# On the Stopping Distance and the Stopping Redundancy of Codes

**Moshe Schwartz**
University of California San Diego
La Jolla, CA 92093, U.S.A.
*moosh@everest.ucsd.edu*

**Alexander Vardy**
University of California San Diego
La Jolla, CA 92093, U.S.A.
*vardy@kilimanjaro.ucsd.edu*

*Abstract*— **It is now well known that the performance of a linear code $\mathbb{C}$ under iterative decoding on a binary erasure channel (and other channels) is determined by the size of the smallest stopping set in the Tanner graph for $\mathbb{C}$. Several recent papers refer to this parameter as the *stopping distance* $s$ of $\mathbb{C}$. This is somewhat of a misnomer since the size of the smallest stopping set in the Tanner graph for $\mathbb{C}$ depends on the corresponding choice of a parity-check matrix. It is easy to see that $s \leqslant d$, where $d$ is the minimum Hamming distance of $\mathbb{C}$, and we show that it is always possible to choose a parity-check matrix for $\mathbb{C}$ (with sufficiently many dependent rows) such that $s = d$. We thus introduce a new parameter, termed the *stopping redundancy* of $\mathbb{C}$, defined as the minimum number of rows in a parity-check matrix $H$ for $\mathbb{C}$ such that the corresponding stopping distance $s(H)$ attains its largest possible value, namely $s(H) = d$. We then derive general bounds on the stopping redundancy of linear codes. We also examine several simple ways of constructing codes from other codes, and study the effect of these constructions on the stopping redundancy. Specifically, for the family of binary Reed-Muller codes (of all orders), we prove that their stopping redundancy is at most a constant times their conventional redundancy. We show that the stopping redundancies of the binary and ternary extended Golay codes are at most 34 and 22, respectively. Finally, we provide upper and lower bounds on the stopping redundancy of MDS codes.**

## I. Introduction

The recent surge of of renewed interest in the binary erasure channel (BEC) is due in large part to the fact that it is the prime example of a channel over which the performance of iterative decoding algorithms can be analyzed precisely. In particular, it was shown in [3] that the performance of an LDPC code (and, in fact, any linear code) under iterative decoding on the BEC is completely determined by certain combinatorial structures called *stopping sets*. A stopping set $\mathcal{S}$ in a code $\mathbb{C}$ is a subset of the variable nodes in a Tanner graph for $\mathbb{C}$ such that all the neighbors of $\mathcal{S}$ are connected to $\mathcal{S}$ at least twice. The size $s$ of the smallest stopping set was termed the *stopping distance* of $\mathbb{C}$ in a number of recent papers [5], [7]. The stopping distance plays an important role in understanding the performance of a code under iterative decoding over the BEC, akin to the role played by the minimum Hamming distance $d$ for maximum-likelihood decoding. Just as one would like to maximize the minimum distance $d$ if maximum-likelihood or algebraic decoding is to be used, so one should try to maximize the stopping distance $s$ in the case of iterative decoding.

There is, however, an important difference between the minimal distance $d$ and the stopping distance $s$. While the former is a property of a code $\mathbb{C}$, the latter depends on the specific Tanner graph for $\mathbb{C}$ or, equivalently, on the specific choice of a parity-check matrix $H$ for $\mathbb{C}$. In order to emphasize this, we will henceforth use $s(H)$ to denote the stopping distance and $d(\mathbb{C})$ to denote the minimum distance.

In algebraic coding theory, a parity-check matrix $H$ for a linear code $\mathbb{C}$ usually has $n - \dim(\mathbb{C})$ linearly independent rows. However, in the context of iterative decoding, it has been already observed in [8], [10] and other papers that adding linearly dependent rows to $H$ can be advantageous. Certainly, this can increase the stopping distance $s(H)$. Thus, throughout this paper, a *parity-check matrix* for $\mathbb{C}$ should be understood as any matrix $H$ whose rows span the dual code $\mathbb{C}^{\perp}$. Then the *redundancy* $r(\mathbb{C})$ of $\mathbb{C}$ may be defined as the minimum number of rows in a parity-check matrix for $\mathbb{C}$. Analogously, we define the *stopping redundancy* $\rho(\mathbb{C})$ of $\mathbb{C}$ as the minimum number of rows in a parity-check matrix $H$ for $\mathbb{C}$ such that $s(H) = d(\mathbb{C})$. This work may be thought of as the first investigation of the trade-off between the parameters $\rho(\mathbb{C})$, $r(\mathbb{C})$, and $d(\mathbb{C})$.

In the next section, we first show that the stopping redundancy $\rho(\mathbb{C})$ is well-defined. That is, given any linear code $\mathbb{C}$, it is always possible to find a parity-check matrix $H$ for $\mathbb{C}$ such that $s(H) = d(\mathbb{C})$. In fact, the parity-check matrix consisting of *all* the nonzero codewords of the dual code $\mathbb{C}^{\perp}$ has this property. Hence $\rho(\mathbb{C}) \leqslant 2^{r(\mathbb{C})} - 1$ for all binary linear codes. We then show that if $d(\mathbb{C}) \leqslant 3$, then *any* parity-check matrix $H$ for $\mathbb{C}$ satisfies $s(H) = d(\mathbb{C})$, so $\rho(\mathbb{C}) = r(\mathbb{C})$ in this case. The main result of Section II is an extension of this simple observation to a general upper bound on the stopping redundancy of linear codes (Theorem 4).

