A Synchronization Technique for Array-based LDPC Codes in Channels With Varying Sampling Rate

Lara Dolecek
EECS Department
University of California
Berkeley, CA 94720, USA
Email: dolecek@eecs.berkeley.edu

Venkat Anantharam
EECS Department
University of California
Berkeley, CA 94720, USA
Email: ananth@eecs.berkeley.edu

***Abstract*— We describe a method for enhancing the synchronization error correction properties of an array-based low density parity check (LDPC) code. The proposed method uses code expurgation: a linear subcode is retained for message encoding and additional input bits are used for protection against synchronization errors. The method is easy to implement and incurs minimal loss in rate.**

I. Introduction

Many communication systems use a substitution-error correcting code to encode a binary input message $\mathbf{x}$ into a coded sequence $\mathbf{c} = C(\mathbf{x})$. The modulated version of this sequence, corrupted by additive noise, arrives at the receiver as a waveform $r(t)$,

$$r(t) = \sum_i c_i h(t - iT) + n(t), \tag{1}$$

where c_i is the i^{th} bit of $\mathbf{c}$, $h(t)$ is the modulating pulse, and $n(t)$ is the noise introduced in the channel.

Upon receiving $r(t)$, the receiver samples it at the times $\{kT_s + \tau_k\}$. The samples are fed into the decoder which produces the most likely input message. In traditional correlation based receivers, for adequate noise rejection, it is essential that the decoder be provided with samples taken at approximately optimal time instances. As the operating requirements under which timing recovery must be performed become more stringent, such as lower signal to noise ratio (SNR) and higher data rates, accurate synchronization becomes critical for the full utilization of the available coding gains.

Several authors have studied the problem of accurate timing recovery in such challenging environments. Proposed solutions include building a more sophisticated timing recovery block [11], a turbo-like approach to iteratively determine both sampling points and encoded data [13], and multiple hypothesis analysis of the sampling instances [9].

As an alternative to more complex and more expensive timing recovery schemes, we propose to instead modify the decoding procedure and the code itself to compensate for imperfect synchronization. The rationale of this approach is that, after a systematic analysis of the robustness of the code to synchronization errors, one could use a subcode of it that would be immune to both substitution as well as synchronization errors. The incurred rate loss in the proposed approach would need to be traded off against the increased complexity and latency associated with the earlier mentioned approaches. The challenge of the proposed approach lies in understanding the synchronization error correction capabilities of given codes of interest by determining high rate subcodes with adequate immunity to synchronization errors.

To emphasize the issues that arise when adequate timing recovery is missing, assume that the modulation scheme in (1) is pulse-amplitude modulation (PAM), and more importantly, that we are operating in the infinite SNR regime where the effect of $n(t)$ is negligible. As a consequence of the initial frequency error, say when $T_s < T$, or of the accumulated phase error in τ_k, some symbol may be sampled more than once (effectively repeated in the infinite SNR regime)[1].

A codeword $\mathbf{c}$ can thus give rise to a whole set of received sampled versions of $r(t)$. We will assume that the number of samples is known, so that codewords can be analyzed in isolation. Thus, for instance, if there is one repetition, $\mathbf{c}$ can give rise to the set $R_1(\mathbf{c})$ of all strings obtained by applying a repetition to $\mathbf{c}$. When two distinct sequences $\mathbf{c_1}$ and $\mathbf{c_2}$ result in the same sampled sequence, it is no longer possible to uniquely determine the codeword or its pre-image $\mathbf{x}$ from the received sequence, *even in the noise-free environment*. We then say that the code $C(n, k)$ suffers from an *identification problem*. We also say that the pair of distinct codewords $\mathbf{c_1}$ and $\mathbf{c_2}$ suffers from an identification problem. For the one repetition case, for instance, this occurs when $R_1(\mathbf{c_1}) \bigcap R_1(\mathbf{c_2})$ is nonempty.

Several authors have studied codes immune to a deletion or an insertion of a bit. For example, the so-called Varshamov-Tenengolts code proposed in [16] and popularized by Levenshtein in [10] has been further studied in [14]. A related construction has been proposed in [8]. While providing immunity

[1]The case $T_s > T$ that may also be of interest is not considered here. See [5] for related work.

1-4244-0504-1/06/$20.00 ©2006 IEEE

to the deletion or insertion of a bit, such constructions do not generally guarantee other desirable properties over a channel that introduces substitution errors, such as linearity, good minimum Hamming distance, and efficient encoding/decoding algorithms. Concatenated codes that correct synchronization and substitution errors have been proposed in [2] and [3], but suffer from a significant loss in rate.

