

RESEARCH STATEMENT

DEANNA DREHER

Digital data transmission is vital in today's world, but inherently prone to corruption due to flaws in the transmission medium. Shannon [16] proved that it is possible to transmit data across virtually any channel in such a way that errors can almost always be detected and corrected. The cost of this detection and correction is in the amount of redundant information transmitted: greater detection and correction requires more redundancy, which is costly in terms of time and physical resources. Thus a major goal of classical coding theory has been to find *good* codes, i.e., codes that maximize the number of errors that can occur and still be corrected, while also minimizing the amount of redundant information transmitted.

Unfortunately, knowing that an error *can* be corrected does not provide a practical means of actually correcting it. Indeed, many straight-forward approaches to correcting errors cannot be implemented in a reasonable time frame. Modern coding theory is concerned with finding good codes that have good decoders; many of these decoders are sub-optimal in the sense that they may not correct every possible error, but their simplicity and speed make them more attractive from a practical implementation standpoint.

In the last twenty years, *iterative message-passing decoding algorithms* [15], [18] have become a prominent class of sub-optimal decoders, due to the fact that they can be applied to certain classes of codes, such as *low-density parity-check codes* [8], [13] and *turbo codes* [4], to provide near-optimal error correction quickly and easily. As these algorithms are sub-optimal, there has been much work [1], [2], [3], [5], [7], [9], [12], [14] done towards identifying and characterizing when these decoders fail, or at least fail to correct errors that are known to be correctable, in hopes of improving the decoders and the codes for which these algorithms are used. My research focuses on understanding and characterizing these so-called *pseudocodewords*.

I. BACKGROUND

A (*binary linear block*) *code* C of length n is a linear subspace of the vector space \mathbb{F}_2^n . There are two main ways of specifying a subspace of a vector space: as the span of a set of basis vectors or as the kernel of a linear map. We will deal primarily with the latter characterization of a code: A *parity-check matrix* H for a code C is an $r \times n$ binary matrix such that $C = \{\mathbf{c} \in \mathbb{F}_2^n : H\mathbf{c}^T = \mathbf{0} \in \mathbb{F}_2^r\}$. The rows of H are called *checks* and the columns of H , which correspond to coordinates of codewords, are called *code bits*. A *low-density parity-check code* is a code defined by a *sparse* parity-check matrix, i.e., a parity-check matrix that has relatively few 1's in each row and column, though we assume that each row and column contains more than one 1.

Sparsity is important for implementing iterative message-passing decoders because the complexity of these algorithms is directly correlated to the number of 1's in a parity-check matrix. However, the notion of sparsity is well-defined only for ensembles of codes [8], and is necessarily vague for a single code. Notice that a low-density parity-check code is defined by a particular representation of that code. The *Tanner graph* of a code is a bipartite graphical representation of the parity-check matrix of a code, with bipartition corresponding to code bits on the one hand and checks on the other.

Definition 1.1. Suppose H is an $r \times n$ parity-check matrix for some code C . The *Tanner graph* $G = G(H)$ associated to H is the bipartite graph $G = (X \cup U, E)$, where $X = \{x_1, \dots, x_n\}$ is the set of *variable nodes*, or *bit nodes*, and $U = \{u_1, \dots, u_r\}$ is the set of *check nodes*, and $x_i u_j \in E$ is an edge in G if and only if $h_{ji} = 1$.

We can think of a codeword $\mathbf{c} \in C$ as an assignment of 0's and 1's to the variable nodes in X so that each check node $u \in U$ has an even number of neighbors with an assignment of 1, since C is the kernel of H over \mathbb{F}_2 . A *configuration* \mathbf{c} on G is an assignment of 0's and 1's to the nodes in X so that each $u \in U$ is adjacent to an even number of nodes that are assigned a 1. This notion of configuration generalizes to any bipartite graph with

variable nodes and check nodes; if X is the set of variable nodes in a bipartite graph G , we write a configuration as $\mathbf{c} = (c_x)_{x \in X}$, where c_x is the value assigned to node x by \mathbf{c} . When graphically indicating an assignment of 0's and 1's to the variable nodes in such a bipartite graph, we will indicate a node with an assignment of 1 by circling it; all uncircled nodes have an assignment of 0.

