Binary Codes for Packet Error and Packet Loss Correction in Store and Forward

Maximilien Gadouleau and Alban Goupil
CReSTIC, Université de Reims Champagne-Ardenne
Reims, France
E-mails: {maximilien.gadouleau, alban.goupil}@univ-reims.fr

Abstract—In this paper, we introduce a novel approach to correct packet errors and packet losses in store and forward by using binary error-correcting codes. Using a framework similar to what has been proposed for error control in random linear network coding, we investigate error control under two scenarios. First, we show that the Hamming metric is suitable for error control in the case of errors intrinsic to the network. Second, we investigate the case of an adversary on the network who erases and injects packets in order to corrupt the communication. Under this setting, we show that error correction is performed using a new metric, referred to as the modified Hamming metric. We then investigate using constant-weight codes and linear codes for error correction in store and forward. We thus show that the traditional approach of indexing the packets in order to recover their original order is a suboptimal restriction of our approach.

I. INTRODUCTION

Store and forward is the traditional means to transmit data through a network. In this scheme, the intermediate nodes simply retransmit the packets they receive towards their destinations without combining them. The traditional techniques to correct packet loss in store and forward include ARQ and erasure codes based on Reed-Solomon proposed in [1], [2], [3], [4]. The approach based on Reed-Solomon codes, referred to as the traditional approach henceforth, distributes a codeword amongst the packets, hence protecting against packet loss. In order to recover the order of packets, a header indicating the index of the packet in the message is added in front of every packet.

In this paper, we propose a novel approach to correct errors and packet losses in store and forward by using binary error-correcting codes. We assume that the message undergoes four possible modifications. We first assume that the packets are randomly permuted, and hence do not arrive at the receiver’s end in the order they were originally sent. This assumption is motivated by networks whose topologies change over time, or where several routes with unequal delays are available to transmit the data. Also, the message can suffer from three types of packet modifications: packets can be lost due to fading, some packets can be injected by an adversary, and others can be corrupted by errors. We hence model the communication of a message using store and forward as the transmission of the set of packets, where the modifications of packets correspond to set alterations. Using a correspondence between subsets of a set and binary vectors, we then model communications using store and forward as a binary channel with additive error. Thus, error control for store and forward can be done using binary error-correcting codes. Although the Hamming metric is appropriate in the case of modifications due to the network, we also introduce an adversarial scenario where error correction is done via a new metric, referred to as the modified Hamming metric.

We then investigate using constant-weight codes, which is equivalent to transmitting messages with the same number of packets. Constant-weight codes have been widely studied because of their applications and their theoretical significance (see [5] and [6] for comprehensive surveys). We show that the traditional approach of packet loss protection based on Reed-Solomon codes can be viewed in our model as using a special class of constant-weight codes, referred to as liftings of Hamming metric codes. Therefore, the traditional approach can be viewed as a suboptimal restriction of our approach to the set of all liftings.

Since the length of the binary codes proposed here increases exponentially with the number of symbols in a packet, encoding and decoding complexities could become an issue. In order to tackle this, we finally investigate using linear codes for error control in store and forward. These codes have a low encoding complexity and many classes of linear codes also have a low decoding complexity. However, other practical issues arise when using these codes. For instance, transmitting the all-zero vector, which is a codeword in any linear code, would result in sending an empty message. Other issues are also investigated, such as the standard deviation and the maximum value of the number of packets in a message.

Our approach offers many advantages, listed below. First, it is based on a simple network protocol, store and forward, where the intermediate nodes do not operate on packets. Our approach is well suited for store and forward, as the error control operations are done at the receiver’s end only, and without any feedback. Second, our codes have a higher rate than the codes previously proposed. Although the rate gain increases with the number of packets, it is always positive and increases rapidly for a number of packets that is small compared to the total number of possible packets. Third, unlike traditional methods which only protect against packet losses, our approach also corrects packet injections and packet errors. Hence the packets need not be coded against link errors, and the data rate can be further increased. Fourth, our scheme is
universal, as it is always based on binary codes, regardless of
the alphabet or the length of the packets.

