Error Floors of LDPC Codes on the Binary Symmetric Channel

Shashi Kiran Chilappagari, Sundararajan Sankaranarayanan and Bane Vasić

Electrical and Computer Engineering Department
University of Arizona
1230 E. Speedway Blvd.
Tucson, AZ 85721
e-mail: {shashic,ssundar,vasic}@ece.arizona.edu

***Abstract*— In this paper, we propose a semi-analytical method to compute error floors of LDPC codes on the binary symmetric channel decoded iteratively using the Gallager B algorithm. The error events of the decoder are characterized using combinatorial objects called trapping sets, originally defined by Richardson. In general, trapping sets are characteristic of the graphical representation of a code. We study the structure of trapping sets and explore their relation to graph parameters such as girth and vertex degrees. Using the proposed method, we compute error floors of regular structured and random LDPC codes with column weight three.**

I. Introduction

Low-density parity-check (LDPC) codes have been the focus of much research over the past decade, and many of their properties are well understood (see [1] for more details). One of the most important open problems is the phenomenon of *error floor*. Roughly, error floor is an abrupt change in the frame error rate (FER) performance of an iterative decoder in the high signal-to-noise ratio (SNR) region (see [2] for more details).

Error floor characterization is important because LDPC decoders, generally, operate in the SNR-region of gracefully degrading FER. Many communication and storage systems require extremely low FER values, which makes estimating error floor by Monte Carlo simulations virtually impossible. With the exception of the binary erasure channel (BEC) for which FER calculation reduces to a combinatorial problem [3], no analytical method for FER calculation is known. Furthermore, the performance in this case has been studied for an ensemble of codes, not for a given (fixed) code.

In his compact yet very insightful paper [2], Richardson proposed a semi-analytical method to compute error floors of LDPC codes and presented results for additive white Gaussian noise (AWGN) channel. He characterized error events of the decoder with the help of *trapping sets*. Another approach, based on the instanton method, was proposed by Chernyak *et al.* in [4]. A common feature of both these approaches is that they strongly depend on the implementation nuances of the decoding algorithm, such as numerical precision of messages. This makes it difficult to study the impact of code's structure on the error floor of the code. In order to understand the relation between code's structure and error floor, we consider the Gallager B algorithm [5]. Since the Gallager B algorithm operates by passing binary messages along the edges of a graph, any concern about the numerical precision of messages is resolved. (It should be noted that for codes with column weight three, the Gallager B algorithm reduces to the simpler Gallager A algorithm.) This approach has enabled us to establish a connection between trapping sets and FER using enumeration of cycles and cycle interactions, and elementary probability theory. In this paper, we propose a method for estimating the FER performance of LDPC codes on the BSC in the error floor region.

The rest of the paper is organized as follows. In Section II we establish the notation, define trapping sets and provide illustrative examples. In Section III we characterize the various parameters associated with trapping sets. In Section IV we present a semi-analytical method to estimate FER in the error floor region and provide numerical results. We highlight the assumptions and approximations inherent in our method in Section V. We conclude with some remarks in Section VI. A special emphasis in every section is on providing an intuition on the results obtained.

II. Trapping Sets: Definition and Examples

A. Graphical Representations of LDPC Codes

The Tanner graph of an LDPC code, $\mathcal{G}$, is a bipartite graph with two sets of nodes: variable (bit) nodes and check (constraint) nodes. The check nodes (variable nodes) connected to a variable node (check node) are referred to as its neighbors. The degree of a node is the number of its neighbors. In a (γ, ρ) regular LDPC code, each variable node has degree of γ and each check node has degree ρ. The girth g is the length of the shortest cycle in $\mathcal{G}$. A check node is said to be satisfied if the sum of all incoming messages has even parity. In this paper, • represents a variable node, □ represents an even degree check node and ■ represents an odd degree check node.

