Soft-In Soft-Out Decoding of Reed–Solomon Codes Based on Vardy and Be'ery's Decomposition

Thomas R. Halford, *Student Member, IEEE*,
Vishakan Ponnampalam, *Member, IEEE*,
Alex J. Grant, *Senior Member, IEEE*, and
Keith M. Chugg, *Member, IEEE*

***Abstract*—This correspondence presents an optimal soft-in soft-out (SISO) decoding algorithm for the binary image of Reed–Solomon (RS) codes that is based on Vardy and Be'ery's optimal soft-in hard-out algorithm. A novel suboptimal list-based SISO decoder that exploits Vardy and Be'ery's decomposition is also presented. For those codes with very high rate, which allows practical decoding with the proposed algorithms, the proposed suboptimal SISO significantly outperforms standard list-based decoding techniques in iteratively decoded systems.**

***Index Terms*—Graphical models, Reed–Solomon (RS) codes, soft-in soft-out (SISO) decoding.**

I. Introduction

THE ubiquity and utility of Reed–Solomon (RS) codes are well established (see, for example, [3]). It has been shown that soft-decision decoding (SDD) algorithms can achieve as much as 3 dB of additional coding gain on the additive white Gaussian noise (AWGN) channel in comparison to hard-decision decoding algorithms; however, SDD algorithms are often much more complex [4]. There has thus been a great deal of recent interest in SDD algorithms for RS codes with practically realizable complexity.

Koetter and Vardy recently presented a soft-in hard-out (SIHO) RS decoder that achieves coding gains on the order of 1 dB compared to the Berlekamp–Massey algorithm with a moderate complexity increase [5]. Extensions of Koetter and Vardy's algorithm proposed by Parvaresh and Vardy [6] and El-Khamy, McEliece, and Harel [7] improve upon these results. Liu and Lin recently presented a SIHO decoder for self-concatenated RS codes based on their binary image [8]. SIHO decoding algorithms are not suitable for iterative decoding, however, and soft-in soft-out (SISO) decoders for RS codes are often desired.

Due to their nonbinary nature, trellis representations of RS codes are in general prohibitively complex [9]. In this correspondence, an optimal SISO decoder for the binary image of RS codes is presented based on the SIHO decoding algorithms of Vardy and Be'ery [10] and Ponnampalam and Vucetic [11]. It is shown that Vardy and Be'ery's decomposition implies a cycle-free factor graph and thus an optimal SISO decoding algorithm [12]. As predicted by the Cut-Set Bound [13], [14], the proposed optimal algorithm is necessarily prohibitively complex for large codes; however, for small, high-rate RS codes the proposed algorithm has reasonable complexity. This proposed optimal SISO decoder was described independently by Ponnampalam and Grant [1] and Halford [2].

Manuscript received December 2, 2004; revised May 26, 2005. The work of T. R. Halford was supported in part by the Powell Foundation and by TrellisWare Technologies Inc., where a portion of this work was completed as part of an internship in the summer of 2003. The material in this correspondence was presented in part at the IEEE International Symposium on Information Theory, Yokohama, Japan, June/July 2003.

T. R. Halford and K. M. Chugg are with the Communication Sciences Institute, University of Southern California, Los Angeles CA 90089-2565 USA (e-mail: halford@usc.edu; chugg@usc.edu).

V. Ponnampalam is with IPWireless Ltd., Chippenham, Wiltshire SN15 1BN, U.K. (e-mail: vponnampalam@ipwireless.com).

A. J. Grant is with the Institute for Telecommunications Research, University of South Australia, Mawson Lakes, SA 5095, Australia (e-mail: Alex.Grant@unisa.edu.au).

Communicated by R. J. McEliece, Associate Editor for Coding Theory.
Digital Object Identifier 10.1109/TIT.2005.859287

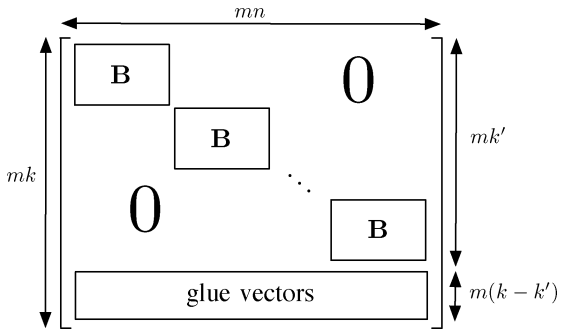

Fig. 1. Structure of the generator matrix of the binary image of $\mathcal{C}_{\rm RS}$ where $\mathbf{B}$ generates $\mathcal{C}_{\rm BCH}$.

Several authors have considered supoptimal SISO RS decoding algorithms. Fossorier and Lin's ordered statistics approach realizes a suboptimal SISO decoding algorithm for the binary image of RS codes [15]. Jiang and Narayan recently presented a suboptimal SISO decoding algorithm for the binary image of RS codes [16]; however, although their algorithm provides soft outputs, there is no indication that these soft outputs are good in an iterative context. The present correspondence proposes a suboptimal list-based SISO decoding algorithm based on Vardy and Be'ery's decomposition. It is shown that for very high-rate RS codes, the proposed algorithm compares favorably to standard list decoding schemes in both complexity and performance.

The remainder of this correspondence is organized as follows. Section II reviews Vardy and Be'ery's decomposition and presents the proposed optimal SISO decoding algorithm. Section III presents the proposed suboptimal list-based SISO decoding algorithm and investigates its performance as a stand-alone decoder and as a constituent decoder in a turbo product code [17]. Section IV gives conclusions and suggests directions for future work.

