Average Min-Sum Decoding of LDPC Codes

Nathan Axvig†, Deanna Dreher†, Katherine Morrison†, Eric Psota‡,
Lance C. Pérez‡, and Judy L. Walker†
University of Nebraska
‡Department of Electrical Engineering
†Department of Mathematics
Lincoln, Nebraska
Email: {s-naxvig1, s-dturk1, s-kmorri11}@math.unl.edu, epsota24@bigred.unl.edu
lperez@unl.edu, jwalker@math.unl.edu

***Abstract*—Simulations have shown that the outputs of min-sum (MS) decoding generally behave in one of two ways: the output either eventually stabilizes at a codeword or eventually cycles through a finite set of vectors that may include both codewords and non-codewords. This inconsistency in MS across iterations has significantly contributed to the difficulty in studying the performance of this decoder. To overcome this problem, a new decoder, average min-sum (AMS), is proposed; this decoder outputs the average of the min-sum output vectors over a finite set of iterations. Simulations comparing MS, AMS, linear programming (LP) decoding, and maximum likelihood (ML) decoding are presented, illustrating the relative performances of each of these decoders. In general, MS and AMS have comparable word error rates, and in the simulation most resembling codes of practical interest AMS is shown to have significantly lower bit error rate, demonstrating the potential benefits of this decoder in its own right. Additionally, the performance of MS and AMS relative to ML and LP decoding is consistent across simulations, indicating that AMS is a valid and potentially important tool for better analyzing MS performance and its relationship to other decoders. Finally, AMS pseudocodewords are introduced and analyzed and their relationship to graph cover and LP pseudocodewords is explored, with particular focus on the AMS pseudocodewords of regular LDPC codes and cycle codes.**

I. Introduction

A major breakthrough in coding theory came with the discovery of turbo codes [4] and the subsequent rediscovery of low-density parity-check (LDPC) codes [6], [11]. Codes from these two classes have been shown to achieve realistic bit error rates, i.e., rates between 10^{-5} and 10^{-12}, with signal-to-noise ratios that are only slightly above the minimum possible for a given channel and code rate established by Shannon's original capacity theorems.

Perhaps the most important commonality between turbo and low-density parity-check codes is that they both utilize iterative message passing decoding algorithms. As successful as these codes and decoders have been in terms of application, several major questions remain before their exceptional performance can be completely understood. In this paper, we focus our attention on one of the primary iterative message passing decoding algorithms used for LDPC codes: the min-sum (MS) decoding algorithm.

The min-sum decoder is a computationally efficient suboptimal decoder for low-density parity-check codes. Its efficiency makes it ideal for implementation, but in order to use it effectively it is important to be able to characterize its non-optimality; in other words, it is necessary to understand the errors that arise in MS decoding. Theoretical analysis of these errors has thus far been scarce (but see, e.g., [7], [14]).

One characteristic of MS decoding that makes it particularly difficult to analyze is its perpetually changing output across iterations. The observed behavior of MS can be characterized in one of two ways. One possibility is that the output stabilizes at a codeword, so that for a sufficiently large number of iterations, the output vector of MS is consistently the same codeword. The other possibility is that the output eventually cycles through a finite set of outputs that may or may not be codewords, as seen in the following example.

Example I.1. Decoding of a single received vector was performed using 800 iterations of min-sum on the Tanner graph given in Figure 1. When the all-zero codeword was sent (modulated as the vector $(-1,-1,-1,-1,-1,-1,-1)$) over an additive white Gaussian noise (AWGN) channel with signal-to-noise ratio 0.0 dB, the channel output was

$$(0.7671, 1.5556, 0.3012, 2.7015, 0.4926, -2.7730).$$

For sufficiently large iterations, the output of the min-sum decoder cycled through six vectors in $\mathbb{F}_2^7$, of which only one was a codeword. For example, the outputs after iterations 783–800 were as follows:

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

Here, the first column gives the outputs after iterations 783–788, the second column corresponds to iterations 789–794, and the third column represents iterations 795–800. Notice the pronounced pattern that these output vectors follow.

