

Error-Correcting Balanced Knuth Codes

Jos H. Weber, *Senior Member, IEEE*, Kees A. Schouhamer Immink, *Fellow, IEEE*, and Hendrik C. Ferreira, *Senior Member, IEEE*

Abstract—Knuth’s celebrated balancing method consists of inverting the first z bits in a binary information sequence, such that the resulting sequence has as many ones as zeroes, and communicating the index z to the receiver through a short balanced prefix. In the proposed method, Knuth’s scheme is extended with error-correcting capabilities, where it is allowed to give unequal protection levels to the prefix and the payload. The proposed scheme is very general in the sense that any error-correcting block code may be used for the protection of the payload. Analyses with respect to redundancy and block and bit error probabilities are performed, showing good results while maintaining the simplicity features of the original scheme. It is shown that the Hamming distance of the code is of minor importance with respect to the error probability.

I. INTRODUCTION

SETS of binary sequences that have a fixed length n and a fixed weight (number of ones) w are usually called *constant-weight codes*. An important sub-class is formed by the so-called *balanced codes*, for which n is even and $w = n/2$, i.e., all codewords have as many zeroes as ones. Such codes have found application in various transmission and (optical/magnetic) recording systems, e.g., in flash memories [11]. A survey on balanced codes can be found in [4].

A simple method for generating balanced codewords, which is capable of encoding and decoding (very) large blocks, was proposed by Knuth [6] in 1986. In his method, an m -bit binary data word, m even, is forwarded to the encoder. The encoder inverts the first z bits of the data word, where z is chosen in such a way that the modified word has equal numbers of zeroes and ones. Knuth showed that such an index z can always be found. The index z is represented by a balanced word of length p . The p -bit prefix word followed by the modified m -bit data word are both transmitted, so that the rate of the code is $m/(m+p)$. The receiver can easily undo the inversion of the first z bits received once z is computed from the prefix. Both encoder and decoder do not require large look-up tables, and Knuth’s algorithm is, therefore, very attractive for constructing long balanced codewords. The redundancy of Knuth’s method is roughly twice the redundancy of a code which uses the full set of balanced words. Since the latter has a prohibitively high complexity in case of large lengths, the factor of two can be considered as a price to

be paid for simplicity. In [5] and [10], modifications to Knuth’s method are presented closing this gap while maintaining sufficient simplicity.

Knuth’s method does not provide protection against errors which may occur during transmission or storage. Actually, errors in the prefix may lead to catastrophic error propagation in the data word. Here, we propose and analyze a method to extend Knuth’s original scheme with error correcting capabilities. Previous constructions for error-correcting balanced codes were given in [2], [9] and [8]. In [9], van Tilborg and Blaum introduced the idea to consider short balanced blocks as symbols of an alphabet and to construct error-correcting codes over that alphabet. Only moderate rates can be achieved by this method, but it has the advantage of limiting the digital sum variation and the runlengths. In [2], Al-Bassam and Bose constructed balanced codes correcting a single error, which can be extended to codes correcting up to two, three, or four errors by concatenation techniques. In [8], Mazumdar, Roth, and Vontobel considered linear balancing sets and applied such sets to obtain error-correcting coding schemes in which the codewords are balanced. In the method proposed in the current paper, we stay very close to the original Knuth algorithm. Hence, we only operate in the binary field and inherit the low-complexity features of Knuth’s method. In our method, the error-correcting capability can be any number. The focus will be on long codes, for which table look-up methods are unfeasible. An additional feature is the possibility to assign different error protection levels to the prefix and the payload, which will be shown to be useful when designing the scheme to achieve a certain required error performance while optimizing the rate. In this context, it turns out that the Hamming distance of the proposed code is of minor importance.

The rest of this paper is organized as follows. In Section II, the proposed method for providing balancing and error-correcting capabilities is presented. In Section III, the redundancy of the new scheme is considered. The block and bit error probabilities are analyzed in Section IV. Case studies are presented in Section V. Finally, the results of this paper are discussed in Section VI.

II. CONSTRUCTION METHOD

The proposed construction method is based on a combination of conventional error correction techniques and Knuth’s method for obtaining balanced words. The encoding procedure consists of four steps which are described below and illustrated in Fig. 1. The input to the encoder is a binary data block \mathbf{u} of length k . Let b^i denote a run of i bits b , e.g., $1^30^5 = 11100000$.

- 1) Encode \mathbf{u} using a binary linear (m, k, d_1) block code \mathcal{C}_1 of dimension k , Hamming distance d_1 , and even length m . The encoding function is denoted by ϕ .

Manuscript received January 05, 2011; revised July 11, 2011; accepted July 20, 2011. Date of publication September 15, 2011; date of current version January 06, 2012. This work was supported by Grant “Theory and Practice of Coding and Cryptography,” Award Number NRF-CRP2-2007-03.

J. H. Weber is with the Delft University of Technology, The Netherlands (e-mail: j.h.weber@tudelft.nl).

K. A. S. Immink is with the Turing Machines BV, The Netherlands, and also with Nanyang Technological University, Singapore (e-mail: immink@turing-machines.com).

H. C. Ferreira is with the University of Johannesburg, Johannesburg, South Africa (e-mail: hcferreira@uj.ac.za).

Communicated by M. Blaum, Associate Editor for Coding Techniques.

Digital Object Identifier 10.1109/TIT.2011.2167954

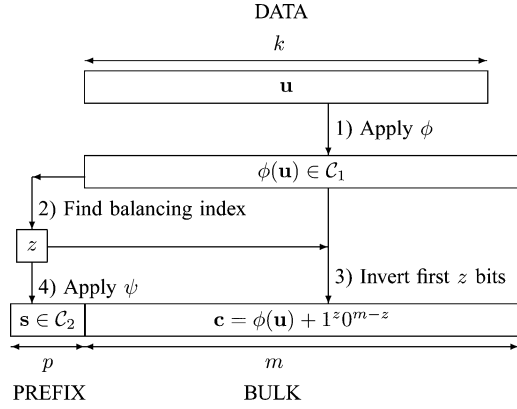


Fig. 1. Encoding procedure.