In Section III, we study several simple ways of constructing codes from other codes, such as the direct-sum construction and code extension by adding an overall parity-check. We investigate the effect of these constructions on the stopping redundancy. Note that although we have limited our discussion to binary codes, most of the results in Sections II and III extend straightforwardly to linear codes over an arbitrary finite field.

We continue in Section IV with an in-depth analysis of the well-known $(u, u+v)$ construction, and in particular its application in the recursive definition [6, p. 374] of binary Reed-Muller codes. By slightly modifying this construction, we establish a strong upper bound on the stopping redundancy of Reed-Muller codes of arbitrary orders. Specifically, we prove that if $\mathbb{C}$ is a Reed-Muller code of length $2^m$ and order $r$, then $\rho(\mathbb{C}) \leqslant d(\mathbb{C})r(\mathbb{C})/2$. Thus for any constant $d(\mathbb{C})$, we have an increase in redundancy by only a constant factor.

In Section V, we study the $(24, 12, 8)$ extended binary Golay code $\mathcal{G}_{24}$. We prove that $\rho(\mathcal{G}_{24}) \leqslant 34$ by providing specific parity-check matrices for this code. We take $\mathcal{G}_{24}$ as a test case, and compare the performance of three different decoders: a maximum-likelihood decoder, an iterative decoder using the conventional $12 \times 24$ double-circulant parity-check matrix of [6, p.65], and an iterative decoder using the $34 \times 24$ parity-check matrix with maximum stopping distance. In each case, exact analytic expressions for the probability of decoding failure are derived using a computer program (see Figure 1).

We conclude in Section VI with a brief discussion and a list of open problems. Some of our results on stopping redundancy of linear codes that are not included here due to space limitations are also briefly mentioned in Section VI.

## II. General Bounds

We begin with rigorous definitions of the stopping distance and the stopping redundancy. Let $\mathbb{C}$ be a binary linear code and let $H = [h_{i,j}]$ be a parity-check matrix for $\mathbb{C}$. The corresponding Tanner graph $T$ for $\mathbb{C}$ is a bipartite graph with each column of $H$ represented by a *variable node* and each row of $H$ represented by a *check node* in such a way that the $j$-th variable node is connected to the $i$-th check node if and only if $h_{i,j} \neq 0$. As already mentioned, a stopping set in $T$ is a subset $\mathcal{S}$ of the variable nodes such that all the check nodes that are neighbors of a node in $\mathcal{S}$ are connected to *at least two nodes* in $\mathcal{S}$. We dispense with this graphical representation of stopping sets in favor of an equivalent definition directly in terms of the underlying parity-check matrix $H$. Thus we say that a *stopping set* is a set of columns of $H$ with the property that the projection of $H$ onto these columns does not contain a row of weight one. The resulting definition of the stopping distance – the smallest size of a stopping set – bears a striking resemblance to the definition of the minimum Hamming distance of a linear code.

Recall that the minimum distance of a linear code $\mathbb{C}$ can be defined as the largest integer $d(\mathbb{C})$ such that every $d(\mathbb{C}) - 1$ or less columns of $H$ are linearly independent. For binary codes, this is equivalent to saying that $d(\mathbb{C})$ is the largest integer such that every set of $d(\mathbb{C}) - 1$ or less columns of $H$ contains at least one row of odd weight.

**Definition 1.** *Let $\mathbb{C}$ be a linear code and let $H$ be a parity-check matrix for $\mathbb{C}$. Then the* ***stopping distance*** *of $H$ is defined as the the largest integer $s(H)$ such that every set of $s(H) - 1$ or less columns of $H$ contains at least one row of weight one.*

The following corollary is an immediate consequence of juxtaposing the definitions of $s(H)$ and $d(\mathbb{C})$ above.

**Corollary 1.** *Let $\mathbb{C}$ be a linear code and let $H$ be an arbitrary parity-check matrix for $\mathbb{C}$. Then $s(H) \leqslant d(\mathbb{C})$.*

Indeed, it is well known [3], [4], [5] that the support of every codeword is a stopping set, which is another way to see that $s(H) \leqslant d(\mathbb{C})$ regardless of the choice of $H$. Thus given a linear code $\mathbb{C}$, the largest stopping distance one could hope for is $d(\mathbb{C})$, no matter how cleverly the Tanner graph for $\mathbb{C}$ is constructed. The point is that this bound can be *always* achieved by adding dependent rows to $H$ (see Theorem 2). This makes the notion of the stopping distance, as a property of a code $\mathbb{C}$, somewhat meaningless: without restricting the number of rows in a parity-check matrix for $\mathbb{C}$, we cannot distinguish between the stopping distance and the conventional minimum distance. This observation, in turn, leads to the following definition.

