positions $1, 2, \ldots, s$ while the fixed 1s are at positions $js + 1$, $(j = 1, 2 \ldots, t)$ satisfying $st \geq \frac{n-1}{2}$. For example, the format of a generalized $F(n = 19, s = 3, t = 3)$ code is $0001xx1xx1xxx1xxxxxxxxx$, where x reprensents an arbitrary information bit and it is either bit 0 or 1. For such a set of fixed-position systematic CFC, we define the code rate as the ratio between the number of arbitrary bits and codeword length, i.e. $\frac{n-s-t}{n}$. The number of fixed positions $s+t$ is minimized when s is equal to t under the condition that $2st \geq n - 1$, i.e, $s = t = \sqrt{\frac{n-1}{2}}$. Therefore, the code rate of the generalized $F(n, s, t)$ code is limited by $\frac{n-2\cdot\sqrt{\frac{n-1}{2}}}{n}$, which approaches $1 - \sqrt{\frac{2}{n}}$ when the codeword length n becomes very large.

Theorem 1: The generalized $F(n, s, t)$ is a comma-free code.

Proof: The positions of 0s are $a_i = i$ $(i = 1, 2, \ldots, s)$ and the positions of 1s are $b_j = js + 1$ $(j = 1, 2, \ldots, t)$ in the proposed generalized $F(n, s, t)$ code. Note that $st \geq 0.5(n - 1)$. Thus, $\{\pm(a_i - b_j)\}$ (mod n) is given by the following.

$$\begin{array}{ll}
b_1 - a_1 = s & a_1 - b_1 = n - s \\
b_1 - a_2 = s - 1 & a_2 - b_1 = n - s + 1 \\
\vdots & \vdots \\
b_1 - a_s = 1 & a_s - b_1 = n - 1 \\
b_2 - a_1 = 2s & a_1 - b_2 = n - 2s \\
\vdots & \vdots \\
b_2 - a_s = s + 1 & a_s - b_2 = n - s - 1 \\
b_t - a_1 = ts & a_1 - b_t = n - ts \\
\vdots & \vdots \\
b_t - a_s = (t-1)s + 1 & a_s - b_t = n - (t-1)s - 1
\end{array}$$

The reason $st + 1 \geq n - st$ is that $st \geq 0.5(n - 1)$ is required for a generalized $F(n, s, t)$ code. Thus, all values between 1 and $(n-1)$ are covered at least once by the residues. Based on the lemma of [19], which states that a fixed-place code with 0s and 1s fixed in the positions a_i $(i = 1, \ldots, s)$ and b_j $(j = 1, \ldots, t)$, respectively, will be synchronous if and only if the set $\{\pm(a_i - b_j)\}$ contains a complete system of nonzero residues (mod n), we demonstrate that the generalized $F(n, s, t)$ code is indeed a systematic comma-free code. □

B. ENCODING AND DECODING OF SYSTEMATIC CFC

The information bit sequence is first divided into blocks, each of which contains k information bits. Then the primer drive and fixed bits are inserted in between the k information bits to construct a systematic CFC codeword. The encoding procedure is repeated for every block of k information bits.

The decoding procedure at the receiver is actually the reverse of the encoding operation. It is very likely that the length of the received sequence is different from that of the transmitted one due to potential insertion/deletion errors. The boundaries between consecutive blocks are not easy to detect any longer and thus, it is impossible to simply pass the decoder one codeword after another. Our aim is to make the decoding procedures as simple as possible. The strategy we chose to solve the synchronization problem of the received sequence is by introducing a decoding window of fixed-length size n, which is simply a codeword length.

The first and the last bit of the received data sequence are known by the decoder but there is no direct information about the boundaries of each codeword. The decoder is able to maintain synchronization at the codeword boundaries with the use of the primer drive sequence and some fixed positions. A codeword is immediately decoded whenever the structure of a systematic CFC codeword is detected by the decoding window. The decoding window then moves n positions to the right and tries to detect the next codeword. It is possible that the n bits in the decoding window cannot construct a CFC codeword due to substitution, deletion and insertion errors. Under this circumstance, the decoding window moves one position to the right at a time until another CFC codeword is detected. The procedures repeat until the decoding window reach the last bit.

By definition, A sequence of bits from two consecutive CFC codes cannot form a CFC. However, a false codeword may appear when (substitution/insertion/deletion) errors occur in CFCs. Supposing one error (insertion, deletion or substitution error) occurs in the ith codeword and no errors occur in the subsequent two codewords, we claim that the receiver can always resynchronize at the $(i+2)$th codeword or maybe at the $(i+1)$th codeword. We will use two examples to illustrate these.

One example to illustrate the latter case is shown in Fig. 1. Suppose that an insertion error occurs in the second codeword (i.e. Codeword 2). When the decoding window is at the first block, the bits contained in the decoding window is the first systematic CFC code and hence this codeword is immediately decoded. The decoding window then moves n positions, however, these n bits in the decoding window cannot form a systematic CFC and hence it moves one position to the right until the decoding window arrives at the third block, at which point the third codeword is decoded. The decoding window moves n positions again after that and this process is continued. The receiver resynchronizes at the third codeword under this condition.

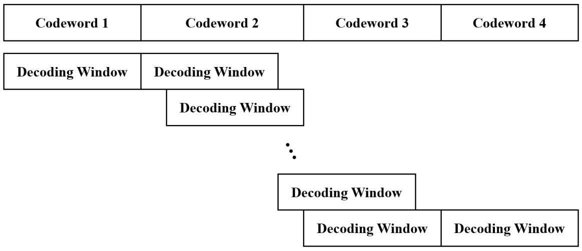


FIGURE 1. Systematic CFC decoding with no false codewords.

For the former case, a CFC codeword is not able to be detected or decoded whenever an insertion error occurs. The tail of the corrupted codeword together with the head of the subsequent codeword may construct a false codeword. The decoding process with the false codeword formed by the second corrupted codeword and the third correct codeword is illustrated in Fig. 2. The tail of the second corrupted codeword and the head of the third correct codeword together form a false codeword caused by an insertion error in the second codeword. The decoding window thinks it is a valid codeword and thus it moves n positions to the right. The content of the decoding window becomes an overlap of two CFCs, which cannot be a codeword. After this, the decoding window will move one position until it decodes the fourth codeword. In this case, synchronization is re-established at the fourth codeword. Even though the third codeword is correct, we cannot decode this codeword due to the existence of a false codeword.

The presence of a false codeword excludes the decoding of the subsequent correct codeword (the 3rd codeword in Fig. 2).

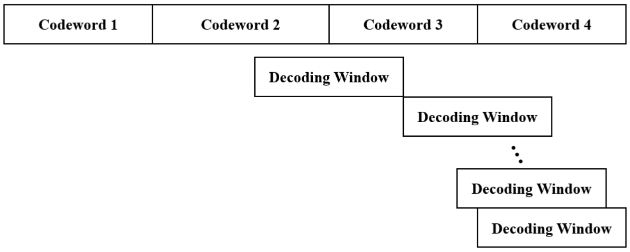


FIGURE 2. Systematic CFC decoding with the existence of false codewords.

Therefore, the false codeword probability when substitution, deletion and insertion errors are introduced is a performance metric of a systematic comma-free code.

III. ERROR TYPES AND CHANNEL MODEL

We define the following synchronization errors and assume a sequence proceeds from left to right.

- A deletion is defined as no detection of a transmitted bit at the receiver. All the following bits after the specific lost bit will shift one position to the left in the sequence.
- An insertion is defined as the detection of an untransmitted bit at the receiver. All the following bits shift one position to the right in the sequence. Random bit might be inserted before each bit of the transmission sequence.

Error detection system requires two components: a precise definition of the communication channel with error characteristics and an analysis of properties of codes that permit error detection or correction for channels in the given model. We will discuss these two aspects in the following.

We consider a channel that suffers from independent substitution, deletion and insertion errors. Both the transmitter and receiver have no information about the specific locations where errors occur. In this paper, only bianary symbols are considered and such a channel is called a binary substitution/insertion/deletion (BSID) channel [20]. The BSID channel model is shown in Fig. 3 [21].

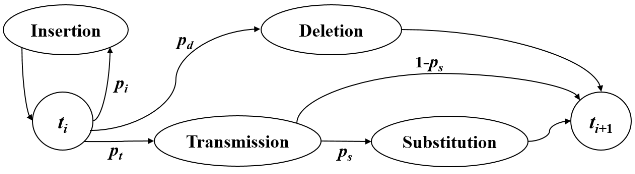


FIGURE 3. BSID channel model.

We consider the BSID channel described by three independent parameters: insertion with probability p_i, where a random bit is inserted; deletion with probability p_d, where the current bit is discarded; and substitution with probability p_s, where the transmitted bit is substituted. At time instant i, the transmission of a bit t_i through the BSID channel can lead to one of the four possible events. One or more bits are inserted before the current bit t_i; The current bit t_i is deleted and subsequent bit t_{i+1} is ready to be transmitted; The current bit t_i is transmitted with probability $p_t = 1 - p_i - p_d$. Further, the transmitted bit get substituted with a probability p_s. For this model, any insertion errors occurring in the boundary between two consecutive codewords are considered as errors of the second codeword.