- 2) Find a balancing index z for the obtained codeword $\phi(\mathbf{u})$, with $1 \leq z \leq m$.
- 3) Invert the first z bits of $\phi(\mathbf{u})$, resulting in the balanced word $\mathbf{c} = \phi(\mathbf{u}) + 1^z 0^{m-z}$.
- 4) Encode the number z into a unique codeword \mathbf{s} from a binary code \mathcal{C}_2 of even length p , constant weight $p/2$, and Hamming distance d_2 . The encoding function is denoted by ψ .

The output of the encoder is the concatenation of the balanced word $\mathbf{s} = \psi(z)$, called the *prefix*, and the balanced word $\mathbf{c} = \phi(\mathbf{u}) + 1^z 0^{m-z}$, called the *bulk* or *payload*. It is obvious that the resulting code \mathcal{C} is balanced and has length $n = m + p$ and redundancy $r = m + p - k$, and thus, code rate $R = k/(m + p)$ and normalized redundancy $\rho = 1 - R = 1 - k/(m + p)$. Its Hamming distance d satisfies the following lower bound.

Theorem 1: The Hamming distance d of code \mathcal{C} is at least

$$\min\{2\lceil d_1/2 \rceil, d_2\}.$$

Proof: Let (\mathbf{s}, \mathbf{c}) and $(\mathbf{s}', \mathbf{c}')$ denote two different codewords of \mathcal{C} and let $z = \psi^{-1}(\mathbf{s})$. If $\mathbf{s} \neq \mathbf{s}'$, then the Hamming distance between the codewords is at least d_2 , since \mathbf{s} and \mathbf{s}' are both in \mathcal{C}_2 . If $\mathbf{s} = \mathbf{s}'$, then the Hamming distance between the codewords is at least $2\lceil d_1/2 \rceil$, which follows from the fact that $\mathbf{c} + 1^z 0^{m-z}$ and $\mathbf{c}' + 1^z 0^{m-z}$ are two different codewords from \mathcal{C}_1 , implying that $d_H(\mathbf{c}, \mathbf{c}') \geq d_1$, and the fact that \mathbf{c} and \mathbf{c}' are both balanced, implying that $d_H(\mathbf{c}, \mathbf{c}')$ is even. ■

Corollary 1: In order to make \mathcal{C} capable of correcting up to t errors, it suffices to choose constituent codes \mathcal{C}_1 and \mathcal{C}_2 with distances $d_1 = 2t + 1$ and $d_2 = 2t + 2$, respectively.

Upon receipt of a sequence (\mathbf{r}, \mathbf{y}) , where \mathbf{r} and \mathbf{y} have lengths p and m , respectively, a simple decoding procedure, illustrated in Fig. 2, consists of the following steps.

- 1) Look for a codeword \mathbf{q} in \mathcal{C}_2 which is closest to \mathbf{r} , and set $\hat{z} = \psi^{-1}(\mathbf{q})$.
- 2) Invert the first \hat{z} bits in \mathbf{y} , i.e., set $\hat{\mathbf{y}} = \mathbf{y} + 1^{\hat{z}} 0^{m-\hat{z}}$.
- 3) Decode $\hat{\mathbf{y}}$ according to a decoding algorithm for code \mathcal{C}_1 , leading to an estimated codeword $\hat{\mathbf{c}}$, and thus, to an estimated information block $\hat{\mathbf{u}} = \phi^{-1}(\hat{\mathbf{c}})$.

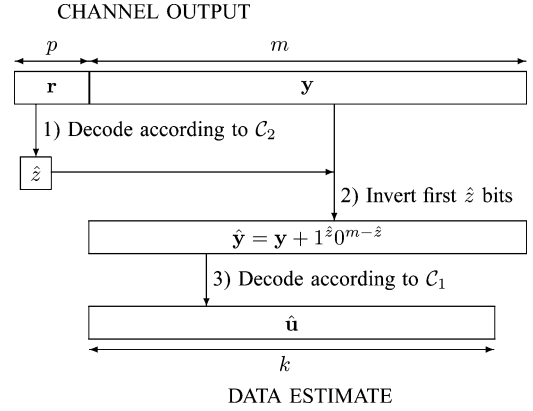


Fig. 2. Decoding procedure.

The following results are immediate.

Theorem 2: The proposed decoding procedure for code \mathcal{C} corrects any error pattern with at most $d_2/2 - 1$ errors in the first p bits and at most $\lceil d_1/2 \rceil - 1$ errors in the last m bits.

Corollary 2: The proposed decoding procedure for code \mathcal{C} corrects up to the number of errors

$$\min\{\lceil d_1/2 \rceil - 1, d_2/2 - 1\}$$

guaranteed by the Hamming distance result from Theorem 1.

Corollary 3: The proposed decoding procedure for code \mathcal{C} corrects up to t errors if $d_1 = 2t + 1$ and $d_2 = 2t + 2$.

Typically, as suggested in Figs. 1 and 2, the length m of the bulk is much larger than the length p of the prefix, and consequently also d_1 is usually (much) larger than d_2 . Hence, from the results presented in this section, the final Hamming distance d is only d_2 in such cases, which may be very small with respect to the overall length $m + p$. However, it will be shown in Section IV, that from the perspective of achieving a certain target decoding error probability, the overall Hamming distance of our scheme is a parameter which is only of minor importance.

III. REDUNDANCY

In this section, we first derive a lower bound on the redundancy of balanced codes with error-correcting capabilities. Then, we compare the redundancy of the code from Section II with this lower bound.