B. Fixed Points of a Decoder and Trapping Sets

Consider an LDPC code of length n, minimum distance d and girth g. Let $\underline{x}$ be a binary vector of length n and $\mathcal{S}(\underline{x})$ be the support of $\underline{x}$. The support of $\underline{x}$ is defined as the set of all positions i where $x_i \neq 0$. Let f denote the

decoding function. (f is general and can correspond to the one step maximum likelihood decoder, or one iteration of a message passing algorithm for the BEC, the BSC or the AWGN channel). $\mathcal{S}(\underline{x})$ is defined to be a *fixed point* of the decoder if $f(\underline{x}) = \underline{x}$.

For the maximum likelihood decoder the codewords are the only fixed points. For the message passing algorithm on the BEC, stopping sets are also the fixed points. For the Gallager B algorithm on the BSC, error vectors other than codewords can be fixed points. We define such fixed points as trapping sets, adopting the terminology from [2]. A (v, c) trapping set $\mathcal{T}$ is a set of v variable nodes whose induced subgraph has c odd degree checks.

C. Illustrative Examples

Example 1: Consider the $(2640, 1320)$ Margulis code which has girth eight and column weight three. Assume that the all zero codeword was transmitted. Let the received vector be a binary vector whose support is the set of four variable nodes which form an eight-cycle. Fig. 1(a) illustrates the subgraph induced by these four variable nodes. In the 1^{st} iteration, the variable nodes pass the received values to their neighbors (shown in Fig. 2(a)). The messages passed by the check nodes to the variable nodes in the 1^{st} iteration are shown in Fig. 2(b). The structure of Margulis code is such that no two of the odd degree check nodes in the above subgraph are connected to a common variable node. From this property and the message passing rules of the Gallager B algorithm, it is clear that the messages passed by the variable nodes in the next iteration are same as the ones passed in previous round. Hence, same messages will be passed in each round. After the algorithm is run for a fixed number of iterations and majority logic is taken for decoding, we see that the decoded vector will be the same as the received vector. Hence an eight cycle is a fixed point for the Margulis code. From the notation above, an eight cycle is a $(4, 4)$ trapping set.

Example 2: Consider the (155,64) code constructed using permutation matrices [6]. This code has girth of eight and column weight three. This code has sets of variable nodes which induce a subgraph as shown in Fig. 1(b). A similar analysis as above shows that this set of variable nodes also form a fixed point of the decoder. This subgraph is an example of a $(5, 3)$ trapping set. This subgraph is union of three eight cycles such that any two eight cycles have three variable nodes in common.

The above two examples clearly indicate that there can exist small weight error patterns which are not correctable by the decoder. At low values of probability of transition α, the number of errors introduced by the channel decreases. But due to the presence of these small weight error patterns, the FER does not degrade as expected, thereby resulting in error floor. More the number of such patterns, the higher is the error floor. Hence, characterization of such error patterns is vital for error floor estimation. The above examples also motivate the definitions of other trapping set parameters introduced in the following section.

III. Characteristics of Trapping Sets

A (v, c) trapping set $\mathcal{T}$ is a subgraph induced by a specific set of v variables of $\mathcal{G}$. The set of all subgraphs that are equivalent (up to labeling) to $\mathcal{T}$ is said to be a class of trapping sets and is denoted by $\mathcal{X}$. The cardinality of $\mathcal{X}$ is denoted by $|\mathcal{X}|$.

From the definition of trapping set, it follows that whenever the support of a received vector is a trapping set, then the final state of the decoder is the trapping set itself thereby resulting in decoding failure or decoding error. An interesting question is what happens if (a) only a few variable nodes of trapping sets are in error or (b) if variable nodes in the trapping set as well as others which are not in the trapping set are in error.