Example I.2. Let H be the parity-check matrix

$$H = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Figure 1 shows a codeword on the corresponding Tanner graph $G = G(H)$.

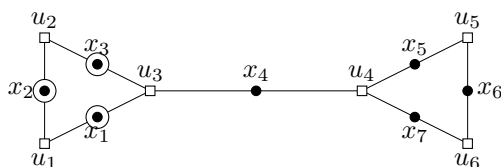


Fig. 1. A codeword configuration for the Tanner graph of Example I.2.

The local nature of iterative message-passing decoding algorithms leads to the consideration of *graph covers*, which are graphs that look locally like a base graph.

Definition I.3. An *unramified graph cover*, or simply a *cover*, of a finite graph $G = (V, E)$ is a graph \tilde{G} along with a surjective graph homomorphism $\pi : \tilde{G} \rightarrow G$, called a *covering map* or *projection map*, such that for each $v \in V$ and each $\tilde{v} \in \pi^{-1}(v)$, the neighborhood of \tilde{v} is mapped bijectively to the neighborhood of v . For a positive integer M , an *M-cover* of G is cover $\pi : \tilde{G} \rightarrow G$ such that for each vertex v of G , $\pi^{-1}(v)$ contains exactly M vertices of \tilde{G} .

Given a cover \tilde{G} of a Tanner graph G , every configuration on G induces a configuration, called a *lifting*, on \tilde{G} , though there may exist configurations on \tilde{G} that are not a lifting of any codeword.

Example I.4. Recall the Tanner graph G from Example I.2. The top half of Figure 2 shows a codeword on G , and the bottom shows a lifting of this codeword configuration to a configuration on a 2-cover of G .

However, the graph cover configuration in Figure 3 is not a lifting of any codeword on G , because it assigns different bit values to copies of the same node, e.g., one copy of x_3 is assigned a 1 while the other is assigned a 0. Notice that the graph cover configuration in Figure 3 assigns a 1 to both copies of x_4 , even though $c_4 = 0$ for any codeword $\mathbf{c} = (c_1, \dots, c_7) \in C$.

Definition I.5. Let $G = (X \cup U, E)$ be a Tanner graph, and let $(\tilde{\mathbf{c}}, \tilde{G})$ be a graph cover configuration on the M -cover $\pi : \tilde{G} \rightarrow G$. The *unscaled graph cover pseudocodeword* corresponding to $(\tilde{\mathbf{c}}, \tilde{G})$ is $\mathbf{p} = (p_x)_{x \in X}$, where, for each $x \in X$, p_x is the number of variable nodes in $\pi^{-1}(x)$ that are assigned a 1. The *normalized graph cover pseudocodeword* corresponding to $(\tilde{\mathbf{c}}, \tilde{G})$ is $\mathbf{f} = \frac{1}{M}\mathbf{p}$.

We say $(\tilde{\mathbf{c}}, \tilde{G})$ is a *realization* of \mathbf{p} and \mathbf{f} .