IV. FALSE CODEWORD PROBABILITIES

A. BLOCK ERROR RATE

We assume that the transmitted sequence is encoded into successive fixed length blocks. Let $\mathbf{t} = (x_1, x_2, \cdots, x_n)$, $x_i \in \{0, 1\}$ for $i = 1, 2, \cdots, n$ and $\mathbf{y} = (x'_1, x'_2, \cdots, x'_r)$, $x'_i \in \{0, 1\}$ for $i = 1, 2, \cdots, r$ be the transmitted and received sequence, respectively. The length of the transmitted sequence n is a constant parameter while the length of the received sequence r is a random variable depending on the deletion and insertion processes. The length of the received sequence is either lengthened or shortened depending on the predominant insertion or deletion errors. Equal number of deletion and insertion errors result in a received sequence of equal length with the transmitted sequence. Such a case is equivalent to having only substitution errors and thus the boundary of the codeword is not affected. However, we cannot always assume the same number of insertion and deletion errors in a codeword. Therefore, we must find a codeword that can be used to synchronize the received sequence whose length is different from the transmitted one.

The BSID channel model does not restrict the maximum number of consecutive insertions. The "overall" probability of insertion errors for a current transmitted bit is $p'_i = p_i + p_i^2 + p_i^3 + \ldots = \frac{p_i}{1-p_i}$. $p'_i \approx p_i$ is valid for small value of the insertion probability p_i. The subsequent bit will be considered when a deletion error or a transmission has occurred. The probability of the current transmitted bit being substituted is $p'_s = (1 - p_i - p_d) \times p_s$. Similar to p'_i, $p'_s \approx p_s$ when the insertion probability p_i and the deletion probability p_d are small values. The operations repeat until the last bit of the sequence has been transmitted and the obtained sequence is the final received sequence. When a codeword of length n bits are transmitted though a BSID channel, we can obtain the block error probability on individual codewords. Based on the BSID channel, we consider the probability $Pr(N_d, N_s)$ with N_d deletion errors and N_s substitution errors and the probability $Pr(N_i)$ with N_i insertion errors individually. The probability of the received codeword with a specific N_d deletion errors and N_s substitution errors through the channel is given by

$$Pr(N_d, N_s) = p_d^{N_d} \cdot (p_t \cdot p_s)^{N_s} \cdot (p_t \cdot (1 - p_s))^{n - N_d - N_s}, \quad where,\ p_t = 1 - p_d - p_i \tag{3}$$

Moreover, the probability of the received codeword with a specific N_i insertion errors though the channel is

$$Pr(N_i) = p_i^{N_i}. \tag{4}$$

All errors are assumed to be independent and thus the probability of the codeword with the specific error pattern containing N_i insertion, N_d deletion and N_s substitution errors is given by the multiplication of these two probabilities, i.e.,

$$\begin{aligned} &Pr(N_i, N_d, N_s) \\ &\quad = Pr(N_d, N_s) \cdot Pr(N_i) \\ &\quad = p_d^{N_d} \cdot (p_t \cdot p_s)^{N_s} \cdot (p_t \cdot (1 - p_s))^{n - N_d - N_s} \cdot p_i^{N_i}. \end{aligned} \tag{5}$$

Since we assume that any insertion errors occurring in the boundary between two codewords is considered as errors of the second codeword, the number of combinations of inserting N_i bits is $\binom{n+N_i-1}{N_i}$. In addition, we have $\binom{n}{N_d}$ combinations of deleting N_d bits from n bits and $\binom{n-N_d}{N_s}$ combinations of substituting N_s bits from $n - N_d$ bits. Therefore, the total number of combinations with the error pattern N_i, N_d and N_s errors is given by

$$C(N_i, N_d, N_s) = \binom{n + N_i - 1}{N_i} \cdot \binom{n}{N_d} \cdot \binom{n - N_d}{N_s}. \tag{6}$$

Finally, the probability of any error pattern containing N_i insertion, N_d deletion and N_s substitution errors equals

$$P(N_i, N_d, N_s) = C(N_i, N_d, N_s) \cdot Pr(N_i, N_d, N_s). \tag{7}$$

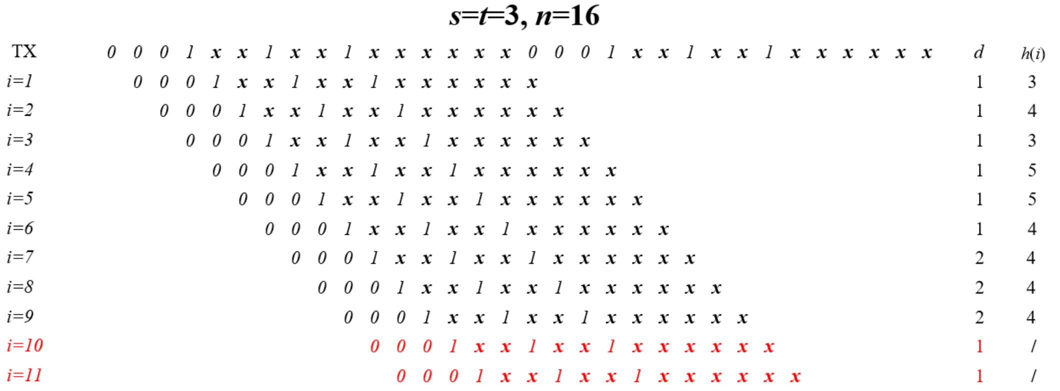
B. FALSE CODEWORD PROBABILITIES

In this section, the false synchronization probabilities will be derived when (i) a single substitution occurs in a codeword, (ii) a single substitution occurs in each of two consecutive codewords, (iii) a single deletion error occurs in a codeword, (iv) a single insertion error occurs in a codeword, under different channel parameters. The following symbols are defined.

- x: An arbitrary information bit which can be either 0 or 1
- $\mathbf{g}$: Generator of the proposed generalized $F(n, s, t)$ code. It is a vector of length n, which consists of s fixed (non-arbitrary) 0s, t fixed (non-arbitrary) 1s and $n - s - t$ arbitrary information bits x.
- $\mathbf{tt}$: Concatenation of two transmitted $F(n, s, t)$ codewords with a total vector length of $2n$.
- $\mathbf{r'r}$: Received vector when $\mathbf{tt}$ is transmitted. The vector length depends on the errors occurring.
- $\mathbf{g}(i)$: A decoding window which is the delay-i version of $\mathbf{g}$, i.e., $\mathbf{g}$ with i empty slots in front.
- $d(\mathbf{tt}, \mathbf{g}(i))$: Hamming distance only between non-arbitrary bits of $\mathbf{tt}$ and $\mathbf{g}(i)$ inside the decoding window. If $d(\mathbf{tt}, \mathbf{g}(i)) = 0$, the content in the decoding window is a valid codeword.
- $h(i)$: Number of positions where the element of $\mathbf{g}(i)$ is 0 or 1 AND the element of $\mathbf{tt}$ is x.

1) SINGLE SUBSTITUTION ERROR

Synchronization will be maintained when a random information bit is substituted. However, synchronization will get lost when one of the fixed bits is substituted and a false codeword may appear under this case. Therefore, we only consider those i for which $d(\mathbf{tt}, \mathbf{g}(i)) = 1$ when evaluating the false synchronization probability caused by a single substitution error. Namely, the transmitted vector $\mathbf{tt}$ and the decoding



s=t=3, n=16

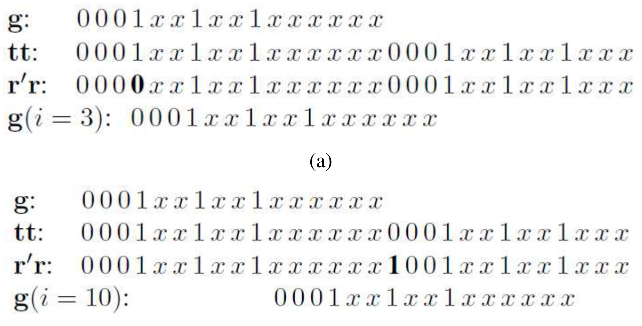
		d	h(i)
TX	0 0 0 1 x x 1 x x 1 x x x x x x 0 0 0 1 x x 1 x x 1 x x x x x x		
i=1	0 0 0 1 x x 1 x x 1 x x x x x x	1	3
i=2	0 0 0 1 x x 1 x x 1 x x x x x x	1	4
i=3	0 0 0 1 x x 1 x x 1 x x x x x x	1	3
i=4	0 0 0 1 x x 1 x x 1 x x x x x x	1	5
i=5	0 0 0 1 x x 1 x x 1 x x x x x x	1	5
i=6	0 0 0 1 x x 1 x x 1 x x x x x x	1	4
i=7	0 0 0 1 x x 1 x x 1 x x x x x x	2	4
i=8	0 0 0 1 x x 1 x x 1 x x x x x x	2	4
i=9	0 0 0 1 x x 1 x x 1 x x x x x x	2	4
i=10	0 0 0 1 x x 1 x x 1 x x x x x x	1	/
i=11	0 0 0 1 x x 1 x x 1 x x x x x x	1	/

FIGURE 4. Complete shifting process of a decoding window.

window vector $\mathbf{g}(i)$ only differ in one fixed position. When the bit at this position is substituted, the false codeword will appear on condition that the fixed bits in $\mathbf{g}(i)$ and the specific aritrary information bits x in $\mathbf{tt}$ are well matched. The probability of the exact matching is given by $(\frac{1}{2})^{h(i)}$.

The generalized systematic $F(n = 16, s = 3, t = 3)$ code is used for explanation. The complete shifting process of the decoding window for the generalized systematic $F(n = 16, s = 3, t = 3)$ is shown in Fig. 4. When $i = 3$, the decoding window moves three positions to the right (or there are three empty slots inserted before the decoding window).

Referring to Fig. 5(a), the first row shows the generator; the second row shows two consecutive transmitted codewords; the third row represents the received vector through the channel, where a substitution error occurs at the first fixed bit 1 in the transmitted codeword; the last row represents the shifted decoding window with $i = 3$. We notice $h(3) = 3$ by comparing the second and fourth rows. The probability of a false codeword appearing is $(\frac{1}{2})^3$ (probability when the 5th, 6th, and 13rd transmitted bits in $\mathbf{tt}$ are 0, 0 and 1 coincidentally) under the condition that the first fixed bit 1 is substituted.



