Failures of the Gallager B Decoder: Analysis and Applications

Sundararajan Sankaranarayanan, Shashi Kiran Chilappagari, Rathnakumar Radhakrishnan, and Bane Vasić, *Senior Member, IEEE*

***Abstract*— In this paper, we study state transitions, induced by low-density parity-check codes, of the Gallager B algorithm in order to characterize failures of the decoder. The failures of the decoder depend on the properties of the underlying graph of the code. Two classes of sets, namely *trapping sets* and *propagating sets*, of variables of the code are defined to categorize failures of the decoder. Such a categorization helps to establish a connection between size and population of these sets in the code and the frame error rate performance of the decoder. This connection is explored to develop a semi-analytical method to estimate the frame error rate of a class of codes from balanced incomplete block designs.**

I. Introduction

Since Gallager's thesis [1] on low-density parity-check (LDPC) codes and iterative decoding algorithms, researchers have developed a whole array of techniques to construct "good" LDPC codes and analyze the performance thresholds [2] of LDPC ensembles under the assumption that the girth goes to infinity. Unfortunately, the same level of maturity has not been attained in the characterization of iterative decoding of finite-length LDPC codes. Wiberg [3] developed techniques by defining computation trees that achieved some success in the regime of finite-length analysis. Frey *et al.* used computation trees to characterize decoding boundaries and pseudocodewords of iterative decoders. The pseudocodewords include words that are not codewords and are local minima of the decoding algorithm. Vontobel and Koetter [4] have analyzed all finite covers of Tanner graphs to explain decision boundaries of linear programming decoders and to characterize local minima of message-passing decoders. Chernyak *et al.* [5] used an instanton approach to analyze finite-length LDPC codes.

Richardson [6], with an aim to estimate the performance of LDPC codes developed a semi-analytical technique that relies on combinatorial objects called trapping sets. The idea of a trapping set is similar to that of near codewords defined in [7]. In this paper, we focus on understanding and characterizing the dynamics of an iterative decoder, namely the Gallager B decoder, for the binary symmetric channel. The state transitions of the decoder are classified into two groups and such a classification is used to characterize failures of the decoder. Finally, we use this knowledge to semi-analytically estimate the performance of LDPC codes from a class of designs.

Manuscript received January 22, 2006. This work was supported by grants from NSF-CCR (grant no. 0208597) and INSIC. A part of the material in this paper will be presented in the *IEEE International Conference on Communications*, June 2006.

S. Sankaranarayanan, S. K. Chilappagari, Rathnakumar Radhakrishnan and B. Vasić are with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721 USA (email: {ssundar, shashic, rathna, vasic}@ece.arizona.edu)

II. Preliminaries

A. Gallager B Decoder

The iterative hard-decision decoder for the BSC is a sub-optimal decoder, and it is one of the simplest decoders that belong to the category of *message-passing decoders*. This decoder, known as the Gallager B decoder, works on the Tanner graph $\mathcal{G}$ of the parity-check matrix $\mathbf{H}$ of the code by sending binary messages over the edges. Since the input and output alphabets are binary, all operations of the decoder are defined over $\mathbb{F}_2$.

The decoding algorithm [8] is described as follows. In round 0, the variable nodes send their received values to the neighboring checks over the incident edges. In round i, the message sent from a check to a neighboring variable is the sum of all incoming messages except the one arriving from the variable. After receiving messages from the check, each variable sends a new message to the neighboring checks. The message sent from a variable to a neighboring check is the majority (if it exists) among all incoming messages except the one arriving from the check. If a majority does not exist, then the received value corresponding to the variable is sent to the check. In the decoding stage, a variable takes a value that is the majority among all incoming messages. If a majority does not exist, then the variable is assigned the corresponding received value.