Let $A(n, d, w)$ denote the maximum cardinality of a code of length n , constant weight w , and even Hamming distance d . Hence, for any balanced code of even length n and Hamming distance d , the redundancy is at least

$$r_{\min} = n - \log_2 A(n, d, n/2). \quad (1)$$

Since

$$A(n, 2, n/2) = \binom{n}{n/2} \quad (2)$$

the minimum redundancy for a balanced code without error correction capabilities [6] is

$$r_0 = n - \log_2 \binom{n}{n/2} \quad (3)$$

$$\approx \frac{1}{2} \log_2 n + \frac{1}{2} \log_2 (\pi/2) \quad (4)$$

$$\approx \frac{1}{2} \log_2 n + 0.326, \quad (5)$$

where the first approximation is due to the well-known Stirling formula

$$n! \approx \sqrt{2\pi n} n^n e^{-n}. \quad (6)$$

No general expression for $A(n, d, w)$ is known, but bounds are available in literature. From Theorem 12 in [1], we have the upper bound

$$A(n, d, n/2) \leq \frac{\binom{n}{n/2-t}}{\binom{n/2}{t}} = \frac{\binom{n}{n/2}}{\binom{n/2+t}{t}} \quad (7)$$

where $t = d/2 - 1$. Note that for $d = 2$, i.e., $t = 0$, this gives the same expression as (2), and thus, the bound is tight in this case. The upper bound (7) can be used to lower bound the minimum redundancy r_{\min} from (1) in case $t \geq 1$, i.e., r_{\min} is at least

$$r_{\min}^* = n - \log_2 \binom{n}{n/2} + \log_2 \binom{n/2+t}{t}. \quad (8)$$

Note that this expression can be decomposed into a contribution r_0 from the balance property and a contribution $\log_2 \binom{n/2+t}{t}$ from the capability of correcting up to t errors. Using Stirling's formula, we obtain the approximation

$$r_{\min}^* \approx \left(t + \frac{1}{2}\right) \log_2 \left(\frac{n}{t}\right) + t(\log_2 e - 1) - 1. \quad (9)$$

Next, we will investigate the difference between the lower bound r_{\min}^* on the redundancy, of which it is unknown whether it is achievable in general, and the redundancy r of the proposed construction method. We know from [6] that the redundancy of the Knuth scheme, without error correction, exceeds the minimum achievable redundancy by a factor of two. Obviously, this is a price to be paid for simplicity. For the proposed method with correction, the redundancy is equal to the sum of the redundancy $m - k$ of code \mathcal{C}_1 , and the length p of the prefix. For neither of these terms a general expression is available. Another complication in the analysis is that the error correction levels $t_1 = \lfloor (d_1 - 1)/2 \rfloor$ of \mathcal{C}_1 and $t_2 = d_2/2 - 1$ of \mathcal{C}_2 may be different. The value of $m - k$ depends on the choice of \mathcal{C}_1 . For example, for BCH codes it is roughly $t_1 \log_2 m$ [7]. The length of the prefix can be decomposed into two parts: a contribution of length $\log_2 m$ identifying the balancing index and a contribution of length roughly $(t_2 + 1/2) \log_2 \log_2 m$, based on the presented bound (9), providing the error correction and balancing properties to the prefix. Hence, assuming $m - k \approx t_1 \log_2 m$, as for BCH codes, the total redundancy $m - k + p$ of the proposed method can be approximated as

$$(t_1 + 1) \log_2 m + (t_2 + 1/2) \log_2 \log_2 m. \quad (10)$$

Further assuming that $t_2 \leq t_1 = t$ and that the length is very large, the comparison of this expression to (9) gives that the redundancy of our method is roughly a factor $(t+1)/(t+1/2) = 1 + 1/(2t+1)$ higher than (the lower bound on) the minimum redundancy for any t -error-correcting balanced code. Although the results presented in this analysis are based on bounds and approximations rather than exact results, it seems safe to conclude that the redundancy of the presented method is within a factor of two of the optimum. This will be illustrated in Section V. The redundancy of the codes presented in [2] is (slightly) lower, but the method presented here is simpler and more general (since the constructions from [2] are for $t \leq 4$ only). The constructions from [9] are of a completely different nature, with much higher redundancies but balancing being established on a very small block scale.

IV. ERROR PROBABILITY ANALYSIS

Since the length of the prefix is considerably shorter than the length of the bulk, the probability of the prefix being hit by a random error is proportionally smaller. Therefore, it may be considered giving the prefix a lower error correcting capability. On the other hand, uncorrectable errors in the prefix will lead to a wrong balancing index z and may thus lead to a huge number of errors in the bulk. Therefore, it may be worthwhile to invest in some extra protection of the prefix. Hence, determining the error correction levels of the codes \mathcal{C}_1 and \mathcal{C}_2 is a delicate issue. This will be investigated from both the block error probability and the bit error probability perspectives, in Sections IV-A and IV-B, respectively. Throughout the analysis, we will assume a memoryless binary symmetric channel with error probability ϵ .

A. Block Error Probability

In this subsection, we will investigate proper choices of the error correction capabilities in the context of the block error probability P_{block} , which is defined as the probability that the decoding result $\hat{\mathbf{u}}$ is different from the original information block \mathbf{u} .

An often-used general expression for the block error probability for a block code of length N and Hamming distance D , thus correcting up to $T = \lceil D/2 \rceil - 1$ errors, is

$$\sum_{i=T+1}^N \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i}. \quad (11)$$

Actually, this is an upper bound, since error correction might also take place in case more than T errors occur. To which extent this could happen depends on the structure of the code and the implementation of its decoder. However, this effect will be neglected throughout this paper. In other words, the performance analysis reflects a worst case scenario: the real error probabilities may be (a little bit) better. A well-known approximation of (11) is obtained by considering only its first term

$$\binom{N}{T+1} \epsilon^{T+1} (1-\epsilon)^{N-T-1} \quad (12)$$

since it dominates all other terms. A further simplification is obtained by ignoring the factor $(1 - \epsilon)^{N-T-1}$ in (12), leading to

$$\binom{N}{T+1} \epsilon^{T+1}. \quad (13)$$

The smaller $N\epsilon$, the better (12) and (13) approximate (11).

Since the scheme presented in Section II is multistep, the P_{block} evaluation is a bit more involved. For a received word of length $n = m + p$, let e_1 and e_2 be the number of errors in the last m bits (the bulk) and the first p bits (the prefix), respectively. We consider various situations.