**Definition 2.** *Let $\mathbb{C}$ be a linear code with minimum Hamming distance $d(\mathbb{C})$. Then the* ***stopping redundancy*** *of $\mathbb{C}$ is defined as the the smallest integer $\rho(\mathbb{C})$ such that there exists a parity-check matrix $H$ for $\mathbb{C}$ with $\rho(\mathbb{C})$ rows and $s(H) = d(\mathbb{C})$.*

The following theorem shows that the stopping redundancy is, indeed, well-defined.

**Theorem 2.** *Let $\mathbb{C}$ be a linear code, and let $H^*$ denote the parity-check matrix for $\mathbb{C}$ consisting of all the nonzero codewords of the dual code $\mathbb{C}^\perp$. Then $s(H^*) = d(\mathbb{C})$.*

*Proof:* Let $[\mathbb{C}^\perp]$ denote the $n \times |\mathbb{C}^\perp|$ matrix consisting of all the codewords of $\mathbb{C}^\perp$. It is well known (cf. [6, p.139]) that $[\mathbb{C}^\perp]$ is an orthogonal array of strength $d(\mathbb{C}) - 1$. This means that any set of $t \leqslant d(\mathbb{C}) - 1$ columns of $[\mathbb{C}^\perp]$ contains all the vectors of length $t$ among its rows, each vector appearing the same number of times. In particular, any set of $d(\mathbb{C}) - 1$ or less columns of $[\mathbb{C}^\perp]$ contains all the vectors of weight one. ■

Theorem 2 also provides a trivial upper bound on the stopping redundancy. In particular, it follows from Theorem 2 that $\rho(\mathbb{C}) \leqslant 2^{r(\mathbb{C})} - 1$ for any binary linear code $\mathbb{C}$. This bound holds with equality in the degenerate case of the single-parity-check code. The next theorem determines $\rho(\mathbb{C})$ exactly for *all* binary linear codes with minimum distance $d(\mathbb{C}) \leqslant 3$.

**Theorem 3.** *Let $\mathbb{C}$ be a binary linear code with minimum distance $d(\mathbb{C}) \leqslant 3$. Then* ***any*** *parity-check matrix $H$ for $\mathbb{C}$ satisfies $s(H) = d(\mathbb{C})$, and therefore $\rho(\mathbb{C}) = r(\mathbb{C})$.*

*Proof:* If $H$ contains an all-zero column, then it is obvious that $s(H) = d(\mathbb{C}) = 1$. Otherwise $s(H) \geqslant 2$, since then every single column of $H$ must contain a row of weight one. Now, if $d(\mathbb{C}) = 3$, then every two columns of $H$ are distinct. This implies that these two columns must contain either the 01 row or the 10 row (or both). Hence $s(H) = 3$. ■

The following theorem, which is our main result in this section, shows that Theorem 3 is, in fact, a special case of a general lower bound on the stopping redundancy of linear codes.

**Theorem 4.** *Let $\mathbb{C}$ be a binary linear code with minimum distance $d(\mathbb{C}) \geqslant 3$. Then*

$$\rho(\mathbb{C}) \leqslant \binom{r(\mathbb{C})}{1} + \binom{r(\mathbb{C})}{2} + \cdots + \binom{r(\mathbb{C})}{d(\mathbb{C}) - 2} \qquad (1)$$

*Proof:* We first prove a slightly weaker result, which is conceptually simpler. Namely, let us show that

$$\rho(\mathbb{C}) \leqslant \binom{r(\mathbb{C})}{1} + \binom{r(\mathbb{C})}{2} + \cdots + \binom{r(\mathbb{C})}{d(\mathbb{C}) - 1} \qquad (2)$$

Let $H$ be an arbitrary parity-check matrix for $\mathbb{C}$ with $r(\mathbb{C})$ linearly independent rows. Construct another parity-check matrix $H'$ whose rows are all the linear combinations of $t$ rows of $H$, for all $t = 1, 2, \dots, d(\mathbb{C}) - 1$. Clearly, the number of rows of $H'$ is given by the right-hand side of (2). Now let $H_t$, respectively $H'_t$, denote a matrix consisting of some $t$ columns of $H$, respectively the corresponding $t$ columns of $H'$. Observe that for all $t \leqslant d(\mathbb{C}) - 1$, the $t$ columns of $H_t$ are linearly independent. This implies that the row-rank of $H_t$ is $t$, and therefore some $t$ rows of $H_t$ must form a basis for $\mathbb{F}_2^t$. Hence the $2^t - 1$ nonzero linear combinations of these $t$ rows of $H_t$ generate all the nonzero vectors in $\mathbb{F}_2^t$, including all the vectors of weight one. But for $t \leqslant d(\mathbb{C}) - 1$, the $2^t - 1$ nonzero linear combinations of *any* $t$ rows of $H_t$ are among the rows of $H'_t$ by construction. This proves that $s(H') = d(\mathbb{C})$ and establishes (2).