II. Optimal SISO Decoding of RS Codes

A. The Vardy–Be'ery Decomposition

Let $\mathcal{C}_{\rm RS}$ be an $(n = 2^m - 1, k, d = n - k + 1)$ RS code defined over GF(2^m) with roots $\{\alpha, \alpha^2, \ldots, \alpha^{d-1}\}$, where α is primitive in GF(2^m). Associated with $\mathcal{C}_{\rm RS}$ is the $(n, k' \le k, d' \ge d)$ binary Bose–Chaudhuri–Hocquenghem (BCH) code $\mathcal{C}_{\rm BCH}$ with roots $\{\alpha, \alpha^2, \ldots, \alpha^{d-1}\}$ and their cyclotomic conjugates over GF(2). Let ϕ : GF$(2^m) \rightarrow$ (GF$(2))^m$ be a GF(2)-linear map with basis $\{\gamma_1, \gamma_2, \ldots, \gamma_m\}$. Any element $c_i \in$ GF(2^m) can be written

$$c_i = \sum_{j=1}^{m} \gamma_j c_i^{(j)}, \qquad \text{where } c_i^{(j)} \in \text{GF}(2) \tag{1}$$

and ϕ thus defines the binary image of $\mathcal{C}_{\rm RS}$.

The SIHO algorithms of [10] and [11] were motivated by structural properties of the generator matrix of the binary image of $\mathcal{C}_{\rm RS}$. Vardy and Be'ery proved in [10] that a generator matrix of the binary image of $\mathcal{C}_{\rm RS}$ can be found with the structure shown in Fig. 1 where $\mathbf{B}$ is $k' \times n$ and generates $\mathcal{C}_{\rm BCH}$. The first mk' rows of this structure are block diagonal. The last $m(k - k')$ rows of this structure are denoted *glue vectors* in [10]. The code generated by the glue vectors is denoted the *glue code*.

The structure of Fig. 1 implies that a codeword in the binary image of $\mathcal{C}_{\rm RS}$ is formed by interleaving m codewords drawn from $\mathcal{C}_{\rm BCH}$ and

adding a glue code codeword. Formally, define codes $\mathcal{B}$ and $\mathcal{L}$ over GF(2^m)[1]

$$\mathcal{B} = \left\{ \boldsymbol{b}(X) \middle| \boldsymbol{b}(X) = \sum_{j=1}^{m} \gamma_j \boldsymbol{c}^{(j)}(X),\ \ \boldsymbol{c}^{(j)}(X) \in \mathcal{C}_{\mathrm{BCH}} \right\} \tag{2}$$

$$\mathcal{L} = \left\{ \boldsymbol{l}(X) \middle| \boldsymbol{l}(X) = \sum_{j=1}^{m} \gamma_j \boldsymbol{l}^{(j)}(X), \boldsymbol{l}^{(j)}(X) \in \mathcal{E} \right. \\ \left. \text{and } \boldsymbol{l}(\beta) = 0 \text{ for } \beta \in \{\alpha, \ldots, \alpha^{d-1}\} \right\} \tag{3}$$

where $\mathcal{E}$ is the set of coset leaders of $\mathcal{C}_{\mathrm{BCH}}$. The diagonal blocks of Fig. 1 correspond to the binary image of $\mathcal{B}$; codewords belonging to $\mathcal{B}$ are formed by interleaving BCH codewords. The glue vectors of Fig. 1 correspond to the binary image of $\mathcal{L}$; codewords belonging to $\mathcal{L}$ are formed by interleaving some combination of coset representatives of $\mathcal{C}_{\mathrm{BCH}}$. The RS code $\mathcal{C}_{\mathrm{RS}}$ is a binary linear combination of $\mathcal{B}$ and $\mathcal{L}$.

Example: $(7,5,3)$ *RS Code:* Let $\mathcal{C}_{(7,5,3)}$ be the $(7,5,3)$ RS code defined over GF(8) with generator polynomial

$$\boldsymbol{g}_{(7,5,3)}(X) = (X+\alpha)(X+\alpha^2)$$

and let ϕ : GF$(8) \to$ GF$(2)^3$ have basis $\{1, \alpha, \alpha^2\}$. Specifically, with this basis: $1 \to 100$, $\alpha \to 010$, and $\alpha^2 \to 001$. The associated BCH code $\mathcal{C}_{(7,4,3)}$ has roots α, α^2, and α^4 and thus has dimension 4 and minimum distance 3. A generator matrix for the binary image of $\mathcal{C}_{(7,5,3)}$ with the structure shown in Fig. 1 is as shown in (4) at the bottom of the page. If the coset leaders of $\mathcal{C}_{(7,4,3)}$ are labeled

$$\begin{aligned} \boldsymbol{\epsilon}_0 &= (0,0,0,0,0,0,0) & \boldsymbol{\epsilon}_4 &= (0,0,1,0,0,0,0) \\ \boldsymbol{\epsilon}_1 &= (1,0,0,0,0,0,0) & \boldsymbol{\epsilon}_5 &= (0,0,0,0,0,0,1) \\ \boldsymbol{\epsilon}_2 &= (0,1,0,0,0,0,0) & \boldsymbol{\epsilon}_6 &= (0,0,0,0,1,0,0) \\ \boldsymbol{\epsilon}_3 &= (0,0,0,1,0,0,0) & \boldsymbol{\epsilon}_7 &= (0,0,0,0,0,1,0) \end{aligned} \tag{5}$$

then the coset configurations that satify (3) are

$$(\boldsymbol{\epsilon}_0, \boldsymbol{\epsilon}_0, \boldsymbol{\epsilon}_0), (\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_6, \boldsymbol{\epsilon}_4), (\boldsymbol{\epsilon}_2, \boldsymbol{\epsilon}_7, \boldsymbol{\epsilon}_3), (\boldsymbol{\epsilon}_3, \boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_7), \\ (\boldsymbol{\epsilon}_4, \boldsymbol{\epsilon}_5, \boldsymbol{\epsilon}_6), (\boldsymbol{\epsilon}_5, \boldsymbol{\epsilon}_3, \boldsymbol{\epsilon}_2), (\boldsymbol{\epsilon}_6, \boldsymbol{\epsilon}_2, \boldsymbol{\epsilon}_5), (\boldsymbol{\epsilon}_7, \boldsymbol{\epsilon}_4, \boldsymbol{\epsilon}_1). \tag{6}$$

[1]Throughout this correspondence, codewords are described interchangeably as n-tuples: $\boldsymbol{c} = (c_0, c_1, \ldots, c_{n-1})$, and as polynomials in an indeterminant X: $\boldsymbol{c}(X) = \sum_{i=0}^{n-1} c_i X^i$.