- If $e_1 \leq t_1 = \lceil d_1/2 \rceil - 1$ and $e_2 \leq t_2 = d_2/2 - 1$, then correction of all errors is guaranteed.
- If $e_2 \leq t_2$ and $e_1 > t_1$, then the balancing index is correctly retrieved from the prefix, but the number of errors in the bulk is beyond the error-correcting capability of \mathcal{C}_1 , thus leading to a wrong decoding result.
- If $e_2 > t_2$, then the balancing index retrieved from the prefix is wrong, i.e., $\hat{z} \neq z$, leading to the introduction of extra errors in the bulk due to the inversion process in step 2 of the decoding procedure. Assuming $e_1 = 0$ for the moment, successful decoding requires $z - t_1 \leq \hat{z} \leq z + t_1$. Since, typically, the bulk length m (and thus, the range of z -values) is very large and the error correcting capability t_1 is very small, the probability of the \mathcal{C}_1 -decoder still being successful is negligible. If $e_1 > 0$, then the successful \hat{z} interval will shrink even further, except when a 'transmission error' coincides with an 'inversion error'. Still, even in the latter case, chances for correct decoding are very low. It could be helpful to design ψ , the mapping of the z -values to codewords in \mathcal{C}_2 , in such a way that distances between codewords corresponding to subsequent z -values are kept as low as possible, but also the impact of this will be limited. In conclusion, we assume that $e_2 > t_2$ implies a wrong decoding result with high probability.

Hence, we approximate the block error probability by

$$P_{\text{block}} \approx P(e_1 > t_1 \text{ or } e_2 > t_2) \quad (14)$$

$$\approx P(e_1 > t_1) + P(e_2 > t_2). \quad (15)$$

$$= P_1 + P_2, \quad (16)$$

where

$$P_1 = \sum_{i=t_1+1}^m \binom{m}{i} \epsilon^i (1 - \epsilon)^{m-i} \quad (17)$$

$$\approx \binom{m}{t_1+1} \epsilon^{t_1+1} \quad (18)$$

and

$$P_2 = \sum_{i=t_2+1}^p \binom{p}{i} \epsilon^i (1 - \epsilon)^{p-i} \quad (19)$$

$$\approx \binom{p}{t_2+1} \epsilon^{t_2+1}. \quad (20)$$

The approximations in (18) and (20) are only valid if $m\epsilon$ and $p\epsilon$, respectively, are sufficiently small, which we will assume throughout the rest of this section.

When designing the scheme such that the redundancy is minimized while achieving a certain Hamming distance, it follows from the results in Section II that the Hamming distances of the constituent codes should be chosen (about) equal, i.e., $t_1 = t_2$. However, since the prefix is mostly much shorter than the bulk, the chances of the bulk being hit by errors is much larger than for the prefix. Therefore, intuitively, it should be beneficial to choose t_2 smaller than t_1 . Indeed, when the focus is on the block error probability rather than the Hamming distance, this turns out to be true. When designing the scheme to achieve a certain target block error probability at maximum rate for a given data length and channel error probability, the focus should not be on the final Hamming distance d , but on a careful choice of the error correction capabilities t_1 and t_2 , where the latter can typically be smaller than the former. Since it follows from (16), (18), and (20) that P_{block} is approximately

$$\binom{m}{t_1+1} \epsilon^{t_1+1} + \binom{p}{t_2+1} \epsilon^{t_2+1} \quad (21)$$

an appropriate design strategy is choosing t_1 and t_2 such that both terms in (21) are in the same order of magnitude as the target block error probability, while their sum is (just) below this target. Note that if $m \gg p$, which is usually the case, the binomial coefficient in the first term in (21) is huge in comparison to the binomial coefficient in the second term, and, therefore, the exponent of ϵ in the second term can be considerably smaller than the exponent in the first term.

If t_1 and t_2 are substantially smaller than p , then a more detailed general analysis is possible. To this end, we evaluate P_2/P_1 for the case $t_2 \leq t_1$. Using (18) and (20), we find

$$\frac{P_2}{P_1} \approx \frac{\binom{p}{t_2+1} \epsilon^{t_2+1}}{\binom{m}{t_1+1} \epsilon^{t_1+1}} \quad (22)$$

$$\approx \frac{(t_1+1)!}{(t_2+1)!} \left(\frac{p}{m}\right)^{t_2+1} (m\epsilon)^{t_2-t_1}, \quad (23)$$

where the second approximation follows from $\binom{a}{b} \approx a^b/b!$, which is quite good if b is much smaller than a . If $t_2 = t_1$, then (23) gives

$$\frac{P_2}{P_1} \approx \left(\frac{p}{m}\right)^{t_1+1} \quad (24)$$

which shows that P_2 is indeed orders of magnitude below P_1 in case $p \ll m$. For any $0 < t_2 \leq t_1$, keeping t_1 fixed and reducing t_2 by 1 causes a growth in P_2/P_1 by a factor

$$\frac{t_2+1}{p\epsilon} \left(\frac{p'}{p}\right)^{t_2} \quad (25)$$

where p and p' are the prefix lengths in the cases that prefix error correcting capabilities are t_2 and $t_2 - 1$, respectively. Note that for ranges of practical interest ϵ^{-1} is the dominating factor in (25). Hence, after several reductions of t_2 the ratio P_2/P_1 may be well above zero, i.e., P_2 may no longer be negligibly small.

B. Bit Error Probability

When the prime interest is the bit error probability rather than the block error probability, then the picture may be different from the one sketched in the previous subsection. A decoding error in the bulk typically results in just a few erroneous bits in the data, but a decoding error in the prefix may completely destroy the data. This would ask for a stronger error protection of the prefix. Therefore, in this subsection, we analyze the bit error probability P_{bit} for the proposed scheme.

In general, for a block code of length N and Hamming distance D , the bit error probability can be well approximated by

$$P_{\text{bit}} \approx \frac{D}{N} P_{\text{block}} \quad (26)$$

where P_{block} is as given in (11). This approximation is based on the fact that in case erroneous decoding occurs it is most likely that the output codeword is still close to the original codeword. Since closest codewords differ in D bits and the total number of bits is N , the result follows.

As in the previous subsection, the analysis of the proposed scheme is more involved due to its multistep character. Again, for a received word of length $n = m + p$, let e_1 and e_2 be the number of errors in the last m bits (the bulk) and the first p bits (the prefix), respectively. We consider various situations.