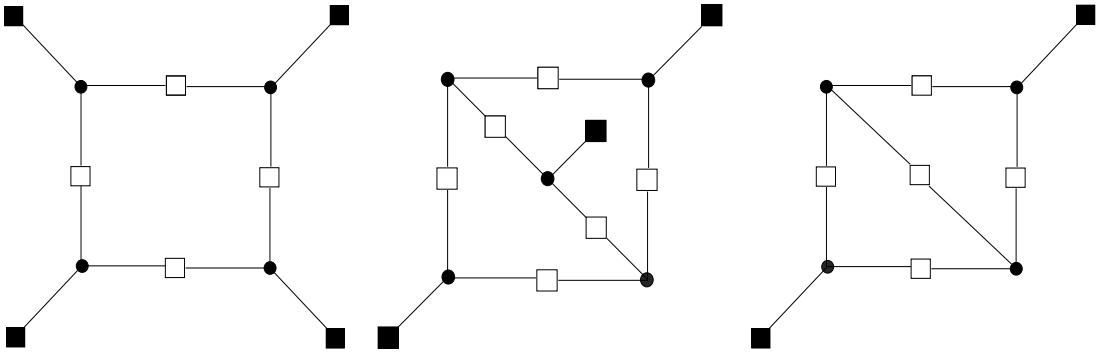
Analysis of the $(4, 4)$ trapping set of *Example 1* shows that the decoder cannot correct an error pattern in which all the four variable nodes are in error but can correct other error patterns with fewer number of errors. In contrast, for the $(5, 3)$ trapping set, as few as three errors in the trapping set can result in a decoding failure. A similar analysis can be carried out for any trapping set to determine the minimum number of nodes in the trapping set that have to be in error to cause a decoding failure. It should be noted that the final state of the decoder need not always be the entire trapping set. Even if the decoder exhibits oscillations, it means that the decoder is not able to correct the error pattern and hence we consider such cases as also ending in a trapping set.

So, with every trapping set $\mathcal{T}$ is associated a *critical number* m defined as the minimum number of nodes in $\mathcal{T}$ that have to be initially in error for the decoder to end in that trapping set. Smaller values of m mean that fewer number of errors can result in decoding failure by ending in that trapping set. However, not all configurations of m errors in a trapping set result in a decoding failure. For example, for the $(5, 3)$ trapping set only one configuration of three errors leads to decoding failure. For the $(4, 2)$ trapping set which is shown in Fig. 1(c) $m = 3$ and all the four combinations of three errors lead to decoding failure. A subset of m nodes $\in \mathcal{T}$ which leads to a decoding failure by ending in a trapping set $\mathcal{T}$ of class $\mathcal{X}$ is called a *failure set*. The number of failure sets of $\mathcal{T}$ is called the *strength of* $\mathcal{T}$ and is denoted by s. Hence, $m = 4$ & $s = 1$ for the $(4, 4)$ trapping set, $m = 3$ & $s = 1$ for the $(5, 3)$ trapping set and $m = 3$ & $s = 4$ for the $(4, 2)$ trapping set. It is clear that a class $\mathcal{X}$ has $s|\mathcal{X}|$ failure sets. Higher values of s and $\mathcal{X}$ imply a greater probability of a set of nodes to contain a failure set.

In the case where m nodes of a trapping set are in error along with some other nodes not in the trapping set, a general conclusion cannot be drawn. However, it is reasonable to assume that even in such cases, the decoder ends in a trapping set. The usefulness and validity of this assumption become clear in the subsequent sections.

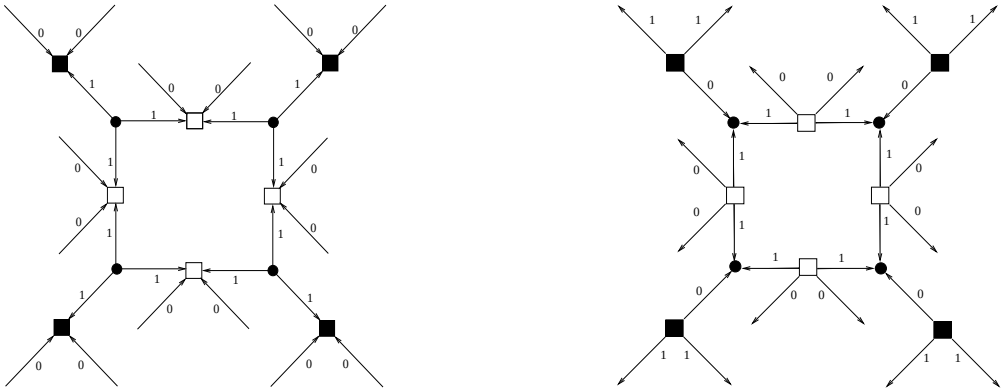
IV. Estimation of FER in the Error Floor Region