We remark that the framework introduced here is similar to
the one introduced in [7], [8] for error correction in random
linear network coding. However, the model for network coding
is defined only in terms of subspaces, while our model is
defined in terms of both subsets and binary vectors. This
allows us to take advantage of the structure of binary vector
spaces and of the wealth of results on binary error-correcting
codes. Also, the modified Hamming metric defined here has
similar properties to the injection metric defined in [8] for
network coding and both arise from adversarial scenarios.
However, the assumptions on the adversary in these two
scenarios are completely different.

The rest of the paper is organized as follows. Section II
introduces the operator channel model for store and forward,
the metrics used for error correction, and demonstrates how
binary codes can be used for that purpose. Section III then
investigates using constant-weight codes and compares it to
the traditional approach. Finally, Section IV discusses the
implementation issues of using linear codes.

II. CHANNEL MODEL AND METRICS

A. Channel model

For any positive integer $q$, we denote $[q] = \{0, 1, \ldots, q-1\}$
henceforth. Note that we do not assume any underlying struc-
ture for $[q]$. Suppose a source wants to transmit a collection
of packets in $[q]^n$, referred to as a message, to one or several
receivers across a network. The protocol used is store and
forward, where the intermediate nodes do not operate on the
packets they receive. We first assume that all network links
and nodes are error-free and that no packets are lost, hence
all the packets sent by the source will be correctly received
by all the receivers. However, we also assume that the packets
may not be received in the order they were originally sent.
We remark that the information carried by the message can
only be a property of this message which remains invariant
after the modifications operated by the channel. Since the
order of packets in the original message is lost through the
communication, store and forward can thus be viewed as the
error-free transmission of only the set of packets, rather than
the ordered sequence of packets.

Let us now consider the case where the network is not
error-free, hence the original message may be modified due to
link errors, packet losses, malfunctioning nodes, an adversary
injecting packets maliciously, etc. All these modifications can
be viewed as alterations of the set of packets $P \in \mathcal{P}([N])$ sent
by the source, where $N = q^n$ represents the total number of
possible packets and $\mathcal{P}(E)$ represents the power set of a set
$E$. These alterations can always be expressed as products of
erasures and disclosures, where an erasure is the deletion of an
element in the subset, and a disclosure is the addition of a
new element into the subset\(^1\).

We now illustrate how message modifications lead to al-
terations of the corresponding set of packets. We consider
three basic alterations of the message. First, a packet loss
leads to either an erasure if no copy of the packet is being
transmitted on the network or no alteration otherwise. Second,
the injection of a new packet leads to either a disclosure if
the newly added packet is distinct to the other packets in the
message or no alteration otherwise. Third, a packet in error can
be viewed as a packet loss immediately followed by a packet
injection. Therefore, a packet error can lead to an erasure, to
a disclosure, or to both an erasure and a disclosure.

After $\epsilon$ erasures and $\delta$ disclosures, the received subset $Q$
can be expressed as

$$Q = (P \setminus E) \cup D,$$

where $E = Q \cap P$ and $D = Q \setminus P$ have cardinalities $\epsilon$ and $\delta$,
respectively. Note that $E$ and $D$ thus represent the packets lost
in the channel, and the error packets injected by the channel,
respectively. Equivalently, denoting $A = P \Delta Q = E \cup D$, where $\Delta$
denotes the symmetric difference between two sets, (1) can be expressed as

$$Q = P \Delta A,$$

where $|A| = \epsilon + \delta$ is the cardinality of $A$. We thus model
communication of a message using store and forward as an
operator channel, where the input $P$ is a subset of $[N]$ and the
output $Q$ is a random subset of $[N]$ related to $P$ by (2), where $\epsilon$ and $\delta$
are upper bounded. We remark that the counterpart of (1) for subspaces was given in [7, (2)], while (2) has no
counterpart for subspaces. This is due to the underlying group
structure of the power set which has no counterpart in the
projective space.