- If $e_1 \leq t_1 = \lceil d_1/2 \rceil - 1$ and $e_2 \leq t_2 = d_2/2 - 1$, then correction of all errors is guaranteed and the original data block is retrieved at the receiver.
- If $e_2 \leq t_2$ and $e_1 > t_1$, then the balancing index is correctly retrieved from the prefix, but the number of errors in the bulk is beyond the error-correcting capability of \mathcal{C}_1 . As argued in the previous paragraph, the resulting fraction of erroneous bits in the data block is d_1/m .
- If $e_2 > t_2$, then the balancing index retrieved from the prefix is wrong, i.e., $\hat{z} \neq z$. As argued in the previous subsection, the final decoding result is wrong with high probability, and, moreover, the number of bit errors can be huge. The actual distribution of the number of errors depends on the implementation of Knuth's algorithm and the mapping ψ . When assuming that both z and \hat{z} are more or less uniformly distributed, it follows from standard probability theory that the expected value of the absolute difference between z and \hat{z} is $m/3$. Hence, the expected fraction of erroneous bits in the data word is $1/3$.

Hence, we approximate the bit error probability by

$$P_{\text{bit}} \approx P'_1 + P'_2 \quad (27)$$

where

$$P'_1 = \frac{d_1}{m} P_1 \quad (28)$$

and

$$P'_2 = \frac{1}{3} P_2 \quad (29)$$

and where P_1 and P_2 are as given in (17) and (19), respectively. It thus follows that P_{bit} is approximately

$$\frac{d_1}{m} \binom{m}{t_1+1} \epsilon^{t_1+1} + \frac{1}{3} \binom{p}{t_2+1} \epsilon^{t_2+1}. \quad (30)$$

Following a similar reasoning as in the previous subsection, an appropriate design strategy is choosing t_1 and t_2 such that both terms in (30) are in the same order of magnitude as the target bit error probability, while their sum is (just) below this target. Again, note that if $m \gg p$, the coefficient in the first term in (30) is huge in comparison to the coefficient in the second term, and, therefore, the exponent of ϵ in the second term can be considerably smaller than the exponent in the first term.

If t_1 and t_2 are substantially smaller than p , then a more detailed general analysis is possible. To this end, evaluate P'_2/P'_1 for the case $t_2 \leq t_1$. Using (28), (29), (18), and (20), we find

$$\frac{P'_2}{P'_1} = \frac{m}{3d_1} \frac{P_2}{P_1} \quad (31)$$

$$\approx \frac{m}{3d_1} \frac{(t_1+1)!}{(t_2+1)!} \left(\frac{p}{m}\right)^{t_2+1} (m\epsilon)^{t_2-t_1} \quad (32)$$

where the approximation comes from (23). If $t_2 = t_1$, then (32) gives

$$\frac{P'_2}{P'_1} \approx \frac{m}{3d_1} \left(\frac{p}{m}\right)^{t_1+1} \quad (33)$$

which shows that P'_2 is indeed orders of magnitude below P'_1 if $p \ll m$ and $t_1 > 0$. For any $0 < t_2 \leq t_1$, keeping t_1 fixed and reducing t_2 by 1 causes a growth in P'_2/P'_1 by the factor given in (25). Hence, after several reductions of t_2 the ratio P'_1/P'_2 may be well above zero, i.e., P'_2 may no longer be negligibly small. Assuming $d_1 < m/3$, this final t_2 value is larger than for the block error probability case, since, for $t_2 = t_1$, P'_2/P'_1 exceeds P_2/P_1 by a factor of $m/(3d_1)$, while the growth factor when reducing t_2 is the same in both cases.

In conclusion, when designing the scheme to achieve a certain P_{bit} at maximum rate, the focus should, as for the P_{block} case, not be on the final Hamming distance d , but on a careful choice of the error correction capabilities t_1 and t_2 . Again, the latter can typically be smaller than the former, but not to the same extent as for the block error probability case.

V. CASE STUDIES

In this section, we illustrate the results obtained in this paper by working out two case studies. In the first case, various options for the constituent codes are considered and studied for one fixed channel. In the second case, one particular code for protecting the payload is evaluated for different channel conditions.

A. Case Study I: Shortened BCH Codes

Let the information block length be $k = 750$. We consider $(750 + 10t_1, 750, 2t_1 + 1)$ codes \mathcal{C}_1 , with $t_1 = 0, 1, \dots, 4$, obtained by shortening $(1023, 1023 - 10t_1, 2t_1 + 1)$ BCH codes [7]. For \mathcal{C}_2 , we consider the shortest known balanced codes with cardinality at least $750 + 10t_1$ and Hamming distance $2t_2 + 2$,

TABLE I
CARDINALITIES OF THE LARGEST KNOWN BALANCED CODES WITH LENGTH
 $p \leq 28$ AND HAMMING DISTANCE $d_2 \leq 10$ [3]

p	$d_2 = 2$	$d_2 = 4$	$d_2 = 6$	$d_2 = 8$	$d_2 = 10$
2	2				
4	6	2			
6	20	4	2		
8	70	14	2	2	
10	252	36	6	2	2
12	924	132	22	4	2
14	3432	325	42	8	2
16	12870	1170	120	30	4
18	48620	3540	320	48	10
20	184756	13452	944	176	38
22	705432	40624	2636	672	46
24	2704156	151484	5616	2576	123
26	10400600	431724	16117	3588	210
28	40116600	1535756	53021	6218	790

TABLE II
CODE PARAMETERS IN THE SETTING OF CASE STUDY I FOR $t_1 = t_2 = t$ WITH
 $0 \leq t \leq 4$

t	C_1			C_2		C		
	m	k	d_1	p	d_2	n	ρ	d
0	750	750	1	12	2	762	0.0157	2
1	760	750	3	16	4	776	0.0335	4
2	770	750	5	20	6	790	0.0506	6
3	780	750	7	24	8	804	0.0672	8
4	790	750	9	28	10	818	0.0831	10

with $t_2 = 0, 1, \dots, 4$. Such balanced codes are tabulated on [3], from which we collected the cardinalities of some short codes in Table I.