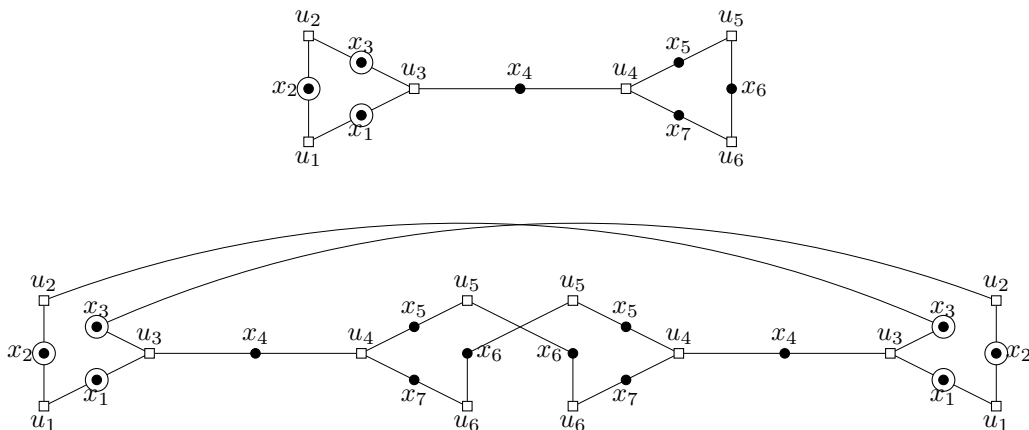


Fig. 2. A codeword and its lifting to a graph cover configuration for the Tanner graph of Example I.4.

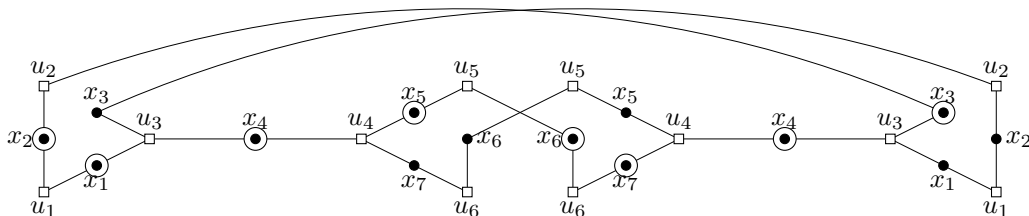


Fig. 3. A graph cover configuration that is not a lifting of any codeword on the Tanner graph G of Example I.4.

Graph cover pseudocodewords have been elegantly characterized by the *fundamental cone* of a Tanner graph [11] in the unscaled case and the *fundamental polytope* of a Tanner graph [17] in the normalized case. While much progress has been made in the realm of graph cover pseudocodewords, Wiberg [18] showed that iterative message-passing decoding algorithms are precisely modeled by *computation trees*, and not graph covers.

Definition I.6. Let $G = (X \cup U, E)$ be a Tanner graph. A *computation tree* R for G is a tree R along with a surjective graph homomorphism $\pi : R \rightarrow G$, called a projection map, such that for each non-leaf node v in R , the neighborhood of v is mapped bijectively to the neighborhood of $\pi(v)$, and every leaf node in R is in $\pi^{-1}(X)$. The nodes in $\pi^{-1}(X)$ are called the *variable nodes* of R and the nodes in $\pi^{-1}(U)$ are called the *check nodes* of R . Each computation tree R will have a variable node designated to be the *root node* of R .

A *computation tree* (π, R) for G of depth m is a computation tree (π, R) such that every path p in R that begins at the root node and ends at a leaf node contains exactly m check nodes.

Example I.7. Figure 4 shows a computation tree configuration on a computation tree of depth 2 for the Tanner graph in Example I.4. The computation tree configuration in Figure 4 is *induced* by the graph cover configuration in Figure 3, by rooting the computation tree at the copy of x_4 on the left hand side of the figure. Notice also that this computation tree configuration does not correspond to a codeword, since it assigns different values to copies of x_2 and x_6 .

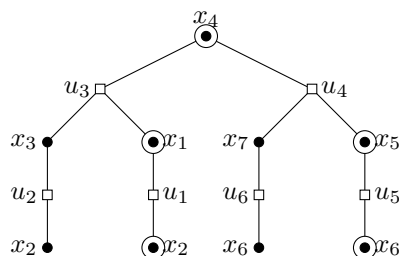


Fig. 4. The computation tree configuration in Example I.7 on a computation tree of depth 2.

II. REALIZATION OF GRAPH COVER PSEUDOCODEWORDS

In using graph covers to analyze iterative message-passing decoding, we are only concerned with graph cover configurations that induce computation tree configurations. Since computation trees are necessarily connected, it follows that only graph cover configurations on connected covers are of interest. We show that for the majority of interesting or practical codes, every normalized graph cover pseudocodeword does in fact have a connected realization.