B. Metrics for correction of erasures and disclosures

The Hamming distance between two subsets $P, Q \in \mathcal{P}([N])$, where $Q$ is obtained from $P$ after $\epsilon$ erasures and
$\delta$ disclosures, is defined as $d_H(P, Q) = \epsilon + \delta = |P \Delta Q|$. The Hamming distance clearly is a metric, and is the appropriate
metric for error correction when the sum of the number of
erasures and disclosures is upper bounded.

We now introduce a scenario where another metric is more
appropriate for error correction in store and forward. Suppose
an adversary who wishes to corrupt the communication is
present on the network. This adversary can erase some existing
packets and inject new ones into the network, but cannot mod-
ify the packets going through the network. In order to search
for malicious adversaries, we suppose that the injection of a
given number of packets or the erasure of the same number of

\(^1\)For random linear network coding, Kötter and Kschischang introduced the
terms “erasure” and “error”, respectively, in order to describe the counterparts
of the alterations to subspaces [7]. However, the terms “erasure” and “disclo-
sure” seem to be more appropriate, as a disclosure represents the gain of some
amount of information. Also, the terms of “errors and erasures” are commonly
referred to describe significantly different alterations to vectors or matrices.
Therefore, in order to avoid confusion, we shall use the terms “erasure” and
“disclosure” henceforth.
packets will invalidate the transmission. Therefore, the adversary will make sure that \( \max\{\epsilon, \delta\} \) is below the alarm bound. The modified Hamming distance between \( P \) and \( Q \) is thus defined as \( d_M(P, Q) = \max\{\epsilon, \delta\} = \max\{|P \setminus Q|, |Q \setminus P|\} \).

**Lemma 1:** The modified Hamming distance is a metric.

**Proof:** For all subsets \( P, Q \in \mathcal{P}(\mathbb{N}) \), we have

\[
d_M(P, Q) = \frac{1}{2}d_H(P, Q) + \frac{1}{2}|P| - |Q|.
\]

Since the Hamming distance is a metric, it follows from (3) that the modified Hamming distance is also a metric. \( \square \)

By (3), we have for all \( P, Q \in \mathcal{P}(\mathbb{N}) \),

\[
d_M(P, Q) \leq d_H(P, Q) \leq 2d_M(P, Q),
\]

which implies that these metrics are equivalent. By (4), the problems of finding optimal codes in the Hamming metric and in the modified Hamming metric are closely related, but not necessarily equivalent. We remark that both metrics introduced above are symmetrical in terms of erasures and disclosures. However, there may be other scenarios for which asymmetrical metrics are more appropriate for error control.

**C. Encoding and decoding using binary codes**

We now demonstrate how the channel model and the metrics introduced for subsets in Sections II-A and II-B, respectively, can be equivalently defined in terms of binary vectors. Any subset \( P \in \mathcal{P}(\mathbb{N}) \) is uniquely represented by a binary vector \( p = (p_0, p_1, \ldots, p_{N-1}) \in \mathbb{F}_2^N \), where \( p_i = 1 \) if and only if \( i \in P \), or equivalently, \( \supp(p) = P \). Therefore, the operator channel can be defined as a channel on \( \mathbb{F}_2^N \), where the input \( p \) and the output \( q \) are related by

\[ q = p + a, \]

where \( |\supp(a) \cap \supp(p)| = \epsilon \) and \( |\supp(a) \setminus \supp(p)| = \delta \). Note that \( |\supp(a)| = w_H(a) = \epsilon + \delta \). Moreover, for all \( p, q \in \mathbb{F}_2^N \), we define the modified Hamming distance between \( p \) and \( q \) as \( d_M(p, q) = \frac{1}{2}d_H(p, q) + \frac{1}{2}|w_H(p) - w_H(q)| \). We thus have

\[
d_H(p, q) = d_H(\supp(p), \supp(q)), \quad d_M(p, q) = d_M(\supp(p), \supp(q)),
\]

for all \( p, q \in \mathbb{F}_2^N \). Therefore, error control in store and forward can be treated a coding theory problem on binary vectors with the Hamming metric or the modified Hamming metric, depending on the scenario considered. We describe below the encoding and decoding processes for the transmission of a message using a binary code.