$\mathbf{g}$: $0001xx1xx1xxxxxx$

$\mathbf{tt}$: $0001xx1xx1xxxxxx0001xx1xx1xxx$

$\mathbf{r'r}$: $000\mathbf{0}xx1xx1xxxxxx0001xx1xx1xxx$

$\mathbf{g}(i = 3)$: $0001xx1xx1xxxxxx$

(a)

$\mathbf{g}$: $0001xx1xx1xxxxxx$

$\mathbf{tt}$: $0001xx1xx1xxxxxx0001xx1xx1xxx$

$\mathbf{r'r}$: $0001xx1xx1xxxxxx\mathbf{1}001xx1xx1xxx$

$\mathbf{g}(i = 10)$: $0001xx1xx1xxxxxx$

(b)

FIGURE 5. Decoding window for the generalized $F(n = 16, s = 3, t = 3)$ code. (a) Decoding window with $i = 3$. (b) Decoding window with $i = 10$.

However, a point needs to be emphasized. Suppose $d(\mathbf{tt}, \mathbf{g}(i)) = 1$ for a certain i and the position where $\mathbf{tt}$ and $\mathbf{g}(i)$ differs is beyond n. For example, we find that the position where the vector $\mathbf{tt}$ and $\mathbf{g}(10)$ differs is at the first bit of the second codeword, as in Fig. 5(b). This means the first codeword is decoded correctly and the only substitution error occurs in the next codeword. Under this case, the decoding window will shift n positions to the right after decoding the first correct codeword. Since the decoding window never contains the tail of the first codeword and the head of the next codeword together, false codeword described in this scenario can never occur.

To calculate the total probability of false synchronization caused by a single substitution error at a fixed position, we must exclude the case where the first codeword is correct and substitution error occurs in the second codeword. Therefore, only the scenario, i.e., one substitution error occurs in the first codeword and the second codeword is correct, needs to be considered. For this case, we require the last bit 1 of the decoding window $\mathbf{g}(i)$ not to go beyond the last bit of the first codeword, i.e., not to exceed the second codeword. Otherwise, the first codeword has been correctly synchronized and detected or $d(\mathbf{tt}, \mathbf{g}(i)) > 1$. Thus, we require $i+st+1 \le n$, i.e., $1 \le i \le n-st-1$. Furthermore, we show that

$$h(i) = \begin{cases} i+t-1 & 1 \le i \le s-1 \\ s+\dfrac{i}{s}-1 & i = s, 2s, \cdots, ts \\ s+t-1 & \text{otherwise.} \end{cases} \tag{8}$$

Combining all the above, we can readily show the probability of false synchronization caused by a single substitution error at a fixed position in the generalized $F(n, s, t)$ code

$$P_s = P'_s \times \sum_{d(\mathbf{tt},\mathbf{g}(i))=1 \ and \ i \le n-st-1} \left(\frac{1}{2}\right)^{h(i)}, \tag{9}$$

where $P'_s = (p_t \cdot p_s) \times (p_t \cdot (1-p_s))^{s+t-1}$. In (9), P'_s represents the probability that exact one fixed position suffers from a substitution error. The expression of P'_s is obtained under the condition $N_s = 1, N_d = 0$ and $N_i = 0$ in (5).

2) ONE SINGLE SUBSTITUTION ERROR IN EACH OF TWO CONSECUTIVE CODEWORDS

In order to calculate the probability of false synchronization caused by more than a single substitution error, we have to find the general expression of $h(i)$. Thus, we define two more symbols.

- $N_0^0(i)$: Number of positions in which both elements in $\mathbf{g}(i)$ and $\mathbf{tt}$ are equal to 0
- $N_1^1(i)$: Number of positions in which both elements in $\mathbf{g}(i)$ and $\mathbf{tt}$ are equal to 1.

The number of fixed positions in $\mathbf{g}(i)$ is $s+t$ and this value also equals $d(\mathbf{tt}, \mathbf{g}(i)) + N_0^0(i) + N_1^1(i) + h(i)$. In other words, $s+t = d(\mathbf{tt}, \mathbf{g}(i)) + N_0^0(i) + N_1^1(i) + h(i)$ and thus $h(i)$ is given by

$$\begin{aligned} h(i) &= s + t - d(\mathbf{tt}, \mathbf{g}(i)) - N_0^0(i) - N_1^1(i) \\ &= (s - N_0^0(i)) + (t - N_1^1(i)) - d(\mathbf{tt}, \mathbf{g}(i)). \end{aligned} \tag{10}$$

The meaning behind $h(i)$ is that there are $h(i)$ occasions where the sequence in the decoding window is not a CFC when the decoding window moves i positions to the right. Previously, we only investigated the scenario when $d(\mathbf{tt}, \mathbf{g}(i)) = 1$ on condition that the subsequent codeword is correct. However, for the scenario where $d(\mathbf{tt}, \mathbf{g}(i)) = 2$, we can no longer assume the subsequent codeword is correct. Each of the two consecutive codewords must contain one substitution error at a fixed position to ensure that the concatenation of the two codewords form a false codeword. Therefore, the scope we consider is enlarged, i.e., the position of the first bit 0 of the decoding window $\mathbf{g}(i)$, i.e., position 1, should never exceed the position of the last bit 1 of the first codeword after shifting i positions of the decoding window to the right. Hence we have $1+i \leq st+1$, i.e., $i \leq st$. Otherwise, the first codeword has been correctly synchronized while the second one is corrupted and thus the false codeword can never arise up.

The values for $d(\mathbf{tt}, \mathbf{g}(i))$, $N_0^0(i)$ and $N_1^1(i)$ and $h(i)$ for the general case have been derived and are listed below:

$$d(\mathbf{tt}, \mathbf{g}(i)) = \begin{cases} 1 & 1 \leq i \leq n - st - 1 \\ 2 & n - st \leq i \leq st; \end{cases} \tag{11}$$

$$N_0^0(i) = \begin{cases} s - i & 1 \leq i \leq s - 1 \\ 0 & s \leq i \leq st; \end{cases} \tag{12}$$

$$N_1^1(i) = \begin{cases} t - \dfrac{i}{s} & i = s, 2s, \cdots, (t-1)s \\ 0 & \text{otherwise}; \end{cases} \tag{13}$$

Therefore, $h(i)$ in (10) can be readily shown equal to

$$h(i) = \begin{cases} i + t - d(\mathbf{tt}, \mathbf{g}(i)) & 1 \leq i \leq s - 1 \\ s + \dfrac{i}{s} - d(\mathbf{tt}, \mathbf{g}(i)) & i = s, 2s, \cdots, (t-1)s \\ s + t - d(\mathbf{tt}, \mathbf{g}(i)) & \text{otherwise}. \end{cases} \tag{14}$$

3) SINGLE DELETION ERROR

We continue to investigate the false synchronization probability due to synchronous errors. We first investigate the false synchronization probability P_d caused by a single deletion error. $\mathbf{r}'_z\mathbf{r}$ is defined as the received vector of length $2n-1$ in which the zth bit $(1 \leq z \leq n)$ of $\mathbf{tt}$ is deleted. In fact, $\mathbf{r}'_z\mathbf{r}$ is the cascading of a codeword with a single deletion error and a correct codeword. Obviously, the distance $d(\mathbf{g}(n-1), \mathbf{r}'_z\mathbf{r}) = 0$ because the sequence in the decoding window is exactly the second correct codeword. $d(\mathbf{g}(i), \mathbf{r}'_z\mathbf{r})$ is usually nonzero for all $z = 1, 2, \cdots, n$ and $i = 1, 2, \cdots, n-2$. In this case, ccodeword cannot be formed caused by a single deletion error. However, a false codeword is detected with probability $(\frac{1}{2})^{h(i)}$ if $d(\mathbf{g}(i), \mathbf{r}'_z\mathbf{r}) = 0$ for certain $i \leq n-2$ and z. Same as the substitution error case, the last bit 1 of the decoding window cannot arrive at the position of the first bit 0 of the second codeword,i.e., $i + st + 1 \leq n - 1$ or $i \leq n - st - 2$.

The problem is further divided into different cases and the corresponding $h(i)$ is derived. The complete shifting process of the decoding window for systematic generalized $F(n, s, t)$ code is shown in Fig. 6.

Case 1 (A Single Bit Among the s Starting 0s Is Deleted): The received vector begins with $(s-1)$ 0s followed by a bit 1 while the decoding window begins with s consecutive 0s. s consecutive 0s cannot be found even though all information bits are 0s. Therefore, a false codeword cannot be formed for this scenario.

Case 2 (The First Bit One After the Sequence of s 0s Is Deleted): A false codeword is formed when $0 \leq i \leq s-1$. We further show that

$$h(i) = \begin{cases} t + i & 0 \leq i \leq s - 2 \\ s & i = s - 1. \end{cases} \tag{15}$$

Case 3 (A Single Bit Between jth Bit 1 and (j + 1) Bit 1 (1 ≤ j ≤ t − 1) Is Deleted): A false codeword can only appear when $i = 0$. We further show that $h(i) = t - j$.

