Iterative Decoding on Multiple Tanner Graphs Using Random Edge Local Complementation

Joakim Grahl Knudsen, Constanza Riera*, Lars Eirik Danielsen, Matthew G. Parker, and Eirik Rosnes
Dept. of Informatics, University of Bergen, Thormøhlensgt. 55, 5008 Bergen, Norway
Email: {joakimk, larsed, matthew, eirik}@ii.uib.no
*Bergen University College, Nygårdsgt. 112, 5008 Bergen, Norway. Email: csr@hib.no

***Abstract*—In this paper, we propose to enhance the performance of the sum-product algorithm (SPA) by interleaving SPA iterations with a random local graph update rule. This rule is known as edge local complementation (ELC), and has the effect of modifying the Tanner graph while preserving the code. We have previously shown how the ELC operation can be used to implement an iterative permutation group decoder (SPA-PD)–one of the most successful iterative soft-decision decoding strategies at small blocklengths. In this work, we exploit the fact that ELC can also give structurally distinct parity-check matrices for the same code. Our aim is to describe a simple iterative decoder, running SPA-PD on distinct structures, based entirely on random usage of the ELC operation. This is called SPA-ELC, and we focus on small blocklength codes with strong algebraic structure. In particular, we look at the extended Golay code and two extended quadratic residue codes. Both error rate performance and average decoding complexity, measured by the average total number of messages required in the decoding, significantly outperform those of the standard SPA, and compares well with SPA-PD. However, in contrast to SPA-PD, which requires a global action on the Tanner graph, we obtain a performance improvement via local action alone. Such localized algorithms are of mathematical interest in their own right, but are also suited to parallel/distributed realizations.**

I. Introduction

Inspired by the success of iterative decoding of low-density parity-check (LDPC) codes, originally introduced by Gallager [1] and later rediscovered in the mid 1990's by MacKay and Neal [2], on a wide variety of communication channels, the idea of iterative, soft-decision decoding has recently been applied to classical algebraically constructed codes in order to achieve low-complexity belief propagation decoding [3–8]. Also, the classical idea of using the automorphism group of the code, $\mathrm{Aut}(\mathcal{C})$, to permute the code, $\mathcal{C}$, during decoding (known as *permutation decoding* (PD) [9]) has been successfully modified to enhance the sum-product algorithm (SPA) in [5]. We will denote this algorithm by SPA-PD. Furthermore, good results have been achieved by running such algorithms on several structurally distinct representations of $\mathcal{C}$ [3, 6]. Both Reed-Solomon and Bose-Chaudhuri-Hocquenghem (BCH) codes have been considered in this context. Certain algebraically constructed codes are known to exhibit large minimum distance and a non-trivial $\mathrm{Aut}(\mathcal{C})$. However, additional properties come into play in modern, graph-based coding theory, for instance, sparsity, girth, and trapping sets [10, 11]. Structural weaknesses of graphical codes are inherent to the particular parity-check matrix, H, used to implement $\mathcal{C}$ in the decoder. This matrix is a non-unique $(n-k)$-dimensional basis for the null space of $\mathcal{C}$, which, in turn, is a k-dimensional subspace of $\{0,1\}^n$. Although any basis (for the dual code, $\mathcal{C}^\perp$) is a parity-check matrix for $\mathcal{C}$, their performance in decoders is not uniform. H is said to be in standard form if the matrix has $n-k$ weight-1 columns. The weight of H is the number of non-zero entries, and the minimum weight is lower-bounded by $(n-k)d_{\min}(\mathcal{C}^\perp)$, where $d_{\min}(\mathcal{C}^\perp)$ denotes the minimum distance of $\mathcal{C}^\perp$. It is well-known that H can be mapped into a bipartite (Tanner) graph, $\mathbf{TG}(H)$, which has an edge connecting nodes v_i and u_j *iff* $H_{ji} \neq 0$. Here, $v_i, 0 \leq i < n$, refers to the bit nodes (columns of H), and $u_j, 0 \leq j < n-k$, refers to the check nodes (rows of H). The local neighborhood of a node, v, is the set of nodes adjacent to v, and is denoted by $\mathcal{N}_v$. The terms standard form and weight extend trivially to $\mathbf{TG}(H)$. In the following, we use bold face notation for vectors, and the transpose of H is written H^T.

