Codebook and Marker Sequence Design for Synchronization-Correcting Codes

Victor Buttigieg
Dept. of Comm. & Comp. Eng., University of Malta
Msida MSD 2080, Malta
Email: victor.buttigieg@um.edu.mt

Johann A. Briffa
Dept. of Computing, University of Surrey
Guildford GU2 7XH, England
Email: j.briffa@surrey.ac.uk

***Abstract*—We propose a construction based on synchronization and error-correcting block codes and a matched marker sequence. The block codes can correct insertion, deletion and substitution errors within each codeword. The marker sequence allows the decoder to maintain synchronization at codeword boundaries even at high error rates. An upper bound is given for the performance of these codes over a channel with random substitutions and synchronization errors. It is shown that the performance is largely dependent on the code's minimum Levenshtein distance. The performance of these codes is verified by simulation and compared to published results. In concatenation with a non-binary outer code we obtain a significant improvement in frame error rate at similar overall code rates.**

***Index Terms*—Insertion-Deletion Correction, Synchronization**

I. Introduction

The use of coding schemes that can correct synchronization (i.e. insertion and deletion) errors has been considered early in the development of coding theory [1]. Practical schemes have recently received considerable attention, due to the emergence of applications where such codes are necessary, for example in image watermarking [2] and bit-patterned magnetic media [3]. A survey of error-correcting codes for synchronization-error channels can be found in [4].

The most promising schemes use an inner code to regain synchronization and a conventional outer code to deal with substitution errors [5], [6]. The Davey-MacKay (DM) inner code was introduced in [5] as part of a concatenated scheme that can correct insertion and deletion, as well as substitution errors. In this construction, a random binary marker sequence is used by the decoder to determine synchronization. A sparse representation of the message is then added to this marker sequence; the choice of sparse representation was optimized in [7].

An optimal symbol-level decoding algorithm was presented in [8] for the DM inner code; at the same decoding complexity, the new decoder takes sums over each possible sparse symbol when determining synchronization. This has two principal advantages: *a*) it allows the decoder to make use of *a priori* information for the sparsely-represented symbols, and *b*) it allows the use of non-sparse representations of the message. Respectively, these enable iterative decoding between the inner and outer codes, and the use of representations with better distance properties.

This paper is organized as follows. Section II begins with a definition of the channel model. It also defines synchronization and error-correcting codes and discusses some of their properties. This is followed by the proposed inner code construction and its design rationale in Section III. We also consider the value of the marker sequence for synchronization at codeword boundaries and propose a design that does not affect the performance of the chosen codebook. The decoding of this code is also outlined in this section, together with a comparison with the DM construction. In Section IV we derive an upper bound for the symbol-error performance of the inner code. This is followed by Section V where we give simulation results that corroborate the theoretical arguments. Finally, we conclude and give suggestions for future work in Section VI.

II. Preliminaries

A. The Channel Model

We refer the reader to the channel model of [5], later called the Binary Substitution, Insertion, and Deletion (BSID) channel. At each *time* i, one bit enters the channel, and one of four events may happen: insertion with probability P_i where a random bit is output; deletion with probability P_d where the input is discarded; or transmission with probability $1-P_d-P_i$ where the input bit is output with probability $1-P_s$ or its negation with probability P_s. In the case of deletion or transmission, we proceed to time $i+1$, otherwise the channel remains at time i and is subject to the same events again. The channel drift at time i is defined as the difference between the number of transmitted bits and the number of received bits before the events of time i are considered. We also follow the notation of [4], referring to insertion and deletion events as *synchronization* errors.