To transition from (2) to (1), observe that we do not need to have all the nonzero vectors of $\mathbb{F}_2^t$ among the rows of $H'_t$; it would suffice to have at least one vector of weight one. Given a set $\mathcal{S} \subseteq \mathbb{F}_2^t$ and a positive integer $m$, let $m\mathcal{S}$ denote the set of all vectors obtained as a linear combination of at most $m$ vectors from $\mathcal{S}$. Define $\mu(t)$ as the smallest integer with the property that for any basis $B$ of $\mathbb{F}_2^t$, the set $\mu(t)B$ contains at least one vector of weight one. Then in the construction of $H'$, it would suffice to take all the linear combinations of at most $\mu(d(\mathbb{C}) - 1)$ rows of $H$. Clearly $\mu(t) \leqslant t - 1$ for all $t$ (in fact, $\mu(t) = t - 1$ for all $t$), and the theorem follows. ■

The bound of (1), while much better than $\rho(\mathbb{C}) \leqslant 2^{r(\mathbb{C})} - 1$, is still too general to be tight for most codes. Nevertheless, we can conclude from Theorem 4 that when $d(\mathbb{C})$ is a constant, the stopping redundancy is only polynomial in the (conventional) redundancy and, hence, in the length of the code.

An obvious question is whether we can do substantially better than Theorem 4. At least in the case of Reed-Muller codes, we shall see in Section IV that the answer is yes.

## III. Constructions of Codes from Other Codes

In this section, we examine several simple ways of constructing codes from other codes. While for most such constructions, it is trivial to determine the redundancy of the resulting code, we find it considerably more difficult to determine the resulting *stopping redundancy*, and resort to bounding it.

We start with two simple examples. The first example (Theorem 5) is the well-known direct-sum construction or, equivalently, the $(u, v)$ construction. The second one (Theorem 6) is the $(u, u)$ construction, or concatenation of a code with itself. Both theorems have simple constructive proofs which we omit.

**Theorem 5.** *Let $\mathbb{C}_1, \mathbb{C}_2$ be $(n_1, k_1, d_1), (n_2, k_2, d_2)$ binary linear codes, respectively. Then $\mathbb{C}_3 = \{(u, v) : u \in \mathbb{C}_1, v \in \mathbb{C}_2\}$ is an $(n_1 + n_2, k_1 + k_2, \min\{d_1, d_2\})$ code with*

$$\rho(\mathbb{C}_3) \leqslant \rho(\mathbb{C}_1) + \rho(\mathbb{C}_2) \tag{3}$$

**Theorem 6.** *Let $\mathbb{C}_1$ be an $(n, k, d)$ binary linear code. Then the code $\mathbb{C}_2 = \{(u, u) : u \in \mathbb{C}_1\}$ is a $(2n, k, 2d)$ code with*

$$\rho(\mathbb{C}_2) \leqslant \rho(\mathbb{C}_1) + n \tag{4}$$

Here is an interesting observation about Theorems 5 and 6. It follows from (3) and (4) that if the constituent codes are optimal, in the sense that their stopping redundancy is equal to their redundancy, then the resulting code is also optimal. This indicates that the bounds in (3) and (4) are tight.

In contrast, the innocuous construction of *extending* a linear code $\mathbb{C}$ by adding an overall parity-check [6, p.27] appears to be much more difficult to handle. The next theorem deals only with the special case where $d(\mathbb{C}) = 3$.

**Theorem 7.** *Let $\mathbb{C}$ be an $(n, k, 3)$ binary linear code. Then the extended code $\mathbb{C}'$ is an $(n + 1, k, 4)$ code with*

$$\rho(\mathbb{C}') \leqslant 2\rho(\mathbb{C}) = 2r(\mathbb{C}') - 2 \tag{5}$$

*Proof:* Let $H$ be an arbitrary $r(\mathbb{C}) \times n$ parity-check matrix for $\mathbb{C}$. We construct a parity-check matrix for $\mathbb{C}'$ as follows

$$H' = \begin{pmatrix} H & \mathbf{0} \\ \overline{H} & \mathbf{1} \end{pmatrix} \tag{6}$$

where $\overline{H}$ is the bitwise complement of $H$, while $\mathbf{0}$ and $\mathbf{1}$ are the all-zero and the all-one column vectors, respectively. Label the columns in $H'$ by $1, 2, \dots, n + 1$, and let $\mathcal{I}$ be a subset of $\{1, 2, \dots, n + 1\}$ with $|\mathcal{I}| \leqslant 3$. In fact, it would suffice to consider the case where $\mathcal{I} \subset \{1, 2, \dots, n\}$ and $|\mathcal{I}| = 3$; all other cases easily follow from the fact that $s(H) = 3$ by Theorem 3.

Let $H(\mathcal{I})$ and $\overline{H}(\mathcal{I})$ denote the projections of $H$ and $\overline{H}$, respectively, on the three positions in $\mathcal{I}$. If $H(\mathcal{I})$ contains a row of weight one, we are done. If $H(\mathcal{I})$ contains a row of weight two, we are also done — then the corresponding row in $\overline{H}(\mathcal{I})$ has weight one. But otherwise, the only rows in $H(\mathcal{I})$ are 000 and 111, which means that the three columns in $H(\mathcal{I})$ are identical, a contradiction since $d(\mathbb{C}) = 3$. ■

The construction in (5) and (6) is not optimal. For example, if $\mathbb{C}'$ is the $(8, 4, 4)$ extended Hamming code, it produces a parity-check matrix for $\mathbb{C}'$ with 6 rows. However, $\mathbb{C}'$ is also the Reed-Muller code $\mathcal{R}(1, 3)$ for which we give in the next section a parity-check matrix $H$ with $s(H) = 4$ and only 5 rows.