This paper is a continuation of our previous work on edge local complementation (ELC) and iterative decoding, in which selective use of ELC (with preprocessing and memory overhead) equals SPA-PD [8]. In this work, we use ELC in a truly random, online fashion, thus simplifying both the description and application of the proposed decoder. The key difference from our previous work is that we do not take measures to preserve graph isomorphism, and explore the benefits of going outside the automorphism group of the code. This means that we alleviate the preprocessing of suitable ELC locations (edges), as well as the memory overhead of storing and sampling from such a set during decoding. Our proposed decoding algorithm can be thought of as a combination of SPA-PD [5] and multiple bases belief propagation [6]. We also discuss the modification of the powerful technique of damping to a graph-local perspective.

II. The ELC Operation

The operation of ELC [12–14], also known as Pivot, is a local operation on a simple graph (undirected with no loops), G, which has been shown to be useful both for code equivalence and classification [13], and for decoding purposes [8]. It has recently been identified as a useful local unitary primitive to be applied to *graph states* [14]–a proposed paradigm for quantum computation [15]. Fig. 1(a) shows $G_{\mathcal{N}_u \cup \mathcal{N}_v}$, the local subgraph of a bipartite graph induced by nodes u, v, and their disjoint neighborhoods which we denote by $\mathcal{N}_u^v \triangleq \mathcal{N}_u \setminus \{v\}$

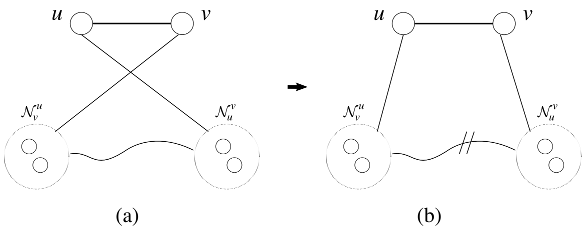


Fig. 1. ELC on edge (u, v) of a bipartite graph. Doubly slashed links mean that the edges connecting the two sets have been complemented.

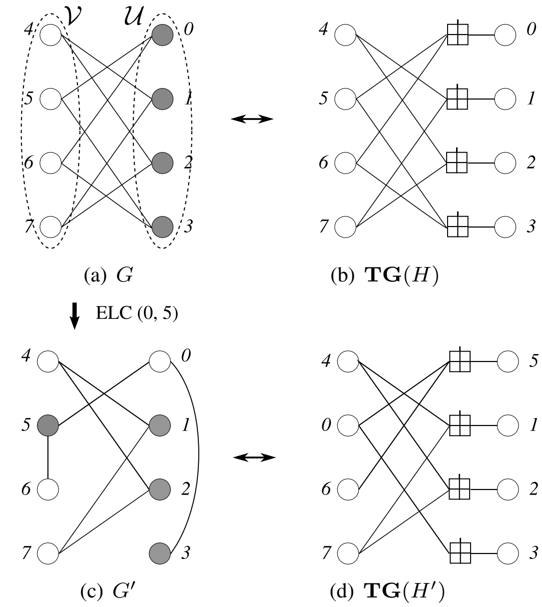


Fig. 2. (a) and (c) show ELC on the edge $(0, 5)$ of a small simple graph G. The corresponding Tanner graphs, in (b) and (d), are distinct structures (e.g., the weight of G and G' is not the same) for the same toy $[8, 4, 2]$ code. This code has a total of three structures in its s-orbit.

and $\mathcal{N}_v^u \triangleq \mathcal{N}_v \setminus \{u\}$, respectively. ELC on a bipartite graph is described as the complementation of edges between these two sets; $\forall v' \in \mathcal{N}_u^v$ and $\forall u' \in \mathcal{N}_v^u$, check whether edge $(u', v') \in G$, in which case it is deleted, otherwise it is created. Finally, the edges adjacent to u and v are swapped – see Fig. 1(b). ELC on G extends easily to ELC on $\mathbf{TG}(H)$ when H is in standard form [8]. Given a bipartite graph with bipartition $(\mathcal{V}, \mathcal{U})$, we then have a one-to-one mapping to a Tanner graph, with check nodes from the set $\mathcal{U}$ and bit nodes from $\mathcal{V} \cup \mathcal{U}$. Fig. 2 shows an example, where the bipartition is fixed according to the sets $\mathcal{V}$ and $\mathcal{U}$. In Fig. 2(a), the left and right nodes correspond to $\mathcal{V}$ and $\mathcal{U}$, respectively, for the simple graph G. $\mathbf{TG}(H)$ may be obtained by replacing grey nodes by a check node singly connected to a bit node, as illustrated in Fig. 2(b). Figs. 2(a) and 2(c) show an example of ELC on the edge $(0, 5)$. Although the bipartition changes (edges adjacent to 0 and 5 are swapped), Figs. 2(b) and 2(d) show how the map to Tanner graphs, in fact, preserves the code.