B. Synchronization and Error-Correcting Codes

The Levenshtein distance between two sequences $\mathbf{a}$ and $\mathbf{b}$, denoted by $d_l(\mathbf{a},\mathbf{b})$, is the minimum number of insertions, deletions and substitutions necessary to transform $\mathbf{a}$ into $\mathbf{b}$ [9]. Consider a code C with minimum Levenshtein distance $d_{l_{\min}}$, where

$$d_{l_{\min}} = \min\left\{d_l(\mathbf{a},\mathbf{b}) : \mathbf{a},\mathbf{b} \in C, \mathbf{a} \neq \mathbf{b}\right\} \tag{1}$$

Then it is easy to show that a minimum Levenshtein distance decoder is able to correct up to a total of t errors (of any

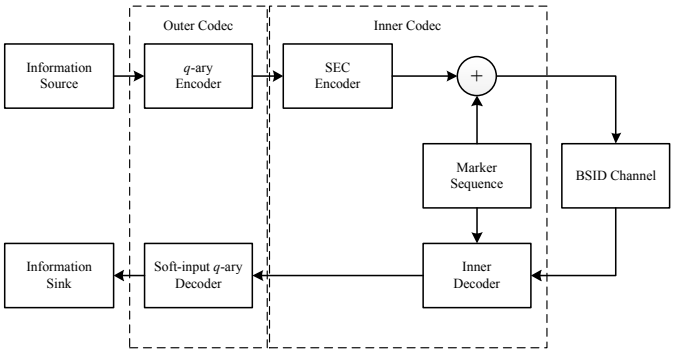

Figure 1. Overall system implementation.

type) per codeword, where $t = \left\lfloor \frac{d_{l_{\min}}-1}{2} \right\rfloor$, provided that the codewords boundaries are known. A maximum Levenshtein distance block code is one which maximizes $d_{l_{\min}}$ for a given fixed codeword length n and number of codewords q. We shall call such a code a $(n, q, d_{l_{\min}})$ Synchronization and Error-Correcting (SEC) code.

C. Offset Codes

A linear block code with minimum Hamming distance $d_{h_{\min}}$can be transformed into a coset with the same minimum Hamming distance by adding a modification vector to all its codewords. Similarly, the minimum Hamming distance is retained when a modification vector is added to a non-linear block code (although in this case the result is not a coset). However, adding a modification vector to a SEC code (linear or non-linear) will not necessarily maintain its $d_{l_{\min}}$. In fact, for most modification vectors, the resultant code produced would have a smaller $d_{l_{\min}}$.

If adding a particular modification vector $\mathbf{v}$ to a block code with $d_{l_{\min}}$ would result in a new code with the same minimum Levenshtein distance then we define the resultant code as an *offset code* of the original code, and we call $\mathbf{v}$ an *allowed modification vector* (AMV).

III. Inner Code Construction and Decoding

A. General System Overview

Consider a concatenated system with an inner code that can correct synchronization and substitution errors on the BSID channel and an outer code to correct any residual substitution errors. This paper focuses on the design of a novel inner code, consisting of a SEC code and a corresponding marker sequence. The choice of the marker sequence depends on the specific SEC code used, as shall be seen in Section III-C. The outer code can be any standard soft-input non-binary error-correcting code. Fig. 1 shows the overall system architecture. The outer code has parameters (N, K), taking K symbols as input and emitting a block of N symbols $\mathbf{d} = (d_0, d_1, \ldots, d_{N-1})$. All input and output symbols for the outer code are in $\mathsf{GF}(q)$.

The SEC code with parameters $(n, q, d_{l_{\min}})$ then maps each q-ary symbol d_ι, $0 \leq \iota < N$ in $\mathbf{d}$ to a binary codeword $\lambda(d_\iota)$ of length n bits. The concatenation of N codewords then forms a codeword sequence $\mathbf{c}$ of Nn bits, which are added modulo 2 to the marker sequence $\mathbf{m}$ (also of length Nn bits) to produce the frame $\mathbf{f} = \mathbf{c} + \mathbf{m}$. The frame $\mathbf{f}$ is then transmitted over the BSID channel, resulting in the received frame $\mathbf{r}$. The marker sequence $\mathbf{m}$ is also available at the inner decoder.

The main objective of the SEC code is to correct for insertion and deletion errors at bit level, although it can also correct for substitution errors. On the other hand the marker sequence is required to maintain synchronization at codeword level. This is necessary since the outer code is unable to correct synchronization errors.