## IV. Reed-Muller Codes

We now focus on the well-known $(u, u + v)$ construction, in particular in connection with the recursive definition of binary Reed-Muller codes. Our goal is to derive a constructive upper bound on the stopping redundancy of $\mathcal{R}(r, m)$ — the binary Reed-Muller code of order $r$ and length $2^m$.

We begin by recalling several well-known facts. First, for all $r = 0, 1, \dots, m$, the dimension of $\mathcal{R}(r, m)$ is $k = \sum_{i=0}^{r} \binom{m}{i}$ and its minimum distance is $d = 2^{m-r}$. Let $G(r, m)$ be a generator matrix for $\mathcal{R}(r, m)$. Then, using the $(u, u + v)$ construction, $G(r, m)$ can be defined recursively, as follows:

$$G(r, m) \overset{\text{def}}{=} \begin{pmatrix} G(r, m-1) & G(r, m-1) \\ \mathbf{0} & G(r-1, m-1) \end{pmatrix} \tag{7}$$

with the recursion in (7) being bootstrapped by $G(m, m) = I_{2^m}$ and $G(0, m) = (11 \cdots 1)$ for all $m$. By convention, the code $\mathcal{R}(-1, m)$ is the set $\{\mathbf{0}\}$ for all $m$. Then

$$\mathcal{R}(r, m)^{\perp} = \mathcal{R}(m - r - 1, m) \tag{8}$$

for all $m$ and all $r = -1, 0, 1, \dots, m$. It follows from (8) that $G(r, m)$ is a parity-check matrix for $\mathcal{R}(m-r-1, m)$, a code with minimum distance $2^{r+1}$. Hence every $2^{r+1} - 1$ columns of $G(r, m)$ are linearly independent.

Our objective in what follows is to construct an alternative parity-check matrix $H(r,m)$ for $\mathcal{R}(m-r-1,m) = \mathcal{R}(r,m)^{\perp}$ such that $s(H(r,m)) = 2^{r+1}$. Then the number of rows in $H(r,m)$ gives an upper bound on the stopping redundancy of $\mathcal{R}(m-r-1,m)$. Here is the recursive construction that we use.

**Recursive Construction A:** For all positive integers $m$ and for all $r = 1,2,\ldots,m-2$, we define:

$$H(r,m) = \left(\begin{array}{c} H_{\text{top}} \\ \hline H_{\text{bot}} \end{array}\right) \overset{\text{def}}{=} \left(\begin{array}{cc} H(r,m-1) & H(r,m-1) \\ \mathbf{0} & H(r-1,m-1) \\ \hline H(r-1,m-1) & \mathbf{0} \end{array}\right) \tag{9}$$

with the recursion in (9) being bootstrapped as follows: for all $m = 0,1,\ldots$, the matrices $H(0,m)$, $H(m-1,m)$, $H(m,m)$ are defined by

$$H(0,m) \overset{\text{def}}{=} G(0,m) = (11\cdots1) \tag{10}$$

$$H(m-1,m) \overset{\text{def}}{=} G(m-1,m) \tag{11}$$

$$H(m,m) \overset{\text{def}}{=} G(m,m) = I_{2^m} \tag{12}$$

We omit the proofs of the next two propositions and lemma.

**Proposition 8.** *$H(r,m)$ is a generator matrix for $\mathcal{R}(r,m)$ and, hence, a parity-check matrix for $\mathcal{R}(m-r-1,m)$.*

**Proposition 9.** *The stopping distance of $H(r,m)$ is $2^{r+1}$ for all positive integers $m$ and for all $r = 0,1,\ldots,m-1$,*

The remaining task is to compute the number of rows in the matrix $H(r,m)$. We denote this number as $g(r,m)$.

**Lemma 10.** *For all $r = 0,1,\ldots,m-1$, the number of rows in $H(r,m)$ is given by*

$$g(r,m) = \sum_{i=0}^{r} \binom{m-r-1+i}{i} 2^i$$

We are now in a position to summarize the results of this section in the following theorem.