By complementing the edges of a local neighborhood of $\mathbf{TG}(H)$, ELC has the effect of row additions on H. The complexity of ELC on (u, v) is $\mathcal{O}(|\mathcal{N}_u||\mathcal{N}_v|)$. The set of vertex-labeled graphs generated by ELC on $\mathbf{TG}(H)$ (or, equivalently, G) is here called the *ELC-orbit* of $\mathcal{C}$. Each information set for $\mathcal{C}$ corresponds to a unique graph in the ELC-orbit [13]. Note that this is a code property, which, as such, is independent of the initial parity-check matrix, H. The set of structurally distinct (unlabeled) graphs generated by ELC is here called the *s-orbit* of $\mathcal{C}$, and is a subset of the ELC-orbit. Graphs are structurally distinct (i.e., non-isomorphic) if the corresponding parity-check matrices are not row or column permutations of each other. Each structure in the s-orbit has a set of $|\mathrm{Aut}(\mathcal{C})|$ isomorphic graphs, comprising an *iso-orbit* [8]. In the following, we will refer to ELC directly on $\mathbf{TG}(H)$, keeping Fig. 2 in mind.

III. Decoding Algorithms

A. SPA

The SPA is an inherently local algorithm on $\mathbf{TG}(H)$, where the global problem of decoding is partitioned into a system of simpler subproblems [16]. Each node and its adjacent edges can be considered as a small constituent code, and essentially performs maximum-likelihood decoding (MLD) based on local information. The key to a successful decoder lies in this partitioning–how these constituent codes are interconnected. The summed information contained in a bit node, v_i, is the *a posteriori* probability (APP), $\hat{x}_i$, at codeword position i. The vector $\hat{\mathbf{x}}$ constitutes a tentative decoding of the received channel vector, $\mathbf{y}$. The decoder input is the log-likelihood ratio (LLR) vector $\mathbf{L} = (2/\sigma^2)\mathbf{y}$, where σ is the channel noise standard deviation on an additive white Gaussian noise (AWGN) channel. Subtracting the input from the APP leaves the extrinsic information, $\hat{x}_i - L_i$, which is produced by the decoder. The message on the edge from node v to u, in the direction of u, $\mu_{v \to u}$, is computed according to the SPA rule on node v. The SPA computation of all check nodes, followed by all bit nodes, is referred to as one *flooding* iteration.

Classical codes, for which strong code properties are known, are typically not very suitable for iterative decoding mainly due to the high weight of their parity-check matrices, which gives many short cycles in the corresponding Tanner graphs.

B. Diversity Decoding

A few recent proposals in the literature have attempted to enhance iterative decoding by dynamically modifying $\mathbf{TG}(H)$ during decoding, so as to achieve diversity and avoid fixed points (local optima) in the SPA convergence process. Efforts to improve decoding may, roughly, be divided into two categories. The first approach is to employ several structurally distinct matrices, and use these in a parallel, or sequential, fashion [3, 6]. These matrices may be either preprocessed, or found dynamically by changing the graph during decoding. However, this incurs an overhead either in terms of memory (keeping a list of matrices, as well as state data), or complexity (adapting the matrix, e.g., by Gaussian elimination [3]). The other approach is to choose a code with a non-trivial $\mathrm{Aut}(\mathcal{C})$, such that diversity may be achieved by permuting the code coordinates [4, 5, 7, 8]. An example is SPA-PD, listed in Al-

Algorithm 1 SPA-PD$(I_1, I_2, I_3, \alpha_0)$ [5]

1: // Input: $(\mathbf{y}, H, \alpha_0, I_1, I_2, I_3)$.
2: // Output: $\Theta^{-1}(\hat{\mathbf{x}})$.