Case 4 (The jth Bit 1 (2 ≤ j ≤ t − 1) Is Deleted): It is obvious that a false codeword may appear when $i = 0$ and $h(i) = t - j + 1$. Besides, a false codeword may also appear when (a) the first bit 0 of the decoding window exceeds $(j-1)$ bit 1, i.e., $1 + i \geq (j-1)s + 1 + 1$ or $i \geq (j-1)s + 1$; AND (b) the first bit 1 of the decoding window does not exceed $(j+1)$th bit 1, i.e., $i + s + 1 \leq (j+1)s + 1 - 1$ and $i \leq js - 1$. Furthermore, it is readily shown that

$$h(i) = \begin{cases} s + t & (j-1)s + 1 \leq i \leq js - 2 \\ s + j & i = js - 1. \end{cases} \tag{16}$$

Case 5 (The Last Bit 1 Is Deleted): Obviously, a false codeword appear when $i = 0$ and $h(i) = 1$. Besides, a false codeword may also appear when the first bit 0 of the decoding window exceeds the fixed bit 1 preceding the deleted bit 1, i.e., $1 + i \geq (t-1)s + 1 + 1$ or $i \geq (t-1)s + 1$. Furthermore, it can be shown that

$$h(i) = \begin{cases} 1 & i = 0 \\ s + t & (t-1)s + 1 \leq i \leq n - st - 2. \end{cases} \tag{17}$$

Case 6 (A Single Bit After the Last Bit 1 Is Deleted): A false codeword may appear only on condition that $i = 0$ and $h(i) = 0$.

```
                                   s=t=3, n=19
TX        0 0 0 1 x x 1 x x 1 x x x x x x x x x 0 0 0 1 x x 1 x x 1 x x x x x x x x x   h(i)
Case 1      0 0 1 x x 1 x x 1 x x x x x x x x x 0 0 0 1 x x 1 x x 1 x x x x x x x x x   /
Case 2      0 0 0 x x 1 x x 1 x x x x x x x x x 0 0 0 1 x x 1 x x 1 x x x x x x x x x
i=0         0 0 0 1 x x 1 x x 1 x x x x x x x x x                                       3
i=1           0 0 0 1 x x 1 x x 1 x x x x x x x x x                                     4
i=2             0 0 0 1 x x 1 x x 1 x x x x x x x x x                                   3
Case 3      0 0 0 1 x 1 x x 1 x x x x x x x x x 0 0 0 1 x x 1 x x 1 x x x x x x x x x
i=0         0 0 0 1 x x 1 x x 1 x x x x x x x x x                                       2
            0 0 0 1 x x 1 x 1 x x x x x x x x x 0 0 0 1 x x 1 x x 1 x x x x x x x x x
i=0         0 0 0 1 x x 1 x x 1 x x x x x x x x x                                       1
Case 4      0 0 0 1 x x x x 1 x x x x x x x x x 0 0 0 1 x x 1 x x 1 x x x x x x x x x
i=0         0 0 0 1 x x 1 x x 1 x x x x x x x x x                                       2
i=4                 0 0 0 1 x x 1 x x 1 x x x x x x x x x                               6
i=5                   0 0 0 1 x x 1 x x 1 x x x x x x x x x                             5
Case 5      0 0 0 1 x x 1 x x x x x x x x x x x 0 0 0 1 x x 1 x x 1 x x x x x x x x x
i=0         0 0 0 1 x x 1 x x 1 x x x x x x x x x                                       1
i=7                         0 0 0 1 x x 1 x x 1 x x x x x x x x x                       6
i=8                           0 0 0 1 x x 1 x x 1 x x x x x x x x x                     6
Case 6      0 0 0 1 x x 1 x x 1 x x x x x x x x 0 0 0 1 x x 1 x x 1 x x x x x x x x x
i=0         0 0 0 1 x x 1 x x 1 x x x x x x x x x                                       0
```

FIGURE 6. Complete shifting process of a decoding window.

We can readily show that the total probability of false sychronization caused by a deletion error is

$$P_d = P'_d \times \sum_{d(\mathbf{g}(i),\mathbf{r}'_z\mathbf{r})=0} \left(\frac{1}{2}\right)^{h(i)}$$

$$= P'_d \times \Bigg\{ \Big[\sum_{i=0}^{s-2} (\frac{1}{2})^{t+i} + (\frac{1}{2})^s \Big] + (s-1) \cdot \sum_{j=1}^{t-1} (\frac{1}{2})^{t-j}$$

$$+ \sum_{j=2}^{t-1} (\frac{1}{2})^{t-j+1} + (\frac{1}{2}) + (n-ts-1) \Bigg\}$$

$$where,\ P'_d = p_d \cdot ((1-p_i-p_d)\cdot(1-p_s))^{n-1} \qquad (18)$$

In (18), $P'_d = p_d \cdot ((1-p_i-p_d)\cdot(1-p_s))^{n-1}$ represents the probability that exact a single bit suffers from a deletion error. The expression of P'_d is obtained under the condition $N_d = 1, N_s = 0$ and $N_i = 0$ in (5).

4) SINGLE INSERTION ERROR

We finally investigate the false synchronization probability P_i caused by a single insertion error. $\mathbf{r}'_q\mathbf{r}$ is referred as the received vector with a random bit inserted before the qth bit of **tt**, $1 \le q \le n$. The length of the received vector is $2n+1$. In fact, $\mathbf{r}'_q\mathbf{r}$ is the cascading of a codeword with a single insertion error and a correct codeword. Obviously, the distance $d(\mathbf{g}(i), \mathbf{r}'_q\mathbf{r}) = 0$ when $i = n+1$ due to the fact that the sequence in the decoding window is the second correct codeword. $d(\mathbf{g}(i), \mathbf{r}'_q\mathbf{r})$ is usually nonzero for all $q = 1, 2, \cdots, n$ and all $i = 1, 2, \cdots, n-2$. In this case, a false codeword can never be formed caused by a single insertion error. However, a false codeword will be detected with probability $(\frac{1}{2})^{h(i)}$ if $d(\mathbf{g}(i), \mathbf{r}'_q\mathbf{r}) = 0$ for certain $i \le n+1$ and q. Same as the substitution and deletion scenarios, the last bit 1 of the decoding window cannot go beyond the position of the first bit 0 of the subsequent codeword, i.e., $i + st + 1 \le n + 1$ or $i \le n - st$.

Similar to dealing with a single deletion error, the problem is divided into several scenarios to derive the corresponding $h(i)$. The complete shifting process of the decoding window for generalized $F(n, s, t)$ code is shown in Fig. 7.

Case 1: Any single bit 0/1 is inserted before any bit in the s starting zeros

Case 2: Any single bit 0/1 is inserted before the first bit 1 after the sequence of s zeros

Case 3: Any single bit 0/1 is inserted before any bits after the last bit 1

Case 4: A single bit 1 is inserted between jth bit 1 and $(j+1)$th bit 1 including the $(j+1)$th bit 1, $1 \le j \le t$

Case 5: A single bit 0 is inserted between jth bit 1 and $(j+1)$th bit 1 including the $(j+1)$th bit 1, $1 \le j \le t$

Based on the complete shifting process of a decoding window, it is impossible to form a false codeword under the first four cases.[1] A false codeword can appear with two conditions under Case 5, i.e., a bit 0 is inserted between jth bit 1 and $(j+1)$th bit 1 including the $(j+1)$ bit 1, $1 \le j \le t$. The first condition is to make sure the sequence in the decoding window is not a codeword when $i = 0$ because the definition of false codeword is the concatenation of two codewords. Only under the first condition will the decoding window shift one position to the right. The second condition is that the first bit 0 of the decoding window must arrive

[1] One point needs to be noticed for Case 1 where any single bit is inserted before the first bit. Even though $d(\mathbf{g}(1), \mathbf{r}'_q\mathbf{r}) = 0$ under this case, the sequence in the decoding window is the first codeword. False codeword is actually the overlap of two consecutive codewords. Based on the definition of the false codeword, this scenario cannot contribute to the false codeword probability caused by a single insertion error.

```
                          s=4, t=2, n=17
TX          0 0 0 0 1 x x x 1 x x x x x x x x 0 0 0 0 1 x x x 1 x x x x x x x x
Case 1    0 0 0 0 0 1 x x x 1 x x x x x x x x 0 0 0 0 1 x x x 1 x x x x x x x x
          0 0 0 0 1 x x x 1 x x x x x x x x
            0 0 0 0 1 x x x 1 x x x x x x x x
                                                0 0 0 0 1 x x x 1 x x x x x x x x
          0 0 0 1 0 1 x x x 1 x x x x x x x x 0 0 0 0 1 x x x 1 x x x x x x x x
          0 0 0 0 1 x x x 1 x x x x x x x x
                              0 0 0 0 1 x x x 1 x x x x x x x x
Case 2    0 0 0 0 1 1 x x x 1 x x x x x x x x 0 0 0 0 1 x x x 1 x x x x x x x x
          0 0 0 0 1 x x x 1 x x x x x x x x
                              0 0 0 0 1 x x x 1 x x x x x x x x
Case 3    0 0 0 0 1 x x x 1 x x x x x x x x x 0 0 0 0 1 x x x 1 x x x x x x x x
          0 0 0 0 1 x x x 1 x x x x x x x x
                                              0 0 0 0 1 x x x 1 x x x x x x x x
                                                0 0 0 0 1 x x x 1 x x x x x x x x
Case 4    0 0 0 0 1 1 x x x 1 x x x x x x x x 0 0 0 0 1 x x x 1 x x x x x x x x
          0 0 0 0 1 x x x 1 x x x x x x x x
                              0 0 0 0 1 x x x 1 x x x x x x x x
Case 5    0 0 0 0 1 0 x x x 1 x x x x x x x x 0 0 0 0 1 x x x 1 x x x x x x x x
  i=0     0 0 0 0 1 x x x 1 x x x x x x x x
  i=5               0 0 0 0 1 x x x 1 x x x x x x x x
```

FIGURE 7. Complete shifting process of a decoding window.