The encoding of a message when using a binary error correcting code \( C \subseteq \mathbb{F}_2^N \) with cardinality \( M \) and minimum Hamming distance \( d \) proceeds as follows. Suppose the source wishes to send a block \( x \in [M] \). It first uses the encoding mapping \( f \) from \([M]\) to \( C \) to determine \( f(x) = c \in C \). Let \( i_0 < i_1 < \ldots < i_{L-1} \) be the nonzero coordinates of \( c \), where \( L = w_H(c) \). Then the message will consist of exactly \( L \) packets \( x_j \in [g]^n \), where \( x_j \) is the representation of the number \( i_j \) in basis \( q \). We remark that this encoding process guarantees that the packets are in lexicographic order.

The decoding thus proceeds: Suppose the channel has applied \( \epsilon \) erasures and \( \delta \) disclosures to the original set of packets, where \( \epsilon + \delta < \frac{d}{2} \). The receiver obtains a different set of \( L' \) packets, which after sorting into lexicographic order, are denoted as \( y_l \in [q]^n \), \( 0 \leq l \leq L' - 1 \). Viewing these words as integers \( k_l \in [N] \), the receiver then determines \( y \in \mathbb{F}_2^N \) for all \( l \) and \( y_l = 0 \) otherwise. Finally, the receiver applies the decoding algorithm for the code \( C \) on \( y \) to determine \( c \in C \), and retrieves \( x = f^{-1}(c) \).

**III. Constant-weight codes and liftings**

**A. Constant-weight codes**

In order to simplify the transmission protocol and to facilitate the decoding process at the receiver end, the source may choose to send messages with a constant number of packets. In the model introduced in Section II-C, this corresponds to using a binary constant-weight code. Constant-weight codes have other advantages listed below.

First, the set of binary vectors with the same Hamming weight is highly structured: when endowed with the Hamming metric, it forms an association scheme, referred to as the Johnson scheme [9], [10]. Constant-weight codes also have many connections to other classes of codes, such as spherical codes [6], binary codes [11], and constant-dimension codes [12]. Because of these features, constant-weight codes have attracted a lot of interest in the literature.

Second, by (3), (6), and (7), we have \( d_H(p, q) \leq 2d_M(p, q) \) for all \( p, q \in \mathbb{F}_2^N \), and equality holds if and only if they have equal Hamming weights. Hence the minimum Hamming distance of a constant-weight code is equal to twice its minimum modified Hamming distance, which is the largest minimum Hamming distance once the minimum modified Hamming distance is fixed. Constant-weight codes can thus be used interchangeably for both scenarios previously considered. The property above also implies that the combinatorial and geometric properties of the Johnson scheme when endowed with the Hamming metric are preserved when it is endowed with the modified Hamming metric instead.

Third, let us denote the maximum cardinality of a binary code of length \( N \) with minimum Hamming distance \( d \) as \( A_H(N, d) \), and the maximum cardinality of a constant-weight code of length \( N \) and weight \( L \) with minimum Hamming distance \( d \) as \( A_H(N, L, d) \). Clearly, \( A_H(N, L, d) \geq A_H(N, L, d) \) for all \( 0 \leq L \leq N \). Also, the Bassalygo-Elias bound [13], [11, Ch. 17, Theorem 33] indicates that

\[
A_H(N, L, d) \geq \binom{N}{L} \frac{\binom{L}{d}}{2^N} \!
\]

for all \( 0 \leq L \leq N \). Following Stirling’s formula, the binomial coefficient \( \binom{N}{L} \) satisfies for all \( N \) and \( 0 \leq L \leq N \) [11, Chapter 7],

\[
\frac{2^{NH(x)}}{\sqrt{8L(1-\lambda)}} \leq \binom{N}{L} \leq \frac{2^{NH(x)}}{\sqrt{2\pi L(1-\lambda)}}.
\]
where $\lambda = \frac{L}{N}$. In particular, for $N$ even and $L = \frac{N}{2}$, (9) and (8) lead to

$$A_H(N, d) \leq A_H \left( N, \frac{N}{2}, d \right) \leq A_H(N, d),$$ (10)

and hence constant-weight codes with weight around half their length form asymptotically optimal binary codes in the Hamming metric. We now derive a similar result for the modified Hamming metric.