Since the objective of the SEC code is to correct for synchronization errors, the code chosen is a block code with minimum Levenshtein distance $d_{l_{\min}}$. Such a code is able to correct t errors per codeword given that the codeword boundaries are known (c.f. Section II-B). Codeword synchronization is ensured by the use of the marker sequence and the design of the inner decoder.

In this work we assume that the system is synchronized at the frame level, i.e. we assume that the decoder knows the position of the frame boundaries. This assumption is justified in some applications, such as image watermarking, where the frame boundaries are known. In other applications, where the start of frame has to be determined from a continuous stream, it is envisaged that this can be determined using techniques similar to those in [5]. This is currently being studied and will be the subject of a future work.

B. Inner Decoder

The inner code resulting from the combination of the SEC code and the marker sequence is decoded using the maximum *a posteriori* (MAP) algorithm of [8]. This MAP decoder determines the likelihoods for each possible q-ary symbol $d_\iota \in \mathbb{F}_q$ at each block index $0 \leq \iota < N$, given by

$$L(d_\iota) = \sum_{x_1, x_2} \left[\begin{matrix} \alpha(\iota, x_1)\beta(\iota+1, x_2) \\ \cdot \Pr\left\{ \mathbf{r}|_{n\iota+x_1}^{n(\iota+1)+x_2} \mid d_\iota \right\} \end{matrix} \right],$$

where $\mathbf{r}|_a^b$ denotes the sub-sequence $(r_a, r_{a+1}, \ldots, r_{b-1})$ of $\mathbf{r}$ and $\alpha(\cdot)$ and $\beta(\cdot)$ are the forward and backward metrics. These are respectively defined by

$$\alpha(\iota, x) = \Pr\left\{ \mathbf{r}|_0^{n\iota+x}, \varsigma_{n\iota} = x \right\} \tag{2}$$

$$\beta(\iota, x) = \Pr\left\{ \mathbf{r}|_{n\iota+x}^{\rho} \mid \varsigma_{n\iota} = x \right\} \tag{3}$$

where $\varsigma_{n\iota}$ represents the assumed channel drift at the beginning of codeword index ι and ρ is the length of $\mathbf{r}$. It can be seen that the forward and backward metrics track the probability of the channel drift at codeword boundaries. These are computed from the drift probabilities at the previously considered boundary and the probabilities of each possible codeword as an explanation for the received bits between these two boundaries. If the channel error rate is low (i.e. the number of errors per codeword is within the capability of the SEC code), the forward and backward metrics correctly track the drift at codeword boundaries.

C. The Marker Sequence

If the number of errors induced on a codeword by the channel exceeds the capabilities of the SEC code, one of three things may occur:

1) The decoder may remain synchronized at codeword boundaries but the codeword is decoded incorrectly. This is seen as a substitution error by the outer decoder.
2) The decoder may lose synchronization by a few bits at both boundaries. This and a number of subsequent codewords may be decoded in error. This is seen as a short burst of errors by the outer decoder.
3) The errors are such that the decoder either deletes or inserts an entire codeword, maintaining synchronization at subsequent codeword boundaries. The outer decoder sees this as a very long burst of errors, as all subsequent codewords, even if correctly decoded, would be shifted.

The first two error types may be correctable by the outer decoder. However, the third type is likely to lead the outer decoder to fail. In order to address this problem, a marker sequence is added (modulo 2) to the codeword sequence emitted by the SEC coder. The effect of this marker sequence is that if a complete codeword is inserted or deleted by the inner decoder, then subsequent codewords would be positioned incorrectly with respect to the assumed marker sequence. Thus none of the possible codewords would provide a good explanation for the received sequence at this position. This increases the likelihood that the decoder would reject this possibility, and therefore that it retains synchronization at the codeword boundary.