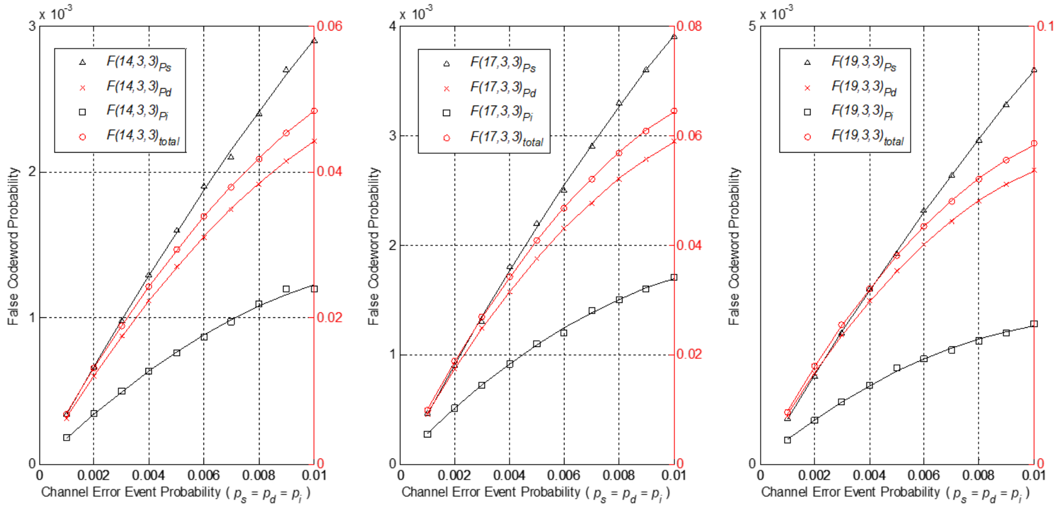


FIGURE 8. Probabilities of false synchronization of generalized $F(n, s, t)$ codes of different lengths with parameters $N = 14, 17, 19$, $s = 3$ and $t = 3$. $p_s = p_d = p_i$ increases from 0.001 to 0.01. Theoretical results are plotted using the solid lines and simulated results are represented by symbols.

at the next bit of the jth bit 1 of the received codewords, i.e., $1 + i = js + 1 + 1$ or $i = js + 1$. Furthermore, we can show that $h(i) = s + j - 1$. Most of the cases do not contribute to false codeword probability. We can readily show that the total probability of false synchronization cased by a single insertion error is

$$P_i = P_i' \times \sum_{j=1}^{t-1} s \left(\frac{1}{2}\right)^{s+j-1},$$
$$where,\ P_i' = \frac{1}{2}((1 - p_i - p_d)(1 - p_s))^n \cdot p_i \quad (19)$$

In (19), P_i' is the probability that exact a single bit 0 is inserted. The expression of P_i' is obtained under the condition $N_i = 1, N_d = 0$ and $N_s = 0$ in (5).

V. RESULTS

The theoretical results of false synchronization for a generalized $F(n, s, t)$ code caused by a single substitution error, a single deletion or a single insertion error have been derived in (9), (18) and (19). In order to verify the accuracy, we perform the simulation as follows: a random sequence of

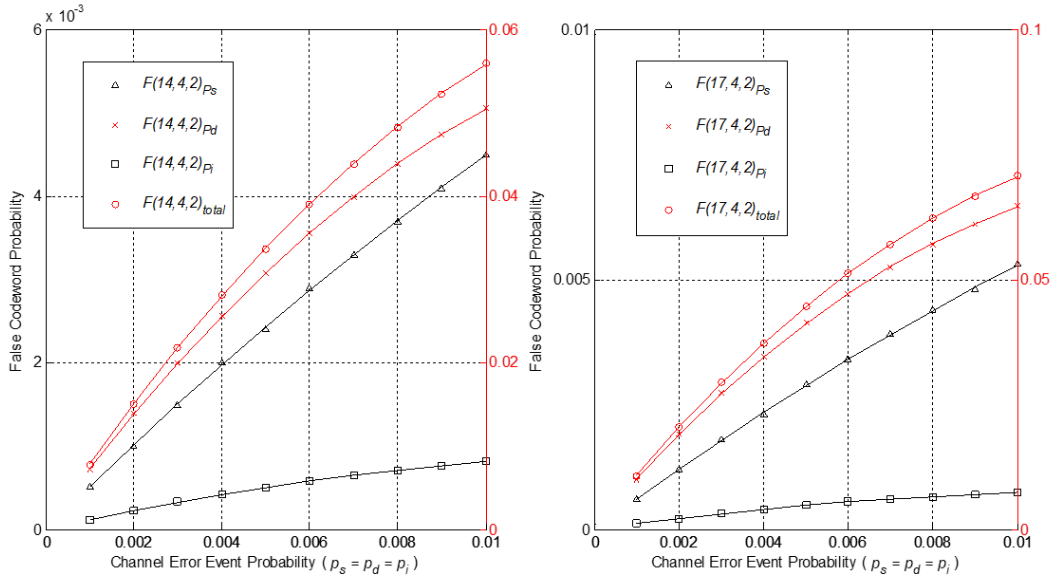


FIGURE 9. Probabilities of false synchronization of generalized $F(n, s, t)$ codes of different lengths with parameters $N = 14, 17$, $s = 4$ and $t = 2$. $p_s = p_d = p_i$ increases from 0.001 to 0.01. Theoretical results are plotted using the solid lines and simulated results are represented by symbols.

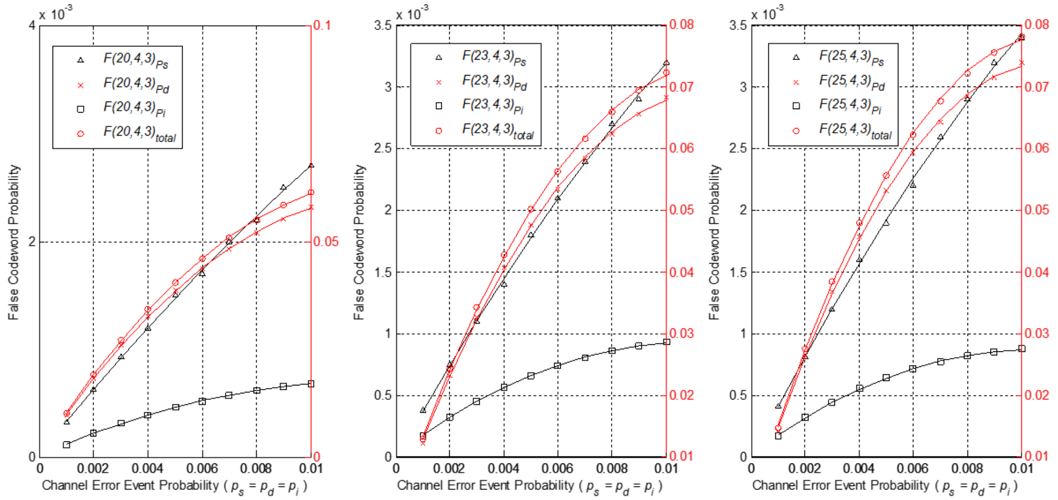


FIGURE 10. Probabilities of false synchronization of generalized $F(n, s, t)$ codes of different lengths with parameters $N = 20, 23, 25$, $s = 4$ and $t = 3$. $p_s = p_d = p_i$ increases from 0.001 to 0.01. Theoretical results are plotted using the solid lines and simulated results are represented by symbols.

10^6 information bits is generated. The information bit sequence is first divided into N blocks, each of which contains k ($k = n - s - t$) information bits based on different generalized $F(n, s, t)$ codes. Then the primer drive and fixed bits are inserted in between the k information bits to form a systematic generalized $F(n, s, t)$ codeword of length n. The encoding procedure is repeated for every block of k information bits, which results in a encoded sequence of length $N \cdot n$.

The encoded sequence, i.e, transmitted sequence $\boldsymbol{t} = \{t_1, t_2, \cdots t_{N \cdot n}\}$ of length $N \cdot n$ is transmitted though the BSID channel model with different parameters p_i, p_d, and p_s. The length of the received sequence is usually different from that of the transmitted sequence due to potential insertion and deletion errors. A fixed-length decoding window with the same length of a codeword is introduced. The decoding window moves from the start of the received sequence until it reaches the last bit. During the whole process, we count the total number of false codewords and count the number of false codewords caused by a single substitution error, a single deletion error and a single insertion error, respectively.

A. PROBABILITIES OF FALSE SYNCHRONIZATION

The simulation results for the probabilities of false synchronization of generalized $F(n, s, t)$ codes with different values

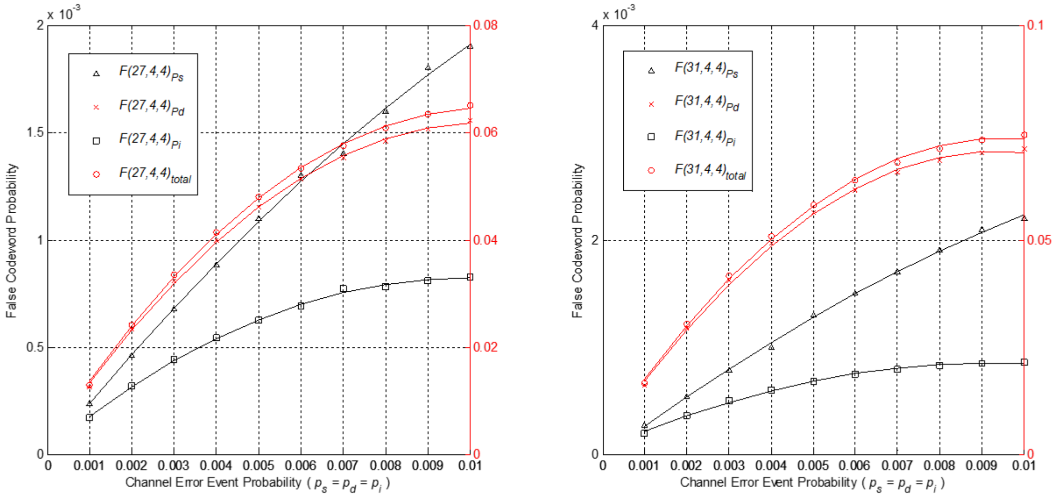


FIGURE 11. Probabilities of false synchronization in generalized $F(n, s, t)$ codes of different lengths with parameters $N = 27, 31$, $s = 4$ and $t = 4$. $p_s = p_d = p_i$ increases from 0.001 to 0.01. Theoretical results are plotted using the solid lines and simulated results are represented by symbols.