**Proposition 1:** The maximum cardinality $A_M(N, d)$ of a binary code of length $N$ with minimum modified Hamming distance $d$ and the maximum cardinality $A_M(N, L, d)$ of a constant-weight code of length $N$ and weight $L$ with minimum modified Hamming distance $d$ satisfy

$$A_M(N, d) \leq \frac{(N + 1) \sqrt{2N}}{2N} \leq A_M \left( N, \frac{N}{2}, d \right) \leq A_M(N, d).$$ (11)

**Proof:** The upper bound being trivial, we now prove the lower bound. Since any constant-weight code has minimum modified Hamming distance $d$ if and only if it has minimum Hamming distance $2d$, we obtain $A_M(N, L, d) = A_H(N, L, 2d)$ for all $0 \leq L \leq N$. Combining with (10), we obtain

$$A_M \left( N, \frac{N}{2}, d \right) = A_H \left( N, \frac{N}{2}, 2d \right) \geq A_H(N, 2d) \sqrt{2N}. $$ (12)

Also, the codewords of weight $L$ in a binary code of minimum distance $d$ form a constant-weight code with minimum distance at least $d$, and hence

$$A_M(N, d) \leq \sum_{L=0}^{N} A_M(N, L, d) \leq A_M \left( N, \frac{N}{2}, 2d \right) \leq (N + 1) A_H(N, 2d).$$ (13)

Combining (12) and (13), we obtain (11). □

Thus constant-weight codes form asymptotically optimal codes for both metrics. As a corollary, there are codes which are nearly optimal for both metrics. However, codes that are simultaneously optimal for both metrics do not necessarily exist.

**B. Liftings of Hamming metric codes**

Fourth, constant-weight codes are related to Hamming metric codes over larger alphabets through the lifting operation, described below.

**Definition 1:** Let $L, M \geq 1$ and $X = (X_0, X_1, \ldots, X_{L-1}) \in [M]^L$. Representing each coordinate $X_i$ into a binary vector $x_i = (x_{i,0}, x_{i,1}, \ldots, x_{i,M-1})$ where $x_{i,i} = 1$ if and only if $X_i = j$, the lifting of $X$, denoted as $I(X)$, is the vector in $GF(2)^{LM}$ obtained by concatenating all the $L$ vectors $x_i$.

Note that since the vector $x_i$ has Hamming weight 1 for all $0 \leq i \leq L - 1$, $I(X)$ has Hamming weight $L$. We remark that although the original word $X$ can have coordinates over any alphabet, its lifting $I(X)$ is always a binary vector. The distance between two liftings is related to the distance between the original words.

**Lemma 2:** For all $X, Y \in [M]^L$, $d_H(I(X), I(Y)) = 2d_M(I(X), I(Y)) = 2d_H(X, Y)$.

**Proof:** Since $w_H(I(X)) = w_H(I(Y))$, we have $d_H(I(X), I(Y)) = 2d_M(I(X), I(Y))$. Following the notations introduced in Definition 1, let $X$ and $Y$ be represented by $L$ binary vectors $x_i$ and $y_i$, respectively. Then we have $d_H(I(X), I(Y)) = \sum_{i=0}^{L-1} d_H(x_i, y_i)$. Clearly, if $X_i = Y_i$ then $d_H(x_i, y_i) = 0$; on the other hand, if $X_i \neq Y_i$ then $d_H(x_i, y_i) = 2$ since $w_H(x_i) = w_H(y_i) = 1$. Therefore,

$$d_H(I(X), I(Y)) = 2|\{i : X_i \neq Y_i\}| = 2d_H(X, Y).$$ □

The definition of lifting is naturally extended to codes in $[M]^L$ as follows: $I(C) = \{I(X) : X \in C\}$. Lemma 2 establishes a strong relation between an $M$-ary Hamming metric code and its lifting, which is a constant-weight code. This relation is summarized in Proposition 2 below.