To illustrate that the glue vectors of (4) generate the coset configurations of (6), consider the binary sum of the three glue vectors

$$(1,1,1,0,0,0,0,0,0,0,0,1,1,1,0,0,1,1,1,0,0)$$

and note that

$$\begin{aligned} &(1,1,1,0,0,0,0) \in \mathcal{C}_{(7,4,3)} + \boldsymbol{\epsilon}_7 \\ &(0,0,0,0,1,1,1) \in \mathcal{C}_{(7,4,3)} + \boldsymbol{\epsilon}_4 \\ &(0,0,1,1,1,0,0) \in \mathcal{C}_{(7,4,3)} + \boldsymbol{\epsilon}_1. \end{aligned} \tag{7}$$

The coset configuration corresponding to the sum of the glue vectors is thus $(\boldsymbol{\epsilon}_7, \boldsymbol{\epsilon}_4, \boldsymbol{\epsilon}_1)$. □

B. An Alternate Definition of the Glue Code

Let $\{\boldsymbol{\epsilon}_0, \boldsymbol{\epsilon}_1, \ldots, \boldsymbol{\epsilon}_{|\mathcal{E}|-1}\} = \mathcal{E}$ be the set of coset leaders of $\mathcal{C}_{\mathrm{BCH}}$ and let $\{\boldsymbol{s}_0, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_{|\mathcal{E}|-1}\} = \mathcal{S}$ be the corresponding set of syndromes where $|\mathcal{E}| = 2^{n-k'}$. The map between coset leaders and syndromes is defined by

$$\boldsymbol{s}_i = \boldsymbol{\epsilon}_i \boldsymbol{H}_{\mathrm{BCH}}^{\top} \tag{8}$$

where $\boldsymbol{H}_{\mathrm{BCH}}$ is an $n - k' \times n$ parity-check matrix for $\mathcal{C}_{\mathrm{BCH}}$ and $\top$ denotes matrix transposition.

Let $\mathcal{T}$ be the code obtained from $\mathcal{L}$ by replacing each coset leader $\boldsymbol{l}^{(j)}$ by the corresponding syndrome $\boldsymbol{t}^{(j)} = \boldsymbol{l}^{(j)} \boldsymbol{H}_{\mathrm{BCH}}^{\top}$

$$\mathcal{T} = \left\{ \boldsymbol{t}(X) \middle| \boldsymbol{t}(X) = \sum_{j=1}^{m} \gamma_j \boldsymbol{t}^{(j)}(X), \boldsymbol{t}^{(j)}(X) \in \mathcal{S} \right. \\ \left. \text{and } \boldsymbol{t}(\beta) = 0 \text{ for } \beta \in \{\alpha, \ldots, \alpha^{d-1}\} \right\} \tag{9}$$

where $\{\gamma_1, \gamma_2, \ldots, \gamma_m\}$ defines the mapping ϕ : GF$(2^m) \to$ $(\mathrm{GF}(2))^m$ and is a subset of a basis for the mapping

$$\phi' : \mathrm{GF}(2^{n-k'}) \to (\mathrm{GF}(2))^{n-k'}.$$

Codewords in $\mathcal{T}$ are thus formed by interleaving m binary syndrome vectors to form an m-tuple of symbols drawn from GF$(2^{n-k'})$.

The code defined by (9) has length m and is defined over GF$(2^{n-k'})$. The codes $\mathcal{L}$ and $\mathcal{T}$ are clearly closely related; the term glue code is

$$\boldsymbol{G}_{(7,5,3)} = \begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} \\ \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{4}$$

used to denote $\mathcal{L}$ and $\mathcal{T}$ interchangeably throughout this correspondence. Codewords $\boldsymbol{l}(X) \in \mathcal{L}$ and $\boldsymbol{t}(X) \in \mathcal{T}$ must both satisfy $\boldsymbol{l}(\beta) = \boldsymbol{t}(\beta) = 0$ for $\beta \in \{\alpha, \ldots, \alpha^{d-1}\}$. When considering the generation or encoding of a codeword, the $\mathcal{L}$ representation of the glue code is more useful as was demonstrated in the example of Section II-A. As will be seen in Section II-C and Section III, however, when considering the decoding of a codeword, the $\mathcal{T}$ representation of the glue code is more useful.

Example: $(7, 5, 3)$ *RS Code:* The glue code $\mathcal{T}$ of the $(7, 5, 3)$ RS code is defined over GF$(2^{n-k'}) =$ GF(8), has roots $\{\alpha, \alpha^2\}$, and is thus the $(7, 5, 3)$ RS code shortened to length 3 with generator matrix

$$\boldsymbol{G}_{\mathcal{T}} = [\,\alpha^3 \quad \alpha^4 \quad 1\,] \tag{10}$$

over GF(8). The columns of the binary image of $\boldsymbol{G}_{\mathcal{T}}$ can be rearranged so that the bits from each syndrome are grouped together yielding a generator matrix for the resulting binary code $\mathcal{T}'$ that is systematic in $\boldsymbol{t}^{(1)}(X)$

$$\boldsymbol{G}_{\mathcal{T}'} = \begin{array}{c} \begin{matrix} t_0^{(1)} & t_1^{(1)} & t_2^{(1)} & t_0^{(2)} & t_1^{(2)} & t_2^{(2)} & t_0^{(3)} & t_1^{(3)} & t_2^{(3)} \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \end{array}. \tag{11}$$