**Theorem 11.** *For all $m = 1,2,\ldots$ and for all $r = 0,1,\ldots,m$, the stopping redundancy of $\mathcal{R}(r,m)$ is upper bounded by*

$$\rho\big(\mathcal{R}(r,m)\big) \leqslant \sum_{i=0}^{m-r-1} \binom{r+i}{i} 2^i \tag{13}$$

*Proof:* Follows immediately from (8), Proposition 8, Proposition 9, and Lemma 10. ■

To see how far Theorem 11 is from the (conventional) redundancy of Reed-Muller codes, let us make a simple calculation. For this, it will be more convenient to work with the dual code $\mathbb{C} = \mathcal{R}(r,m)^{\perp}$. Recall that the redundancy of $\mathbb{C}$ is $\sum_{i=0}^{r}\binom{m}{i}$. Comparing this to the bound on $\rho(\mathbb{C})$ in (13), we find that

$$\rho(\mathbb{C}) \leqslant \sum_{i=0}^{r} \binom{m-r-1+i}{i} 2^i \leqslant 2^r \sum_{i=0}^{r} \binom{m}{i} = 2^r r(\mathbb{C})$$

Therefore, for any fixed order $r$, the stopping redundancy of $\mathcal{R}(r,m)^{\perp}$ is at most the redundancy of $\mathcal{R}(r,m)^{\perp}$ times a constant. Alternatively, if we take $\mathbb{C} = \mathcal{R}(r,m)$, then Theorem 11 implies that $\rho(\mathbb{C}) \leqslant d(\mathbb{C})r(\mathbb{C})/2$. Thus for any fixed $d(\mathbb{C})$, the increase in redundancy is by a constant factor.

## V. Golay Codes

The $(24,12,8)$ binary Golay code $\mathcal{G}_{24}$ is arguably the most remarkable binary block code. It is often used as a benchmark in studies of code structure and decoding algorithms.

There are several "canonical" parity-check matrices for $\mathcal{G}_{24}$, see [1], [2], [9] and other papers. Our starting point is the systematic double-circulant matrix $H_{24}$ given in MacWilliams and Sloane [6, p.65] and shown in Table I. It can be readily verified that $s\big(H_{24}\big) = 4$, which means that $H_{24}$ achieves only half of the maximum possible stopping distance. Curiously, the stopping distance of the two "trellis-oriented" parity-check matrices for $\mathcal{G}_{24}$, given in [9, p. 2060] and [1, p.1441], is also 4.

Computing the bound of Theorem 4 for the special case of $\mathcal{G}_{24}$ produces the rather weak result: $\rho\big(\mathcal{G}_{24}\big) \leqslant 2509$. Having tried several methods to construct a parity-check matrix for $\mathcal{G}_{24}$ with stopping distance 8, our best result was achieved using a greedy (lexicographic) computer search. Specifically, with the 4095 nonzero vectors of $\mathcal{G}_{24}$ listed lexicographically, we iteratively construct the parity-check matrix $H'_{24}$, at each iteration adjoining to $H'_{24}$ the first vector on the list with the highest score. Each vector receives $i$ points to its score for each yet uncovered $i$-set it covers, where $i \in \{1,2,\ldots,7\}$. The resulting matrix is given in Table I.

TABLE I

TWO PARITY-CHECK MATRICES FOR THE $(24,12,8)$ GOLAY CODE $\mathcal{G}_{24}$

$$H_{24} = \begin{pmatrix}
1&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1&1&0&0&0&1&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1&1&0&0&0&1\\
1&0&0&1&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1&0&0\\
1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1&0\\
1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1\\
1&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&1&0&1&1&0&1&1\\
1&0&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&0&1&0&1&1&0&1\\
1&0&0&0&0&0&0&0&0&1&0&0&0&1&1&1&0&0&0&1&0&1&1&0\\
1&0&0&0&0&0&0&0&0&0&1&0&0&0&1&1&1&0&0&0&1&0&1&1\\
1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&1&1&1&1
\end{pmatrix}$$