**Proposition 2:** Let $C$ be a code in $[M]^L$ with minimum Hamming distance $d$, then its lifting $I(C)$ is a constant-weight code in $GF(2)^{ML}$ with weight $L$ and minimum Hamming distance $2d$.

We now compare our subset approach to the traditional way of transmitting data across a channel which permutes the packets. Let the source send a message of $L$ packets in $[q]^L$, which is viewed as a binary vector $p \in GF(2)^N$ with Hamming weight $L$. By definition, our approach chooses amongst all $\binom{N}{L}$ such vectors. However, in the traditional approach, the source adds a header to the packets indicating their index in the original order for the receivers to determine the original order of packets. Proposition 3 that such a message corresponds to the lifting of some word.

**Proposition 3:** A message of $L$ packets in $[q]^n$ transmitted using the traditional method corresponds to the lifting of a word in $[q^{2^L-n}]^L$, where $l = \log_2 L$.

**Proof:** Let $x_0, x_1, \ldots, x_{L-1} \in [q]^n$ form a message of $L$ packets transmitted using the traditional method. Since each packet $x_i$ can be represented as $(i, X_i)$, where $X_i \in [q^{2^L-n}]$, for $0 \leq i \leq L-1$, the corresponding binary vector with weight $L$ has ones at the coordinates $iq^{2^L-n} + X_i$. This is hence the lifting of the vector $(X_0, X_1, \ldots, X_{L-1}) \in [q^{2^L-n}]^L$. □

Conversely, it is easily shown that any lifting corresponds to a message where the header gives the index of the packet. Therefore, the traditional index method restricts itself to the set of all liftings. Since there are $L$ packets in every message, the header takes $\lceil L \rceil$ symbols, leading to a total overhead of $L \lceil L \rceil$ symbols. Therefore, only $q^{L(L+1)/2} \leq \frac{q^L}{L + 1} < \frac{q^L}{L}$ vectors can be obtained through the traditional index method. Thus, the traditional approach only considers a proper subset of all vectors with weight $L$ and is hence suboptimal.

We want to emphasize the gain in terms of data rate of our binary approach over the traditional approach by comparing their respective asymptotic rates. We remark that the asymptotic rates of constant-weight codes and Hamming metric codes are still unknown for all distances [11]. Therefore, the gain cannot be derived for all possible distances. Instead, we
only consider the alphabets on which these codes are based, which is equivalent to setting the minimum distance to 1. Although this is only a special case, we believe it provides an interesting insight on the rate improvement obtained by using the binary vector approach over the traditional approach. Thus consider the combinatorial rates \( R_b(L) = \frac{\log_q (N \choose L)}{Ln} \), \( R_t(L) = \frac{L(n-\lfloor t \rfloor)}{L(n-\lfloor t \rfloor)} \) of the binary approach and of the traditional approach, respectively. We also introduce the gain

\[
G(L) = \frac{R_b(L)}{R_t(L)} = \frac{\log_q (N \choose L)}{L(n-\lfloor t \rfloor)}
\]

of the binary approach over the traditional approach. Proposition 4 determines the asymptotic form of the gain, denoted as \( g(\lambda) = \lim_{N \to \infty} G(L) \), where \( \lambda = \frac{t}{n} \).

**Proposition 4:** The asymptotic gain of the binary approach over the traditional approach, where all messages consist of \( L \) packets in \([q]^n\) is given by

\[
g(\lambda) = -\frac{H(\lambda)}{\lambda \log_2 \lambda},
\]

where \( \lambda = \frac{t}{n} \) and \( H(\lambda) \) is the binary entropy function.