An overview of the parameters of codes obtained by choosing $t_1 = t_2 = t$ is provided in Table II. If $t = 0$, then it is found from Table I that a prefix of length 12 is required to represent the 750 possible balancing positions without error correction capabilities, as in the original Knuth case, leading to a code rate of $750/(750 + 12) = 0.9843$, i.e., a normalized redundancy of $1 - 0.9843 = 0.0157$. If $t = 1$, then it is found from Table I that a prefix of length 16 is required to represent the 760 possible balancing positions in the BCH codeword with a single error correction capability, leading to a higher normalized redundancy of $1 - 750/(760 + 16) = 0.0335$. Further increasing the value of t leads to higher distances at the expense of higher redundancies, as can be checked from the table. For $t \leq 4$, C_1 has length $m = 750 + 10t$ and Hamming distance $2t + 1$, while C_2 has length $p = 12 + 4t$ and Hamming distance $2t + 2$. Thus, the code C has length $n = m + p = 762 + 14t$, redundancy $r = 12 + 14t$, normalized redundancy $\rho = (12 + 14t)/(762 + 14t)$, and Hamming distance $d = 2t + 2$. In Table III, we compare the normalized redundancy ρ of our scheme to ρ_{\min}^* , which is the lower bound on the normalized redundancy of any scheme with the same length n and error correction level t . Note that ρ/ρ_{\min}^* is, as for Knuth's original method, close to 2, in fact a little bit less in case we have error-correcting capabilities. The factor

TABLE III
REDUNDANCY COMPARISON IN THE SETTING OF CASE STUDY I FOR
 $t_1 = t_2 = t$ WITH $0 \leq t \leq 4$

t	n	$\rho = 1 - 750/n$	$\rho_{\min}^* = r_{\min}^*/n$	ρ/ρ_{\min}^*
0	762	0.0157	0.0067	2.34
1	776	0.0335	0.0177	1.89
2	790	0.0506	0.0271	1.87
3	804	0.0672	0.0355	1.89
4	818	0.0831	0.0432	1.92

TABLE IV
CODE PARAMETERS IN THE SETTING OF CASE STUDY I FOR $t_1 = 3$ AND
 $0 \leq t_2 \leq 4$

t_1	t_2	C_1			C_2		C		
		m	k	d_1	p	d_2	n	ρ	d
3	0	780	750	7	12	2	792	0.0530	2
3	1	780	750	7	16	4	796	0.0578	4
3	2	780	750	7	20	6	800	0.0625	6
3	3	780	750	7	24	8	804	0.0672	8
3	4	780	750	7	28	10	808	0.0718	8

TABLE V
NUMERICAL EVALUATIONS OF P_1 AND P_2 , WITH $m = 750 + 10t_1$,
 $p = 12 + 4t_2$, $0 \leq t_1 = t_2 \leq 4$, AND $\epsilon = 10^{-4}$

t_1, t_2	P_1	P_2
0	7.2×10^{-2}	1.2×10^{-3}
1	2.7×10^{-3}	1.2×10^{-6}
2	7.2×10^{-5}	1.1×10^{-9}
3	1.4×10^{-6}	1.1×10^{-12}
4	2.4×10^{-8}	9.8×10^{-16}

TABLE VI
NUMERICAL EVALUATIONS OF P'_1 AND P'_2 , WITH $m = 750 + 10t_1$,
 $d_1 = 2t_1 + 1$, $p = 12 + 4t_2$, $0 \leq t_1 = t_2 \leq 4$, AND $\epsilon = 10^{-4}$

t_1, t_2	P'_1	P'_2
0	9.6×10^{-5}	4.0×10^{-4}
1	1.1×10^{-5}	4.0×10^{-7}
2	4.6×10^{-7}	3.8×10^{-10}
3	1.3×10^{-8}	3.5×10^{-13}
4	2.7×10^{-10}	3.3×10^{-16}

ρ/ρ_{\min}^* , the price to be paid for simplicity, may even be smaller since ρ_{\min}^* is only a lower bound on ρ_{\min} of which it is unknown whether it is achievable in case $t \geq 1$.

The proposed scheme also offers the option to provide unequal error protection to the bulk and the prefix. An overview of the parameters of codes obtained by fixing $t_1 = 3$ and varying t_2 is provided in Table IV. Choosing $t_2 = 0$, i.e., providing no error correction capability to the prefix, gives a Hamming distance $d = 2$ and a normalized redundancy $1 - 750/(780 + 12) = 0.0530$. Note that increasing t_2 up to the value of 3 increases both the Hamming distance (since $d = \min\{8, 2t_2 + 2\} = 2t_2 + 2$) and the redundancy. However, also note that further increasing t_2 from 3 to 4 (or beyond) increases the redundancy, without the reward of an improved distance d (since it is stuck at $d = 8$ due to the fact that $t_1 = 3$).

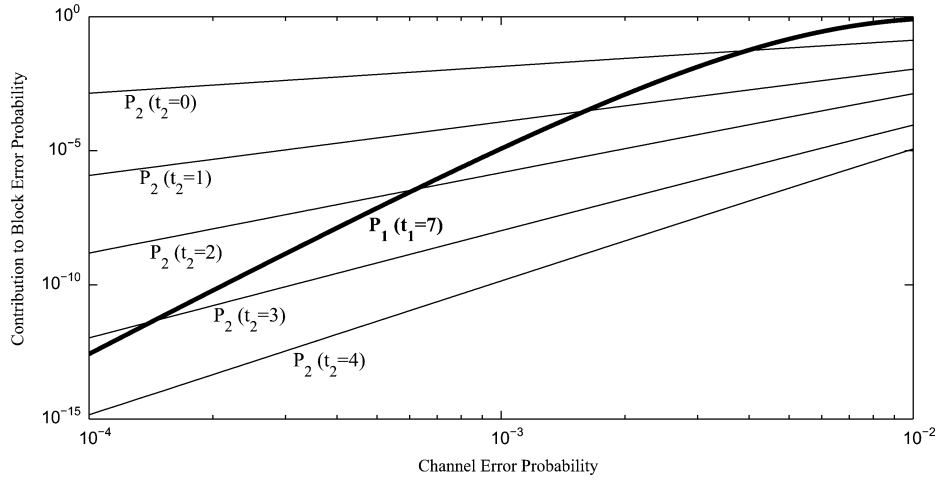


Fig. 3. Probabilities P_1 and P_2 (for $t_2 = 0, 1, \dots, 4$), as a function of the channel error probability ϵ , for Case Study II.