In this section we establish the relation between trapping sets and the FER performance of LDPC codes at low values of α of the BSC by proposing a semi-analytical method of



(a) A (4,4) trapping set (b) A (5,3) trapping set (c) A (4,2) trapping set

Fig. 1. Trapping sets



(a) (b)

Fig. 2. Illustration of message passing in the Gallager B algorithm

computation of FER. We then present some results obtained by the application of our method to different codes.

A. A Semi-Analytical Method

The method we propose consists of the following three main steps.

1) Identifying the relevant classes of trapping sets: In [2], Richardson gave the necessary and sufficient conditions for a set of variable nodes to form a trapping set for serial flipping decoder for a (3,6) regular LDPC code. He also proved that the trapping sets for serial flipping algorithm are also trapping set for the Gallager A and the Gallager B algorithm. Using this fact we have identified some structures which can be trapping sets (see [7] for more details). The trapping sets for the Gallager B algorithm turn out to be either cycles or union of cycles. In general, for column-weight-three codes with girth g, cycles of length g and $g+2$ and their unions form the most significant trapping sets. As pointed out earlier, the $(4,4)$ trapping set is an eight cycle, the $(5,3)$ trapping set is a union of three eight cycles and $(4,2)$ trapping set is a union of two six cycles.

2) Calculating the number of trapping sets of each class: The size of a class $\mathcal{X}$ is found by searching the graph. For column-weight-three codes (that are the main focus of this paper) this is particularly easy because the search is equivalent to counting cycles of minimum length and their unions. For example, enumerating $(4,4)$ and $(5,5)$ trapping sets is equivalent to counting eight cycles and ten cycles respectively. The $(5,3)$ trapping set is union of three eight cycles such that any two eight cycles have three variable nodes in common. All the $(5,3)$ trapping sets are identified by listing the eight cycles and checking for sets of three eight cycles which have three nodes in common when taken two at a time. Enumeration is even simpler for structured codes [8], since the underlying symmetries can be exploited.

3) Calculating the contribution of each class of trapping set: The contribution of a class of trapping sets $\mathcal{X}$ (with critical number m and strength s), $\Pr\{\mathcal{X}\}$, to the FER is calculated by:

$$\Pr\{\mathcal{X}\} = \sum_{r=m}^{M} \Pr\{\mathcal{X}|\ r\text{-errors}\}\Pr\{r\text{-errors}\} \quad (1)$$

$$\Pr\{\mathcal{X}|\ r\text{-errors}\} = \frac{s\,|\mathcal{X}|\,\binom{r}{m}}{\binom{n}{m}} \quad (2)$$

$$\Pr\{r \text{ errors}\} = \binom{n}{r}\alpha^r(1-\alpha)^{n-r} \quad (3)$$

$\Pr\{\mathcal{X}|r\text{-errors}\}$ is the probability that the decoder ends in a failure set of the class $\mathcal{X}$, given that the channel introduced r errors. We assume that if m of the r errors belong to a failure

set, the decoder ends in that trapping set (the justification of this assumption is discussed in detail in Section V). $s\,|\mathcal{X}|\,/\binom{n}{m}$ is the probability that a given set of m variable nodes is a failure set of class $\mathcal{X}$. There are $\binom{r}{m}$ of subsets with cardinality m for a set with r elements. Hence, $s|\mathcal{X}|\binom{r}{m}/\binom{n}{m}$ is the probability that this set of r errors consists of at least one failure set of $\mathcal{X}$. Pr$\{r$-errors$\}$ is the probability that the channel introduced r errors and is calculated using binomial expansion as in (3).