Similar marker sequences were already previously used in the literature. The first such use was in [1] where a pseudo-random sequence was added to the output of a convolutional encoder. A more recent use of a pseudo-random sequence was in [5]. Here, we cannot use such a random sequence. When the marker sequence is added to the SEC code, this is effectively adding modification vectors to the code. However, as highlighted in Section II-C, adding a modification vector to a maximum Levenshtein distance code may change its $d_{l_{\min}}$, unless it is an AMV that results in an offset code. So the marker sequence that must be used is one consisting of a sequence of AMVs. This transforms the maximum Levenshtein distance block code into a sequence of offset codes. We have found that a sequential use of these AMVs performs adequately. However, the performance of the system at high error rates is improved as the number of unique AMVs used in the marker sequence is increased.

D. SEC Code Construction

We can use any construction that gives a good $(n, q, d_{l_{\min}})$ SEC code as part of our inner code. It is also desirable, especially at high error rates, that the code has a maximum number of AMVs (and hence offset codes).

In this work we have used two techniques to construct such codes. The first uses Varshamov-Tenegolts codes [10] with the modifications introduced by Varshsamov [11] and Levenshtein [12]. These consist of all the binary vectors of length n satisfying $\sum_{i=1}^{n} i \cdot x_i \equiv 0 \mod 2n$. This construction gives a SEC code with parameters $(n, M, 3)$. In general M will not be equal to the required q; in this case a selection of q codewords are chosen (assuming that $M > q$). A second technique uses simulated annealing to construct SEC codes with the required parameters. The algorithm described in [13] was modified by replacing Hamming distance with Levenshtein distance. This was used to construct SEC codes with $d_{l_{\min}} > 3$.

The construction of good $(n, q, d_{l_{\min}})$ SEC codes with a maximum number of AMVs is still an open problem. So far we have found the AMVs for a SEC code through an exhaustive search, which is feasible for small n.

E. Comparison with the DM Construction

The system described here is similar to the DM construction [5], however its principle of operation is substantially different. In [5] the SEC code is replaced by a sparse code consisting of codewords with minimum Hamming weight. There is therefore minimal distance (Hamming or Levenshtein) between these codewords. Synchronization is entirely achieved through a pseudo-random marker sequence. If the inner decoder is out of synchronization, there will be a large difference between the received sequence and the known marker sequence. This difference is used by the decoder to track synchronization using a forward-backward algorithm. A good marker sequence in this case therefore has an autocorrelation that is almost zero for any non-zero shift. This explains why a pseudo-random sequence was found to perform best in [5].

The use of a sparse code in [5] ensures that the encoded data causes minimal changes to the marker sequence, allowing the decoder to recover synchronization. The changes due to the sparse code are treated as substitution 'errors' in the marker sequence, in addition to the actual substitution errors from the BSID channel. A sparser code increases the probability that the decoder correctly synchronizes with the marker sequence.

A principal disadvantage of the DM construction is that the decoder finds it difficult to distinguish between the sparse codewords due to their poor distance properties. Also, the sparse code is a liability in the synchronization process, in contrast to the SEC code.

The inner decoder in the DM construction was replaced in [8] by a MAP decoder that is aware of the sparse codebook. Thus, for any given sparse codebook the decoder in [8] always results in a lower symbol-error rate than the one used in [5]. Taking advantage of the MAP decoder, the sparse code was replaced in [8] by one with a higher minimum Hamming distance. Although this improved performance, the system still relied on the marker sequence to maintain bit-level synchronization.

IV. Bound on SEC Code Performance

Since the inner decoder feeds the outer one with q-ary symbols, it is of interest to deduce a bound on the symbol error probability of the inner code.

As we have seen in the previous section, with the use of the marker sequence it is easier for the decoder to maintain synchronization at the codeword boundaries of the SEC code. So assuming that the decoder is synchronized at codeword boundaries we can bound the symbol-error probability by considering the effect of errors on individual codewords.