B. Balanced Incomplete Block Design

A *balance incomplete block design* (BIBD) is defined as a collection of k-subsets of a v-set P, $k < v$, such that pair of elements of P occur together in exactly λ of the blocks. Each k-subset is called a *block* and each element of P is called a *point*. A BIBD is referred to as a design with parameters 2-(v, k, λ). The *incidence matrix* of a 2-(v, k, λ) design with b blocks is a $b \times v$ matrix $\mathbf{A} = (a_{ij})$ such that a_{ij} is 1 if the i^{th} block contains the j^{th} point or 0 otherwise. The parity-check matrix $\mathbf{H}$ of a code from BIBD is the transpose of the incidence matrix. The parity-check matrix obtained from the design has uniform column and row weights.

The projective geometry PG(2,2^m) and affine geometry AG(2,2^m) codes considered in this paper are constructed from the incidence matrix of $2{-}(q^2{+}q{+}1, q{+}1, 1)$ and $2{-}(q^2, q, 1)$ BIBDs, respectively, where q is a power of prime. The column

weight of a PG$(2,q)$ code is $q+1$ and its minimum distance is $d_{\min} = q+2$. The column weight of an AG$(2,q)$ is q and its minimum distance is $d_{\min} = q+1$.

It is easy to show that one iteration of the Gallager B algorithm is similar to a majority-logic decoder and the number of estimates used in the decoding stage is clarified in the following facts.

Fact 1: For a code with column weight γ, the decoding decision for a variable after one iteration is based on γ estimates, if γ is odd, and $\gamma+1$ estimates (including the received value of the bit), if γ is even.

Using the derivation given in [9] by Rudolph, it is possible to compute the error correction capability of one iteration of the Gallager B algorithm (or a one-step majority logic decoder) working on a code from $2-(v,k,1)$ BIBD.

Theorem 2 (Extension of Rudolph's Result): For a code from $2-(v,k,1)$ BIBD with column weight γ, a single iteration of the Gallager B decoder can correct up to $\frac{\gamma-1}{2}$ errors, if γ is odd, and $\frac{\gamma}{2}$ errors, if γ is even. ■

Therefore, for codes from PG$(2, q = 2^m)$, the decoder (in one step) can correct all error patterns up to weight $t = \lfloor \frac{q+1}{2} \rfloor = \lfloor \frac{d_{\min}-1}{2} \rfloor$. For codes from AG$(2, q = 2^m)$, the decoder can correct all error patterns up to weight $t = \lfloor \frac{q}{2} \rfloor = \lfloor \frac{d_{\min}-1}{2} \rfloor$. In fact, for error patterns of higher weights, the behavior of the decoder can be completely characterized for codes from projective plane and consequently the bit error rate can be analytically determined. The probability of a bit being in error after decoding can be determined as follows

$$p_{\mathrm{b}} = \sum_{N_e} \Pr(\text{bit decoded incorrectly}|N_e \text{ errors})\Pr(N_e \text{ errors}). \tag{1}$$

For sake of simplicity, let b_0 and $\hat{b}_0$ be the channel and decoder output of the bit and let its transmitted value be 0. Then,

$$\begin{aligned} p_{\mathrm{b}} = \sum_{N_e} \Big[&\Pr\left(\hat{b}_0 = 1 | \{N_e - 1 \text{ other errors}, b_0 = 1\}\right) \cdot \\ &\Pr(b_0 = 1|N_e \text{ errors}) + \Pr\left(\hat{b}_0 = 1 | \{N_e \text{ errors}, b_0 = 0\}\right) \cdot \\ &\Pr(b_0 = 0|N_e \text{ errors}) \Big] \cdot \Pr(N_e \text{ errors}), \end{aligned} \tag{2}$$

The expression $\Pr(\text{bit decoded incorrectly}|N_e \text{ errors})$ can be calculated combinatorially for these codes. Since, any two elements of P occur together in exactly 1 block, every variable node is connected to every other variable node through exactly one check node. Specifically, the γ check equations used to decode variable node b_0 comprises of all the other variable nodes. Therefore, any error pattern of a given weight has to be distributed in some way among the nodes associated to these γ check equations. This specific characteristic of the code enables a purely combinatorial evaluation of the above expression. The BER for various codes derived from projective geometry is shown in Fig. 1. Unfortunately, this analytical technique cannot be easily extended to characterize decoder behavior for 2 or more iterations, since the errors patterns are highly correlated after the 1st iteration.