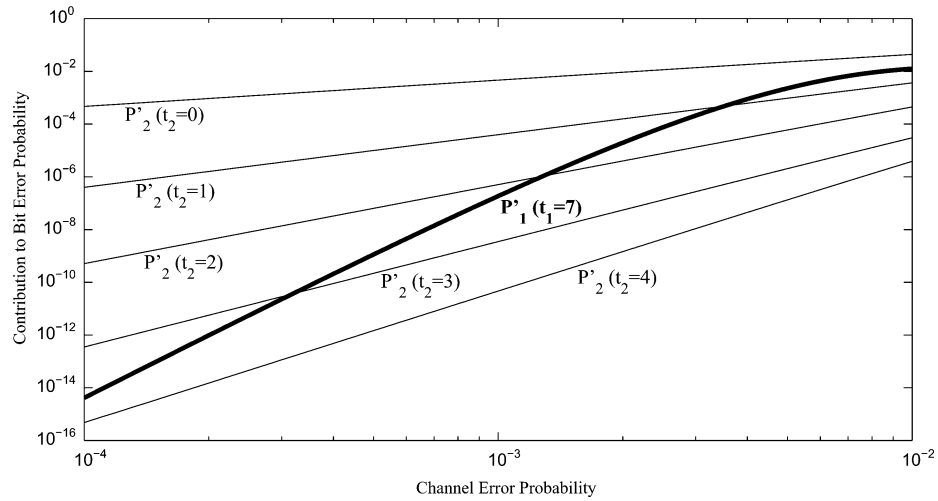


Fig. 4. Probabilities P'_1 and P'_2 (for $t_2 = 0, 1, \dots, 4$), as a function of the channel error probability ϵ , for Case Study II.

Next, we study the block error probability, where we assume that the channel error probability ϵ is 10^{-4} . In Table V, the values of P_1 and P_2 are displayed for various choices of t_1 and t_2 . Let the target block error probability for our application be 10^{-5} , i.e., the error protection levels t_1 and t_2 should be chosen in such a way that $P_1 + P_2$ does not exceed 10^{-5} . From Table V, we conclude that t_1 should be (at least) equal to 3. From Table II, we see that choosing $t_2 = 3$ as well would result in a code \mathcal{C} with normalized redundancy 0.0672 and Hamming distance 8, while we see from Table V that $P_{\text{block}} = 1.4 \times 10^{-6} + 1.1 \times 10^{-12} < 10^{-5}$. However, from Tables IV and V, we can also conclude that keeping $t_1 = 3$ and lowering t_2 from 3 to only 1 results in (i) a normalized redundancy decrease to 0.0578, i.e., a reduction by 14%, (ii) a Hamming distance decrease to 4, and (iii) a block error probability still meeting the target: $P_{\text{block}} = 1.4 \times 10^{-6} + 1.2 \times 10^{-6} < 10^{-5}$. Hence, we obtain a rate increase while still meeting the performance requirement, in spite of the distance drop. Note that a further rate increase is not possible within this scheme, since the performance requirement is not met by the choice $t_2 = 0$, i.e., by giving the prefix no error protection at all. Similar observations can be made for the case that the target P_{block} is 10^{-7} , where

we find that $t_1 \geq 4$ is required, but that $t_2 = 2$ is sufficient for the error protection of the prefix.

Finally, we investigate the bit error probability, still assuming $\epsilon = 10^{-4}$. In Table VI, the values of P'_1 and P'_2 are displayed for various choices of t_1 and t_2 . Let the target bit error probability for a certain application be 10^{-7} , i.e., the error protection levels t_1 and t_2 should be chosen in such a way that $P'_1 + P'_2$ does not exceed 10^{-7} . Note that the choice $t_1 = 3$ and $t_2 = 2$ satisfies the requirement, but that t_2 cannot be reduced any further. This indicates that when designing the scheme to achieve a certain target bit error probability, t_2 could still be chosen smaller than t_1 , but not to the same extent as for the block error probability.

B. Case Study II: Reed-Muller Code RM(6,10)

In this subsection, we use the Reed Muller Code RM(6,10) [7] as the code \mathcal{C}_1 protecting the payload. The code length $m = 2^{10} = 1024$, the data block length $k = \sum_{i=0}^6 \binom{10}{i} = 848$, and the Hamming distance $d_1 = 2^{10-6} = 16$, and thus, the error correction level $t_1 = \lfloor (16 - 1)/2 \rfloor = 7$. We investigate the choice of an appropriate code \mathcal{C}_2 protecting the prefix in the proposed scheme, for different channel error probabilities ϵ .

The balanced code \mathcal{C}_2 should have a size of at least $m = 1024$ and a length p as small as possible. For small values of the Hamming distance d_2 , the minimum length can be determined from Table I. For Hamming distances 2, 4, 6, 8, and 10 the minimum lengths are 14, 16, 22, 24, and 30, respectively. The error protection level $t_2 = d_2/2 - 1$.

The block error probability can be approximated by the sum of $P_1 = \sum_{i=8}^{1024} \binom{1024}{i} \epsilon^i (1-\epsilon)^{1024-i}$ and P_2 as given in (19). For $10^{-4} \leq \epsilon \leq 10^{-2}$, the values of P_1 and P_2 (for $t_2 = 0, 1, \dots, 4$) are given in Fig. 3. When choosing the error protection level $t_2 = t_1 = 7$, then P_2 is many orders of magnitudes below P_1 . Hence, in order to lower the redundancy, t_2 should rather be chosen smaller. Also when lowering t_2 from 7 to 4, and thus, considerably decreasing the redundancy, P_2 is still negligible compared to P_1 for the whole range covered in the figure, so further reductions may be possible. The figure shows that the channel error probability ϵ determines to which extent t_2 can be reduced. The higher ϵ , the lower t_2 can be chosen while keeping P_2 well below P_1 .

The bit error probability can be approximated by the sum of P'_1 and P'_2 , which, according to (28) and (29), are $16P_1/1024 = P_1/64$ and $P_2/3$, respectively. For $10^{-4} \leq \epsilon \leq 10^{-2}$, the values of P'_1 and P'_2 (for $t_2 = 0, 1, \dots, 4$) are given in Fig. 4. Similar conclusions as for the block error probability can be drawn, i.e., the higher ϵ , the lower t_2 can be chosen while keeping P'_2 well below P'_1 . However, the resulting value of t_2 for the bit error probability case is higher than for the block error probability case, due to the fact that P'_2/P'_1 exceeds P_2/P_1 by a factor of $m/(3d_1) = 1024/48 \approx 21$.