$$H'_{24} = \begin{pmatrix}
0&0&0&0&0&0&0&0&0&0&1&1&0&1&1&0&0&1&0&0&1&1&1&0\\
0&0&0&0&0&0&0&0&0&0&1&1&1&0&0&1&1&0&1&1&0&0&0&1\\
0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&1&0&0&1&1&1&0&1&1\\
0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&1&1&0&0&0&1&0&0&1\\
0&0&0&0&0&0&1&0&0&1&1&1&1&1&1&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&1&1&0&0&1&1&1&0&0&0&0&0&0\\
0&0&0&0&1&0&1&1&0&1&0&0&0&0&0&1&0&0&1&0&0&1&1&0\\
0&0&0&0&1&1&1&0&1&1&0&1&1&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0&1&0&0&0&1&0&1&0&0&0&0&1\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&1&1&1&1&0&0&0&1&1\\
0&0&0&1&1&0&0&0&0&0&0&0&0&1&1&1&0&1&1&0&0&1&0&0\\
0&0&0&1&1&0&1&1&1&0&0&1&0&0&0&0&0&1&1&0&0&0&0&0\\
0&0&0&1&1&1&0&0&0&0&0&1&1&1&0&0&0&0&1&1&0&0&0&0\\
0&0&1&0&0&0&1&1&0&1&0&0&0&0&0&0&1&0&0&0&1&0&1&1\\
0&0&1&0&1&1&1&1&1&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&1&1&1&0&0&1&0&0&1&1&0&1&1&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&1&0&1&0&1&0&1&0&0&1&0&1&1&0\\
0&1&0&0&0&0&0&0&1&0&1&1&0&1&1&0&1&0&0&0&0&0&0&1\\
0&1&0&0&0&1&0&0&1&0&1&0&0&0&1&0&0&0&1&0&1&0&1&0\\
0&1&0&0&0&1&1&1&1&0&0&1&0&0&0&0&1&0&0&0&1&0&0&0\\
0&1&1&0&0&1&0&0&1&0&0&0&0&0&1&0&1&1&0&1&0&0&0&0\\
0&1&1&1&0&0&0&1&0&0&0&0&0&1&0&0&0&1&1&1&0&0&0&0\\
1&0&0&0&0&0&1&0&0&0&1&1&1&0&0&1&0&0&0&0&0&1&1&0\\
1&0&0&0&0&1&0&0&0&0&1&1&0&1&1&1&0&0&1&0&0&0&0&0\\
1&0&0&1&1&0&0&0&1&0&0&0&0&0&0&1&0&1&0&0&1&0&0&1\\
1&0&1&0&0&0&0&0&0&1&1&0&1&0&0&0&0&0&0&1&0&0&1&1\\
1&0&1&1&0&0&0&0&0&0&0&1&0&0&1&1&0&0&0&0&1&1&0&0\\
1&0&1&1&0&1&0&0&0&0&0&0&1&0&0&0&0&1&0&1&1&0&0&0\\
1&0&1&1&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&1&0&1\\
1&1&0&0&0&1&1&0&0&0&0&0&1&0&0&0&1&1&0&0&0&1&0&0\\
1&1&0&0&1&0&1&0&0&1&1&0&0&0&0&0&0&0&0&1&0&1&0&0\\
1&1&0&1&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&1&1&0&0\\
1&1&1&0&1&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
1&1&1&1&0&0&0&0&1&1&0&0&0&0&0&1&0&0&0&1&0&0&0&0
\end{pmatrix}$$

To evaluate the effect of increasing the stopping distance, it would be interesting to compare the performance of iterative decoders for $\mathcal{G}_{24}$ based on $H_{24}$ or $H'_{24}$, respectively. As a baseline for such a comparison, it would be also useful to have the performance of a maximum-likelihood decoder for $\mathcal{G}_{24}$. In what follows, we give analytic expressions for the performance of the three decoders on the binary erasure channel (BEC).

Clearly, a maximum-likelihood decoder fails to decode (recover) a given erasure pattern if and only if this pattern contains the support of (at least one) nonzero codeword of $\mathcal{G}_{24}$.

TABLE II

NUMBER OF UNDECODABLE ERASURE PATTERNS BY WEIGHT $w$ IN THREE DECODERS FOR $\mathcal{G}_{24}$

| $w$ | Total Patterns | $\Psi_{\mathrm{ML}}(w)$ | $\Psi_{H_{24}}(w)$ | $\Psi_{H'_{24}}(w)$ |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 |
| 1 | 24 | 0 | 0 | 0 |
| 2 | 276 | 0 | 0 | 0 |
| 3 | 2024 | 0 | 0 | 0 |
| 4 | 10626 | 0 | 110 | 0 |
| 5 | 42504 | 0 | 2277 | 0 |
| 6 | 134596 | 0 | 19723 | 0 |
| 7 | 346104 | 0 | 100397 | 0 |
| 8 | 735471 | 759 | 343035 | 3598 |
| 9 | 1307504 | 12144 | 844459 | 82138 |
| 10 | 1961256 | 91080 | 1568875 | 585157 |
| 11 | 2496144 | 425040 | 2274130 | 1717082 |
| 12 | 2704156 | 1313116 | 2637506 | 2556402 |
| $\geqslant 13$ | $\binom{24}{w}$ | $\binom{24}{w}$ | $\binom{24}{w}$ | $\binom{24}{w}$ |

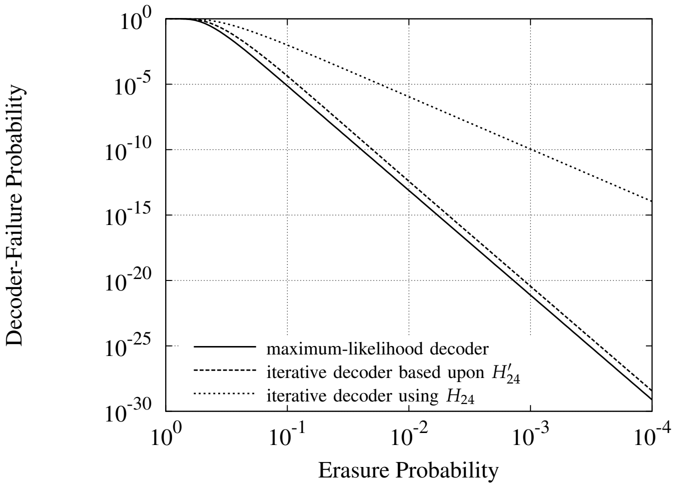

Fig. 1. The decoding failure probability of three decoders for $\mathcal{G}_{24}$: a maximum-likelihood decoder and iterative decoders based upon $H_{24}$ and $H'_{24}$