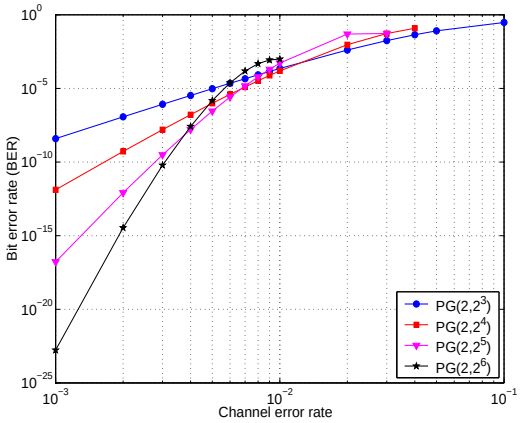


Fig. 1. Analytically calculated BER for codes from AG(2,2^4) and PG(2,2^4) when decoded using one-step majority logic decoder

III. State Transitions of the Gallager B Decoder

Let the i^{th} state of the Gallager B decoder be the decoded word obtained after the i^{th} round or iteration. The transition from an i^{th} state to an $(i+1)^{\text{th}}$ state, given an initial state, depends on the Tanner graph representation of the code. The state transitions of the decoder can be *periodic* or *aperiodic*.

A. Periodic Transitions

A periodic transition of states with period $T \neq 0$ is said to occur if the $(i+kT)^{\text{th}}$ states, for some $i \geq i_0$ and $\forall\ k \geq 0$, are equal. Let us first focus on the periodic transition with $T = 1$.

The initial state of the decoder that causes periodic transition with $T = 1$ and $i_0 = 0$ is referred to as the *fixed point*. A fixed point of the decoder is characterized by defining a set called the *trapping set*. The trapping set of a decoder with a given graph representation is the support of an initial state that is a fixed point of the decoder. For example, supports of all codewords are trapping sets of the decoder. But there are also other trapping sets as shown in the example below. A (v,c) trapping set $\mathcal{T}$ is a set of v variable nodes whose induced subgraph has c unsatisfied checks.

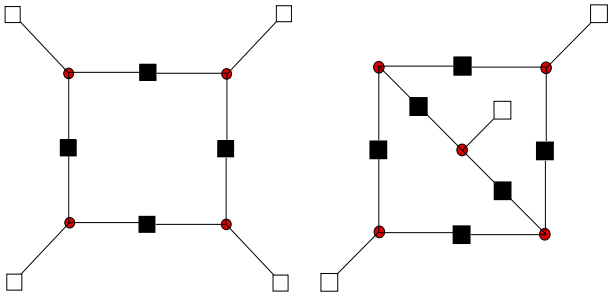
Example 1: Consider the $(2640, 1320)$ Margulis code which has girth 8 and column weight 3. From the structure of the code we can calculate the number of eight-cycles in the graph to be 1320. Fig. 2(a) illustrates the structure of an eight-cycle. Let the support of the initial state of the decoder be the variable nodes of the eight cycle. Each variable node receives a message of 1 from two check nodes and a message of 0 from one check node. If majority logic is used for decoding, these variable nodes will be decoded as 1. The same messages are passed at every iteration and the decoder fails to correct these errors. Hence, an eight cycle is a $(4,4)$ trapping set for the Margulis code. Similarly, Fig. 2(b) illustrates another trapping set of the code. The $(12,4)$ trapping set for the AWGN channel from [6] is also a trapping set for the Gallager B algorithm.