Eight syndrome 3-tuples, or *syndrome configurations*, are generated by $\boldsymbol{G}_{\mathcal{T}'}$

$$(\boldsymbol{s}_0, \boldsymbol{s}_0, \boldsymbol{s}_0), (\boldsymbol{s}_1, \boldsymbol{s}_6, \boldsymbol{s}_4), (\boldsymbol{s}_2, \boldsymbol{s}_7, \boldsymbol{s}_3), (\boldsymbol{s}_3, \boldsymbol{s}_1, \boldsymbol{s}_7), (\boldsymbol{s}_4, \boldsymbol{s}_5, \boldsymbol{s}_6), (\boldsymbol{s}_5, \boldsymbol{s}_3, \boldsymbol{s}_2), (\boldsymbol{s}_6, \boldsymbol{s}_2, \boldsymbol{s}_5), (\boldsymbol{s}_7, \boldsymbol{s}_4, \boldsymbol{s}_1). \tag{12}$$

The set of syndrome configurations in (12) are in one-to-one correspondence with the set of coset configurations in (6) via (8). Note that not every syndrome 3-tuple corresponds to a codeword in $\mathcal{T}$. For example, $(\boldsymbol{s}_1, \boldsymbol{s}_1, \boldsymbol{s}_1) \notin \mathcal{T}$. Syndrome configurations in $\mathcal{T}$ are denoted *valid syndrome configurations*. The corresponding coset configurations in $\mathcal{L}$ are denoted *valid coset configurations*. □

C. RS Code Factor Graph

The generator matrix structure seen in Fig. 1 implies a cycle-free factor graph for RS codes. The RS factor graph consists of m parallel n-stage binary trellises and an additional glue node as illustrated in Fig. 2, where variables are represented by circular vertices, state variables by double circles, and local constraints by square vertices. The binary trellises correspond to $\mathcal{C}_{\mathrm{BCH}}$ and are constructed using the Wolf method [18]. The final trellis stage is a $2^{n-k'}$-ary variable node corresponding to the cosets, or equivalently the syndromes, of $\mathcal{C}_{\mathrm{BCH}}$. The node connecting the final trellis stages corresponds to the glue code.

Coded bits are labeled $\{c_i^{(j)}\}_{i=0,\ldots,n-1;j=1,\ldots,m}$ and uncoded bits are similarly labeled $\{a_i^{(j)}\}_{i=0,\ldots,k-1;j=1,\ldots,m}$. If there is no *a priori* soft information on uncoded bits then the corresponding sections of the factor graph are ignored. If there is *a priori* soft information on uncoded bits and if a systematic encoder is used then the equality constraints in the corresponding sections of the factor graph enforce $a_i^{(j)} = c_i^{(j)}$ for $i = 0, \ldots, k-1$ and $j = 1, \ldots, m$.

In [10], Vardy and Be'ery noted that generator matrices with the structure shown in Fig. 1 can be found for any code containing a subfield subcode. Accordingly, there exist factor graph representations similar to that shown in Fig. 2 for codes containing subfield subcodes. Specifically, such factor graphs can be found for shortened and extended RS codes [19].

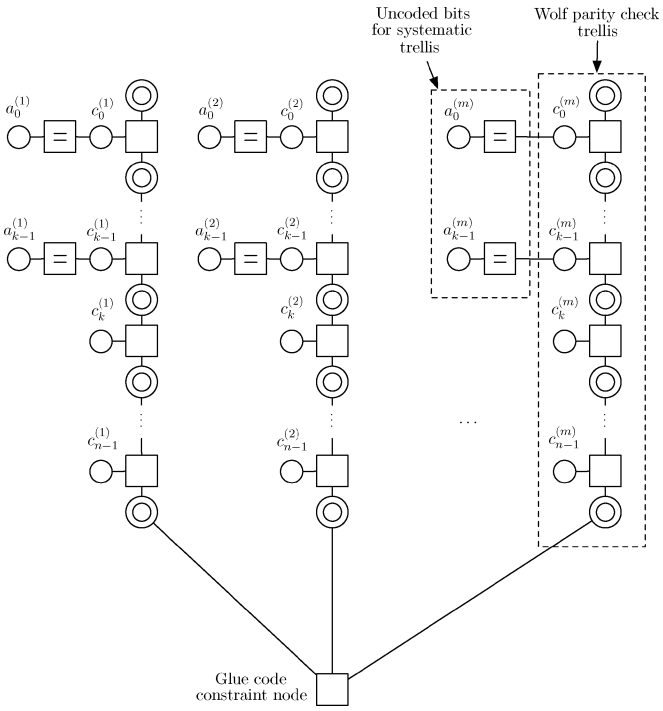

Fig. 2. RS code factor graph based on the Vardy–Be'ery decomposition.

D. Optimal SISO Decoding

Using the factor graph shown in Fig. 2, the following (nonunique) message-passing schedule ensures optimal SISO decoding. An inward recursion is performed on each trellis in parallel corresponding to the forward recursion of the standard forward–backward algorithm (FBA) on a trellis. The forward state messages for the the final state then act as soft input to an optimal SISO decoding of the glue code. The backward state metrics of the trellises are initialized with the soft output of the glue code SISO decoder. The outward recursion on each trellis is then performed in parallel corresponding to the backward recursion of the standard FBA. After the forward and backward metrics are computed at each trellis state, soft-out information on coded (and possibly uncoded) bits is obtained as per the FBA.