Theorem II.1. *Let $G = (X \cup U, E)$ be a Tanner graph with average variable node degree a_X and average check node degree a_U . If $a_U \left(1 - \frac{1}{a_X}\right) \geq 2$, then any normalized graph cover pseudocodeword that can be realized on an M -cover can be realized on a connected M -cover.*

In Theorem II.1, we required $a_U \left(1 - \frac{1}{a_X}\right) \geq 2$, which is equivalent to $a_U \geq 2 + \frac{|X|}{|U|} = 2 + \frac{n}{r}$, since $a_X = \frac{|E|}{|X|} = \frac{|U||E|}{|X||U|} = \frac{|U|}{|X|}a_U = \frac{r}{n}a_U$. We will see in Theorem II.2 that if we restrict to the class of cycle codes, then we can loosen this restriction on a_U to $a_U \geq 2 + \frac{2}{|U|}$, which is equivalent to requiring $r \leq n - 1$ for cycle codes. One can check that, with this restriction on a_U , the fundamental group of a Tanner graph G has rank at least two, and the following theorem shows that we still have connected realizations of all points in the fundamental polytope. However, not every normalized graph cover pseudocodeword may be *minimally* realizable on a connected cover, i.e. there may be a smaller degree cover that does not admit a connected realization.

Theorem II.2. *Let $G = (X \cup U, E)$ be the Tanner graph of a cycle code. If the fundamental group of G has rank at least two, then every normalized graph cover pseudocodeword has a realization on a connected cover.*

More precisely, every normalized graph cover pseudocodeword that can be realized on an M -cover can be realized on a connected cover of degree M or $2M$.

Notice that the cover degrees given in Theorems II.1 and II.2 assume that the normalized graph cover pseudocodeword \mathbf{f} under inspection is realizable on an M -cover. This leads to examine for which M there exists a realization of \mathbf{f} on an M -cover.

Theorem II.3. *Let $G = (X \cup U, E)$ be a Tanner graph, let \mathcal{P} be the fundamental polytope of G , and let $\mathbf{f} \in \mathcal{P}$. Let $M \in \mathbb{N}$ be such that $M\mathbf{f}$ is an integer vector that reduces to a codeword modulo 2. Then \mathbf{f} is realizable on an M -cover of G .*

An immediate consequence of this theorem and a result in [11] is a characterization of the minimum degree cover needed to realize an arbitrary normalized graph cover pseudocodeword.

Corollary II.4. *Let G be a Tanner graph, let \mathcal{P} be the fundamental polytope of G , and let $\mathbf{f} \in \mathcal{P}$. Let $M \in \mathbb{N}$ be any positive integer such that each coordinate of $M\mathbf{f}$ is a non-negative integer.*

Then \mathbf{f} is realizable on a $2M$ -cover, and \mathbf{f} is realizable on an M -cover if and only if $M\mathbf{f}$ reduces to a codeword modulo 2.

In particular, if M is the smallest positive integer such that $M\mathbf{f}$ is an integer vector, then the minimum degree cover needed to realize \mathbf{f} is M if $M\mathbf{f}$ reduces to a codeword modulo 2, and is $2M$ otherwise.

III. CYCLE CODES AND PSEUDOWEIGHT

Koetter and Vontobel introduce *graph cover decoding* and show that it is equivalent to *linear programming decoding* as introduced by Feldman [6]. They also show that the performance of linear programming decoding is largely determined by the minimum *pseudoweight*, over all graph cover pseudocodewords, of the Tanner graph of the code. Although the precise definition of pseudoweight is channel dependent, the minimum pseudoweight is always achieved at a *minimal vertex* of the fundamental polytope of the Tanner graph of the code, i.e., a vertex of the fundamental polytope that lies on an edge of the fundamental cone. Moreover, the minimum pseudoweight of a code is generally accepted as a good predictor of performance in iterative decoding [10].

For the class of cycle codes, i.e. codes whose parity-check matrices have constant column weight two, we show that there is an elegant graphical characterization of these minimal vertices of the fundamental polytope that is reminiscent of the characterization of *bad pseudo-cycles* in Horn's analysis of the min-sm algorithm on cycle codes [9].