3: $\alpha \leftarrow \alpha_0$.
4: **for** I_3 times **do**
5: $\quad \mathbf{L} \leftarrow (2/\sigma^2)\mathbf{y}$ and $\Theta \leftarrow \pi_0$ // identity permutation.
6: $\quad$ **for** I_2 times **do**
7: $\quad\quad \mu_{v \to u} \leftarrow L_v, \ \forall (u, v) \in \mathbf{TG}(H)$.
8: $\quad\quad$ Do I_1 flooding iterations, $\hat{\mathbf{x}} \leftarrow \mathrm{SPA}(\mathbf{TG}(H))$.
9: $\quad\quad$ Take the hard decision of $\hat{\mathbf{x}}$ into $\mathbf{c}$, **stop** if $\mathbf{c}H^T = \mathbf{0}$.
10: $\quad\quad L_i \leftarrow (\hat{x}_i - L_i)\alpha + L_i, \ \ 0 \le i < n$.
11: $\quad\quad$ Draw random permutation $\pi \in \mathrm{Aut}(\mathcal{C})$ [17].
12: $\quad\quad \mathbf{L} \leftarrow \pi(\mathbf{L})$ and $\Theta \leftarrow \pi(\Theta)$.
13: $\quad$ **end for**
14: $\quad \alpha \leftarrow \alpha_0 + (1 - \alpha_0)\frac{I_3}{I_3 - 1}$.
15: **end for**

Algorithm 2 SPA-ELC$(p, I_1, I_2, I_3, \alpha_0)$

1: // Input: $(\mathbf{y}, H, \alpha_0, I_1, I_2, I_3, p)$.
2: // Output: $\hat{\mathbf{x}}$.

3: $\alpha \leftarrow \alpha_0$.
4: **for** I_3 times **do**
5: $\quad \mathbf{L} \leftarrow (2/\sigma^2)\mathbf{y}$.
6: $\quad \mu_{v \to u} \leftarrow L_v, \ \forall (u, v) \in \mathbf{TG}(H)$.
7: $\quad$ **for** I_2 times **do**
8: $\quad\quad$ Do I_1 flooding iterations, $\hat{\mathbf{x}} \leftarrow \mathrm{SPA}(\mathbf{TG}(H))$.
9: $\quad\quad$ Take the hard decision of $\hat{\mathbf{x}}$ into $\mathbf{c}$, **stop** if $\mathbf{c}H^T = \mathbf{0}$.
10: $\quad\quad$ **for** p times **do**
11: $\quad\quad\quad$ Select random edge $e = (u, v) \in \mathbf{TG}(H)$.
12: $\quad\quad\quad \mathbf{TG}(H) \leftarrow \mathrm{ELC}(\mathbf{TG}(H), e)$.
13: $\quad\quad\quad \mu_{v' \to u'} \leftarrow (\hat{x}_{v'} - L_{v'})\alpha + L_{v'}$,
$\quad\quad\quad \forall (u', v') \in \mathbf{TG}(H), \ u' \in \mathcal{N}_v^u, \ v' \in \mathcal{N}_u^v$.
14: $\quad\quad$ **end for**
15: $\quad$ **end for**
16: $\quad \alpha \leftarrow \alpha_0 + (1 - \alpha_0)\frac{I_3}{I_3 - 1}$.
17: **end for**

gorithm 1, where $\mathrm{Aut}(\mathcal{C})$ is represented by a small set of generators, and uniformly sampled using an algorithm due to Celler *et al.* [17]. These permutations tend to involve all, or most, of the code coordinates, making it a global operation. Note that line 7 in Algorithm 1 is to compensate for the fact that permutations are applied to $\mathbf{L}$ in line 12, rather than to the columns of H, after which the messages on the edges no longer 'point to' their intended recipients. This is yet another global stage. The extrinsic information is damped by a coefficient α, $0 < \alpha \le 1$, in line 10 before being used to re-initialize the decoder. Each time α is incremented, the decoder re-starts from the channel vector, $\mathbf{y}$.