Optimal SISO decoding of $\mathcal{C}_{\mathrm{RS}}$ requires optimal SISO decoding of the glue code. One such optimal SISO decoder uses a trellis. When $|\mathcal{T}|$ is small, a trellis need not be used and the SISO decoding of the glue code proceeds as follows. The metric associated with each valid syndrome configuration $\boldsymbol{t} \in \mathcal{T}$ is computed by combining soft-input information on the individual syndromes. The soft output on each syndrome is then found by marginalizing over all configurations consistent with that syndrome. This process is denoted *exhaustive combination and marginalization*.

E. Complexity of the Optimal SISO

The complexity of SISO decoding of RS codes using a Wolf trellis grows as the number of trellis states [20]

$$O(2^{m(n-k)}).$$

For most RS codes, the complexity of the proposed optimal SISO decoder is dominated by the complexity of the glue code SISO decoder. If the glue code is decoded via exhaustive combination and marginaliza-tion, then the complexity of the glue code SISO grows as the number of glue codewords $2^{m(k-k')}$. If the glue code is decoded with a trellis then the complexity of the glue code SISO grows as the

TABLE I
COMPLEXITY COMPARISON OF PROPOSED OPTIMAL SISO AND WOLF TRELLIS DECODERS FOR VARIOUS REED-SOLOMON CODES. COMPLEXITY IS MEASURED BY THE BASE-2 LOGARITHM OF REAL OPERATIONS PER CODEWORD

m	n	k	k'	Wolf Trellis Complexity	Proposed SISO Complexity
3	7	3	1	19.0	13.1
		5	4	13.0	10.0
4	15	9	5	32.5	19.8
		11	7	24.5	19.2
		13	11	16.5	13.0
5	31	25	16	39.9	38.8
		27	21	29.9	28.2
		29	26	19.9	17.5

number of states in the glue code trellis representation $2^{m(n-k)}$. The complexity of the proposed optimal SISO decoder thus grows as

$$O(\min(2^{m(n-k)}, 2^{m(k-k')})).$$

For RS codes where $k - k' \leq n - k$, the proposed optimal SISO decoder is much less complex than trellis decoding. This set includes the $(7,3,5)$, $(7,5,3)$, $(15,9,7)$, $(15,11,5)$, and $(15,13,3)$ codes. For larger RS codes, the complexity of the Wolf trellis and proposed SISO decoders both grow as $O(2^{m(n-k)})$ and neither decoder is practically realizable in accordance with the Cut-Set Bound [13], [14]. Moreover, the Cut-Set Bound precludes the existence of any practically realizable optimal SISO decoding algorithms for large RS codes.

A more specific complexity comparison is made by first estimating the number of real operations required by the FBA on an N-stage, S-state Wolf trellis. The forward and backward recursions each require 1 add–compare–select operation, or three real operations, per state, per stage [11]. The completion step requires $2\log_2 S$ real operations per stage. Exhaustive combination and marginalization of a length-m code requires $2m$ real operations per codeword. Table I compares the complexity of the proposed optimal SISO and Wolf trellis decoders for a number of RS codes. Complexity is given as the base-2 logarithm of real operations per codeword.

III. SUBOPTIMAL SISO DECODING OF RS CODES

For high-rate RS codes, the complexity of the proposed optimal SISO decoder is dominated by the complexity of the optimal glue code SISO decoder. Practically realizable *suboptimal* SISO decoding algorithms for high-rate RS codes can be developed by replacing the optimal glue code SISO decoder by a suboptimal glue code SISO decoder. This correspondence proposes such a decoder that uses a list-based glue code SISO decoder.

List-based SISO decoders for linear block codes have been examined extensively in the literature (see, for example, [21] and the references therein). List-based SISO decoders first produce a list of likely codewords $\mathcal{K} \subset \mathcal{C}$ and then perform the marginalization described in Section II-D over $\mathcal{K}$ rather than all codewords $\mathcal{C}$. The complexity of list-based SISO decoders depends on $K = |\mathcal{K}|$ and can thus be controlled.

A. Generic Glue Code List Generation

The generation of $\mathcal{K}$ is the most difficult design aspect of list-based SISO decoders [21]. The following presents a generic approach to list generation for the glue code.

As described in Section II-B, codewords of $\mathcal{T}$ are BCH syndrome m-tuples (or configurations). Specifically, let $\boldsymbol{w}$ be the syndrome configuration

$$\boldsymbol{w} = \left(\boldsymbol{s}_{\boldsymbol{w}}^{(1)}, \ldots, \boldsymbol{s}_{\boldsymbol{w}}^{(m)}\right) \in \mathcal{S}^m \tag{13}$$

Input: $\mathrm{MI}[\mathbf{s}^{(i)}]$ for all $\mathbf{s} \in \mathcal{S}$ and $i = 1, \ldots, m$.
Output: List of glue codewords with K-shortest syndrome configuration metrics.
$\mathcal{K} \leftarrow \emptyset$;
$codewords\_found \leftarrow 0$;
repeat
  $\mathbf{w} \leftarrow$ syndrome configuration with next smallest syndrome configuration metric;
  if $\boldsymbol{w} \in \mathcal{T}$ **then**
    $\mathcal{K} \leftarrow \mathcal{K} \cup \{\mathbf{w}\}$;
    $codewords\_found \leftarrow codewords\_found + 1$;
  end
until $codewords\_found = K$;

Algorithm 1. Generic glue code list generation.

where $\mathcal{S}$ is the set of BCH syndromes. Associated with $\boldsymbol{w}$ is the *syndrome configuration metric*

$$\mathrm{M}[\boldsymbol{w}] = \sum_{i=1}^{m} \mathrm{MI}\left[\boldsymbol{s}_{\boldsymbol{w}}^{(i)}\right] \tag{14}$$

where $\mathrm{MI}[\boldsymbol{s}_{\boldsymbol{w}}^{(i)}]$ is the final state metric corresponding to the syndrome $\boldsymbol{s}_{\boldsymbol{w}}^{(i)}$ in the ith parallel BCH trellis.[2] Recall from Section II-B that the set of all syndrome configurations is a superset of the glue code $\mathcal{T}$. **Algorithm 1** generates a list of likely codewords by generating a list of likely syndrome configurations and throwing out those configurations not contained in $\mathcal{T}$. The next shortest path algorithm described in [23] is used to obtain the likely syndrome configurations; this algorithm was used successfully for list detection in multiple-access channels in [24].