A minimal trapping set $\mathcal{T}_{\mathrm{M}}$ of a code is a trapping set with the smallest possible cardinality. A bound on the cardinality of minimal trapping set of a regular code with girth g and column weight γ is given in (3), where $x = \gamma$ if γ odd and

$$|\mathcal{T}_{\mathrm{M}}| \geq \begin{cases} 2 & g = 4 \\ \lceil \frac{x}{2} \rceil + 1 & g = 6 \\ 2\lceil \frac{x}{2} \rceil & g = 8 \\ 1 + \sum_{i=0}^{\frac{g-6}{4}} \lceil \frac{x}{2} \rceil (\lceil \frac{x}{2} \rceil - 1)^i & g \geq 10, \frac{g}{2} \text{ odd} \\ 1 + \sum_{i=0}^{\frac{g-8}{4}} \lceil \frac{x}{2} \rceil (\lceil \frac{x}{2} \rceil - 1)^i + (\lceil \frac{x}{2} \rceil - 1)^{\frac{g-4}{4}} & g \geq 10, \frac{g}{2} \text{even}, \end{cases} \tag{3}$$

TABLE I
SMALL TRAPPING SETS OF REGULAR CODES

Code	No. of Variables	No. of Checks	Girth	Trapping Sets	No. of Trapping Sets
MacKay-1	1008	504	6	(3,3);(4,4);(5,3)	165; 1215; 14
MacKay-2	816	408	6	(3,3);(4,4);(5,3)	132; 1372; 41
Margulis code	2640	1320	8	(4,4);(5,5)	1320; 11088
Tanner code	155	93	8	(5,3)	155
QC code one	900	450	8	(5,3);(4,4)	50;675
QC code two	900	450	6	(4,2);(4,4)	150;1125
QC code three	900	450	6	(6,2);(4,4);(3,3);	150;1025;200



(a) A $(4,4)$ trapping set (b) A $(5,3)$ trapping set

Fig. 2. Illustration of Trapping sets

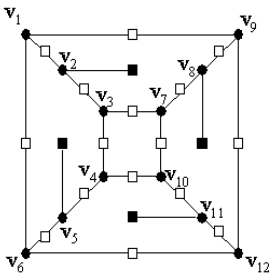


Fig. 3. A $(12,4)$ trapping set of the $(2640, 1320)$ Margulis code

$x = \gamma + 1$ if γ even. In Table I, we show the size and number of small trapping sets of certain $(3,6)$ regular codes.

Note that subgraphs induced by trapping sets are either simple cycles or unions of simple cycles. For example, a $(12,4)$ trapping set of the $(2640, 1320)$ Margulis code [6] is a set of 12 variable nodes and the induced subgraph has 4 odd-degree check nodes. The illustration of the subgraph induced by the trapping set (see Fig. 3), shows that the subgraph can be factored into a set of four simple 12 cycles, namely $\{v_1 - v_2 - v_3 - v_4 - v_5 - v_6,\ v_1 - v_2 - v_3 - v_7 - v_8 - v_9,\ v_7, v_8 - v_9 - v_{10} - v_{11} - v_{12},\ v_4 - v_5 - v_6 - v_{10} - v_{11} - v_{12}\}$. Also, it can be visualized as the interaction of two simple eight cycles, namely $\{v_1 - v_9 - v_{12} - v_6\}$ and $\{v_3 - v_7 - v_{10} - v_4\}$.

The periodic transitions with $T > 1$ can be explained based on trapping sets. An initial state that induces such a periodic transition has the support which is a subset of some trapping set. Also, the union of supports of decoder states over an entire period of an oscillation (or transition) is a trapping set.

B. Aperiodic Transitions

Any sequence of state transitions that do not exhibit periodicity is classified as *aperiodic*. Note that if the support of the initial state is a codeword, then aperiodic state transitions are not possible. Thus if a channel error vector, also the initial state, is nonzero, then aperiodic transitions lead to the propagation of errors across iterations. The support of an initial state that propagates errors is called a *propagating set*. Quite contrary to popular belief, subgraphs induced by minimal propagating sets have been observed to be cycle-free. For example, at high SNR most decoder failures of a $(3195, 2844)$ array code with $\gamma = 5$ are due to cycle-free subgraphs isomorphic to those illustrated in Fig. 4. The variable nodes of these cycle-free subgraphs form propagating sets.