A *closed promenade* in a graph G is a closed walk $\gamma = e_{i_0} \dots e_{i_{k-1}}$ such that $e_{i_j} \neq e_{i_{j-1} \pmod{k}}^{-1}$ for $j = 0, \dots, k-1$, i.e. γ makes no U-turns. We say γ is *irreducible* if there do not exist $k > 1$ closed promenades $\gamma_1, \dots, \gamma_k$ such that $\gamma = \gamma_1 \dots \gamma_k$, up to rotational equivalence. Given a closed promenade $\gamma = e_{i_0} \dots e_{i_{k-1}}$, (\tilde{c}, \tilde{G}) is a *realization* of γ if it is a graph cover configuration such that the subgraph of \tilde{G} induced by the variable nodes that are assigned a 1 and their neighbors is a cycle $\tilde{e}_{i_0} \dots \tilde{e}_{i_{k-1}}$ such that \tilde{e}_{i_j} projects to e_{i_j} for all $j = 0, \dots, k-1$. A *minimal realization* of γ is a realization of γ on an M -cover such that γ is not realizable on an N -cover for any $N < M$.

Theorem III.1. *Let $G = (X \cup U, E)$ be the Tanner graph of a cycle code, let \mathcal{P} be its fundamental polytope and let $\mathbf{f} \in \mathcal{P}$. Then \mathbf{f} is a minimal vertex of \mathcal{P} if and only if \mathbf{f} is the normalized graph cover pseudocodeword corresponding to a minimal realization of some irreducible closed promenade γ in G such that γ uses each node of G at most twice and either γ is a cycle or $\gamma = c_1 p c_2 p^{-1}$ for some cycles c_1, c_2 and some path p that touches c_1 and c_2 only at its endpoints.*

This characterization can be used to prove that the minimum pseudoweight of a cycle code is the same as the minimum distance of the code, and is achieved only by codewords.

Theorem III.2. *If C is a cycle code with parity check matrix H , then the minimum pseudoweight of a H is the minimum distance of C . Moreover, if \mathbf{f} is a minimal vertex of the fundamental polytope \mathcal{P} , and \mathbf{f} is not a codeword, then $w(\mathbf{f}) \geq 2d_{\min}$.*

The asymptotic performance of linear programming decoding is determined not only by the minimum pseudoweight of a code, but also by the number of vertices of \mathcal{P} with that weight [17], [19]. Since there are no non-codeword vertices of \mathcal{P} with weight at most d_{\min} , it follows that LP decoding and ML decoding will agree asymptotically (i.e., at high signal-to-noise ratios), as pointed out in [19].

IV. REALIZATIONS OF COMPUTATION TREE PSEUDOCODEWORDS ON GRAPH COVERS

Returning to the foundational work of Wiberg [18], we know that for iterative message-passing decoding algorithms such as min-sum and sum-product, there is a simple cost function on computation tree configurations such that the minimum cost computation tree configurations rooted at each variable node determine the output

of the decoder. Thus, if we are to use graph cover configuration in our analysis of iterative message-passing decoding algorithms, we need to know somehow compare the set of graph cover configurations with the set of computation tree configurations. One means of achieving this comparison is to look at *consistent* computation tree configurations, as defined by [10]. While it can be seen that there is no meaningful vector representation of an inconsistent computation tree configuration, we prove that inconsistent computation tree configurations are still induced by graph cover configurations, in the sense discussed in Example I.7.

Given a graph cover configuration $(\tilde{\mathbf{c}}, \tilde{G})$, a *configured subgraph* of $(\tilde{\mathbf{c}}, \tilde{G})$ is a subgraph of \tilde{G} with the assignment of 0's and 1's at its variable nodes inherited from $\tilde{\mathbf{c}}$. When discussing subgraphs of graph covers and computation trees, nodes can be labeled by the node in a base graph that they project onto, and so we say such subgraphs are *isomorphic* if they are isomorphic as graphs with a graph isomorphism that respects these labels.