Consider an n-bit codeword being transmitted over a BSID channel. Then it can be shown [14] that the probability of having an error pattern with, respectively, n_i, n_d and n_s insertion, deletion and substitution errors is

$$\begin{aligned}\Pr\{n_i, n_d, n_s\} =& P_i^{n_i}(1-P_i)^n \cdot P_d^{n_d} \cdot ((1-P_d)P_s)^{n_s} \\ & \cdot ((1-P_d)(1-P_s))^{n-n_d-n_s}.\end{aligned} \tag{4}$$

Now, there are $\binom{n+n_i-1}{n_i}$ ways of inserting n_i bits (note that no bits are inserted after the last codeword bit, since these, if present, are considered to be inserted at the start of the next codeword). Also, there are $\binom{n}{n_d}$ ways of deleting n_d bits and $\binom{n}{n_s}$ ways of substituting n_s bits. The total combinations of having n_i, n_d and n_s errors is given by

$$C_{n_i,n_d,n_s} = \binom{n+n_i-1}{n_i}\binom{n}{n_d}\binom{n}{n_s}. \tag{5}$$

Therefore, the probability of having n_i, n_d and n_s errors anywhere within an n-bit codeword is given by

$$P(E) = C_{n_i,n_d,n_s}\Pr\{n_i, n_d, n_s\}. \tag{6}$$

Given that the code can correct up to t errors and assuming that it cannot correct any error pattern containing more than t errors (which could be any combination of insertion, deletion and substitution errors), then the probability of symbol error, $P_S(E)$ is bounded by

$$P_S(E) \leq \sum_{n_i+n_d+n_s>t} C_{n_i,n_d,n_s}\Pr\{n_i, n_d, n_s\}. \tag{7}$$

Now if $P_i = P_d = P_s = p \ll 1$ we can approximate (7) by

$$P_S(E) \leq \sum_{n_i+n_d+n_s=t+1} C_{n_i,n_d,n_s}p^{t+1}. \tag{8}$$

Similarly, with $P_s = 0$ and $P_i = P_d = p \ll 1$ (7), may be approximated by

$$P_S(E) \leq \sum_{n_i+n_d=t+1} \binom{n+n_i-1}{n_i}\binom{n}{n_d}p^{t+1}. \tag{9}$$

From (8) and (9) one may observe that similar to error-correcting codes for substitution errors, the performance of a SEC code at low error rates mainly depends on its $d_{l_{\min}}$.

V. Results

A critical feature of the inner decoder is that it can maintain synchronization at codeword boundaries. At low error rates the SEC code alone should be sufficient; at higher error rates, the use of an appropriate marker sequence is necessary. To demonstrate this we consider an $(8, 16, 3)$ SEC code, and measure the symbol-error rate (SER) performance using the MAP decoder for a block size $N = 667$. Results are shown in Fig. 2, where the SER is measured using both the conventional Hamming distance (SER_h) and the Levenshtein distance (SER_l). In each case, of course, the distances are taken between the original input symbol sequence and the decoded symbol sequence. If the decoded sequence contains insertion and deletion errors, $\text{SER}_l < \text{SER}_h$. Otherwise the two would be approximately equal.

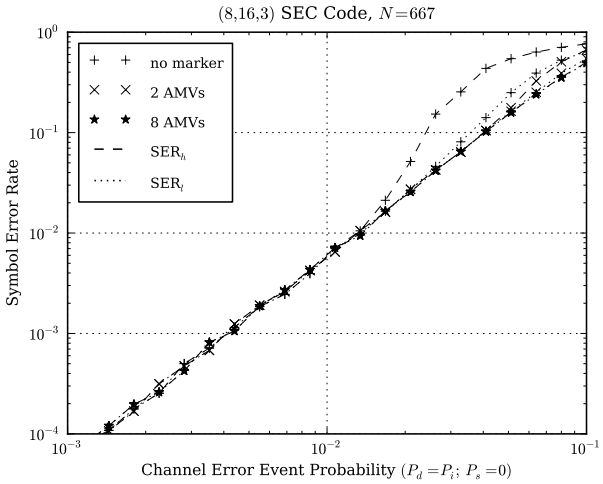

Figure 2. Inner code performance without marker and with a repeating sequence of AMVs, in terms of SER_l (dotted) and SER_h (dashed).