In this paper, we consider the oscillatory behavior of the outputs of the min-sum algorithm. As such, we propose and study the *average min-sum (AMS) decoder*, which aims to capture this behavior by giving as output the average of the outputs of the min-sum algorithm. In Section II, we present background material on graph cover decoding and linear programming

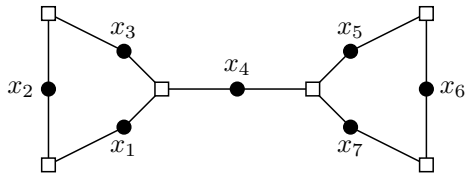

Fig. 1. The Tanner graph of Example I.1.

(LP) decoding. The definitions of the AMS decoder and AMS pseudocodewords are given in Section III. In Section IV, we present simulation results on several codes of small length. We include, when practical, a list of the AMS pseudocodewords that arose in the simulations as well as comparisons between AMS, MS, LP and maximum likelihood (ML) decoding. Also in Section IV, we provide a simulation on a larger code that has parameters closer to those of codes of practical interest. Finally, Section V contains a discussion of the simulation results and some directions for future investigation.

II. Graph Cover Decoding and Linear Programming Decoding

Previous attempts to understand the behavior of the min-sum algorithm have often built upon the intuition that, since the algorithm operates locally on the Tanner graph, it does not distinguish between the Tanner graph itself and any finite, unramified cover of the Tanner graph. This leads to the notion of *graph cover pseudocodewords*, which correspond to codewords in the codes defined by finite, unramified covers of the Tanner graph. Such pseudocodewords are the object of much study; see, e.g. [1], [3], [8], [9], [10], and [12]. In an effort to formalize this intuition, Vontobel and Koetter [13] define *graph cover decoding*; this decoder simultaneously considers all codewords on all covers of the Tanner graph and then returns the pseudocodeword corresponding to the one which, in a certain precise sense, provides the best explanation of the channel output. They show that graph cover decoding is equivalent to *linear programming decoding*, as defined by Feldman [5]. Hence, although graph cover decoding is by its very nature not amenable to simulation, one can investigate its performance (and, in particular, compare it to MS) by running simulations with LP decoding instead. As such, we now turn to the formal definition of LP decoding.

Definition II.1 ([5])**.** Let $H = (h_{j,i})$ be the $r \times n$ parity check matrix with corresponding Tanner graph T, and, for $1 \leq j \leq r$, set

$$N(j) = \{i \,|\, h_{j,i} = 1\} \subseteq \{1, \ldots, n\}$$

so that $N(j)$ is the set of variable nodes adjacent to check node j in T. The *fundamental polytope* $P = P(H)$ is the subset of the unit hypercube $[0,1]^n \subset \mathbf{R}^n$ consisting of all vectors $\mathbf{x} = (x_1, \ldots, x_n)$ such that for $1 \leq i \leq n$, $1 \leq j \leq r$, and each subset $S \subseteq N(j)$ with $|S|$ odd, we have

$$\sum_{i \in S} x_i + \sum_{i \in N(j) \setminus S} (1 - x_i) \leq |N(j)| - 1.$$

For a given vector of log-likelihoods $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n)$ determined by the channel output and for any $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$, the *cost* $\gamma(\mathbf{x})$ of $\mathbf{x}$ is given by

$$\gamma(\mathbf{x}) = \boldsymbol{\lambda} \cdot \mathbf{x} = \sum_{i=1}^{n} \lambda_i x_i.$$

Linear programming (LP) decoding is defined to be the task of minimizing $\gamma(\mathbf{x})$ over all $\mathbf{x} \in P$.

Since the cost function is linear and the polytope is defined by linear inequalities, the output of linear programming decoding may always be taken to be a vertex of the fundamental polytope. Feldman [5] shows that a vector in $\{0,1\}^n$ is a vertex of the fundamental polytope if and only if it is a codeword. This motivates the following definition.