C. SPA-ELC

Our proposed local algorithm is a two-stage iterative decoder, interleaving the SPA with random ELC operations. We call this SPA-ELC, and say that it realizes a local diversity decoding of the received codeword. Our algorithm is listed in Algorithm 2. Both SPA-PD and SPA-ELC perform a maximum of $T \triangleq I_1 I_2 I_3$ iterations. SPA update rules ensure that extrinsic information remains summed in bit nodes, such that an edge may be removed from $\mathbf{TG}(H)$ without loss of information. New edges, (u', v'), should be initialized according to line 13 in Algorithm 2. Although neutral (i.e., LLR 0) messages will always be consistent with the convergence process, our experiments clearly indicate that this has the effect of 'diluting' the information, resulting in an increased decoding time and worse error rate performance.

The simple SPA-ELC decoder requires no preprocessing or any complex heuristic or rule to decide when or where to apply ELC. As ELC generates the s-orbit of $\mathcal{C}$, as well as the iso-orbit of each structure, diversity of structure can be achieved even for random codes, for which $|\mathrm{Aut}(\mathcal{C})|$ is likely to be 1 while the size of the s-orbit is generally very large. However, going outside the iso-orbit means that we change the properties of H, most importantly in terms of density and number of short cycles. Ideally, the SPA-ELC decoder operates on a set of structurally distinct parity-check matrices, which are all of minimum weight. With the exception of codes with very strong structure, such as the extended Hamming code, the ELC-orbit of a code will contain structures of weight greater than the minimum. SPA-ELC should take measures against the negative impact of increased weight. In this paper, we adapt the technique of damping to our graph-local perspective. Damping with the standard SPA, where $\mathbf{TG}(H)$ is fixed, does not work, so we only want to damp the parts of the graph which change. As opposed to SPA-PD, only a subgraph of $\mathbf{TG}(H)$ is affected by ELC, so we restrict damping to new edges in line 13. Note that SPA-ELC simplifies to a version without damping, denoted by SPA-ELC(p, I_1, T), when $\alpha_0 = 1$, $I_2 = T/I_1$, and $I_3 = 1$. This is, simply, flooding iterations interspersed with random ELC operations, where new edges are initialized with the adjacent APP (line 13).

Currently, the SPA stopping criterion (i.e., the parameters used to flag when decoding should stop) is still implemented globally. However, a reasonable local solution would be to remove the syndrome check ($\mathbf{c}H^T = \mathbf{0}$) from the stopping criterion, and simply stop after $\hat{T}$ SPA-ELC iterations, where $\hat{T}$ can be empirically determined. However, this has obvious implications for complexity and latency. In some scenarios a stopping criterion can be dispensed with anyway–for instance when using the decoder as some form of distributed process controller, or for a pipelined implementation in which the iterations are rolled out.

IV. Results

We have compared SPA-ELC against standard SPA, and SPA-PD. Extended quadratic residue (EQR) codes were chosen for the comparison, mainly due to the fact that for some of these codes, $\mathrm{Aut}(\mathcal{C})$ can be generated by 3 generators [18]. In fact, our experiments have shown that EQR codes

TABLE I
OPTIMIZATION OF CODES USED IN SIMULATIONS

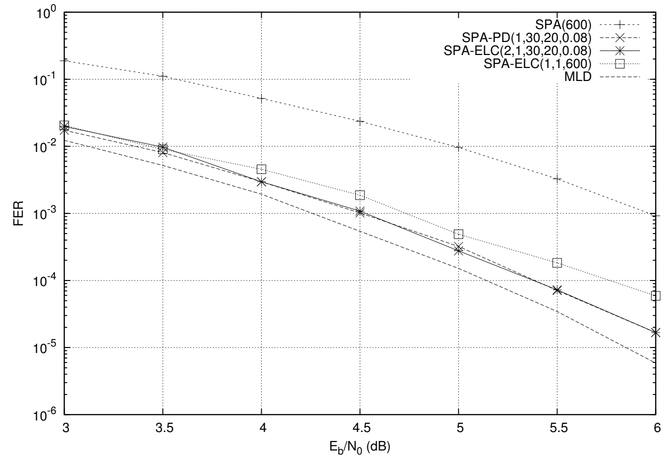
	Initial		Reduced		Reduced IP	
	W	C	W	C	W	C
$[24, 12, 8]$	* 96	366	* 96	147	* 96	366
$[48, 24, 12]$	320	4936	* 288	897	* 288	2672
$[104, 52, 20]$	1344	89138	1112	16946	1172	49839