Theorem IV.1. *Let $G = (X \cup U, E)$ be the Tanner graph of a cycle code. Let R be a computation tree of G , and let $M = \max_v m_v$, where M_v is the number of copies of v on R for each v in $X \cup U$. Then there exists a connected $2M$ -cover \tilde{G} of G such that for any configuration \mathbf{c} on R , there exists a graph cover configuration $(\tilde{\mathbf{c}}, \tilde{G})$ containing a configured subgraph isomorphic to (\mathbf{c}, R) .*

Theorem IV.2. *Let $G = (X \cup U, E)$ be a Tanner graph and suppose (\mathbf{c}, R) is a computation tree configuration on a depth d computation tree (π_R, R) of G . For each $u \in U$, let $M_u = |\pi_R^{-1}(u)|$ and set $M = \max_u M_u$.*

Then there exists a (valid) graph cover configuration $(\tilde{\mathbf{c}}, \tilde{G})$ on a $4M$ -cover such that

- $(\tilde{\mathbf{c}}, \tilde{G})$ induces the configuration (\mathbf{c}, R) ,
- $(\tilde{\mathbf{c}}, \tilde{G})$ contains a configured subgraph that is isomorphic to $(\bar{\mathbf{c}}, \bar{R})$, where \bar{R} is the tree R with all the leaf nodes at depth d deleted and $\bar{\mathbf{c}}$ is the restriction of \mathbf{c} to \bar{R} , and
- the normalized graph cover pseudocodeword corresponding to $(\tilde{\mathbf{c}}, \tilde{G})$ is $(\frac{1}{2}, \dots, \frac{1}{2})$.

We also show that there exists a graph cover configuration on a connected $4(M+1)$ -cover that satisfies all the conditions in Theorem IV.2. An interesting corollary to Theorem IV.2 is that for every computation tree configuration (\mathbf{c}, R) of G , there is a connected realization of $(\frac{1}{2}, \dots, \frac{1}{2})$ that induces (\mathbf{c}, R) . We also show that we can achieve graph cover realizations that induce computation tree pseudocodewords on covers with degree much smaller than M , where $M = \max_v M_v$ and M_v is the number of times a vertex v appears on the computation tree, if we take into consideration a particular configuration on the computation tree.

V. FUTURE WORK

A natural question to ask when investigating the relationship between computation tree configurations and graph cover configurations is how the *cost* of the associated configurations are related, since the cost of these configurations is what determines the output of their respective decoders. Traditionally, the cost of a graph cover pseudocodeword \mathbf{f} is the inner product $\langle \mathbf{f}, \boldsymbol{\lambda} \rangle$ for some suitable vector $\boldsymbol{\lambda}$. This notion of cost provides no insight into the cost of computation tree configurations induced by a certain graph cover pseudocodeword, as we have seen that $(\frac{1}{2}, \dots, \frac{1}{2})$ induces every possible computation tree configuration, regardless of cost. One question of interest is whether there is a better notion of cost, or one that captures more information about a graph cover configuration, that might relate to the cost of induced computation tree pseudocodewords.

Another open area lies in the concept of a graph cover pseudocodeword *inducing* a computation tree pseudocodeword, which may also explain the above issue of inconsistent cost between graph cover pseudocodewords and the computation trees they induce. Perhaps inducing a computation tree configuration is not enough to draw relationships between configurations; we might rather look at minimal graph cover configurations that induce a particular computation tree configuration, or minimal graph cover configurations that contain a particular computation tree configuration.

Additionally, since we generally see relatively quick convergence of iterative message-passing algorithms when they do converge, one might even bound the degree of the cover necessary to realize a graph cover pseudocodeword by the depth of the computation tree. Thus the assumption that the algorithm is taking into consideration the full graph cover configuration may be valid and put a considerable restriction on the graph cover configurations that induce a particular computation tree configuration.