Definition II.2. A *linear programming pseudocodeword* of a code defined by the parity-check matrix H is any vertex of the fundamental polytope $P(H)$. A *nontrivial* linear programming pseudocodeword is a linear programming pseudocodeword that is not a codeword.

Additionally, Feldman [5] establishes that linear programming decoding has the *ML certificate property*: if linear programming decoding outputs a codeword then that codeword is necessarily the maximum likelihood codeword. Vontobel and Koetter [13] show that the collection of rational points in the fundamental polytope is precisely the collection of graph cover pseudocodewords. Thus, with the definitions here, every linear programming pseudocodeword is a graph cover pseudocodeword, but not vice versa.

III. The Average Min-Sum Decoder

As mentioned in the introduction, the observed behavior of the min-sum algorithm is that the output vector either eventually stabilizes at a codeword or eventually cycles through a finite set of outputs that may or may not be codewords. Example I.1 gives a concrete example of MS repeatedly cycling through a set of outputs that includes both codewords and non-codewords, even after more than 700 iterations have been performed. If the min-sum algorithm were stopped in this example at iterations 787, 793, or 799, the decoder would output the codeword $(0,0,0,0,1,1,1)$. Within the range shown, however, the codeword $(0,0,0,0,1,1,1)$ only represents one sixth of all possible outputs of MS. In particular, for all non-codeword outputs in the range shown, the binary value assigned to the fourth coordinate is 1. In an oscillatory case such as this, we see that the outputs of MS can vary drastically even between consecutive iterations. We propose the following decoding algorithm with the aim of capturing the overall behavior of MS, rather than simply a snapshot of a particular iteration.

Definition III.1. The *average min-sum (AMS) decoder* is given by the following rule: After m iterations, the decoder outputs

$$\hat{\mathbf{x}}^{\text{AMS}} := \frac{1}{m}\sum_{i=1}^{m}\hat{\mathbf{x}}^{(i)},$$

where $\hat{\mathbf{x}}^{(i)}$ is the output of the min-sum decoder after i iterations.

Example III.2. Again, consider the $[7,3,2]$ code of Example I.1, defined by the Tanner graph of Figure 1. A simulation of 800 iterations of MS decoding on the AWGN channel with SNR of 0.0 dB was performed to obtain the AMS output. It was observed that over these iterations, MS always reached a steady oscillatory pattern, which resulted in an output of AMS that was a vector of "nice" rational numbers. In particular, four common non-codeword outputs were observed. These four outputs were extremely close to the following rational vectors:

$$\left\{ \left(\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2},1,\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2}\right), \quad \left(\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2},\tfrac{5}{6},\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2}\right), \right.$$

$$\left. \left(\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2},\tfrac{2}{3},\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2}\right), \quad \left(\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2}\right) \right\}.$$

Notice that the vector $(\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{5}{6},\frac{1}{2},\frac{1}{2},\frac{1}{2})$ is obtained, for example, in the situation of Example I.1. Meanwhile, there is only one nontrivial LP pseudocodeword for this code, namely, $\left(\frac{1}{2},\frac{1}{2},\frac{1}{2},1,\frac{1}{2},\frac{1}{2},\frac{1}{2}\right)$.

Because the observed behaviors of the min-sum decoder imply that the outputs always either stabilize at a codeword or eventually repeatedly cycle through some finite set of vectors, it is reasonable to believe that the output of the average min-sum decoder always approaches some limit, i.e., that for any channel input, the limit

$$\lim_{m\to\infty}\hat{\mathbf{x}}^{\text{AMS}} = \lim_{m\to\infty}\frac{1}{m}\sum_{i=1}^{m}\hat{\mathbf{x}}^{(i)} \tag{III.1}$$

always exists. This motivates the next definition.