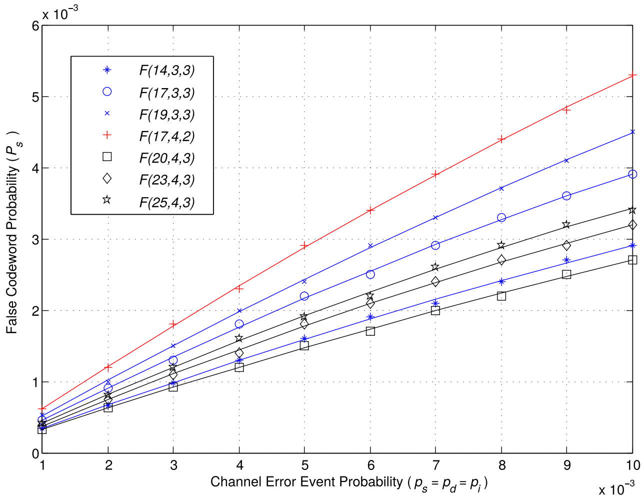


FIGURE 12. The false synchronization probability P_s in a generalized $F(n, s, t)$ code caused by a single substitution error at fixed positions as $p_s = p_d = p_i$ increase from 0.001 to 0.01. Theoretical results are plotted using the solid lines and simulated results are represented by symbols.

of s and t are shown in Fig. 8, Fig. 9, Fig. 10 and Fig. 11. (i) The probability of false synchronization caused by a single substitution error and (ii) the probability of false synchronization caused by a single insertion error are plotted using the left black ordinate axis while (iii) the probability of false synchronization caused by a single deletion error and (iv) the total probability of false synchronization are plotted using the right red ordinate axis. The simulation results represented by symbols match with those theoretical results represented by lines derived in (9), (18) and (19). The figures also show that deletion errors dominate the false synchronization probability while insertion errors have the least influence for the generalized $F(n, s, t)$ codes.

B. SINGLE SUBSTITUTION FALSE SYNCHRONIZATION

In order to analyze the false synchronization of the generalized $F(n, s, t)$ codes further, we show the simulation results of false synchronization in a generalized $F(n, s, t)$ code caused by a single substitution error at the fixed positions, a single deletion error and a single insertion error respectively.

The simulation result for probability of false synchronization in a generalized $F(n, s, t)$ code caused by a single substitution error at the fixed positions is shown in Fig. 12. From the simulation results, we observe that the false synchronization caused by a single substitution error at the fixed positions is not always increasing as the codeword length increases. This probability is closely related to the values of s and t. This can be demonstrated by the simulation result, in which the curve for $F(25, 4, 3)$ code is always lower than those of $F(17, 3, 3)$, $F(19, 3, 3)$ and $F(17, 4, 2)$ codes under the same channel error event probability. However, one point needs to be emphasized is that the false codeword probability caused by a single substitution error at the fixed positions increases as the codeword length increases with the same values of s and t, e.g., comparing results for $F(20, 4, 3)$, $F(23, 4, 3)$ and $F(25, 4, 3)$.

In addition, we investigate the performance of generalized $F(n, s, t)$ codes with different parameters in terms of false codeword probability at different values of p_s caused by a single substitution error at fixed positions in Fig. 13. For all generalized $F(n, s, t)$ codes with different values of s and t, the false codeword probability P_s is slightly decreased when p_d and p_i increase. It is because when p_d increases, more bits are deleted and hence fewer bits are transmitted. When fewer bits are sent, fewer substitution errors can occur. On the other hand, when p_i increases, more bits are inserted and hence the false codeword, if occurs, is more likely to occur due to insertion errors. Therefore, the false codeword probability P_s due to "substitution errors" becomes lower. Moreover,

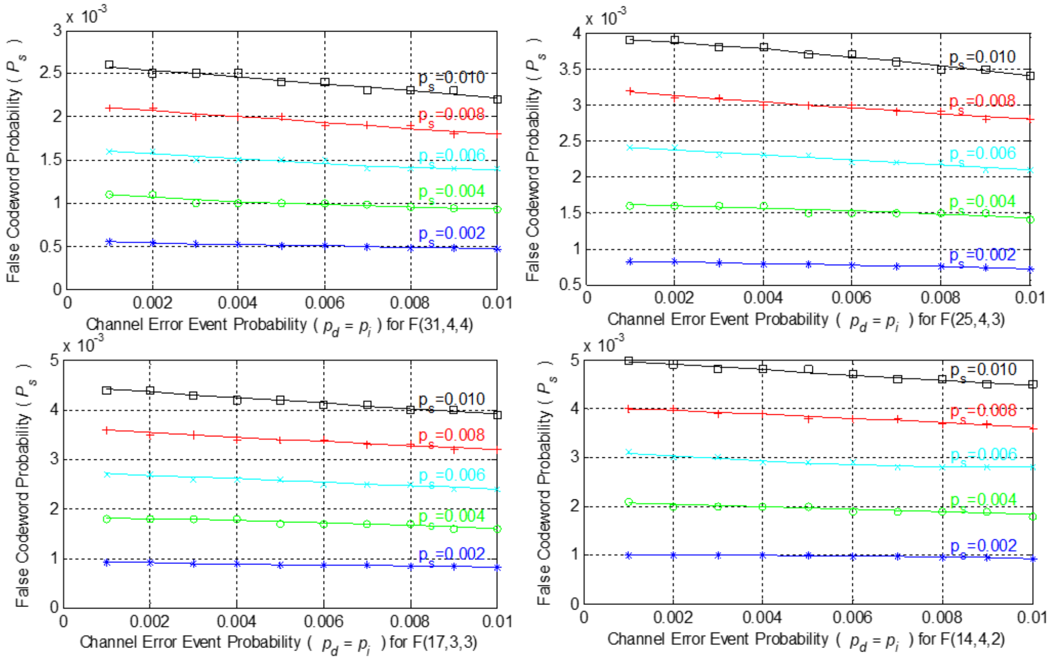


FIGURE 13. The false synchronization probability P_s in a generalized $F(n, s, t)$ code caused by a single substitution error at fixed positions with different values of p_s as $p_i = p_d$ increase from 0.001 to 0.01. Theoretical results are plotted using the solid lines and simulated results are represented by symbols.

P_s changes a lot as p_s increases under the same insertion and deletion probabilities. For instance, the false codeword probability of generalized $F(25, 4, 3)$ is 4.14×10^{-4} under the channel error event $\{p_s = 10^{-3}, p_d = 10^{-3}, p_i = 10^{-3}\}$, and 3.91×10^{-3} under $\{p_s = 10^{-2}, p_d = 10^{-3}, p_i = 10^{-3}\}$.

C. ONE SINGLE SUBSTITUTION ERROR IN EACH OF TWO CONSECUTIVE CODEWORDS FALSE SYNCHRONIZATION

The theoretical false synchronization in a generalized $F(n, s, t)$ code caused by one substitution error at each of two consecutive codewords' fixed positions are shown in Table 1. For a fixed combination of (s, t), false codeword cannot be formed for the maximum codeword length, i.e., $2st + 1$, for specific s and t such as $F(19, 3, 3)$, $F(17, 4, 2)$, $F(25, 4, 3)$ and $F(33, 4, 4)$.

TABLE 1. Theoretical probability of false synchronization in a generalized $F(n, s, t)$ code caused by one single substitution error at fixed positions in each of two consecutive codewords.

n	s	t	p_s	p_d	p_i	P_s
14	3	3	0.01	0.01	0.01	4.44×10^{-6}
17	3	3	0.01	0.01	0.01	8.87×10^{-6}
19	3	3	0.01	0.01	0.01	0
13	4	2	0.01	0.01	0.01	1.77×10^{-5}
17	4	2	0.01	0.01	0.01	0
20	4	3	0.01	0.01	0.01	1.46×10^{-5}
23	4	3	0.01	0.01	0.01	4.18×10^{-6}
25	4	3	0.01	0.01	0.01	0
31	4	3	0.01	0.01	0.01	1.97×10^{-6}
33	4	4	0.01	0.01	0.01	1.07×10^{-6}

Comparing results in Fig. 12 and Table 1, we notice the probability of false synchronization due to two substitution errors in two consecutive codewords is much smaller than that due to a single substitution error. The explanation is as follows: $N_0^0(i)$ is nonzero for $i = 1, 2, \cdots, s - 1$ and $N_1^1(i)$ is nonzero for $i = s, 2s, \cdots, (t - 1)s$. The probability of false synchronization caused by substitution errors at fixed positions is the largest when $d(\mathbf{tt}, \mathbf{g}(i)) = 1$ and the sum of

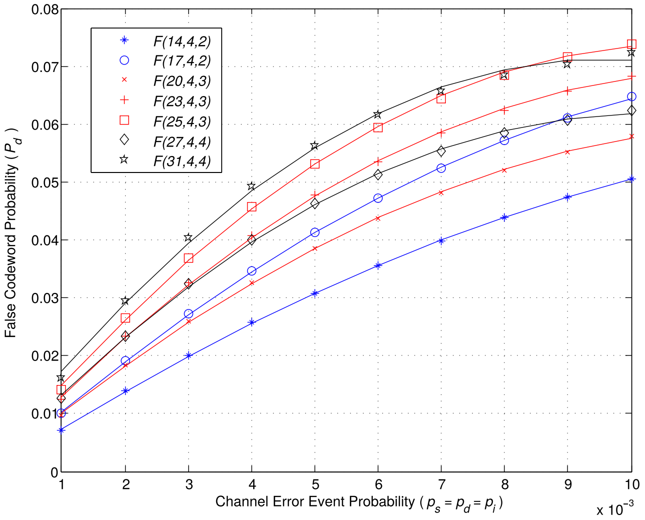


FIGURE 14. The false synchronization probability P_d in a generalized $F(n, s, t)$ code caused by a single deletion error as $p_s = p_d = p_i$ increase from 0.001 to 0.01. Theoretical results are plotted using the solid lines and simulated results are represented by symbols.