M is the maximum number of errors which can end up in a trapping set. It is defined as the *susceptibility* of a trapping set to additional errors. If more than M errors are introduced by the channel, the decoder still fails to correct these errors but in this case the decoder does not end in a trapping set. For a given α, the number of errors introduced by the channel will be binomially distributed with mean $n\alpha$. Hence most of the error vectors have weights centered around $n\alpha$. For the values of α for which $n\alpha$ is much less than M, most of the error events will be due to trapping sets. For values of α such that $n\alpha$ is comparable to M, the FER is worse but the contribution of trapping set errors is smaller. In other words, M determines the value of α at which trapping set errors start dominating. The calculated FER approaches the real value for lower and lower values of α. Determining M is non trivial, and is found by a combination of heuristics, simulations and code properties like minimum distance.

B. Numerical Results

We present results of our FER estimation method applied to different classes of codes. Table I summarizes the parameters of the codes considered along with the types of dominant trapping sets and their cardinality. We consider regular random and structured codes with different lengths and girths and possessing different types of trapping sets. We consider three quasi-cyclic (QC) codes constructed from circulant permutation matrices [9]. These codes are abbreviated as QC Codes. The corresponding FER plots are provided in Figs. 3(a-g). These codes are the examples where the dominant error events are due to trapping sets. From the plots it is clear that the method is successful in predicting the performance at low values of α and capturing the error floor.

V. Assumptions and Approximations

In this section we highlight some of the approximations and assumptions made in the method outlined in Section IV. We also present a brief overview of the computational complexity issues of the proposed method.

The probability δ that a set of m variable nodes is a failure set of class $\mathcal{X}$ is

$$\delta = \frac{s|\mathcal{X}|}{\binom{n}{m}}.$$

In a set of cardinality r, there are $\binom{r}{m}$ subsets with m elements. The probability that a set of r elements consists of at least one failure set of class $\mathcal{X}$ is

$$1-(1-\delta)^{\binom{r}{m}} \approx \binom{r}{m}\delta.$$

This approximation is made in (2) and is valid since the value of δ is small and the subsequent terms of binomial expansion are negligible.

We assumed that if m of the r errors form a failure set, the decoder ends in that trapping set. It is possible that the channel introduces r errors of which m belong to a failure set and the decoder still corrects these errors. To account for this we introduce sensitivity β of a trapping set. Let $E_{\mathcal{T}}^{m+1}$ denote the set of all $m+1$ weight error patterns of which m belong to $\mathcal{T}$. Let $U_{\mathcal{T}}^{m+1} \subseteq E_{\mathcal{T}}^{m+1}$ denote the set of error patterns which end in $\mathcal{T}$. The sensitivity of $\mathcal{T}$ to addition of one extra node, β_1 is defined as

$$\beta_1 = \frac{|U_{\mathcal{T}}^{m+1}|}{|E_{\mathcal{T}}^{m+1}|}$$

Sensitivity of trapping set to addition of more errors can be defined similarly. In general, β_1 dominates in the neighborhood of m and for considerably higher values of r the sensitivity can be taken as 1.

Example 3: Consider the $(4,4)$ trapping set $\mathcal{T}$ of the $(2640, 1320)$ Margulis code. For this $\mathcal{T}$, $m=4$ and $|E_{\mathcal{T}}^{m+1}| = 2640-4 = 2636$. Each of the satisfied checks of $\mathcal{T}$ is connected to $\rho-2=4$ variable nodes outside the trapping set as shown in Fig.4. If the fifth error is in one of these $4*4=16$ nodes, the decoder corrects this error pattern. For all other $2636-16=2620$ cases, the decoder ends in $\mathcal{T}$. Therefore $|U_{\mathcal{T}}^{m+1}| = 2620$. Hence, β_1 for $\mathcal{T}$ is

$$\beta_1 = 2620/2636 = 0.9939.$$

Similarly, β_1 for the $(5,3)$ trapping set is found to be 0.8816. In all our calculations, we assumed β equal to 1. The results clearly indicate that this is a reasonable assumption.