Definition III.3. An *average min-sum pseudocodeword* is a limiting value of the output vectors of the average min-sum decoding algorithm. A *nontrivial* average min-sum psuedocodeword is an average min-sum pseudocodeword that is not a codeword.

If the limit of (III.1) exists, then

$$\begin{aligned}\lim_{m\to\infty}\hat{\mathbf{x}}^{\text{AMS}} &= \lim_{m\to\infty}\frac{1}{m}\sum_{i=1}^{m}\hat{\mathbf{x}}^{(i)} \\ &= \lim_{m\to\infty}\frac{1}{(m-(\ell-1))}\sum_{i=\ell}^{m}\hat{\mathbf{x}}^{(i)}\end{aligned}$$

exists for any $\ell \in \mathbb{N}$. Since the first several outputs of MS typically jump around before the eventual behavior described above appears, using a larger value of ℓ may actually improve the rate of convergence in the above limits. Because of this, the implementation of AMS used in Section IV computes

$$\frac{1}{(m-599)}\sum_{i=600}^{m}\hat{\mathbf{x}}^{(i)},$$

where m is chosen uniformly at random from the integers $\{800, 801, \ldots, 900\}$. We note that applying this implementation to Example III.2 results in the same performance and set of observed outputs.

IV. Simulation Results

In this section, we examine simulation results to explore the performance of the average min-sum decoder. The focus is on the relationship between MS and AMS performance as well as a comparison of these two decoders with LP/graph cover decoding. In particular, when practical, the set of nontrivial AMS pseudocodewords is examined in hopes of elucidating the oscillatory nature of MS across iterations. Additionally, examination of AMS pseudocodewords may further explain a link or perhaps a disparity between MS and LP/graph cover decoding.

The following simulations are performed on a variety of codes, and different parity-check representations of the same code. Both H_1 and H_2 define cycle codes, and H_3, H_4, and H_5 are different (non-cycle) realizations of the code defined by H_2. By studying different realizations of the same code, we hope to understand the effect of parity-check regularity/irregularity on AMS decoding. Finally, simulation results are given for an length 10 irregular code defined by H_6, and a larger $(6,3)$-regular code of length 1080.

Let H_1 be the semi-regular parity-check matrix

$$H_1 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

and let

$$C_1 = \begin{array}{ll} \{(0,0,0,0,0,0,0), & (0,0,0,0,1,1,1), \\ (1,1,1,0,0,0,0), & (1,1,1,0,1,1,1)\} \end{array}$$

be the code of length $N = 7$ and dimension $K = 2$ determined by H_1; this is the same code considered in Examples I.1 and III.2 above. Each of the decoders; AMS, MS and LP demonstrates nearly identical performance during simulations of this code with respect to both word and bit error rate, and each is very close to ML. When AMS outputs a codeword, it always matched the LP output and, as a result of the ML certificate property of LP decoding, this was the ML codeword. The sets of LP and common AMS pseudocodewords are discussed in Example III.2.

Now let H_2 be the parity-check matrix

$$H_2 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

and let C_2 be the cycle code determined by H_2. Then

$$C_2 = \begin{matrix} \{(0,0,0,0,0,0), & (0,0,0,0,1,1), \\ (0,1,1,1,1,0), & (1,1,0,0,0,0), \\ (1,1,0,0,1,1), & (1,0,1,1,0,1), \\ (1,0,1,1,1,0), & (0,1,1,1,0,1)\} \end{matrix}$$

has length $N = 6$ and dimension $K = 3$. In simulations, we observed very few distinct AMS pseudocodewords, and all of them were approximations of simple rational vectors in the LP polytope. While the only nontrivial LP pseudocodewords determined by H_2 are $\left(\frac{1}{2},\frac{1}{2},1,0,\frac{1}{2},\frac{1}{2}\right)$ and $\left(\frac{1}{2},\frac{1}{2},0,1,\frac{1}{2},\frac{1}{2}\right)$, the following AMS pseudocodewords were commonly seen in simulations:

$$\{\left(\tfrac{1}{2},\tfrac{1}{2},1,0,\tfrac{1}{2},\tfrac{1}{2}\right), \quad \left(\tfrac{1}{2},\tfrac{1}{2},0,1,\tfrac{1}{2},\tfrac{1}{2}\right),$$

$$\left(\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{4},0,\tfrac{1}{2},\tfrac{1}{2}\right), \quad \left(\tfrac{1}{2},\tfrac{1}{2},0,\tfrac{1}{4},\tfrac{1}{2},\tfrac{1}{2}\right),$$

$$\left(\tfrac{1}{2},\tfrac{1}{2},\tfrac{3}{4},0,\tfrac{1}{2},\tfrac{1}{2}\right), \quad \left(\tfrac{1}{2},\tfrac{1}{2},0,\tfrac{3}{4},\tfrac{1}{2},\tfrac{1}{2}\right)\}.$$

Decoding simulations demonstrated similar relative performance to the previous cycle code; AMS, MS and LP perform identically and are close to ML, and there was a small consistent set of AMS pseudocodewords. The performance results from the previous two cycle codes agree with Feldman's comment [5] that the performance of LP and MS agree when the parity check matrix has constant column weight two.

Consider next the $(3,3)-$regular LDPC code C_3 of length $N = 6$ and dimension $K = 3$ determined by the parity-check matrix

$$H_3 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

Note that, as subspaces of $\mathbb{F}_2^6$, we have $C_3 = C_2$. Min-sum and AMS perform similarly in simulation, as do LP and ML, though LP and ML perform much better than either MS or AMS. The following set of commonly witnessed AMS pseudocodewords is still simple, in that it is small and consists of vectors in the LP polytope that appear to be approximations of fractions with small denominators:

$$\{\left(\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2},\tfrac{1}{2}\right), \quad \left(\tfrac{1}{2},\tfrac{1}{2},0,0,0,0\right),$$

$$\left(0,0,0,0,\tfrac{1}{2},\tfrac{1}{2}\right), \quad \left(\tfrac{1}{2},\tfrac{1}{2},0,0,1,1\right),$$

$$\left(1,1,0,0,\tfrac{1}{2},\tfrac{1}{2}\right)\}.$$

In contrast to the case of the cycle code determined by H_2, where two of the six nontrivial AMS pseudocodewords appearing in simulations were the nontrivial LP pseudocodewords, in this case the nontrivial LP pseudocodewords $\left(\frac{1}{2},\frac{1}{2},1,0,\frac{1}{2},\frac{1}{2}\right)$ and $\left(\frac{1}{2},\frac{1}{2},0,1,\frac{1}{2},\frac{1}{2}\right)$ are not present in the set of AMS pseudocodewords for H_3.

As with the parity-check matrices of column weight two above, the regularity condition on the low-density parity-check code may be loosened to semi-regularity so as to further examine the impact of regularity on the behavior of the average min-sum decoder. The matrix

$$H_4 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

defines a semi-regular LDPC code C_4 of length $N = 6$ and dimension $K = 3$ with constant column weight 3. Notice that, as subsets of $\mathbb{F}_2^6$, we have $C_4 = C_3 = C_2$. Min-sum and AMS again performed similarly. The performances of LP and ML are still close, but now they are only slightly better than MS and AMS. With this description of the code, however, the set of AMS pseudocodewords is now large and unwieldy; even at large SNRs the non-codeword AMS outputs no longer appear to be simple rational vectors. Four such AMS outputs obtained at 5.0 dB are

$$\begin{matrix} (0.50, 0.50, 0.27, 0.47, 0.28, 0.33), \\ (0.14, 0.14, 0.14, 0.17, 0.81, 0.80), \\ (0.45, 0.33, 0.41, 0.49, 0.50, 0.50), \\ (0.11, 0.11, 0.04, 0.07, 0.02, 0.08). \end{matrix}$$