have Tanner graphs well-suited to SPA-ELC, at least for short blocklengths. The codes considered have parameters $[24, 12, 8]$ (the extended Golay code), $[48, 24, 12]$ (EQR48), and $[104, 52, 20]$ (EQR104). Parity-check matrices for the codes were preprocessed by heuristics to minimize the weight and the number of 4-cycles. The results are listed in Table I, where columns marked 'W' and 'C' show the weight and the number of 4-cycles, respectively. Columns marked 'Initial' show the weight and the number of 4-cycles of the initial Tanner graph constructions. 'Reduced' and 'Reduced IP' refer to optimized Tanner graphs, where the latter is restricted to Tanner graphs in standard form. Entries marked by an asterisk correspond to minimum weight parity-check matrices.

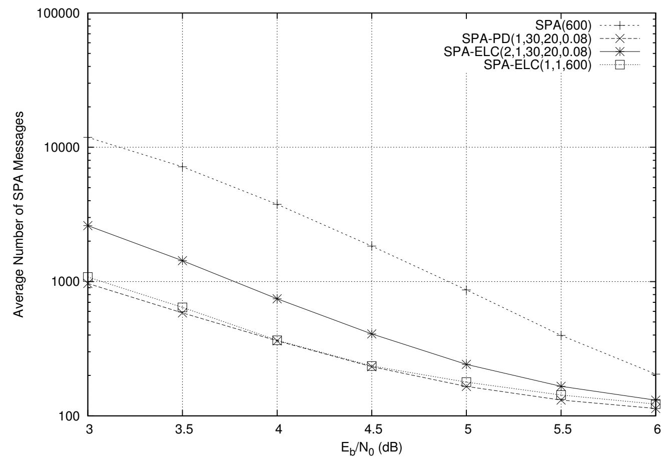
In Figs. 3-5, we show the frame error rate (FER) performance and the average number of SPA messages of SPA, SPA-PD, and SPA-ELC for the extended Golay code, the EQR48 code, and the EQR104 code, respectively, on the AWGN channel versus the signal-to-noise ratio, E_b/N_0.

The specific parameters used are indicated in the figure legends. For the extended Golay code and the EQR48 code, we set a maximum at $T = 600$ iterations, which we increased to $T = 2000$ to accommodate the larger EQR104 code. For SPA-ELC we have also included results without damping. Since SPA-ELC changes the weight of $\mathbf{TG}(H)$, we can not compare complexity by simply counting iterations. Since the complexity of one ELC operation is much smaller than the complexity of a SPA iteration, the total number of SPA messages may serve as a common measure for the complexity of the decoders. We have no initial syndrome check, so the number of iterations approaches 1 at high E_b/N_0. In the same way, the complexity approaches the average weight of the matrices encountered during decoding. Each FER point was simulated until at least 100 frame errors were observed.

From the figures, we observe that the SPA-ELC decoder outperforms standard SPA decoding, both in terms of FER and decoding complexity. The extended Golay code is a perfect example for demonstrating the benefits of SPA-ELC. The s-orbit of this code contains only two structures, where one is of minimum weight (weight 96) and the other only slightly more dense (weight 102), while the iso-orbit of the code is very large. Thus, we can extend SPA-PD with multiple Tanner graphs (two structures) while keeping the density low. Not surprisingly, SPA-ELC achieves the FER performance of SPA-PD, albeit with some complexity penalty. Note that the simple SPA-ELC decoder, without damping, approaches closely the complexity of SPA-PD at the cost of a slight loss in FER. For the larger codes, the sizes of the s-orbits are very large, and many structures are less suited for SPA-PD. Still, the same tradeoff between FER performance and complexity holds, based on whether or not we use damping. For the EQR48 code, we have observed a rich subset of the s-orbit containing minimum weight structures (weight 288). The optimum value of p (see line 10 in Algorithm 2) was determined empirically.