We also gave a graphical characterization of the most likely errors the minimal vertices of the fundamental polytope of code under graph cover decoding of cycle codes. While the graphical characterization and its related results are interesting in their own right, we also use this characterization to extend work by Horn [9] to the BSC in showing that any minimal vertex of the fundamental polytope of a code that is not a codeword has minimum pseudoweight at least twice the minimum distance of the code, as well as to obtain the same result for the AWGN channel. A major area of open research is whether there is a clean graphical characterization of these minimal vertices for more general low-density parity-check codes. Another open question is whether we can graphically classify other interesting subsets of the vertices of the fundamental polytope, e.g. is there a graphical characterization of *all* vertices of the fundamental polytope?

REFERENCES

- [1] N. Axvig, D. Dreher, K. Morrison, E. Psota, L. C. Pérez, and J. L. Walker. Analysis of connections between pseudocodewords. Submitted to *IEEE Transactions on Information Theory*, March 2008.
- [2] N. Axvig, K. Morrison, E. Psota, D. Turk, L. C. Pérez, and J. L. Walker. Average min-sum decoding of LDPC codes. In preparation for *2008 International Symposium on Turbo Codes and Related Topics*, September 2008.
- [3] N. Axvig, K. Morrison, E. Psota, D. Turk, L. C. Pérez, and J. L. Walker. Universal cover decoding. In preparation, 2008.
- [4] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding. In *Proceedings of the 1993 IEEE International Conference on Communications*, pages 1064–1070, Geneva, Switzerland, 1993.
- [5] D. Changyan, D. Proetti, I. E. Telatar, T. J. Richardson, and R. L. Urbanke. Finite length analysis of low-density parity-check codes on the binary erasure channel. *IEEE Transactions on Information Theory*, 48:1570–1579, June 2002.
- [6] J. Feldman. *Decoding Error-Correcting Codes via Linear Programming*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 2000.
- [7] G. D. Forney, Jr., R. Koetter, F. R. Kschischang, and A. Reznik. On the effective weights of pseudocodewords for codes defined on graphs with cycles. In *Codes, systems, and graphical models (Minneapolis, MN, 1999)*, volume 123 of *IMA Vol. Math. Appl.*, pages 101–112. Springer, New York, 2001.
- [8] R. G. Gallager. *Low-Density Parity Check Codes*. MIT Press, Cambridge, MA, 1963.
- [9] G.A. Horn. *Iterative Decoding and Pseudocodewords*. PhD thesis, California Institute of Technology, Pasadena, CA, 1999.
- [10] C. Kelley and D. Sridhara. Pseudocodewords of Tanner graphs. *IEEE Transactions on Information Theory*, 53:4013–4038, November 2007.
- [11] R. Koetter, W.-C. W. Li, P. O. Vontobel, and J. L. Walker. Characterizations of pseudo-codewords of LDPC codes. *Advances in Mathematics*, 213:205–229, 2007.
- [12] R. Koetter and P. O. Vontobel. Graph covers and iterative decoding of finite-length codes. In *Proceedings of the IEEE International Symposium on Turbo Codes and Applications*, pages 75–82, Brest, France, Sept. 1-5 2003.
- [13] D. J. C. MacKay and R. M. Neal. Near Shannon limit performance of low-density parity check codes. *IEE Electronic Letters*, 32(18):1645–1646, August 1996.
- [14] T. Richardson, A. Shokrollahi, and R. Urbanke. Design of capacity-approaching irregular low-density parity check codes. *IEEE Transactions on Information Theory*, February 2001.
- [15] T. Richardson and R. Urbanke. The capacity of low-density parity check codes under message-passing decoding. *IEEE Transactions on Information Theory*, February 2001.
- [16] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423 and 623–656, July and October 1948.
- [17] P. Vontobel and R. Koetter. Graph-cover decoding and finite-length analysis of message-passing iterative decoding of LDPC codes. To appear in *IEEE Transactions on Information Theory*.
- [18] N. Wiberg. *Codes and Decoding on General Graphs*. PhD thesis, Linköping University, Linköping, Sweden, 1996.
- [19] Shu-Tao Xia and Fang-Wei Fu. Minimum pseudo-codewords of LDPC codes. pages 109–113, Oct. 2006.