Comparing the SER_h and SER_l results for the SEC code with no marker sequence it is clear that there are symbol-level synchronization errors at high error rates. SER_h and SER_l results coincide at low channel error rates, indicating that there are only substitution errors at the decoder output. The performance at higher channel error rates improves when using a marker consisting of a repeating sequence of AMVs. Further, when using eight AMVs instead of two, the SER_h and SER_l results coincide throughout. This shows that the more varied marker sequence is better at maintaining synchronization at codeword boundaries.

We next compare the performance of SEC codes with varying $d_{l_{\min}}$. The SER_h performance curves are shown in Fig. 3 for codes with a block size $N = 50$. For such a small block size, the SEC code is able to maintain synchronization at codeword boundaries for the whole range of channel error rates considered. Thus, no marker sequence is used in these results. These new codes are also compared with the $(8, 16)$ Z1 code of [8] and the $(7, 8)$ sparse code of [5], both with a random marker sequence. For these codes, the use of the marker sequence is mandatory for maintaining synchronization. We have already seen that at low channel error rates an SEC code is able to maintain synchronization without a marker sequence even for larger block sizes. In Fig. 3 we also plot the approximate upper bound given in (9) for these codes. The MAP decoder performs better than this bound because it is able to correct some events with $t + 1$ or more errors. It is clear from these results that codes with a larger $d_{l_{\min}}$ perform markedly better. Indeed the bound demonstrates how the exponent in the polynomial depends entirely on $d_{l_{\min}}$. This

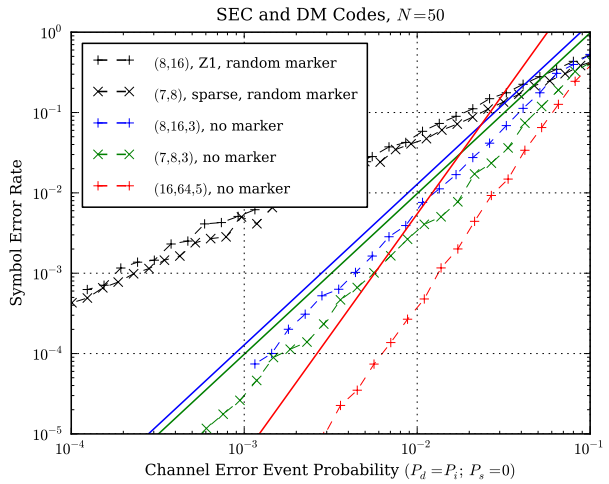

Figure 3. Performance and bounds of SEC codes with varying $d_{l_{\min}}$, and comparison with DM codes. Solid lines represent each respective bound.

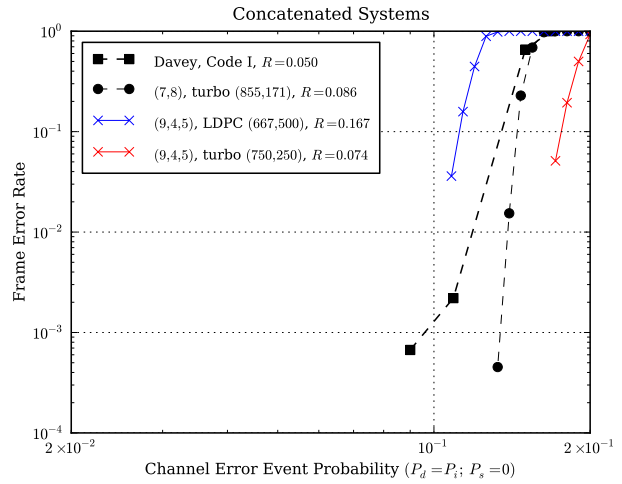

Figure 4. Performance of $(9, 4, 5)$ SEC code with 40 AMVs concatenated with non-binary outer codes, and comparison with published results. Both the $(7, 8)$ sparse code and Davey's Code I use a random marker sequence.

translates to a steeper slope on logarithmic scale plots as $d_{l_{\min}}$ increases.