(a) FER performance



(b) Average number of SPA messages

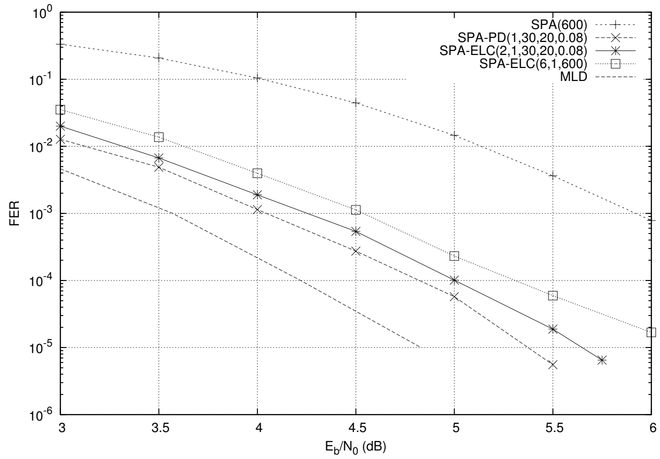
Fig. 3. $[24, 12, 8]$ extended Golay code

V. CONCLUSION AND FUTURE WORK

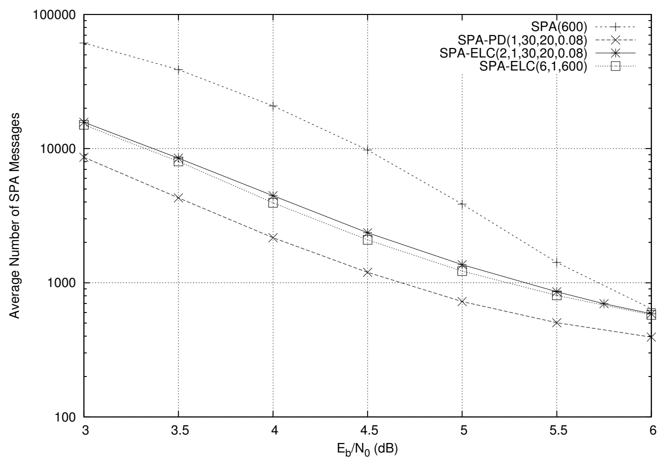
We have described a local diversity decoder, based on the SPA and the ELC operation. The SPA-ELC algorithm outperforms the standard SPA both in terms of error rate performance and complexity, and compares well against SPA-PD, despite the fact that SPA-PD uses global operations. Ongoing efforts are devoted to further improvements, and include; selective application of ELC, rather than random; devise techniques such that diversity may be restricted to sparse structures in the s-orbit; identify a code construction suited to SPA-ELC, for which the s-orbit contains several desirable structures even for large blocklengths.

ACKNOWLEDGMENT

The authors wish to thank Alban Goupil for providing the MLD curve for the EQR48 code.