In this paper we first present a brief overview of the array-based LDPC codes and discuss their identification properties. In Section III we propose a general technique for constructing collections of binary strings immune to multiple repetitions. Having established several useful ancillary results in Section IV, we then describe in Section V how the array-based LDPC code can be modified to eliminate the identification problem for the single repetition model. A decoding algorithm appropriate for channels with a single repetition and substitution errors is developed in Section VI.

II. Array-based LDPC codes

Array based LDPC codes are regular LDPC codes parameterized by integers j and p, where $1 \leq j \leq p$, and p is an odd prime, having the parity check matrix $H_{p,j}$ given by ([12])

$$H_{p,j} = \begin{bmatrix} I & I & I & \dots & I \\ I & \sigma & \sigma^2 & \dots & \sigma^{p-1} \\ I & \sigma^2 & \sigma^4 & \dots & \sigma^{2(p-1)} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ I & \sigma^{j-1} & \sigma^{(j-1)2} & \dots & \sigma^{(j-1)(p-1)} \end{bmatrix} \quad (2)$$

where σ denotes a $p \times p$ permutation matrix circularly shifted by 1 position, i.e.

$$\sigma = \begin{bmatrix} 0 & 0 & \dots & 0 & 1 \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}. \quad (3)$$

We let $C_{p,j}$ denote the linear code with the parity check matrix given in (2). Note that $H_{p,j}$ is of rank $pj - j + 1$.

Array-based LDPC codes have good performance [7] and low encoding complexity [12]. They have been proposed for a variety of applications, including digital subscriber lines [6] and magnetic recording applications [15]. These codes permit efficient parallel decoding. However, as explained below, they suffer from an identification problem.

Lemma 1: Under the single repetition model, there are at least $2^{p-2} - 1$ identification problem causing codeword pairs in $C_{p,j}$ for $1 < j < p$.

Proof: Let $\mathbf{c_1}$ denote $[a_2 a_3 ... a_{p-2} a_{p-1} \overline{a}_p a_1]$ and $\mathbf{c_2}$ denote $[a_3 a_4 ... a_{p-1} a_p \overline{a}_1 a_2]$, where a_i ($\overline{a}_i$) denotes a string of length p bits with a single 1 (0) in the i^{th} position and 0's (1's) everywhere else.

We see that $\mathbf{c_1}$ and $\mathbf{c_2}$ can both give rise to the same string after one repetition, namely $[a_3 a_4 ... a_{p-1} a_p \overline{a}_1 a_2 0]$ (same as $[0 a_2 a_3 ... a_{p-2} a_{p-1} \overline{a}_p a_1]$).

We now prove that $\mathbf{c_1}, \mathbf{c_2}$ are in fact codewords of $C_{p,p-1}$.

Let $\mathbf{c_1}^{\langle kp \rangle}$ denote the string obtained by cyclically shifting $\mathbf{c_1}$ to the right by kp positions. Since $C_{p,j}$ is quasi-cyclic [12], it suffices to verify that $\mathbf{c_1}^{\langle 2p \rangle} = [\overline{a}_p a_1 a_2 a_3 ... a_{p-2} a_{p-1}]$ and $\mathbf{c_2}^{\langle 2p \rangle} = [\overline{a}_1 a_2 a_3 a_4 ... a_{p-1} a_p]$ satisfy $\mathbf{c_1}^{\langle 2p \rangle} H_{p,p-1}^T = 0$ and $\mathbf{c_2}^{\langle 2p \rangle} H_{p,p-1}^T = 0$.

It is easily seen that $\mathbf{c_1}^{\langle 2p \rangle} [II...I]^T = 0$. Now consider a row-wise submatrix $[I \sigma^l \sigma^{2l} \dots \sigma^{l(p-1)}]$ of $H_{p,p-1}$, for some l, $1 \leq l \leq p-2$. Write $\mathbf{c_1}^{\langle 2p \rangle} [I \sigma^l \sigma^{2l} \dots \sigma^{l(p-1)}]^T$ as $\overline{a}_p + \sum_{i=1}^{p-1} a_i [\sigma^{il}]^T = \overline{a}_p + \sum_{i=1}^{p-1} a_{[i+il]_p}$, where $[x]_p$ indicates x mod p. Since $1 \leq i \leq p-1$ and $1 \leq l \leq p-2$, $(i + il)$ mod $p \neq 0$, and no term in the summation is a_p. In addition, all terms in the summation are distinct, as otherwise there would exist i, i', $i' < i$ such that $(i - i')(l + 1) \equiv 0$ mod p, which is impossible for p prime, $i, i' \leq p-1$ and $l + 1 \leq p - 1$. Therefore, $\mathbf{c_1}^{\langle 2p \rangle} [I \sigma^l \sigma^{2l} \dots \sigma^{(p-1)l}]^T = 0$. The proof for $\mathbf{c_2}^{\langle 2p \rangle} H_{p,p-1}^T = 0$ is analogous.