Let $\Psi_{\mathrm{ML}}(w)$ denote the number of such erasure patterns as a function of their weight $w$. Then

$$\Pr{}_{\mathrm{ML}}\{\text{decoding failure}\} \;=\; \sum_{w=0}^{24} \Psi_{\mathrm{ML}}(w)\, p^{w}(1-p)^{24-w}$$

where $p$ is the erasure probability of the BEC. In contrast, an iterative decoder (based on $H_{24}$ or $H'_{24}$) fails if and only if the erasure pattern contains a stopping set. Thus

$$\Pr{}_{H_{24}}\{\text{decoding failure}\} \;=\; \sum_{w=0}^{24} \Psi_{H_{24}}(w)\, p^{w}(1-p)^{24-w}$$

$$\Pr{}_{H'_{24}}\{\text{decoding failure}\} \;=\; \sum_{w=0}^{24} \Psi_{H'_{24}}(w)\, p^{w}(1-p)^{24-w}$$

where $\Psi_{H_{24}}(w)$ and $\Psi_{H'_{24}}(w)$ denote the number of erasure patterns of weight $w$ that contain a stopping set of $H_{24}$ and $H'_{24}$, respectively. It remains to compute $\Psi_{H_{24}}$, $\Psi_{H'_{24}}$, and $\Psi_{\mathrm{ML}}$.

Obviously, $\Psi_{\mathrm{ML}}(w) = 0$ for $w \leqslant 7$ and $\Psi_{\mathrm{ML}}(w) = \binom{24}{w}$ for $w \geqslant 13$ (any 13 columns of a parity-check matrix for $\mathcal{G}_{24}$ are linearly dependent). For the other values of $w$, we have

$$\Psi_{\mathrm{ML}}(w) \;=\; \begin{cases} \dbinom{16}{w-8}759 & 8 \leqslant w \leqslant 11 \\[2ex] 1771(20+720)+2576 & w = 12 \end{cases}$$

where we made use of Table IV of [2]. To find $\Psi_{H_{24}}(\cdot)$ and $\Psi_{H'_{24}}(\cdot)$, we used exhaustive computer search. These functions are given in Table II. The resulting probabilities of decoding failure are plotted in Figure 1. Note that while we may add rows to $H'_{24}$ to eliminate more stopping sets, this would have negligible effect since the slope of the performance curve is dominated by the smallest $w$ for which $\Psi_{H'_{24}}(w) \neq 0$.

## VI. FURTHER RESULTS AND OPEN PROBLEMS

This paper only scratches the surface of the many interesting and important problems that arise in the investigation of stopping redundancy. Here is a representative sample:

- Determine the stopping redundancy of well-known codes with substantial algebraic and/or combinatorial structure. In particular, is it true that $\rho(\mathcal{G}_{24}) = 34$? It appears that proving lower bounds on the stopping redundancy, even for specific codes such as $\mathcal{G}_{24}$, is quite difficult.
- Is the construction devised for binary Reed-Muller codes in Section IV optimal? More generally, for which families of codes can one find parity-check matrices with only $O\big(r(\mathbb{C})\big)$ rows and stopping distance equal to $d(\mathbb{C})$?
- Are there codes with non-vanishing rate and normalized distance, whose stopping redundancy is $O\big(r(\mathbb{C})\big)$? We can answer this with a no, in the case where the dual codes also have non-vanishing normalized distance.

## REFERENCES

[1] A.R. Calderbank, G.D. Forney, Jr., and A. Vardy, "Minimal tail-biting trellises: the Golay code and more," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1435–1455, July 1999.

[2] J.H. Conway and N.J.A. Sloane, "Orbit and coset analysis of the Golay and related codes," *IEEE Trans. Inform. Theory*, vol. 36, pp. 1038–1050, September 1990.

[3] C. Di, D. Proietti, I.E. Telatar, T.J. Richardson, and R. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inform. Theory*, vol. 48, pp. 1570–1579, Jun 2000.

[4] J. Feldman, *Decoding Error-Correcting Codes via Linear Programming* Ph.D. Thesis, Massachusetts Institute of Technology, September 2003.

[5] N. Kashyap and A. Vardy, "Stopping sets in codes from designs," in *Proc. IEEE Internat. Symp. Information Theory*, Yokohama, Japan, July 2003.

[6] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Co-des*. Amsterdam: North-Holland, 1978.

[7] A. Orlitsky, K. Viswananathan, and J. Zhang, "Stopping set distribution of LDPC code ensembles," *IEEE Trans. Inform. Theory*, vol. 51, pp. 929–953, March 2005.

[8] N. Santhi and A. Vardy, "On the effect of parity-check weights in iterative decoding," in *Proc. IEEE International Symposium Information Theory*, Chicago, IL., July 2004.

[9] A. Vardy, "Trellis structure of codes," Chapter 24 in the HANDBOOK OF CODING THEORY, V. Pless and W.C. Huffman (Editors), Elsevier, 1998.

[10] J.S. Yedidia, J. Chen, and M. Fossorier, "Generating code representations suitable for belief propagation decoding," in *Proc. 40-th Allerton Conference Commun., Control, and Computing*, Monticello, IL., October 2002.