**Proof:** Incorporating the bounds on the binomial coefficient in (9) into (14), we obtain

\[
\frac{N H(\lambda)}{L(n-\lfloor t \rfloor) \log_2 q} - \frac{1}{2L} \log_q (8L(1-\lambda)) \\
\leq G(L) \\
\leq \frac{N H(\lambda)}{L(n-\lfloor t \rfloor) \log_2 q} - \frac{1}{2L} \log_q (2\pi L(1-\lambda)).
\]

Since \( \log_2 \lambda = (1-n) \log_2 q \), (16) easily leads to (15). \( \blacksquare \)

The gain \( g(\lambda) \) is plotted in Figure 1 for \( \lambda \) ranging from 0 to 1. Recall that the parameter \( \lambda \) represents the ratio of the number of packets in a message over the number of possible packets. We remark that the range of \( \lambda \) can be split into two regions. The first region, where \( \lambda \) is low, represents the case where the messages consist of a small number of large packets. The other region, where \( \lambda \) is high, represents the case where many small packets are transmitted in each message. It is clear from Figure 1 that our approach, although it always improves upon the traditional approach, is more suitable for the second region. However, because of its definition, \( \lambda \) would typically have a small value in applications. Nonetheless, we remark that because \( g(\lambda) \) has an infinite derivative at 0, the gain increases rapidly for small values of \( \lambda \), hence our approach offers some gain, even for very small values of \( \lambda \). In order to illustrate this gain for small values of \( \lambda \), suppose the source wishes to send \( L = 64 \) packets of 16 bytes, i.e. \( N = 2^{128} \) and \( \lambda = 2^{-122} \). Then the total number of bits sent is \( L \log_2 N = 8192 \), and the number of useful bits sent using the traditional approach is given by \( L(\log_2 N - \log_2 L) = 7808 \), while the number of useful bits sent using the subset approach is around 7892. Therefore, our approach saves 84 bits, hence nearly a whole packet.

**IV. LINEAR CODES**

As shown in Section II-C, any binary code can be used for error control in store and forward. In Section III, we described the advantages and issues of using constant-weight codes. In this section, we focus on linear codes instead. The main advantages of linear codes are the ease of encoding and the fact that many classes of linear codes have efficient decoding algorithms as well. This is particularly desirable, as the code length increases exponentially with \( n \). However, a practical issue arises when using linear codes, as described below.

Suppose the source wants to transmit a vector \( x \in GF(2)^K \) using an \((N,K)\) linear code \( C \) with minimum Hamming distance \( d \). The encoding function of \( C \) is a matrix multiplication: \( c = xG \), where \( G \) is the generator matrix of the code. According to Section II-C, the corresponding message consists of \( w_t(c) \) packets. In particular, if the source wishes to send the all-zero vector, then \( c = 0 \) and the received message consists of \( w_t(0) = 0 \) packet. In other words, the source does not transmit anything, and hence the receiver is unaware of the transmission.

If \( C \) is not a perfect code, then this issue can be solved by transmitting \( xG + b \) instead, where \( b \) is not a codeword (or, without loss of generality, \( b \) is a coset leader) at distance greater than \( t = \left\lceil \frac{d-2}{2} \right\rceil \) from the code. We then have \( w_t(xG+b+a) > 0 \) for any \( x \in GF(2)^K \) and any \( a \) with weight no more than \( t \), and hence the messages always have a positive number of packets. Note that this new encoding is equivalent to using the code \( C+b \), which has the same minimum distance and the same distance distribution as \( C \). On the other hand, if \( C \) is a perfect code, then no such vector \( b \) exists. However, we remark that perfect linear binary codes exist for limited sets of parameters only [14, Theorem 11.2.2], and hence most of the codes that would be used in our setting are not perfect. Also, if such a code should be used, then selecting a translate vector \( b \) with Hamming weight \( t \) may reduce the probability of obtaining an empty message at the receiver’s end. Finally, if a large perfect code is used, then removing one codeword
from the codebook would make it non-perfect with only a slight decrease in rate.