Provided that both $\mathbf{c_1}^{\langle kp \rangle}$ and $\mathbf{c_2}^{\langle kp \rangle}$ have the same starting and ending bits, they too suffer from the identification problem in the single repetition model. This occurs as long as k is not congruent to 1 or to 2 mod p. Let B_1 (B_2) be the set of codewords obtained by cyclically shifting $\mathbf{c_1}$ ($\mathbf{c_2}$) by kp positions for k ranging from 3 to p. One can directly check that the pair comprised of any nontrivial linear combination of elements in B_1 and the same linear combination of their counterparts in B_2 also suffers from the identification problem.

Since, by construction, $C_{p,j} \supseteq C_{p,j+1}$, in each $C_{p,j}$ there are therefore at least $2^{p-2} - 1$ pairs of codewords suffering from the identification problem.

III. Construction of a multiple repetitions correcting set

For a binary string $\mathbf{s}$, let $R_t(\mathbf{s})$ denote the set of all strings obtained by applying t repetitions to $\mathbf{s}$. We call a collection S of strings t-repetitions correcting if the sets $R_t(\mathbf{s_1})$ and $R_t(\mathbf{s_2})$ are disjoint for all distinct elements $\mathbf{s_1}, \mathbf{s_2}$ of S. In this section we describe a method for constructing a t-repetitions correcting collection of strings, building on the $t = 1$ case [10], [14]. Given a code, this can in principle be used to develop a subcode that does not suffer from the identification problem for t repetitions, along the lines developed for $t = 1$ in Sections V and VI.

Let us first introduce a useful transformation in which we express the number of runs of a string in terms of the weight of a string in the transformed domain. For a string $\mathbf{c}$ of length n, let the string $\tilde{\mathbf{c}}$ of length $n - 1$ be defined as $\mathbf{c}T_n$, where T_n is a $n \times (n-1)$ matrix satisfying

$$T_n(i,j) = \begin{cases} 1, & \text{if } i = j, j+1 \\ 0, & \text{else.} \end{cases} \quad (4)$$

If $\mathbf{c}$ has r runs, then $\tilde{\mathbf{c}}$ has weight $r - 1$, and vice versa. Both $\mathbf{c}$ and its complement $\overline{\mathbf{c}}$ result in the same $\tilde{\mathbf{c}}$.

If C is a linear code of length n with a generator matrix G, its image under T_n is a linear code generated by $\tilde{G} = GT_n$. If the all-ones is not a codeword in C, then $\tilde{G}$ is full rank.

A repetition in $\mathbf{c}$ corresponds to an insertion of a zero in its counterpart $\tilde{\mathbf{c}}$. Therefore, to construct a collection of strings that is $t = 1$ repetition correcting, it suffices to construct a

collection of strings that is single insertion of a zero correcting in the transformed domain.

For $w \geq 1$, consider the set $S(m,w,a,r)$ defined as:

$$S(m,w,a,r) = \{ \quad \mathbf{s} = (s_1, s_2, ...s_m) \in \{0,1\}^m : \sum_{i=1}^{m} s_i = w, \sum_{i=1}^{m} i s_i \equiv a \bmod r\}. \tag{5}$$

The set $S(m,0,0,r)$ contains just the all zeros string by convention. Let $a_0 = 0$, and let $S\left(m,(a_1,r_1),(a_2,r_2),...,(a_m,r_m)\right)$ be defined as

$$S\left(m,(a_1,r_1),(a_2,r_2),...,(a_m,r_m)\right) = \bigcup_{l=0}^{m} S(m,l,a_l,r_l). \tag{6}$$

Lemma 2: Provided that $r_l > l \ \forall l \in [0,m]$, the set $S\left(m,(a_1,r_1),(a_2,r_2),...,(a_m,r_m)\right)$ is single insertion of a zero correcting.

Proof: If each set in the disjoint union in (6) is single insertion of a zero correcting, so is their (disjoint) union. Consider $\mathbf{x} \in S(m,l,a_l,r_l)$ for $r_l > l$. Following the analysis in [10], [14], suppose the insertion of a zero occurs in the L^{th} position (which is unknown). Let $\mathbf{x}'$ denote the resulting string. Compute $a' \equiv \sum_{i=1}^{m} i x'_i \bmod r_l$;

$$\begin{aligned} a' &\equiv \textstyle\sum_{i=1}^{m} i x'_i \bmod r_l \\ &\equiv \left(\textstyle\sum_{i=1}^{L-1} i x_i + \sum_{i=L}^{m} (i+1) x_i\right) \bmod r_l \\ &\equiv (a_l + R) \bmod r_l, \end{aligned} \tag{7}$$

where R denotes the number of ones to the right of the inserted 0. Since $R \leq l < r_l$, the offset R mod r_l can be uniquely determined from a_l and a' mod r_l, and the string $\mathbf{x}$ is recovered by deleting a zero immediately preceding the R^{th} 1 in $\mathbf{x}'$ counting from the right. ■