B. Glue Code List Generation for $(n, n-2, 3)$ RS Codes

Algorithm 1 is inefficient because in order to generate a list of K codewords, many more than K syndrome configurations must be generated. The following presents a reduced-complexity list generation algorithm for the glue codes corresponding to the $(n, n-2, 3)$ RS codes that exploits the algebraic structure of $\mathcal{T}$. The authors have developed similar reduced-complexity glue code list generation algorithms for the glue codes corresponding to the $(n, n-4, 5)$ RS codes and the $(15, 9, 7)$ code [19]; these are omitted for the sake of brevity.

Let $\mathcal{C}_{\mathrm{RS}}$ be an $(n = 2^m - 1, n-2, 3)$ RS code with roots α and α^2 where α is primitive in GF(2^m). Since $\alpha^{2^m} = \alpha$ in GF(2^m), the union of the sets of cyclotomic conjugates of α and α^2 over GF(2) is $\{\alpha, \alpha^2, \ldots, \alpha^{2^m-1}\}$. The associated BCH code $\mathcal{C}_{\mathrm{BCH}}$ thus has dimension

$$k' = k - (m+2) = n - m. \tag{15}$$

For the $(n, n-2, 3)$ RS codes, the glue code $\mathcal{T}$ is, therefore, a length-m code defined over GF(2^m) with roots $\{\alpha, \alpha^2\}$. Since $|\mathcal{T}| = 2^{m(k-k')} = 2^{m(m-2)}$, the dimension of the glue code is $m-2$ and $\mathcal{T}$ is a shortened $(n, n-2, 3)$ RS code.

As per the example of Section II-B, let $\boldsymbol{G}_{\mathcal{T}}$ generate $\mathcal{T}$ over GF(2^m) and let $\boldsymbol{G}_{\mathcal{T}'}$ be the binary image of $\boldsymbol{G}_{\mathcal{T}}$ with columns reordered sothat bits from each syndrome are grouped together. For the specific $(n, n-2, 3)$ RS codes considered in this correspondence, binary generators matrices that are systematic in the bits corresponding to $\boldsymbol{t}^{(1)}, \ldots, \boldsymbol{t}^{(m-2)}$ were obtained.

[2]Note that min-sum or min*-sum processing is assumed and combination is achieved via addition of metrics rather than multiplication of probabilities. Metrics are negative logartithms of probabilities [22].

Input: $\mathrm{MI}[\mathbf{s}^{(i)}]$ for all $\mathbf{s} \in \mathcal{S}$ and $i = 1, \ldots, m-2$.
Output: List of glue codewords with K-shortest syndrome sub-configuration metrics.
$\mathcal{K} \leftarrow \emptyset$;
for $k = 1, \ldots, K$ **do**
  $\mathbf{w}_{|m-2} \leftarrow$ syndrome sub-configuration with next smallest syndrome sub-configuration metric;
  $\mathbf{w}'_{|m-2} \leftarrow$ binary image of $\mathbf{w}_{|m-2}$;
  $\mathbf{t}' \leftarrow \mathbf{w}'_{|m-2}\mathbf{G}_{\mathcal{T}'}$;
  $\mathbf{t} \leftarrow$ glue code codeword corresponding to $\mathbf{t}'$;
  $\mathcal{K} \leftarrow \mathcal{K} \cup \{\mathbf{t}\}$;
end

Algorithm 2. List generation for $(n, n-2, 3)$ RS codes.

Define a *syndrome subconfiguration*, $\boldsymbol{w}_{|m-2}$, as the first $m-2$ elements of a syndrome configuration $\boldsymbol{w} \in \mathcal{S}^m$

$$\boldsymbol{w}_{|m-2} = \left(\boldsymbol{s}_{\boldsymbol{w}}^{(1)}, \ldots, \boldsymbol{s}_{\boldsymbol{w}}^{(m-2)}\right) \in \mathcal{S}^{m-2} \tag{16}$$

Associated with $\boldsymbol{w}_{m-2}$ is the *syndrome subconfiguration metric*:

$$\mathrm{M}[\boldsymbol{w}_{|m-2}] = \sum_{i=1}^{m-2} \mathrm{MI}\left[\boldsymbol{s}_{\boldsymbol{w}}^{(i)}\right] \tag{17}$$

where $\mathrm{MI}[\boldsymbol{s}_{\boldsymbol{w}}^{(i)}]$ is defined as per Section III-A. **Algorithm 2** generates a list of likely codewords by generating a list of likely syndrome subconfigurations and exploits the structure of $\mathcal{T}$ to determine a glue code codeword corresponding to each subconfiguration. Note that only K subconfigurations need be generated since each subconfiguration generates a single codeword. Also note that the lists of codewords produced by Algorithms 1 and 2 may not be identical since Algorithm 2 uses metrics from only $m-2$ trellises in its shortest path search.

Since RS codes are maximum distance separable (MDS) over GF(2^m), Algorithm 2 can readily be adapted to generate a list of likely RS codewords by considering configurations of GF(2^m) symbols and the corresponding symbol-level soft information. The resulting list-based SISO decoder is denoted the standard list-based SISO in the following section since it does not exploit the Vardy–Be'ery decomposition.

C. Simulation Results and Discussion

In this subsection, the performance of the proposed suboptimal SISO decoder is compared to that of both the proposed optimal SISO decoder and the standard list-based SISO decoder. Note that the list generation algorithms described in Section III-B are used rather than Algorithm 1. Binary antipodal signaling over AWGN channels is assumed throughout.