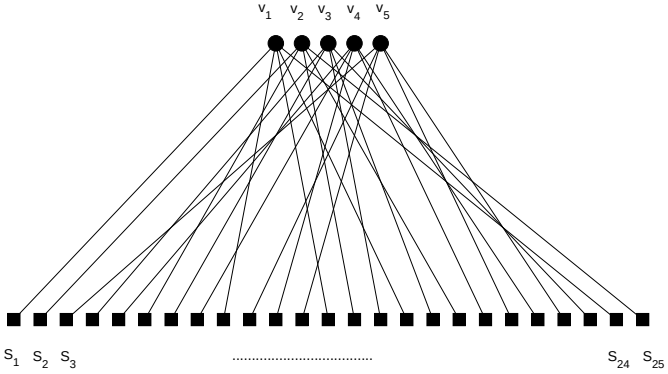


Fig. 4. A $(5,25)$ propagating set of a regular LDPC code with $\gamma = 5$

IV. Failures of the Gallager B Decoder

A *decoding failure* is said to have occurred when the decoded word is different from the transmitted codeword. The characterization of failure events is essential to quantify the performance of a decoder. For example, a maximum likelihood decoder fails if and only if the received word is closer to a codeword that is not the same as the transmitted one. The analysis of failures in Gallager B decoder is dependent on the properties of the code and its graph representation. The failures of the decoder can be due to minimum distance error patterns or low (relative to minimum distance) weight error patterns. These failures can be characterized using trapping sets and propagating sets.

Assume that the all-zero codeword is transmitted over the channel. An error vector (introduced by the channel) can result in one of the following possibilities at the decoder.

1) The decoder, after a finite number of iterations l, corrects the error and arrives at the all-zero state. This is a favorable situation where the decoder exhibits a periodic state transition with $T = 1$ from $i = l$.
2) The decoder, after a finite number of iterations l, exhibits a periodic state transition with $T \geq 1$ from $i = l$ and it never reaches the all-zero state. This is not a favorable situation because this constitutes a decoder failure. Such failures are characterized, as discussed in Section III-A, by trapping sets.
3) The decoder, after a finite number of iterations l, exhibits an aperiodic state transition from $i = l$ and it does not reach the all-zero state. Such a failure is characterized, as discussed in Section III-A, by a propagating set.

The situation described in Item 2 occurs when errors are located (as a result of the channel, i.e. $l = 0$) or transferred (as a result of l iteration) to a set of variable nodes that is contained in a trapping set. For example, it is sufficient for a select three variable nodes of a $(5, 3)$ trapping set to be in error to result in periodic state transition with $T = 2$. Although minimum distance errors are accounted for in this category, "good" codes exhibit trapping set failures that are not minimum distance failures. Thus the probability of such failures depends on the size of trapping sets and on the population of trapping sets.

The situation described in Item 3 occurs when errors are located in a set of variable nodes that contains a propagating set. It is common for propagating set failures to corrupt approximately half of all the variable nodes. The probability of such failures depends on the size of propagating sets and on the population of propagating sets.

V. Application of Failure Analysis

The application of our interest is the estimation of frame error rate (FER) performance of the Gallager B algorithm. We have shown that such an estimation is dependent on size and population of trapping and propagating sets. Although identifying all trapping sets and propagating sets of a code is a daunting task, especially if the code is moderately long and pseudo-randomly constructed, we will show that it is possible to considerably reduce the complexity of estimating the performance. The reduction in complexity is achieved by identifying that in the regime of high signal-to-noise ratio (SNR), performance of the decoder is dominated by trapping and propagating sets of smaller sizes. Similar approaches have been used in estimating the performance of conventional decoders with parameters relating to minimum distance codewords in the regime of high SNR.

A. Performance of $(3, 6)$ LDPC Codes

In [10], we presented a semi-analytical method to estimate the performance of a class of $(3, 6)$ LDPC codes whose decoding failures are dominated by small trapping sets. For example, the performance of the Margulis code, listed in Table I, is dominated by $(4, 4)$ and $(5, 5)$ trapping sets. Similarly, the performance of one of the quasi cyclic codes is dominated by $(5, 3)$ and $(4, 4)$ trapping sets. It is not difficult to identify the dominant trapping sets by running short simulations that track decoder state transitions. The total number of the dominant trapping sets in the code can be easily computed, owing to the small sizes of these sets, by running a search algorithm. For more details, see [10].