The construction given in (5) and (6) can be generalized for the correction of multiple repetitions as follows:

Let w denote the weight of $\mathbf{s}$, let $b_{i+1} = b_{i+1}(\mathbf{s})$, $1 \leq i \leq w$ be the size of the run of zeros immediately following the i^{th} 1 in $\mathbf{s}$, and let $b_1 = b_1(\mathbf{s})$ be the size of the run of zeros preceding the leftmost 1. If the i^{th} 1 is immediately followed by another 1, $b_{i+1} = 0$, and if the leftmost bit in $\mathbf{s}$ is 1, $b_1 = 0$. Moreover, if $\mathbf{s}$ consists only of zeros, $b_1 =$ length of $\mathbf{s}$. We call b_i the size of the i^{th} bin of zeros of $\mathbf{s}$.

Let $\mathbf{a} = (a_1, a_2, ..., a_t)$ for $t \geq 1$, and consider the set $\hat{S}(m,w,\mathbf{a},p)$ for $w \geq 1$ defined as

$$\begin{aligned} \hat{S}(m,w,\mathbf{a},p) = \{ \quad & \mathbf{s} = (s_1, s_2, ...s_m) \in \{0,1\}^m : \\ & \textstyle\sum_{i=1}^{m} s_i = w, \\ & \textstyle\sum_{i=1}^{w+1} i b_i \equiv a_1 \bmod p, \\ & \textstyle\sum_{i=1}^{w+1} i^2 b_i \equiv a_2 \bmod p, \\ & \qquad \vdots \\ & \textstyle\sum_{i=1}^{w+1} i^t b_i \equiv a_t \bmod p\}. \end{aligned} \tag{8}$$

The set $\hat{S}(m,0,\mathbf{0},p)$ contains just the all-zeros string by convention. Let $\mathbf{a_0} = \mathbf{0}$ and let $\hat{S}\left(m,(\mathbf{a_1},p_1),(\mathbf{a_2},p_2),...,(\mathbf{a_m},p_m)\right)$ be defined as

$$\hat{S}\left(m,(\mathbf{a_1},p_1),(\mathbf{a_2},p_2),...,(\mathbf{a_m},p_m)\right) = \bigcup_{l=0}^{m} \hat{S}(m,l,\mathbf{a_l},p_l). \tag{9}$$

Lemma 3: If each p_l is prime and $p_l > max(t,l)$, the set $\hat{S}\left(m,(\mathbf{a_1},p_1),(\mathbf{a_2},p_2),...,(\mathbf{a_m},p_m)\right)$ is t-insertions of zeros correcting.

Proof: It suffices to show that each set $\hat{S}(m,l,\mathbf{a_l},p_l)$ is t-insertions of zeros correcting. Consider $\mathbf{x} \in \hat{S}(m,l,\mathbf{a_l},p_l)$. After experiencing t insertions of zeros, it becomes string $\mathbf{x}'$. We now show that $\mathbf{x}$ is always uniquely determined from $\mathbf{x}'$.

Let $i_1 \leq i_2 \leq ... \leq i_t$ be the (unknown) indices of the bins of zeros that have experienced insertions. For each j, $1 \leq j \leq t$, compute $a'_j \equiv \sum_{i=1}^{w+1} i^j b'_i \bmod p_l$, where b'_i is the size of the i^{th} bin of zeros of $\mathbf{x}'$,

$$\begin{aligned} a'_j &\equiv \textstyle\sum_{i=1}^{w+1} i^j b'_i \bmod p_l \\ &\equiv a_j + (i_1^j + i_2^j + ... + i_t^j) \bmod p_l, \end{aligned} \tag{10}$$

where a_j is the j^{th} entry in the residue vector $\mathbf{a_l}$ (to lighten the notation the subscript l in a_j is omitted).