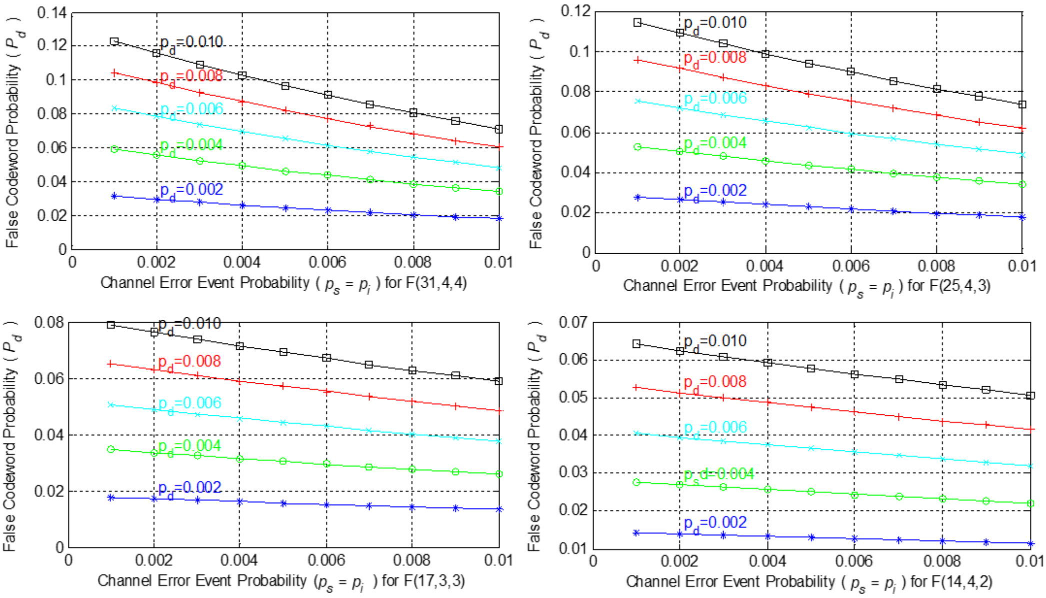


FIGURE 15. The false synchronization probability P_d in a generalized $F(n, s, t)$ code caused by a single deletion error with different values of p_d as $p_i = p_s$ increase from 0.001 to 0.01. Theoretical results are plotted using the solid lines and simulated results are represented by symbols.

$N_0^0(i)$ and $N_1^1(i)$ is large. Under this condition, $h(i)$ is small and $(\frac{1}{2})^{h(i)}$ is large. During the shifting process of the decoding window, we may find $d(\mathbf{tt}, \mathbf{g}(i)) = 1$ is approved under most cases. This is the reason that the probability of false synchronization caused by two substitution errors is much smaller, which is in accordance with the results shown when $d(\mathbf{tt}, \mathbf{g}(i)) = 1$ and $d(\mathbf{tt}, \mathbf{g}(i)) = 2$ in Table 1.

D. SINGLE DELETION FALSE SYNCHRONIZATION

The simulation result for the probability of false synchronization in a generalized $F(n, s, t)$ code caused by a single deletion error is shown in Fig. 14. Similar to false synchronization due to substitution errors, the false synchronization caused by a single deletion increases as the codeword length increases with the same set of $\{s, t\}$, e.g., comparing results for $F(20, 4, 3)$, $F(23, 4, 3)$ and $F(25, 4, 3)$. However, the false synchronization due to a single deletion error is not always increasing as the codeword length increases. This probability is closely related to the values of s and t. This can be verified by the simulation result, in which the curve for $F(27, 4, 4)$ is always lower than that of $F(23, 4, 3)$ and $F(25, 4, 3)$ under the same channel error event probability. This can also be verified by the better performance of $F(27, 4, 4)$ compared to $F(17, 4, 2)$ when the channel error event probability is equal to or greater than 9×10^{-3}.

In addition, we plot the performance of generalized $F(n, s, t)$ codes with different parameters in terms of false codeword probability at different values of p_d due to a single deletion error in Fig. 15. For all four generalized $F(n, s, t)$ codes with different values of s and t, the false codeword probability decreases when p_s and p_i increases. Under the condition p_d is a small value like $p_d = 0.002$ or $p_d = 0.004$, P_d changes very little under the same deletion error probability p_d when both p_s and p_i increase ten times from 10^{-3} to 10^{-2}. When p_d is large, however, P_d can change quite significantly. Moreover, P_d changes significantly as p_d increases under the same insertion and deletion probabilities. For instance, the false synchronization probabilities of generalized $F(25, 4, 3)$ are 0.0142 and 0.1144 under the channel error events $\{p_s = 10^{-3}, p_d = 10^{-3}, p_i = 10^{-3}\}$ and $\{p_s = 10^{-3}, p_d = 10^{-2}, p_i = 10^{-3}\}$, respectively.

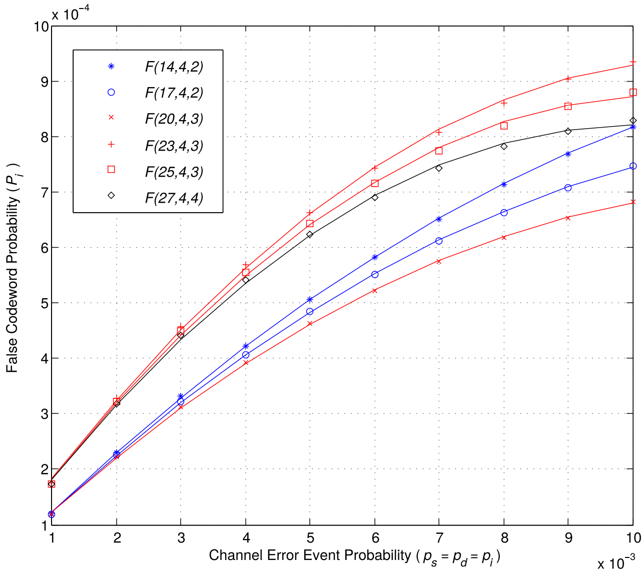


FIGURE 16. The false synchronization probability P_i in a generalized $F(n, s, t)$ code caused by a single insertion error as $p_s = p_d = p_i$ increase from 0.001 to 0.01. Theoretical results are plotted using the solid lines and simulated results are represented by symbols.

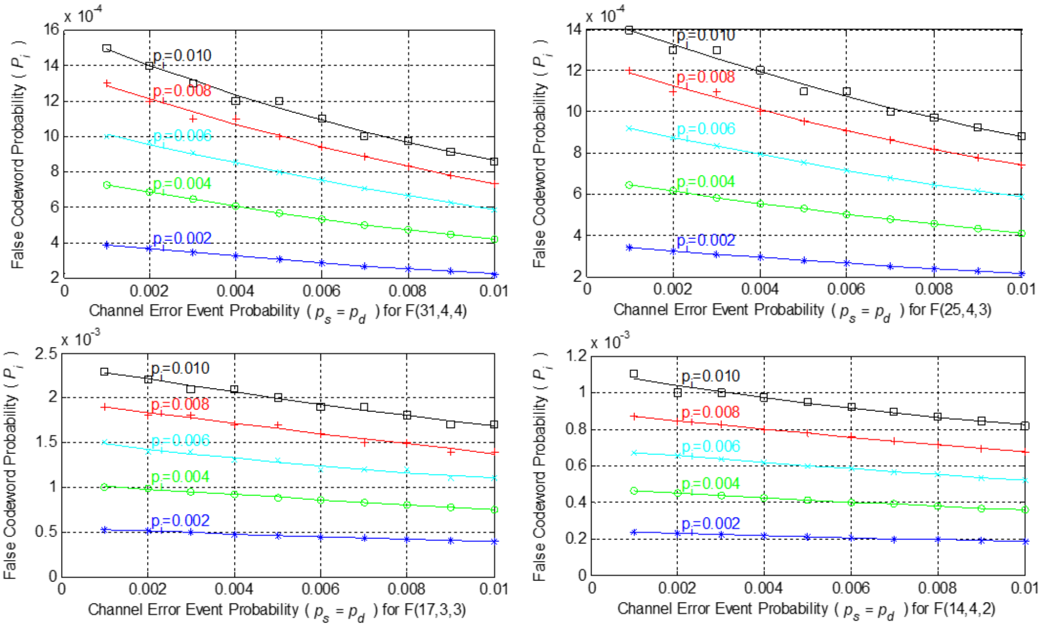


FIGURE 17. The false synchronization probability P_i in a generalized $F(n, s, t)$ code due to a single insertion error with different values of p_i as $p_d = p_s$ increase from 0.001 to 0.01. Theoretical results are plotted using the solid lines and simulated results are represented by symbols.