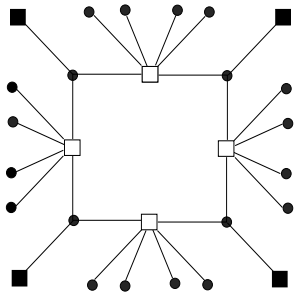


Fig. 4. Illustration of sensitivity of a trapping set

A note on the complexity of the method: As mentioned in Section IV-A.1, trapping sets for the Gallager B algorithm turn out to be cycles and union of cycles. The Margulis code has girth of 8 and eight cycles are trapping sets for this code. Each variable node is involved in two eight cycles. From the automorphism group of the code the total number of eight cycles can be calculated as $(2640\times 2)/4 = 1320$. Similar calculation can be carried out for counting $(5,5)$ trapping sets (ten-cycles).

For the $(155, 64)$ Tanner code, each node is involved in 12 eight cycles and hence the total number of eight cycles is 465. The $(5,3)$ trapping set is formed by the union of three eight

TABLE I
DIFFERENT CODES AND THEIR PARAMETERS

Code	No. of Variables	No. of Checks	Girth	Relevant Trapping Sets	No. of Trapping Sets	M
MacKay's code one	1008	504	6	(3,3);(4,4);(5,3)	165; 1215; 14	≈ 30
MacKay's code two	816	408	6	(3,3);(4,4);(5,3)	132; 1372; 41	≈ 25
Margulis code	2640	1320	8	(4,4);(5,5)	1320; 11088	≈ 95
Tanner code	155	93	8	(5,3)	155	≈ 10
QC code one	900	450	8	(5,3);(4,4)	50;675	≈ 25
QC code two	900	450	6	(4,2);(4,4)	150;1125	≈ 25
QC code three	900	450	6	(6,2);(4,4);(3,3);	150;1025;200	≈ 25

cycles. Again the symmetry of the code helps us in calculating the number of $(5,3)$ trapping sets to be $465/3 = 155$.

Even though counting cycles and cycle interactions is relatively easy for structured codes, our method can be applied to regular random codes as well. We considered two random codes taken from MacKay's webpage [10], both having girth 6. The counting of all trapping sets by searching took about five minutes on a 1.4GHz processor computer.

VI. CONCLUSION

In this paper, we have established a relation between the trapping sets and the FER of a code. We characterized the trapping sets in terms of cycles and their unions. We then presented a semi-analytical method to compute error floors of LDPC codes transmitted over the BSC. The method provides an insight into the effects of code structure on its performance. Although the method is similar in principle to that proposed by Richardson [2], it is much simpler due to its combinatorial nature. The method established the relationship between the cycles in the code representation and the code's performance. The method was successfully applied in evaluating the error floors of column-weight-three LDPC codes. The problem left for future research is a rigorous characterization of trapping set susceptibility to additional errors.

APPENDIX
GALLAGER B ALGORITHM

The Gallager B algorithm on the the BSC operates by passing binary messages along the edges of the Tanner graph of an LDPC code. Every round of message passing (iteration) starts with sending messages from variable nodes and ends by sending messages from check nodes to variable nodes. For a variable node v (check node c), let $E(v)$ ($E(c)$) denote the edges incident on v (c). Also, let $r(v)$ denote the received value of node v. Let the degree of a variable node be j. A threshold $b_{i,j}$ is determined for every iteration i and variable degree j. Let $\overrightarrow{\mathrm{m_i}}\ (e)$ and $\overleftarrow{\mathrm{m_i}}\ (e)$ represent the messages passed on an edge e from variable node to check node and check node to variable node in iteration i respectively. Then for each node v, the Gallager B algorithm passes the following messages in i^{th} iteration.