By collecting the resulting expressions over all j, and setting $R_j \equiv a'_j - a_j \bmod p_l$, we arrive at

$$E_t = \begin{cases} R_1 \equiv i_1 + i_2 + ... + i_t \bmod p_l \\ R_2 \equiv i_1^2 + i_2^2 + ... + i_t^2 \bmod p_l \\ \qquad \vdots \\ R_k \equiv i_1^t + i_2^t + ... + i_t^t \bmod p_l. \end{cases} \tag{11}$$

The terms on the right hand side of the congruency constraints are known as power sums in t variables. Let Λ_k denote the k^{th} elementary symmetric function of $\{i_1, i_2, ..., i_t\}$ mod p_l,

$$\Lambda_k \equiv \sum_{v_1 < v_2 < ... < v_k} i_{v_1} i_{v_2} \cdots i_{v_k} \bmod p_l. \tag{12}$$

Using Newton's identities over $GF(p_l)$ which relate power sums to symmetric functions of the same variable set, and are of the type

$$R_k - \Lambda_1 R_{k-1} + \Lambda_2 R_{k-2} - ... + (-1)^{k-1} \Lambda_{k-1} R_1 + (-1)^k k \Lambda_k = 0, \tag{13}$$

for $k \leq t$, we can obtain an equivalent system of t equations:

$$\widetilde{E}_t = \begin{cases} d_1 \equiv \sum_{j=1}^{t} i_j \bmod p_l \\ d_2 \equiv \sum_{j<k} i_j i_k \bmod p_l \\ \qquad \vdots \\ d_t \equiv \prod_{j=1}^{t} i_j \bmod p_l, \end{cases} \tag{14}$$

where each residue d_k is computed recursively from $\{d_1, ..., d_{k-1}\}$ and $\{R_1, R_2, ..., R_k\}$. This may be done because, in each k^{th} equation of the t equations of type (13) we use, the coefficient of Λ_k is nonzero.

Consider the expression:

$$\prod_{j=1}^{t} (x - i_j) \equiv 0 \bmod p_l, \tag{15}$$

and expand it into the form

$$x^t + c_{t-1} x^{t-1} + ... + c_1 x + c_0 \equiv 0 \bmod p_l. \tag{16}$$

Since (15) equals (16), by comparison with (14) we see that $d_k \equiv (-1)^k c_{t-k} \bmod p_l$. We may then solve for the roots of (16) to get the desired set of indices $\{i_1, i_2, ..., i_t\}$. Thus $\mathbf{x}$ is always uniquely recovered from $\mathbf{x}'$. ■

One can check that for $t = 1$, $p_l > l > 0$ and p_l prime, $S(m, l, a_l, p_l) = \hat{S}(m, l, d - a_l, p_l)$, where $d = (l + 1)(2m - l)/2$.

IV. Auxiliary Results

Due to space constraints we state the results without proof. For the omitted proofs please refer to [4].

Let P be the set of binary strings of length $n = p^2$ defined as $P = \{\mathbf{s} : \mathbf{s} = 0^{(p-t)p}1^{tp}$ or $\mathbf{s} = 1^{tp}0^{(p-t)p}\}$ where p is an odd prime, t is an even integer, $1 \le t \le p - 1$, and the notation 0^k1^l denotes a binary string comprised of a run of k zeros followed by a run of l ones.

Lemma 4: The set P is a $p-1$-dimensional set of linearly independent binary strings.

Lemma 5: For all $\mathbf{s} \in P$, $\mathbf{s}H_{p,j}^T = 0$, for $H_{p,j}$ given in (2) and $j \le p$.

For $j < p$, as a consequence of the previous two Lemmas, we can form a generator matrix $G_{p,j}$ of the array-based LDPC code $C_{p,j}$, such that

$$G_{p,j} = \begin{bmatrix} G_p^s \\ G_{p,j}^m \end{bmatrix} \tag{17}$$

where G_p^s is a $p - 1 \times p^2$ matrix whose rows are all distinct elements of the set P. By applying only row manipulations to a generator matrix, the matrix $G_{p,j}^m$ (which is $(K-p+1)\times p^2$, where K is $p(p - j) + j - 1$ ([12]), and thus nonempty for $j < p$) has each qp^{th} column, for $1 \le q \le p$, equal to the $(qp + 1)^{\text{th}}$ column.

Let $\tilde{G}_{p,j} = G_{p,j}T_{p^2}$ where T_{p^2} is given by (4), and observe that the top $p - 1$ rows of $\tilde{G}_{p,j}$ are all distinct and are of the form $0^{tp-1}10^{p^2-tp-1}$, for $1 \le t \le p - 1$.

Let $\tilde{C}_{p,j}$ be the code generated by $\tilde{G}_{p,j}$. Since the all-ones string is not a codeword in $C_{p,j}$, the matrix $\tilde{G}_{p,j}$ has full rank.

Lemma 6: No codeword in $\tilde{C}_{p,j}$ has weight p^2-1 or p^2-2.

We complete the section with the following result.

Lemma 7: For p an odd prime, and for each j, $0 \le j \le p-1$, there exists a subset of $S = \{1, p+1, 2p+1, \ldots, (p-1)p+1, p^2+1\}$ the sum of whose elements equals j mod p.