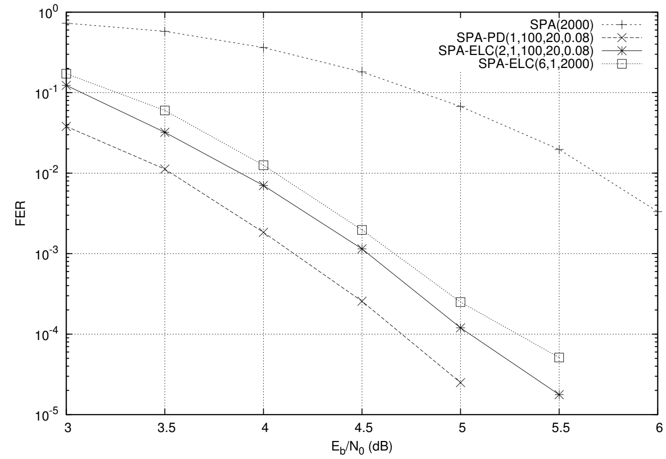


(a) FER performance

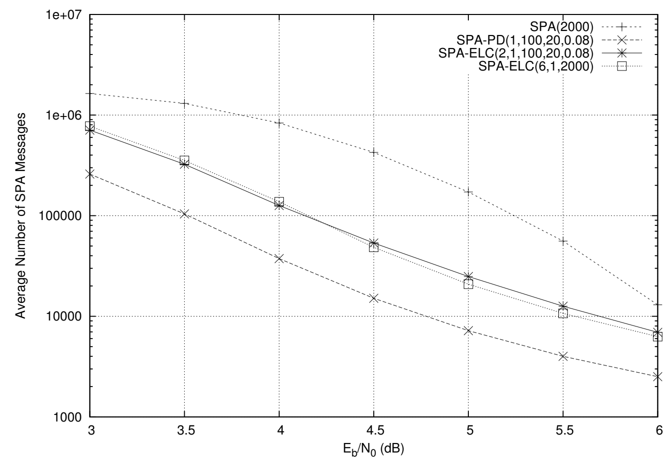


(b) Average number of SPA messages

Fig. 4. $[48, 24, 12]$ EQR48 code



(a) FER performance



(b) Average number of SPA messages

Fig. 5. $[104, 52, 20]$ EQR104 code

REFERENCES

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[2] D. J. C. MacKay and R. M. Neal, "Good codes based on very sparse matrices," in *Proc. 5th IMA Conf. Cryptography and Coding (Lecture Notes in Computer Science)*, vol. 1025, Royal Agricultural College, Cirencester, UK, Dec. 1995, pp. 100–111.

[3] J. Jiang and K. R. Narayanan, "Iterative soft-input soft-output decoding of Reed-Solomon codes by adapting the parity-check matrix," *IEEE Trans. Inform. Theory*, vol. 52, no. 8, pp. 3746–3756, Aug. 2006.

[4] ——, "Iterative soft decoding of Reed-Solomon codes," *IEEE Commun. Lett.*, vol. 8, no. 4, pp. 244–246, Apr. 2004.

[5] T. R. Halford and K. M. Chugg, "Random redundant iterative soft-in soft-out decoding," *IEEE Trans. Commun.*, vol. 56, no. 4, pp. 513–517, Apr. 2008.

[6] T. Hehn, J. B. Huber, S. Laendner, and O. Milenkovic, "Multiple-bases belief-propagation for decoding of short block codes," in *Proc. IEEE Int. Symp. Inform. Theory*, Nice, France, Jun. 2007, pp. 311–315.

[7] I. Dimnik and Y. Be'ery, "Improved random redundant iterative HDPC decoding," *IEEE Trans. Commun.*, to appear.

[8] J. G. Knudsen, C. Riera, M. G. Parker, and E. Rosnes, "Adaptive soft-decision decoding using edge local complementation," in *Proc. 2nd Int. Castle Meeting on Coding Theory and Appl. (2ICMCTA) (Lecture Notes in Computer Science)*, vol. 5228, Castillo de la Mota, Medina del Campo, Spain, Sep. 2008, pp. 82–94.

[9] F. J. MacWilliams, "Permutation decoding of systematic codes," *Bell System Tech. J.*, vol. 43, pp. 485–505, 1964.

[10] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.

[11] T. Richardson, "Error floors of LDPC codes," in *Proc. 41st Annual Allerton Conf. Commun., Control, and Computing*, Monticello, IL, Oct. 2003, pp. 1426–1435.

[12] A. Bouchet, "Isotropic systems," *European J. Comb.*, vol. 8, no. 3, pp. 231–244, Jul. 1987.

[13] L. E. Danielsen and M. G. Parker, "Edge local complementation and equivalence of binary linear codes," *Designs, Codes and Cryptography*, vol. 49, no. 1-3, pp. 161–170, Dec. 2008.

[14] C. Riera and M. G. Parker, "On pivot orbits of Boolean functions," in *Proc. 4th Int. Workshop on Optimal Codes and Related Topics*, Pamporovo, Bulgaria, Jun. 2005, pp. 248–253.

[15] M. Hein, W. Dür, J. Eisert, R. Raussendorf, M. Van den Nest, and H.-J. Briegel, "Entanglement in graph states and its applications," in *Proc. Int. School of Physics "Enrico Fermi" on "Quantum Computers, Algorithms and Chaos"*, Varenna, Italy, Jul. 2005. [Online]. Available: http://arxiv.org/abs/quant-ph/0602096

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

[17] F. Celler, C. R. Leedham-Green, S. H. Murray, A. C. Niemeyer, and E. A. O'Brien, "Generating random elements of a finite group," *Commun. in Algebra*, vol. 23, pp. 4931–4948, 1995.

[18] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. North Holland, 1977, ch. 16.