$$\overrightarrow{\mathrm{m_i}}\ (e) \quad = \quad r(v), \qquad i = 1$$

For $i > 1$

$$\overrightarrow{\mathrm{m_i}}\ (e) \quad = \quad \begin{cases} 1, & \text{if } \sum_{e' \in E(v)\backslash\{e\}} \overleftarrow{\mathrm{m_{i-1}}}\ (e') \geq b_{i,j}, \\ 0, & \text{if } j - 1 - \sum_{e' \in E(v)\backslash\{e\}} \overleftarrow{\mathrm{m_{i-1}}}\ (e') \leq b_{i,j} \\ r(v), & \text{otherwise} \end{cases}$$

For each check node c, the messages passed in i^{th} iteration are

$$\overleftarrow{\mathrm{m_i}}\ (e) = \left(\sum_{e' \in E(v)\backslash\{e\}} \overrightarrow{\mathrm{m_i}}\ (e') \right) \ mod\ 2$$

ACKNOWLEDGMENT

This work was supported by grants from INSIC-EHDR and NSF-CCR (grant no. 0208597). The authors would like to thank Gianluigi Liva for providing codes constructed from circulant permutation matrices.

REFERENCES

[1] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 619–637, Feb 2001.

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

[3] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. Spielman, "Efficient Erasure Correcting Codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 569–584, Feb 2001.

[4] 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.

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

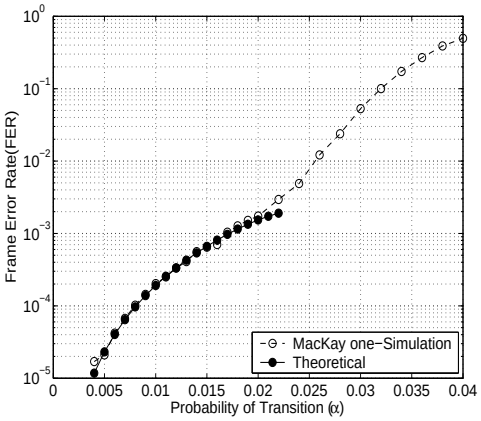
[6] R. M. Tanner, D. Sridhara, and T. Fuja, "A Class of Group-Structured LDPC Codes," in *ISCTA 2001*, Ambleside, England, 2001.

[7] B. Vasic, "Error Floors of LDPC Codes on Binary Symmetric Channel." [Online]. Available: http://www.ece.arizona.edu/∼shashic/ctw_05_m.pdf

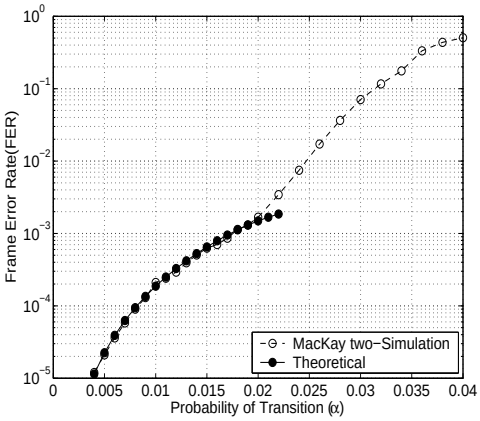
[8] B. Vasic and O. Milenkovic, "Combinatorial Constructions of Low-Density Parity-Check Codes for Iterative Decoding," *IEEE Trans. Inform. Theory*, vol. 50, no. 6, pp. 1156–1176, June 2004.

[9] M. P. C. Fossorier, "Quasi-Cyclic Low-Density Parity-Check Codes from Circulant Permutation Matrices," *IEEE Trans. Inform. Theory*, vol. 50, no. 8, pp. 1788–1793, 2004.