B. Performance of Codes from BIBDs

In Section II, the analytical expression for calculating BER for codes derived from affine and projective planes when decoded using a one-step majority logic decoder (or a Gallager B algorithm with 1 iteration) was shown. It was pointed out that the methodology could not be extended to Gallager B algorithm with 2 or more iterations. In this paper, we focus on the problem of estimating the FER of codes from PG$(2, 2^m)$ and AG$(2, 2^m)$. The FER estimate p_{CW} for such codes after l iterations of the Gallager B algorithm can be written as

$$p_{\text{CW}} = \sum_{N_{\text{e}}=t+1}^{n} \Pr(\text{decoding failure}|N_{\text{e}}) \Pr(N_{\text{e}}|\alpha), \qquad (4)$$

where α is the parameter of the binary symmetric channel and N_{e} is the number of errors introduced by the channel. The size of minimal trapping sets of these codes is $t + 1$. For example, the subgraph induced by one such minimal trapping set is illustrated in Fig. 5, where each line corresponds to a variable node of the code from PG$(2, 2^4)$ and each point corresponds to a check node of the code. From simulations we observe that the dominant error events are due to propagating sets and not trapping sets. Due to the large sizes of trapping and propagating sets, it is not possible to count them and hence, we resort to a semi-analytical method (SAM) to estimate the FER of these codes.

The conditional probability of decoding failures is obtained from simulations. A fixed number of errors, say N_{e}, is introduced at random bit positions of a codeword, and the resultant word is decoded using the Gallager B algorithm. Note that these simulations are independent of α and hence, these probabilities can be applied to estimate FERs at very high SNRs. This experiment is repeated to compute the probability of decoder failure for different values of N_{e}. In Fig. 6, observe that conditional probabilities for PG$(2, 2^5)$ and

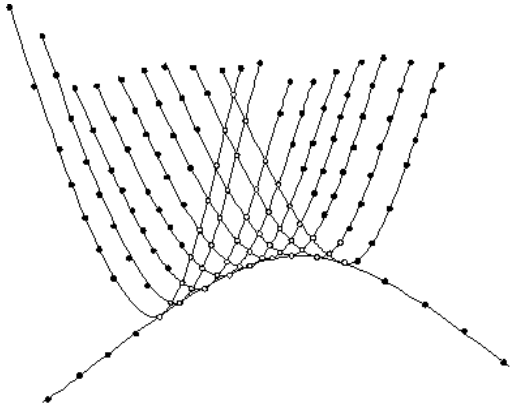


Fig. 5. A minimal trapping set of PG$(2, 2^4)$

AG$(2, 2^5)$ codes approach unity for values of $N_e \approx t+4$. This behavior holds for all codes from projective and affine planes. Therefore, the FER in (4) is modified as

$$p_{\text{CW}} \approx \sum_{N_e=t+1}^{M} \Pr(\text{decoding failure}|N_e)\Pr(N_e|\alpha) + \sum_{N_e=M+1}^{n} 1 \Pr(N_e|\alpha), \quad (5)$$

for some value of M in the neighborhood of t.