V. Modified Encoding

Consider the code $C_{p,j}$. Let $\mathbf{m_u}$ be a binary string of length $(K - p + 1)$ bits provided by the user. Denote by $\mathbf{m_s}$ an auxiliary binary string of length $p - 1$. Let $\mathbf{c} = [\mathbf{m_s}\mathbf{m_u}]G_{p,j}$, and let s_1 and s_2 be additional single bits.

The values of $\mathbf{m_s}$, s_1 and s_2 are chosen such that

$$f(\mathbf{v}) = \sum_{i=1}^{p^2+1} iv_i \equiv a \bmod p^2, \tag{18}$$

is satisfied for some arbitrary but fixed constant a, where $\mathbf{v}$ is defined as $[s_1\mathbf{c}s_2]T_{p^2+2}$ for T_{p^2+2} given in (4). By Lemma 6, the string $\mathbf{v}$ in (18) has weight at most $p^2 - 1$. By Lemma 2 with $a_i = a$ and $r_i = p^2$ for all $1 \le i \le p^2 - 1$, the set of $\mathbf{v}$ satisfying (18) is single insertion of a zero correcting. We transmit $[s_1\mathbf{c}s_2]$. The set of such strings is single repetition correcting.

To show that for every $\mathbf{m_u}$ it is possible to find appropriate values of s_1, s_2 and $\mathbf{m_s}$ such that (18) holds, let $\tilde{\mathbf{u}}$ be $[0^{p-1}\mathbf{m_u}]\tilde{G}_{p,j}$ and let $a' \equiv \sum_{i=1}^{p^2-1}(i + 1)\tilde{u}_i \bmod p^2$. Also let $\tilde{\mathbf{s}}$ be $[\mathbf{m_s}0^{K-p+1}]\tilde{G}_{p,j}$, where $\tilde{\mathbf{c}} = \tilde{\mathbf{u}} + \tilde{\mathbf{s}}$, for $\tilde{\mathbf{c}} = \mathbf{c}T_{p^2}$. By construction every entry in $\tilde{\mathbf{u}}$ in a position whose index is a multiple of p is precisely zero, and the only non-zero entries in $\tilde{\mathbf{s}}$ are in positions with indices that are multiples of p. Expand $f(\mathbf{v})$ as

$$\begin{aligned} f(\mathbf{v}) &= \hat{s}_1 + \textstyle\sum_{i=1}^{p^2-1}(i + 1)\tilde{c}_i + (p^2 + 1)\hat{s}_2 \\ &= \hat{s}_1 + \textstyle\sum_{i=1}^{p^2-1}(i + 1)\tilde{u}_i + \sum_{i=1}^{p^2-1}(i + 1)\tilde{s}_i + \\ &\quad (p^2 + 1)\hat{s}_2 \end{aligned} \tag{19}$$

Then, $f(\mathbf{v}) \equiv a' + \sum_{i=0}^{p}(ip + 1)z_i \bmod p^2$, where $\mathbf{z} = [\hat{s}_1\mathbf{m_s}\hat{s}_2]$, and $\hat{s}_1 = s_1 + c_1$, $\hat{s}_2 = s_2 + c_{p^2}$.

It follows by Lemma 7 that irrespective of the value a' (determined by the user's input message) it is always possible to choose $\mathbf{z}$ such that $f(\mathbf{v}) \equiv a \bmod p^2$. ■

Therefore, by reducing the dimension of the input space by $p - 1$ and introducing two guard bits, we are able to ensure that the transmitted codeword (plus the guard bits) does not suffer from the identification problem for the single repetition model. The overall rate loss is then $\frac{K}{p^2} - \frac{K-(p-1)}{p^2+2}$, which is asymptotically zero as the blocklength tends to infinity.

local domain	local function $\varphi(\cdot)$
$\{G\}$	1
$\{G, L_i\}$	$1[L_i = 1 \cdot 1(G \le i - 1) +$ $0 \cdot 1(G = i) + (-1) \cdot 1(G \ge i + 1)]$
$\{L_i, x_i\}$	$P(y_i\|x_i)1(L_i = -1) + P(y_{i+1}\|x_i)1(L_i = 1)$ $+P(y_i\|x_i)P(y_{i+1}\|x_i)1(L_i = 0)$
$\{x_i\}$	1
$\{c_k, (x_j, j \in \mathcal{N}_k)\}$	$1(c_k = \oplus_{j\in\mathcal{N}_k} x_j)$

Fig. 1. Local domains and functions.

VI. Modified decoding algorithm

We conclude the paper with an outline of a message passing decoding algorithm appropriate for our scheme.