Fig. 3 compares the performance of the three algorithms when used as stand-alone decoders for the $(15, 13, 3)$ and $(15, 11, 5)$ codes. The glue codes of the $(15, 13, 3)$ and $(15, 11, 5)$ contain 256 and 65 536 codewords, respectively. A negligible performance loss is incurred by the proposed suboptimal decoders when respective glue code list sizes of 32 and 1024 are used. Observe that the standard list-based SISO decoder with list size 1024 incurs a 0.5-dB loss with respect to the proposed algorithms. Generating 1024 codewords of the $(15, 11, 5)$ glue code, which is a length–4 code over GF(256), is substantially less complex than generating 1024 codewords of the $(15, 11, 5)$ code over GF(16).

The proposed suboptimal SISO was also compared to the proposed optimal SISO for the $(31, 29, 3)$ and $(63, 61, 3)$ codes. It was found that respective glue code list sizes of 64 and 128 were required in order to approximate optimal performance for these codes.

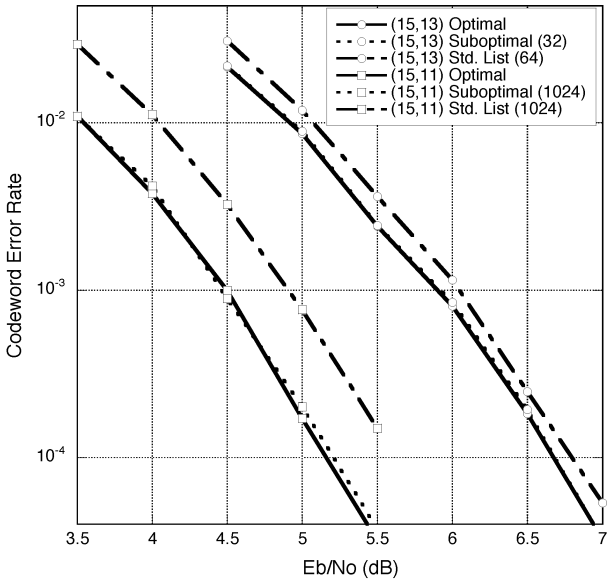

Fig. 3. Codeword error rate performance comparison of the proposed optimal SISO, proposed suboptimal SISO and standard list-based SISO for the $(15, 13, 3)$ and $(15, 11, 5)$ RS codes. List sizes appear in parentheses.

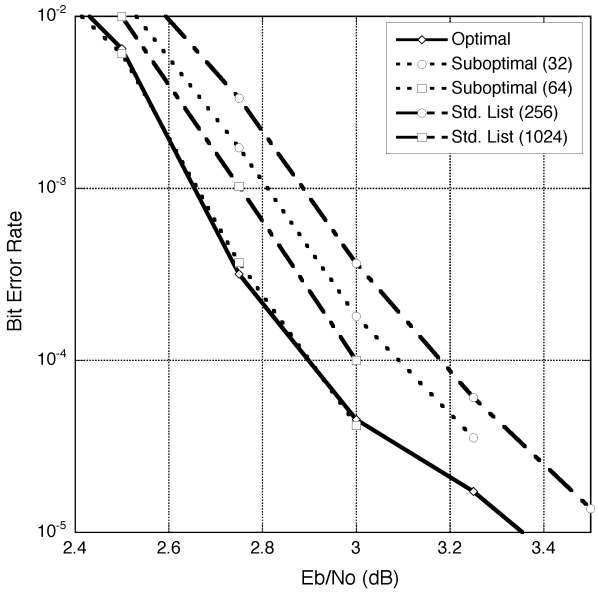

Fig. 4. Bit-error rate performance comparison of the proposed optimal SISO, proposed suboptimal SISO, and standard list-based SISO for the rate $\frac{13}{17}$ $(15, 13, 3) \times (15, 13, 3)$ RS turbo product code. Ten decoding iterations were performed. List sizes appear in parentheses.

In order to investigate the quality of soft-out information produced by the proposed suboptimal SISO decoder, its performance was compared to that of the proposed optimal and standard list-based SISO decoders in an iteratively decoded system. Specifically, a rate $\frac{13}{17}$ $(15, 13, 3) \times (15, 13, 3)$ RS turbo product code was considered with input block length 2704 bits. A high-spread pseudorandom bit-level interleaver was constructed using the real-relaxation optimization method described in [25].

Fig. 4 illustrates the performance of the iterative turbo product decoder after ten iterations using five different RS SISO decoders: the proposed optimal SISO, the proposed suboptimal SISO with glue code

list sizes 32 and 64 and the standard list-based SISO with list sizes 256 and 1024. With a list size of 64, the decoder employing the suboptimal SISO incurs a negligible loss with respect to the decoder employing the optimal SISO. With a list size of 256, the decoder employing the standard list-based SISO performs approximately 0.3 dB worse than the decoder employing the optimal SISO at a bit-error rate of 10^{-4}; increasing the list size to 1024 narrows, but does not close, this performance gap. Generating 64 codewords of the $(15, 13, 3)$ codeword, which is a length4 code defined over GF(16), is much less complex than generating 1024 codewords of the $(15, 13, 5)$ code over GF(16).

IV. Conclusion and Future Work

In this correspondence, an optimal SISO decoding algorithm for RS codes has been proposed. The proposed optimal SISO decoder employs a cycle-free graphical representation that is an alternative to conventional trellis-based decoding. As predicted by the Cut-Set Bound, the proposed optimal SISO is of reasonable complexity only for small, high-rate codes. Suboptimal SISO decoding algorithms for RS codes were thus motivated.