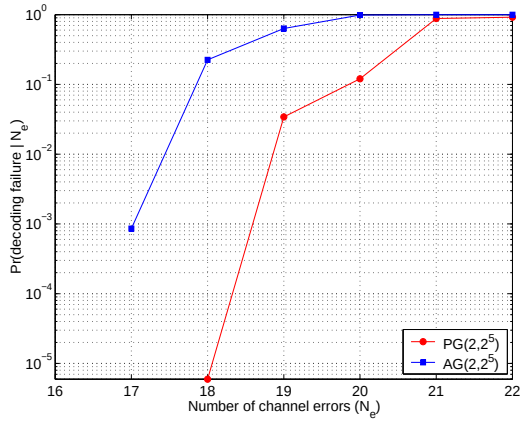
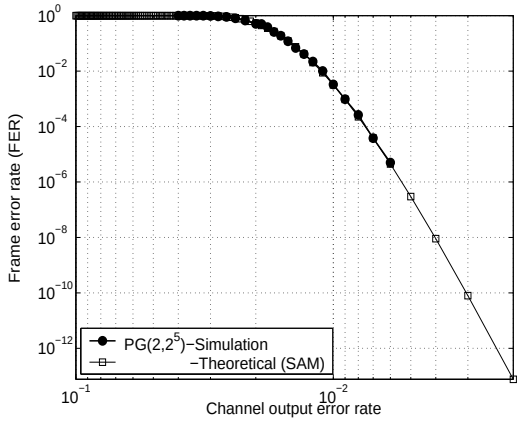


Fig. 6. Probabilities of decoding failures given N_e channel errors

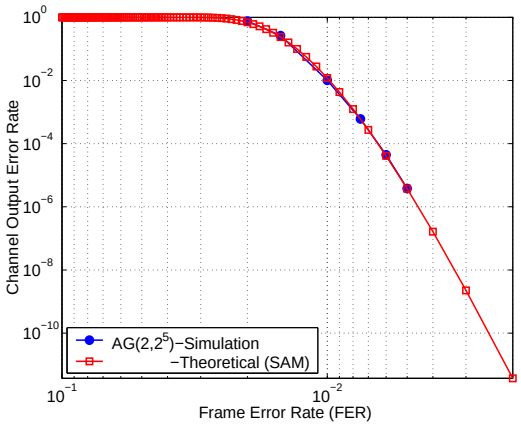
In Fig. 7, we illustrate the success of this approach in long codes such as $(1057, 813)$ projective plane code and $(1056, 813)$ affine plane code. The estimates from the semi-analytical method agree with those obtained from simulations.

VI. Conclusion

In this paper, we successfully applied the knowledge on decoder state transitions to estimate the performace of two class of LDPC codes. The error events in these classes are dominated by trapping sets or propagating sets. An extension of this work to other classes of LDPC codes whose error events are dominated by both trapping and propagating sets.



(a) Comparison between FER estimates and simulated FER for PG$(2, 2^5)$



(b) Comparison between FER estimates and simulated FER for AG$(2, 2^5)$

Fig. 7. FER estimates of codes computed using SAM

References

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

[2] T. J. Richardson and R. L. Urbanke, "The Capacity of Low-Density Parity-Check Codes under Message-Passing Decoding," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 599–618, Feb 2001.

[3] N. Wiberg, *Codes and Decoding on General Graphs*. Univ. Linkoping, Linkoping, Sweden: Ph.D. Dissertation, 1996.

[4] P. Vontobel and R. Koetter, "Graph-cover decoding and finite-length analysis of message-passing iterative decoding of ldpc codes," *IEEE Trans. Inform. Theory*, submitted 2005.

[5] V. Chernyak, M. Chertkov, M. Stepanov, and B. Vasic, "Error Correction on a Tree: An Instanton Approach, Physical Review," *Physical Review Letters*, vol. 93, no. 19, pp. 198 702–1–4, Nov 2004.

[6] T. J. Richardson, "Error Floors of LDPC Codes," in *Proc. 41st Annual Allerton Conf. on Communications, Control and Computing*, 2003.

[7] D. MacKay and M. Postol, "Weaknesses of margulis and ramanujan-margulis low-density parity-check codes," *Electronic Notes in Theoretical Computer Science*, vol. 74, 2003.

[8] A. Shokrollahi, "LDPC Codes: An Introduction." [Online]. Available: http://www.ipm.ac.ir/IPM/homepage/Amin2.pdf

[9] L. D. Rudolph, "A class of mjority logic decodable codes," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 305–307, Apr 1967.

[10] S. K. Chilappagari, S. Sankaranarayanan, and B. Vasic, "Error floors of ldpc codes on binary symmetric channel," in *IEEE Intl. Conference on Communications*, Istanbul, June 2006.