Suppose the sequence y of length $n + 1$ bits is received as a result of transmitting a codeword of length n bits through a noisy channel that also causes a repetition error. For each coded bit x_i we wish to compute $P(x_i|y_1^{n+1})$. We introduce auxiliary variables G, which takes values in $\{1, ..., n\}$, and L_i, for $\forall\ i \in [1, n]$, such that $L_i \in \{-1, 0, 1\}$. The variable G denotes the position of the repetition, and L_i denotes the relative location of the i^{th} bit with respect to the repetition. Write

$$P(x_i|y_1^{n+1}) \propto \sum_G \sum_{L_1^n} \sum_{x_1^n \setminus x_i} P(x_1^n, L_1^n, G, y_1^{n+1}). \tag{20}$$

Group the variables as shown in Fig. 1., for $1 \le i \le n$ and $1 \le k \le M$, where M is the total number of checks.

The junction graph corresponding to these local domains has the bidirectional edges between:

- $\{G\}$ and $\{G, L_i\}$ for each $1 \le i \le n$,
- $\{G, L_i\}$ and $\{L_i, x_i\}$ for each $1 \le i \le n$,
- $\{L_i, x_i\}$ and $\{c_k, (x_i, i \in \mathcal{N}_k)\}$, for each pair (i, k) such that $i \in \mathcal{N}_k$, and
- $\{x_i\}$ and $\{L_i, x_i\}$ for each $1 \le i \le n$.

Then,

$$\sum_G \sum_{L_1^n} \sum_{x_1^n \setminus x_i} P(x_1^n, L_1^n, G, y_1^{n+1}) \approx \sum_{L_i} \varphi(L_i, x_i) \sum_G \varphi(G, L_i) \prod_{j=1, j\neq i}^{n} \sum_{L_j} \varphi(G, L_j) \times \sum_{x_j} \varphi(L_j, x_j) \prod_{k \in \mathcal{N}_i} 1(c_k = \oplus_{j \in \mathcal{N}_k} x_j) \quad (21)$$

where $\varphi(\cdot)$ denotes the local function of the appropriate variables listed in Fig. 1 and the approximation comes from ignoring the cycles in the graph. We may thus use a message passing algorithm as in [1] to try to find $P(x_i|y_1^{n+1})$.

Let all messages be initialized to 1, and let $\alpha_j(L_j)$ be the message sent from $\{L_j, x_j\}$ to $\{G, L_j\}$ at some stage.

The message $\beta_j(G)$ from $\{G, L_j\}$ to $\{G\}$ is then $\beta_j(G) = \sum_{L_j} \varphi(G, L_j)\alpha_j(L_j)$, and the message $\gamma_i(G)$ sent from $\{G\}$ to $\{G, L_i\}$ is $\prod_{j\in\{1,n\}\setminus i} \beta_j(G)$. Finally, the message from $\{G, L_i\}$ to $\{L_i, x_i\}$ is $\delta_i(L_i) = \sum_G \varphi(G, L_i)\gamma_i(G)$.

The message $\tau_{j,k}(x_j)$ sent from $\{L_j, x_j\}$ to $\{c_k(x_j, j \in \mathcal{N}_k)\}$ is $\sum_{L_j} \varphi(L_j, x_j)\delta_j(L_j) \prod_{l\in\mathcal{N}_j\setminus k} \eta_{l,j}(x_j)$, where $\eta_{l,j}(x_j)$ is the message from $\{c_l, (x_j, j \in \mathcal{N}_l)\}$ to $\{L_j, x_j\}$ and is $\sum_{x_i, i\in\mathcal{N}_l\setminus j} \varphi(c_l, (x_i, i \in \mathcal{N}_l)) \prod \tau_{i,l}(x_i)$.

The message $\alpha_j(L_j)$ is updated to $\sum_{x_j} \varphi(L_j, x_j) \prod_{k\in\mathcal{N}_j} \eta_{k,j}(x_j)$, and message exchange continues as above. Thus, (20) is

$$P(x_i|y_1^{n+1}) \approx \sum_{L_i} \varphi(L_i, x_i)\delta_i(L_i) \prod_{k\in\mathcal{N}_i} \eta_{k,i}(x_i). \quad (22)$$

The number of steps needed in each global iteration of the updates is reduced from $O(N^2)$ to $O(N)$ by proper organization of the updates between the messages between $\{G, L_i\}$ and $\{L_i, x_i\}$. For details, see [4].

We tested our ideas on $C_{23,3}$, thinned along the lines in Section V for the choice $a = 0$ in (18) with appropriate choices for the auxiliary $p + 1$ bits (s_1, s_2, and $\mathbf{m_s}$ for a given $\mathbf{m_u}$). This was decoded after one repetition over an AWGN channel. The simulations so far are very limited in scope [2], but we nevertheless present the results in Fig. 2. The x-axis gives the SNR per message bit. If similar results hold up under more extensive simulations, this will illustrate the benefits of our approach for thinning the code to handle the identification problem in the absence of adequate synchronization.