VI. CONCLUSION

We have extended Knuth's balancing scheme with error-correcting capabilities. The approach is very general in the sense that any block code can be used to protect the payload, while the prefix of length p is protected by a constant-weight code where the weight is $p/2$. It has been demonstrated that, in order to meet a certain target block or bit error probability in an efficient way, the distances of the constituent codes may preferably be unequal. Hence, from the performance perspective, the overall Hamming distance is of minor importance. As for the original Knuth algorithm, the scheme's simplicity comes at the price of a somewhat higher redundancy than the most efficient but prohibitively complex code. Therefore, the proposed scheme is an attractive simple alternative to achieve (long) balanced sequences with error correction properties.

REFERENCES

- [1] E. Agrell, A. Vardy, and K. Zeger, "Upper bounds for constant-weight codes," *IEEE Trans. Inf. Theory*, vol. 46, no. 7, pp. 2373–2395, Nov. 2000.
- [2] S. Al-Bassam and B. Bose, "Design of efficient error-correcting balanced codes," *IEEE Trans. Computers*, vol. 42, no. 10, pp. 1261–1266, Oct. 1993.
- [3] A. E. Brouwer, Bounds for Binary Constant Weight Codes [Online]. Available: <http://www.win.tue.nl/~aeb/codes/Andw.html>
- [4] K. A. S. Immink, *Codes for Mass Data Storage Systems*, 2nd ed. Eindhoven, The Netherlands: Shannon Foundation Publishers, 2004.

- [5] K. A. S. Immink and J. H. Weber, "Very efficient balanced codes," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 2, pp. 188–192, Feb. 2010.
- [6] D. E. Knuth, "Efficient balanced codes," *IEEE Trans. Inf. Theory*, vol. IT-32, no. 1, pp. 51–53, Jan. 1986.
- [7] S. Lin and D. J. Costello, Jr., *Error Control Coding*, 2nd ed. Upper Saddle River, NJ: Pearson Prentice-Hall, 2004.
- [8] A. Mazumdar, R. M. Roth, and P. O. Vontobel, "On linear balancing sets," in *Proc. IEEE Int. Symp. Information Theory*, Seoul, South Korea, Jun.-Jul. 2009, pp. 2699–2703.
- [9] H. van Tilborg and M. Blaum, "On error-correcting balanced codes," *IEEE Trans. Inf. Theory*, vol. 35, no. 5, pp. 1091–1095, Sep. 1989.
- [10] J. H. Weber and K. A. S. Immink, "Knuth's balanced code revisited," *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1673–1679, Apr. 2010.
- [11] H. Zhou, A. Jiang, and J. Bruck, "Error-correcting schemes with dynamic thresholds in nonvolatile memories," in *Proc. IEEE Int. Symp. Information Theory*, Saint Petersburg, Russia, Jul.-Aug. 2011, pp. 2109–2113.

Jos H. Weber (S'87–M'90–SM'00) was born in Schiedam, The Netherlands, in 1961. He received the M.Sc. (in mathematics, with honors), Ph.D., and MBT (Master of Business Telecommunications) degrees from Delft University of Technology, Delft, The Netherlands, in 1985, 1989, and 1996, respectively.

Since 1985, he has been with the Faculty of Electrical Engineering, Mathematics, and Computer Science of Delft University of Technology. Currently, he is an associate professor at the Wireless and Mobile Communications Group. He is the chairman of the WIC (Werkgemeenschap voor Informatie-en Communicatietheorie in de Benelux) and the secretary of the IEEE Benelux Chapter on Information Theory. He was a Visiting Researcher at the University of California at Davis, USA, the University of Johannesburg, South Africa, and the Tokyo Institute of Technology, Japan. His main research interests are in the areas of channel and network coding.

Kees A. Schouhamer Immink (M'81–SM'86–F'90) received his Ph.D. degree from the Eindhoven University of Technology. He founded and was named president of Turing Machines, Inc., in 1998. He has been, since 1994, an adjunct professor at the Institute for Experimental Mathematics, Essen University, Germany, and is affiliated with the Nanyang Technological University of Singapore. Immink designed coding techniques of a wealth of digital audio and video recording products, such as Compact Disc, CD-ROM, CD-Video, Digital Compact Cassette system, DCC, DVD, Video Disc Recorder, and Blu-ray Disc. He received a Knighthood in 2000, a personal "Emmy" award in 2004, the 1996 IEEE Masaru Ibuka Consumer Electronics Award, the 1998 IEEE Edison Medal, 1999 AES Gold and Silver Medals, and the 2004 SMPTE Progress Medal. He was named a fellow of the IEEE, AES, and SMPTE, and was inducted into the Consumer Electronics Hall of Fame, and elected into the Royal Netherlands Academy of Sciences and the U.S. National Academy of Engineering. He served the profession as President of the Audio Engineering Society, Inc., New York, in 2003.

Hendrik C. Ferreira (SM'08) was born and educated in South Africa where he received the D.Sc. (Eng.) degree from the University of Pretoria in 1980.

From 1980 to 1981, he was a postdoctoral researcher at the Linkabit Corporation in San Diego, CA. In 1983, he joined the Rand Afrikaans University, Johannesburg, South Africa where he was promoted to professor in 1989 and served two terms as Chairman of the Department of Electrical and Electronic Engineering, from 1994 to 1999. He is currently a research professor at the University of Johannesburg. His research interests are in Digital Communications and Information Theory, especially Coding Techniques, as well as in Power Line Communications.

Dr. Ferreira is a past chairman of the Communications and Signal Processing Chapter of the IEEE South Africa section, and from 1997 to 2006 he was Editor-in-Chief of the *Transactions of the South African Institute of Electrical Engineers*. He has served as chairman of several conferences, including the international 1999 IEEE Information Theory Workshop in the Kruger National Park, South Africa, as well as the 2010 IEEE African Winter School on Information Theory and Communications.