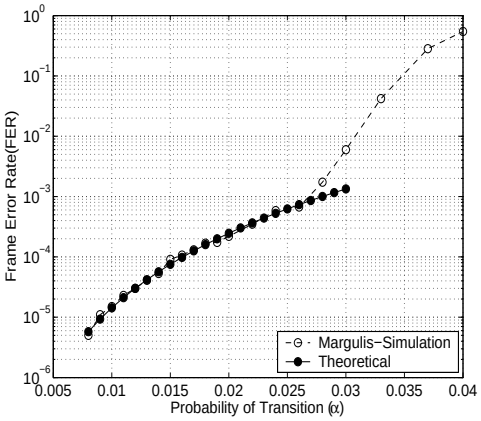
[10] D. J. C. MacKay, "Encyclopedia of Sparse Graph Codes." [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html



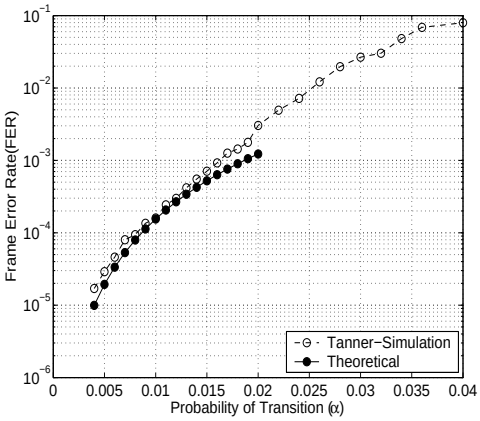
(a) MacKay's random code one



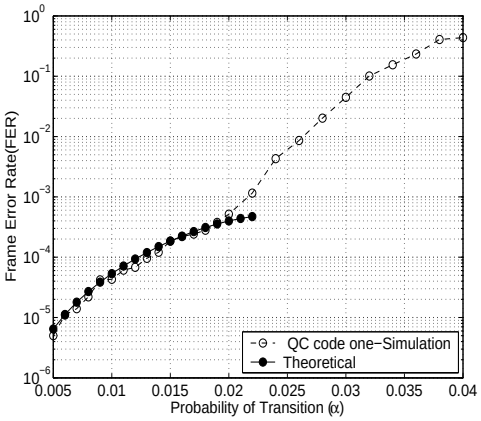
(b) MacKay's random code two



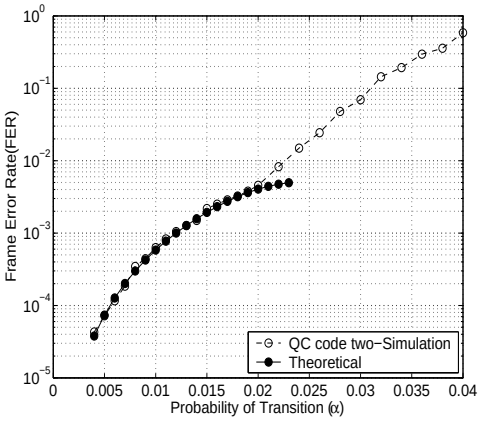
(c) Margulis code



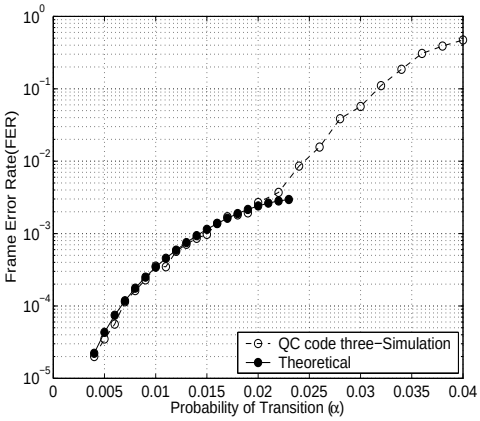
(d) Tanner code



(e) QC code one



(f) QC code two



(g) QC code three

Fig. 3. FER plots for different classes of codes