We finally compare the performance of the $(9, 4, 5)$ SEC code in concatenation with non-binary outer codes. The SER_h performance curves for the concatenated systems are shown in Fig. 4, together with the best concatenated codes in [5], [8]. At a similar overall code rate, our new inner codes perform significantly better than the best published results. With a less powerful outer code, it can still be favourably compared with existing results for a significant gain in code rate.

VI. Conclusions

We have presented a novel construction for channels with synchronization errors, consisting of a SEC code together with a matched marker sequence. This can be used in concatenation with a non-binary outer code for very low error rates that improve on previously published results. We have shown theoretically and verified experimentally that the performance at low error rate is mainly dependent on the SEC code's $d_{l_{\min}}$. We have also shown that at high error rates it is important to have marker sequences consisting of a number of different AMVs. The increase in performance has been achieved without increasing the decoder complexity compared to similar systems [5], [8]. Our construction allows us to directly control the trade-off between code rate, performance and complexity by varying the parameters of the SEC code. Specifically, to increase $d_{l_{\min}}$ one may either increase the codeword size n or reduce the rate $\frac{\log_2 q}{n}$.

We are still investigating the optimal configuration of the concatenated system, in particular the use of iterative decoding between the inner and outer codes. Although we have not presented any results with non-zero P_s, we can report that the performance of the system degrades gracefully with increasing P_s. This can also been inferred from the upper bound given in (8).

Unfortunately, no useful capacity bounds yet exist for the type of channel considered here, so it is difficult to assess how close we are to the channel capacity.

References

[1] R. G. Gallager, "Sequential decoding for binary channels with noise and synchronization errors," Massachusetts Inst. of Tech. Lexington Lincoln Lab, Tech. Rep. 2502, Oct. 27th, 1961.

[2] D. Bardyn, J. A. Briffa, A. Dooms, and P. Schelkens, "Forensic data hiding optimized for JPEG 2000," in *Proc. IEEE Intern. Symp. on Circuits and Systems*, Rio de Janeiro, Brazil, May 15–18, 2011, to be published.

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

[4] H. Mercier, V. Bhargava, and V. Tarokh, "A survey of error-correcting codes for channels with symbol synchronization errors," *Communications Surveys Tutorials, IEEE*, vol. 12, no. 1, pp. 87–96, 2010.

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

[6] E. A. Ratzer, "Marker codes for channels with insertions and deletions," *Annals of Telecommunications*, 2005.

[7] J. A. Briffa and H. G. Schaathun, "Improvement of the Davey-MacKay construction," in *Proc. IEEE Intern. Symp. on Inform. Theory and its Applications*, Auckland, New Zealand, Dec. 7–10, 2008, pp. 235–238.

[8] J. A. Briffa, H. G. Schaathun, and S. Wesemeyer, "An improved decoding algorithm for the Davey-MacKay construction," in *Proc. IEEE Intern. Conf. on Commun.*, Cape Town, South Africa, May 23–27, 2010.

[9] J. B. Kruskal, "An overview of sequence comparison: Time warps, string edits, and macromolecules," *SIAM Review*, vol. 25, no. 2, pp. 201–237, Apr. 1983.

[10] R. R. Varshamov and G. M. Tenegolts, "Codes which correct single asymmetric errors," *Automation and Remote Control*, vol. 26, no. 2, pp. 282–292, 1965.

[11] R. R. Varshamov, "An arithmetic function applicable to coding theory," *Sov. Phys. Doklady*, vol. 10, pp. 185–187, 1965.

[12] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Sov. Phys. Doklady*, vol. 10, no. 8, pp. 707–710, 1966.

[13] A. A. E. Gamal, L. A. Hemachandra, I. Shperling, and V. K. Wei, "Using simulated annealing to design good codes," *IEEE Trans. Inform. Theory*, vol. IT-33, no. 1, pp. 116–123, January 1987.

[14] V. Buttigieg, "Using variable-length codes to correct insertion, deletion and substitution errors," in *Canadian Workshop on Information Theory (CWIT 2011)*, Kelowna, British Columbia, May 2011, submitted for publication.