We now investigate some of the desirable properties of the translate vector $b$ and how to determine it once the property is fixed. Note that for any linear code $C$ and any coordinate $0 \leq i \leq N - 1$, then either all codewords of $C$ have a 0 in coordinate $i$ or half have a 0 and the other half have a 1. In the first case, the coordinate $i$ is not coded and hence need not be transmitted. Therefore, without loss of generality, we assume the code $C$ satisfies the following property: for all coordinates $0 \leq i \leq N - 1$, half the codewords in $C$ have a 0 in coordinate $i$ and half have a 1. Also, we assume that $C$ is not a perfect code, and we denote the set of coset leaders of $C$ with weight greater than $t$ as $B(C)$. Since the distribution of the number of packets received at the receiver’s end depends on the distribution of the error patterns and hence on the channel, we only consider the distribution of the number of packets in a message sent by the source. This assumption is consistent with our choice of studying constant-weight codes in Section III, which is equivalent to considering a constant number of packets in the messages sent by the source.

First, we remark that the average number of packets in a message, given by $\mu(b) = \frac{1}{|C|} \sum_{c \in C} w_H(c + b)$, is independent of $b$ and is hence denoted by $\mu$. Denoting the weight distribution of $C$ as $A_i$ for $0 \leq i \leq N$, we have $\mu = \sum_{i=0}^{N} i A_i$.

Second, one of the main advantages of using constant-weight codes is the constant number of packets in all messages. In other words, the standard deviation of the number of packets in a message is equal to zero. Accordingly, the translate vector $b$ may be chosen as to minimize the standard deviation of the number of packets in a message, so that the receiver usually expects a number of packets that is near the average. The minimum standard deviation of the number of packets over all possible choices for $b$ is given by $\sigma = \min_{b \in B(C)} \sigma(b)$, where

$$\sigma(b) = \sqrt{\frac{1}{|C|} \sum_{c \in C} (w_H(c + b) - \mu)^2}. \quad (17)$$

Proposition 5 below gives bounds on $\sigma$.

**Proposition 5:** The minimum standard deviation $\sigma$ of a linear code $C \subseteq GF(2)^N$ with error correction capability $t$ satisfies

$$\sqrt{(t+1)^2 - \frac{\mu^2}{|C|}} \leq \sigma \leq \max\{\mu - t + 1, N - \mu\}. \quad (18)$$

**Proof:** First, for any $c \in C$ and $b \in B(C)$, we have $t + 1 \leq w_H(c + b) \leq N$, and hence $(w_H(c + b) - \mu)^2 \leq (\max\{\mu - t + 1, N - \mu\})^2$. Combining with (17), we obtain the upper bound in (18). Second, we have

$$\sigma = \sqrt{\frac{\Sigma - \mu^2}{|C|}}, \quad (19)$$

where

$$\Sigma = \min_{b \in B(C)} \sum_{c \in C} w_H(c + b)^2. \quad (20)$$

Since $w_H(c + b) \geq t + 1$ for any $c \in C$ and $b \in B(C)$, we have $\Sigma \geq (t+1)^2|C|$ by (20). Combining with (19), we obtain the lower bound in (18).

We remark that by (20), the value of $\Sigma$ (and by (19), the value of $\sigma$) can be determined by solving the following binary linear program. Denote the elements of $B(C)$ as $b_i$ for $0 \leq i \leq |B(C)| - 1$ and consider the vector $s = (\sigma(b_0), \sigma(b_1), \ldots, \sigma(b_{|B(C)|-1}))$.

$$\min \ s \cdot x \quad (21)$$

subject to $\ 1 \cdot x = 1 \quad (22)$

$x \in \{0, 1\}^{|B(C)|}. \quad (23)$

According to Chebyshev’s inequality [15], minimizing the standard deviation of the number of packets minimizes the probability that the messages have a large number of packets. Such restriction on the size of large messages can be made more strict by finally choosing $b$ to minimize the maximum number of packets in a message. This choice is equivalent to determining $m = \min_{b \in B(C)} \max_{c \in C} w_H(c + b)$. By definitions of $\mu$ and $m$, we easily obtain that $m \geq \mu$. We remark that $m$ can also be viewed as the optimal solution of a linear program similar to (21)-(23).

**REFERENCES**