It is important to ensure good Hamming distance between post repetition codewords of the thinned code. The minimum distance of the array codes $C_{p,j}$ is as yet largely unknown [12], [17]. As an example, we looked at the code $C_{7,5}$. This has minimum distance 12. The post repetition distance of this code is zero. With a judicious assignment of s_1, s_2 and $\mathbf{m_s}$ in (19) for $a = 0$, the post repetition Hamming distance of the thinned code is at least 5, except for one pair of codewords which has a minimum distance of 3.

VII. Concluding remarks

We proposed a technique for modifying array-based LDPC codes when varying sampling rate may cause repetition of symbols. Allowing a small loss in rate, we systematically expurgate the code to get a thinned code with significantly improved synchronization error correction properties. We also gave a scheme for constructing t-repetitions correcting families of sequences. Incorporating multiple synchronization error correction capabilities in array-based LDPC codes and other codes of interest is a topic for future research.

[2] At each SNR point on each curve, 25 codewords were selected at random. For every one of the possible repetition locations, the codeword after repetition was passed once through an AWGN channel and decoded using the proposed message passing algorithm for 100 iterations.

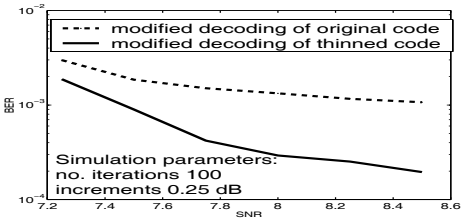

Fig. 2. Performance of LDPC (529,462) over AWGN with one repetition

Acknowledgment

The authors would like to thank Marvell Semiconductor Inc. and U.C. MICRO program for supporting their research.

References

[1] S. Aji and R. McEliece, "The generalized distributive law", *IEEE Trans. Inform. Theory* vol. 46(2), pp. 325–43, March 2000.

[2] G. Chen, M. Mitzenmacher, C. Ng and N. Varnica, "Concatenated codes for deletion channels," *Int. Symp. Inform. Theory*, 2003, p. 218.

[3] M.C. Davey and D.J.C. MacKay, "Reliable communication over channels with insertions, deletions and substitutions," *IEEE Trans. Inf. Theory*, vol. 47(2), pp. 687–98, Feb. 2001.

[4] L. Dolecek and V. Anantharam, "On array-based LDPC codes in channels with varying sampling rate," available at www.eecs.berkeley.edu/~dolecek/papers

[5] L. Dolecek and V. Anantharam, "Using Reed-Muller codes in channels with synchronization and substitution errors," available at www.eecs.berkeley.edu/~dolecek/papers

[6] E. Eleftheriou and S. Ölçer, "Low density parity check codes for digital subscriber lines," *Int. Conf. on Comm.*, 2002, pp. 1752–57.

[7] J. L. Fan, "Array-codes as low-density parity-check codes," *Second Int. Symp. on Turbo Codes and Related Topics*, 2000, pp. 543–46.

[8] T. Kløve, "Codes correcting a single insertion/deletion of a zero or a single peak-shift," *IEEE Trans. Inf. Theory*, vol. 41(1), pp. 279–83, Jan. 1995.

[9] P. Kovintavewat, J. R. Barry, M. F. Erden and E. Kurtas, "Per-survivor timing recovery for uncoded partial response channels,"*Int. Conf. on Comm.*, 2004, pp. 2715–19.

[10] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Sov. Phys.-Dokl.*, vol. 10(8), pp. 707–10, Feb. 1966.

[11] J. Liu, H. Song and B.V.K.V. Kumar, "Symbol timing recovery for low-SNR partial response recording channels," *Globecom* 2002, pp. 1129–36.

[12] T. Mittelholzer, "Efficient encoding and minimum distance bounds of Reed-Solomon-type array codes," *Int. Symp. Inform. Theory*, 2002, p. 282.

[13] A. R. Nayak, J. Barry and S. W. McLaughlin, "Joint timing recovery and turbo equalization for coded partial response channels," *IEEE Trans. On Magnetics*, vol. 38(5), pp. 2295–97, Sept. 2002.

[14] N.J.A. Sloane, "On single deletion correcting codes," 2000. Available at http://www.research.att.com/~njas/doc/dijen.pdf

[15] H. Song and B.V.K.V. Kumar, "Low density parity check codes for partial response channels," *IEEE Signal Proc. Magazine*, vol. 21(1), pp. 56–66, Jan 2004.

[16] R. R. Varshamov and G.M. Tenengolts, "Codes which correct single asymmetric errors," *Avtomatika i Telemehkanika*, vol. 26(2), pp. 288–92,1965.

[17] K. Yang and T. Helleseth, "On the minimum distance of array codes as LDPC codes," *IEEE Trans. Inf. Theory*, vol. 49(12), pp. 3268–71, Dec. 2003.