A suboptimal SISO decoder for high-rate RS codes that exploits Vardy and Be'ery's decomposition of the binary image of RS codes [10] was also proposed. This suboptimal SISO was found to outperform a standard list-based SISO decoding algorithm as a stand-alone decoder and as a constituent SISO decoder in a turbo product code. Furthermore, the complexity of glue code list generation is less than that of standard list generation for the RS code because the glue codes have length m whereas the full RS codes have length $2^m - 1$. An interesting area for future work is to investigate the use of SISO ordered statistics decoding [15] of the glue code and to compare the resulting suboptimal SISO with a SISO employing ordered statistics decoding of the full RS code.

The proposed suboptimal algorithm is of practically realizable complexity only for very high-rate codes. However, very high-rate RS codes are highly relevant as component codes in iteratively decodable systems. Generating 64 codewords of the $(15, 13, 3)$ glue code, which is a length-4 code defined over GF(16), is much less complex than generating 1024 codewords of the full $(15, 13, 3)$ code over GF(16).

References

[1] V. Ponnampalam and A. Grant, "An efficient SISO for reed-solomon codes," in *Proc. IEEE Symp. Information Theory*, Yokohama, Japan, Jun./Jul. 2003, p. 204.

[2] T. R. Halford, "Optimal Soft-In Soft-Out Decoding of Reed-Solomon Codes," Communication Sciences Inst., USC, Los Angeles, CA, Tech. Rep. CSI-04-05-03, Apr. 2003.

[3] S. B. Wicker and V. K. Bhargava, *Reed-Solomon Codes and Their Applications*. New York: Wiley, 1999.

[4] J. G. Proakis, *Digital Communications*, 3rd ed. New York: McGraw-Hill, 1995.

[5] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes," *IEEE Trans. Inf. Theory*, vol. 49, no. 11, pp. 2809–2825, Nov. 2003.

[6] F. Parvaresh and A. Vardy, "Multiplicity assignments for algebraic soft-decoding of Reed-Solomon codes," in *Proc. IEEE Symp. Information Theory*, Yokohama, Japan, Jun./Jul. 2003, p. 205.

[7] M. El-Khamy, R. J. McEliece, and J. Harel, "Performance enhancements for algebraic soft decision decoding of Reed-Solomon codes," in *Proc. IEEE Symp. Information Theory*, Chicago, IL, Jun./Jul. 2004, p. 419.

[8] C. Y. Liu and S. Lin, "Turbo encoding and decoding of Reed-Solomon codes through binary decomposition and self-concatenation," *IEEE Trans. Communun.*, vol. 52, no. 9, pp. 1484–1493, Sep. 2004.

[9] S. K. Shin and P. Sweeney, "Soft decision decoding of Reed-Solomon codes using trellis methods," *Proc. Inst. Elec. Eng.–Communications*, vol. 141, no. 5, pp. 303–308, Oct. 1994.

[10] A. Vardy and Y. Be'ery, "Bit level soft-decision decoding of Reed-Solomon codes," *IEEE Trans. Communun.*, vol. 39, no. 3, pp. 440–444, Mar. 1991.

[11] V. Ponnampalam and B. Vucetic, "Soft decision decoding of Reed-Solomon codes," *IEEE Trans. Communun.*, vol. 50, no. 11, pp. 1758–1768, Nov. 2002.

[12] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[13] N. Wiberg, "Codes and Decoding on General Graphs," Ph.D. dissertation, Linköping Univ., Linköping, Sweden, 1996.

[14] G. D. Forney Jr., "Codes on graphs: Normal realizations," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 520–548, Feb. 2001.

[15] M. P. C. Fossorier and S. Lin, "Soft-input soft-output decoding of linear block codes based on ordered statistics," in *Proc. GLOBECOM Conf.*, Sydney, Australia, Nov. 1998, pp. 2828–2833.

[16] J. Jiang and K. R. Narayanan, "Iterative soft decision decoding of Reed-Solomon codes based on adaptive parity check matrices," in *Proc. IEEE Symp. Inf. Theory*, Chicago, IL, Jun./Jul. 2004, p. 261.

[17] R. M. Pyndiah, "Near optimum decoding of product codes: Block turbo codes," *IEEE Trans. Communun.*, vol. 46, no. 8, pp. 1003–1010, Aug. 1998.

[18] J. K. Wolf, "Efficient maximum-likelihood decoding of linear block codes using a trellis," *IEEE Trans. Inf. Theory*, vol. IT-24, no. 1, pp. 76–80, Jan. 1978.

[19] T. R. Halford, "Systematic Extraction of Good Cyclic Graphical Models for Inference," Ph.D. dissertation proposal, Communication Sciences Institute, USC, Los Angeles, CA, Nov. 2004. Tech. Rep. CSI-05-01-02.

[20] S. Lin, T. Kasami, T. Fujiwara, and M. Fossorier, *Trellises and Trellis-Based Decoding Algorithms for Linear Block Codes*. Norwell, MA: Kluwer Academic, 1998.

[21] P. A. Martin, D. P. Taylor, and M. P. C. Fossorier, "Soft-input soft-output list-based decoding algorithm," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 252–262, Feb. 2004.

[22] K. M. Chugg, A. Anastasopoulos, and X. Chen, *Iterative Detection: Adaptivity, Complexity Reduction, and Applications*. Norwell, MA: Kluwer Academic, 2001.

[23] D. Eppstein, "Finding the k shortest paths," Dept. Info. and Comp. Sci., Univ. California, Irvine, CA, Tech. Rep, Mar. 1997.

[24] A. B. Reid, A. J. Grant, and P. D. Alexander, "List detection for multi-access channels," in *Proc. GLOBECOM Conf.*, vol. 2, Taipei, Taiwan, R.O.C., Nov. 2002, pp. 1083–1087.

[25] S. Crozier, "New high-spread high-distance interleavers for turbo-codes," in *Proc. 20th Biennial Symp. Communications*, Kingston, ON, Canada, May 2000, pp. 3–7.