Additionally, enforcing regularity at the check nodes, i.e., maintaining constant row weight, does not appear to simplify the set of AMS pseudocodewords any more than enforcing regularity at the variable nodes, as seen in the next example. Let H_5 be the parity check matrix

$$H_5 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

and let C_5 be the code determined by H_5. Then C_5 is a code of length $N = 6$ and dimension $K = 3$ and H_5 has a constant row weight of three. As subspaces of $\mathbb{F}_2^6$, we have $C_5 = C_4 = C_3 = C_2$. The relative performance is similar to that of C_4: MS and AMS are close, with LP and ML performing similarly and only slightly better than MS and AMS. As was the case with C_4, however, we also observe very little discernible structure in the large set of nontrivial AMS pseudocodewords.

The final example that we consider for which it is practical to examine the set of AMS pseudocodewords is an irregular LDPC code C_6 of length $N = 10$ and dimension $K = 1$

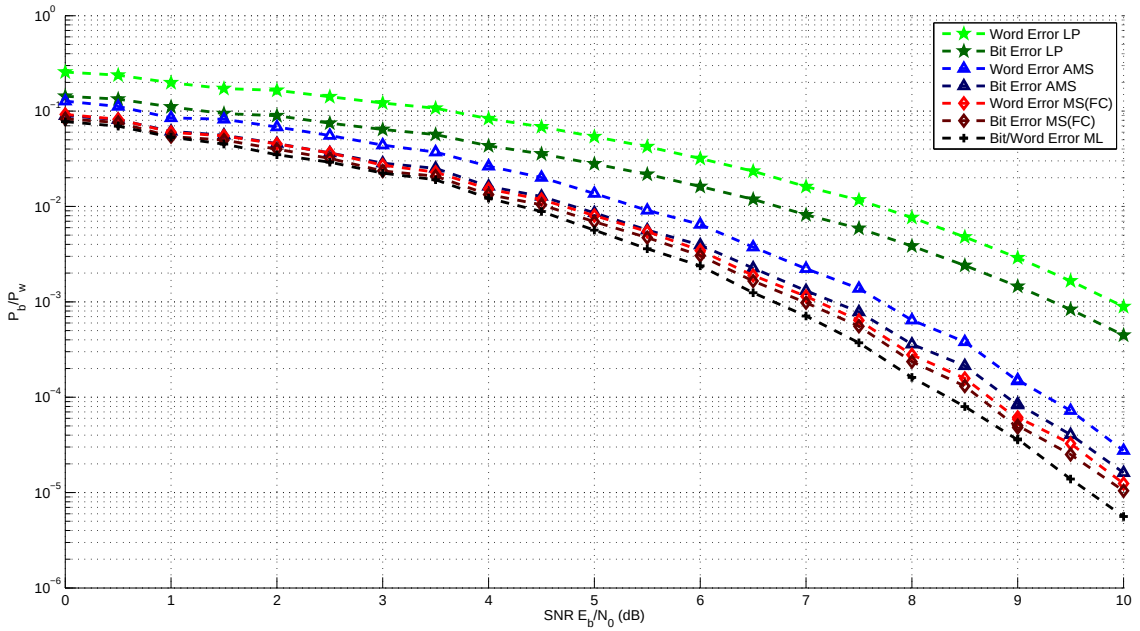

Fig. 2. Performance of the irregular LDPC code C_6 with maximum likelihood (ML), linear programming (LP), min-sum (MS) and average min-sum (AMS) decoding.

defined by the parity-check matrix

$$H_6 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

The performance of this code under various decoding algorithms is shown in Figure 2. In this simulation, MS is implemented to terminate when it reaches its first codeword. The performance of MS is slightly better than AMS, and both are reasonably close to ML and far better than that of LP. The nontrivial AMS pseudocodewords are extremely irregular and again it is hard to parse any structure.