E. SINGLE INSERTION FALSE SYNCHRONIZATION

The simulation result for the probability of false synchronization in a generalized $F(n, s, t)$ code caused by a single insertion is shown in Fig. 16. One point need to be emphasized is that different from the previous two cases, the false codeword probability caused by a single insertion may not always increase as the codeword length increases even under the same set of $\{s, t\}$. This can be demonstrated as better performance of $F(25, 4, 3)$ compared to $F(23, 4, 3)$.

Same as the first two cases, we plot the performance of four generalized $F(n, s, t)$ codes with different parameters in terms of false codeword probability at different values of p_i caused by a single insertion in Fig. 17. We notice this probability decreases gradually as p_s and p_d increases for all four generalized $F(n, s, t)$ codes. For example, the false codeword probabilities for $F(33, 4, 4)$ under the channel error events $\{p_s = 0.001, p_d = 0.001, p_i = 0.01\}$ and $\{p_s = 0.01, p_d = 0.01, p_i = 0.01\}$ are 0.0150 and 8.1×10^{-4}, respectively.

VI. CONCLUSION

In this paper, we have proposed a new class of systematic comma-free code, namely generalized $F(n, s, t)$ code. For a classical $F(n, s, t)$ code of length n, it requires that is the minimum interger greater or equal to $\sqrt{2(n-1)}$, s is the minimum integer greater or equal to $\frac{r}{2}$ and $t = r - s$. However, for the generalized $F(n, s, t)$ code, the condition $st \geq 0.5(n-1)$ is enough to ensure the generalized $F(n, s, t)$ code is a synchronous code. In addition, we provided the proof. As a result, more combinations of (s, t) can be chosen for a specific codeword length n in constructing the proposed generalized $F(n, s, t)$ code. For instance, $F(17, 4, 2)$ and $F(25, 4, 4)$ are the generalized $F(n, s, t)$ but they are not the classical F codes. The generalized $F(n, s, t)$ code not only provides more choices but also a higher code performance in terms of false synchronization under some specific cases. For example, the generalized $F(17, 4, 2)$ has a better performance compared with $F(14, 4, 2)$ caused by a single insertion error.

The theoretical probabilities of false synchronization caused by three types of errors, namely substitution error, deletion error and insertion error have been derived. Simulations are performed on generalized $F(n, s, t)$ codes with different parameters and corresponding probabilities of false synchronization are recorded under different channel parameters. We observe the simulation results match closely with the theoretical ones. Moreover, deletion errors have a large effect on the probability false synchronization while insertion errors have the least influence.

It has been shown that the probability of false synchronization caused by a single substitution error or a single deletion error increases as the codeword length increases under the same set of $\{s, t\}$. We conclude that shorter length of generalized $F(n, s, t)$ codes perform better in terms of false synchronization under the same set of $\{s, t\}$ for these two cases. However, shorter length codes require more redundancy, which decrease the code rate and efficiency of generalized $F(n, s, t)$ code. Different from the previous two cases, the false codeword probability caused by a single insertion error may not always increase as the codeword length increases even under the same set of $\{s, t\}$. The generalized $F(n, s, t)$ code of the maximum codeword length, i.e., $2st + 1$ for a fixed set of $\{s, t\}$ usually perform better against the false synchronization caused by a single insertion error.

Therefore, the false synchronization of the generalized $F(n, s, t)$ code is not only related to the parameters of the channel but also to the number of fixed bits and codeword length. Thus, a specific generalized $F(n, s, t)$ code should be chosen based on different applications and requirements, i.e., channel parameters, code rate, codeword length and so on.

VII. ACKNOWLEDGMENT

This paper was presented in part at the IEEE Asia Pacific Conference on Circuits and Systems 2017 [1].

REFERENCES

[1] T. Xue and F. C. M. Lau, "A generalized systematic comma free code," in *Proc. 23rd Asia–Pacific Conf. Commun. (APCC)*, Dec. 2017, pp. 1–5.

[2] J. Hu, T. M. Duman, E. M. Kurtas, and M. F. Erden, "Bit-patterned media with written-in errors: Modeling, detection, and theoretical limits," *IEEE Trans. Magn.*, vol. 43, no. 8, pp. 3517–3524, Aug. 2007.

[3] D. Bardyn, J. A. Briffa, A. Dooms, and P. Schelkens, "Forensic data hiding optimized for JPEG 2000," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2011, pp. 2657–2660.

[4] G. Sharma and D. J. Coumou, "Watermark synchronization: Perspectives and a new paradigm," in *Proc. 40th Annu. Conf. Inf. Sci. Syst.*, Mar. 2006, pp. 1182–1187.

[5] C. T. Clelland, V. Risca, and C. Bancroft, "Hiding messages in DNA microdots," *Nature*, vol. 399, no. 6736, pp. 533–534, Jun. 1999.

[6] C. Bancroft, T. Bowler, B. Bloom, and C. T. Clelland, "Long-term storage of information in DNA," *Science*, vol. 293, no. 5536, pp. 1763–1765, 2001.

[7] G. M. Church, Y. Gao, and S. Kosuri, "Next-generation digital information storage in DNA," *Science*, vol. 337, no. 6102, p. 1628, 2012.

[8] N. Goldman *et al.*, "Towards practical, high-capacity, low-maintenance information storage in synthesized DNA," *Nature*, vol. 494, pp. 77–80, Jan. 2013.

[9] G. M. Tenengol'ts, "A class of codes correcting bit loss and errors in the preceding symbol," *Avtomatika Telemekhanika*, vol. 37, no. 5, pp. 174–179, 1976.

[10] G. Zhou and Z. Zhang, "Synchronization recovery of variable-length codes," *IEEE Trans. Inf. Theory*, vol. 48, no. 1, pp. 219–227, Jan. 2002.

[11] E. N. Gilbert and E. F. Moore, "Variable-length binary encodings," *Bell Syst. Tech. J.*, vol. 38, no. 4, pp. 933–967, Jul. 1959.

[12] E. Gilbert, "Synchronization of binary messages," *IRE Trans. Inf. Theory*, vol. 6, no. 4, pp. 470–477, Sep. 1960.

[13] S. W. Golomb, B. Gordon, and L. R. Welch, "Comma-free codes," *Can. J. Math*, vol. 10, no. 2, pp. 202–209, 1958.

[14] S. W. Golomb, L. R. Welch, and M. Delbrück, "Construction and properties of comma-free codes," *Biol. Medd. Dan. Vid. Selsk.*, vol. 23, no. 9, pp. 1–34, 1958.

[15] S. Golomb, "Efficient coding for the desoxyribonucleic channel," in *Proc. Symp. Appl. Math.*, vol. 14, 1962, pp. 87–100.

[16] B. Jiggs, "Recent results in comma-free codes," *Can. J. Math*, vol. 15, nos. 178–187, p. 377, 1963.

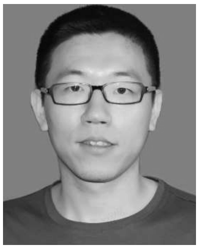
[17] W. Eastman, "On the construction of comma-free codes," *IEEE Trans. Inf. Theory*, vol. IT-11, no. 2, pp. 263–267, Apr. 1965.

[18] W. Eastman and S. Even, "On synchronizable and PSK-synchronizable block codes," *IEEE Trans. Inf. Theory*, vol. 10, no. 4, pp. 351–356, Oct. 1964.

[19] D. J. Clague, "New classes of synchronous codes," *IEEE Trans. Electron. Comput.*, vol. EC-16, no. 3, pp. 290–298, Jun. 1967.

[20] J. A. Briffa and H. G. Schaathun, "Improvement of the Davey–MacKay construction," in *Proc. IEEE Int. Symp. Inf. Theory Appl.*, Dec. 2008, pp. 1–4.

[21] M. C. Davey and D. J. C. MacKay, "Reliable communication over channels with insertions, deletions, and substitutions," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 687–698, Feb. 2001.



TIANBO XUE received the B.E. degree in measurement, control technique, and instruments from the Harbin Institute of Technology, China, and the M.Sc. degree in electrical engineering from Syracuse University, USA. He is currently pursuing the Ph.D. degree with The Hong Kong Polytechnic University, Hong Kong.

FRANCIS C. M. LAU (M'93–SM'03) received the B.Eng. degree (Hons.) in electrical and electronic engineering from King's College London and the Ph.D. degree from the University of London, U.K. He is currently a Professor and an Associate Head of the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong. He is also a fellow of the IET.

He has co-authored *Chaos-Based Digital Communication Systems* (Heidelberg: Springer-Verlag, 2003) and *Digital Communications with Chaos: Multiple Access Techniques and Performance Evaluation* (Oxford: Elsevier, 2007). He is also a co-holder of five U.S. patents. He has published around 300 papers. His main research interests include channel coding, cooperative networks, wireless sensor networks, chaos-based digital communications, applications of complex-network theories, and wireless communications. He was the Chair of Technical Committee on Nonlinear Circuits and Systems, IEEE Circuits and Systems Society, from 2012 to 2013. He is also serving as the General Co-Chair of the International Symposium on Turbo Codes & Iterative Information Processing 2018. He served as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II from 2004 to 2005, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS I from 2006 to 2007, and the *IEEE Circuits and Systems Magazine* from 2012 to 2015. He has been a Guest Associate Editor of the *International Journal and Bifurcation and Chaos* since 2010 and an Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II since 2016.