Finally, we offer in Figure 3 results of a simulation of the average min-sum decoder on a larger regular code that is more similar to codes actually used in practice. We see that in this simulation AMS performs the same as MS with respect to word error rate but better than MS with respect to bit error rate. Unfortunately, however, it is impractical to examine the set of AMS pseudocodewords or to compare this performance to that of LP or ML with such a large block length.

V. Discussion and Conclusion

We have observed through simulation with the small parity-check matrices $H_1, H_2, H_3, H_4,$ and H_5 in Section IV that min-sum and average min-sum have similar performance with respect to both bit and word error rate. Furthermore, on the large code of length $N = 1080$, MS and AMS again have comparable word error rates, but AMS has a significantly better bit error rate than MS (see Figure 3). The question of whether AMS typically outperforms MS with respect to bit error rate for codes with reasonable parameters is an object of future investigations.

Also of interest is the set of nontrivial average min-sum pseudocodewords. In Section IV it was observed that for codes defined by parity-check matrices of column weight two or uniform row and column weight, the set of nontrivial AMS pseudocodewords is a small set of vectors resembling "nice" rational vectors that lie within the fundamental polytope. In the cases where the parity-check matrix is irregular with at least one column having weight different from two, the set of nontrivial AMS pseudocodewords was extremely large and it was difficult to find any apparent structure in the vectors. For parity-check matrices with a uniform column weight two, perhaps the phenomenon of "nice" rational vectors can be explained by Feldman's comment that MS and LP have identical performance on cycle codes [5]. As for the regular LDPC code defined by H_3 in Section IV, it may be that the pleasant set of nontrivial pseudocodewords can be explained by the fact that the distribution of copies of variable nodes on a

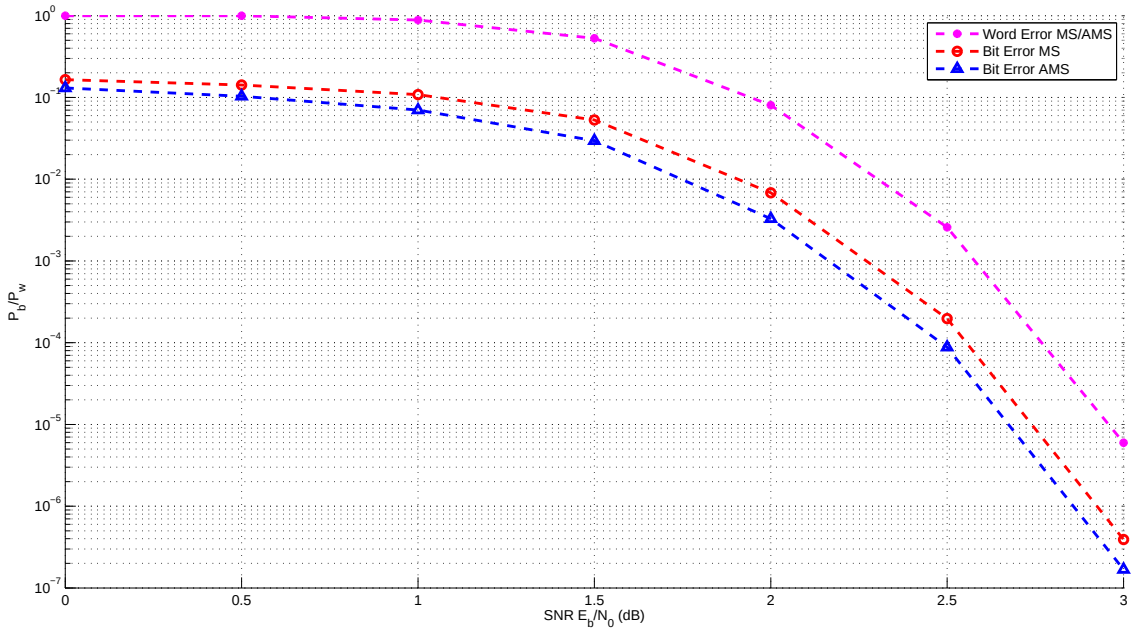

Fig. 3. Performance of a rate $1/2$, (6,3)-regular LDPC code of length $N = 1080$ with min-sum (MS) and average min-sum (AMS) decoding.

computation tree of the Tanner graph T approaches uniformity as the depth of the computation tree increases. The following proposition makes this precise.

Proposition V.1 ([2])**.** *Let T be a connected Tanner graph of a regular LDPC code of length n with row and column weights at least three. Let $R_v^{(m)}$ be the computation tree of T of depth m rooted at the variable node v of T, and let $X(R_v^{(m)})$ be the set of variable nodes in $R_v^{(m)}$. For each variable node x in T, let $\mu_v^{(m)}(x)$ be the number of copies of variable node x contained in $R_v^{(m)}$. Then*

$$\lim_{m\to\infty} \frac{\mu_v^{(m)}(x)}{|X(R_v^{(m)})|} = \frac{1}{n}.$$

Intuitively, it seems there should be a link between the nice distribution of variable nodes in computation trees of large depth, as guaranteed by Proposition V.1, and regularity in the outputs of the min-sum algorithm. Possible connections between the distribution of copies of variable nodes in computation trees and the outputs of MS remain under investigation.

In summary, the average min-sum decoder performs analogously in most cases with min-sum, and in the simulation on a code most similar to codes used in practice AMS displayed a significant improvement in bit error rate over MS. Thus, AMS is interesting in its own right as well as being an instrument with which to study the long-term behavior of MS. Additionally, in the simulations performed with regular LPDC codes and cycle codes, the set of nontrivial AMS pseudocodewords approximated rational points in the fundamental polytope. Moreover, some of these nontrivial AMS pseudocodewords were not vertices of the fundamental polytope, and hence are not possible outputs of (the standard implementations of) LP/graph cover decoding. These results suggest that the study of AMS decoding may shed light on relationships that exist between MS and LP/graph cover decoding.

VI. Acknowledgements

This work was supported in part by NSF grant DMS-0602332.

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. Dreher, L.C. Pérez, and J. L. Walker. Towards a universal theory of decoding and pseudocodewords. SGER Technical Report 0801, University of Nebraska-Lincoln, March 2008.

[3] N. Axvig, E. Price, E. Psota, D. Turk, L.C. Pérez, and J.L. Walker. A universal theory of pseudocodewords. In *Proceedings of the 45th Annual Allerton Conference on Communication, Control, and Computing*, September 2007.

[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] J. Feldman. *Decoding Error-Correcting Codes via Linear Programming*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 2003.

[6] R. G. Gallager. *Low-Density Parity Check Codes*. MIT Press, Cambridge, MA, 1963.

[7] C. Kelley and D. Sridhara. Pseudocodewords of Tanner graphs. *IEEE Transactions on Information Theory*, 53:4013–4038, November 2007.

[8] C. Kelley, D. Sridhara, J. Xu, and J. Rosenthal. Pseudocodeword weights and stopping sets. In *Proceedings of the 2004 IEEE International Symposium on Information Theory*, page 150, Chicago, IL, Jun.-Jul. 27-3 2004.

[9] R Koetter, W.-C. W. Li, P. O. Vontobel, and J. L. Walker. Pseudo-codewords of cycle codes via zeta functions. In *Proc. IEEE Inform. Theory Workshop*, pages 7–12, San Antonio, TX, USA, 2004.

[10] 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.

[11] 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.

[12] P. O. Vontobel R. Smarandache. Pseudo-codeword analysis of Tanner graphs from projective and euclidean planes. *IEEE Transactions on Information Theory*, IT-53(7):2376–2393, July 2007.

[13] 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*.

[14] N. Wiberg. *Codes and Decoding on General Graphs*. PhD thesis, Linköping University, Linköping, Sweden, 1996.