The Distribution of Loop Lengths in Graphical Models for Turbo Decoding

Xianping Ge, David Eppstein, Padhraic Smyth

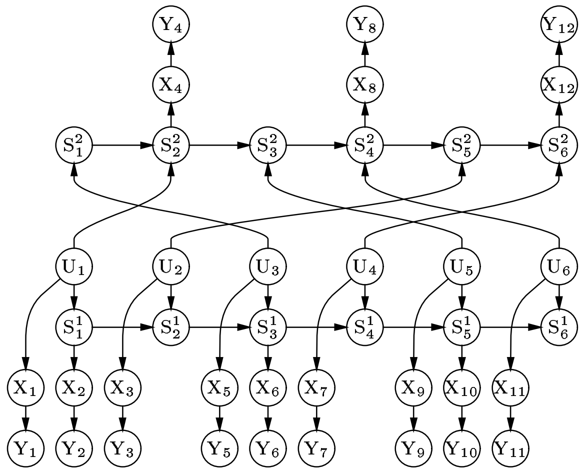


Fig. 1
THE ADG MODEL FOR A $K = 6$, $N = 12$, RATE $1/2$ TURBO CODE.

Abstract— **This paper analyzes the distribution of loop lengths in graphical models for turbo decoding. The properties of such loops are of significant interest in the context of iterative decoding algorithms based on belief propagation. We estimate the probability that there exist no loops of length less than or equal to c at a randomly chosen node in the acyclic directed graphical (ADG) model for turbo decoding, using a combination of counting arguments and approximations. When K, the number of information bits, is large, this probability is approximately $e^{-\frac{2^{c-1}-4}{K}}$, for $c \geq 4$, where nodes for input information bits are ignored for convenience. The analytical results are validated by simulations. For example, for turbo codes with $K = 64000$, a randomly chosen node has a less than 1% chance of being on a loop of length less than or equal to 10, but has a greater than 99.9% chance of being on a loop of length less than or equal to 20.**

Keywords— **Belief propagation, graphical models, iterative decoding, turbo code.**

I. Introduction

Turbo codes are a new class of coding systems that offer near optimal coding performance while requiring only moderate decoding complexity [1]. It is known that the widely-used iterative decoding algorithm for turbo codes is in fact a special case of Pearl's local message-passing algorithm [2] for efficiently computing posterior probabilities in acyclic directed graphical (ADG) models (also known as "belief networks") [3], [4]. Figure 1 shows the ADG model for a rate 1/2 turbo code with $N = 12$ codeword bits and $K = 6$ information bits. (For real turbo codes, K can be much larger, e.g. $K = 64000$.)

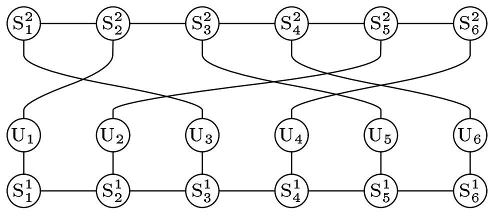
Pearl's algorithm is provably convergent to the true posterior probabilities provided that the graph structure of the ADG model does not contain any loops in the undirected version of the ADG (i.e., the graph where directionality of the edges is dropped). This "loop-less" condition ensures that there is only one path of message-passing from one node to another. Otherwise, the information from one node can be over-counted at another node. It is well-known that message-passing in such graphs with loops can converge to incorrect posterior probabilities (e.g., [5]). Thus, we have the "mystery" of turbo decoding: why does a provably incorrect algorithm produce an extremely useful and practical decoding algorithm? In the remainder of this paper we take a step in understanding this by characterizing the distribution of loop lengths. The motivation is as follows: if it turns out that loop lengths are "long enough" then there may be a well-founded basis for believing that message-passing in graphs with loops of the appropriate length are not susceptible to the "over-counting" problem (i.e., that the effect of long loops in practice may be negligible). This is somewhat speculative and we will return to this point in Section V. An additional motivating factor is that the characterization of loop length distributions in turbo codes is of fundamental interest by itself. For example, the loop length distributions may be related to the weight distributions of the code, although it must be pointed out that the loop length distributions, by themselves, are not sufficient for designing codes with good weight distributions.

In Section II, we derive analytical approximations for the probability that there are no loops of length c or less at a randomly chosen node in the ADG model for turbo codes. In Section III, we provide numerical results based on the analytical approximations and compare them with the simulation results. In Section IV we briefly discuss extending the techniques to various extensions of turbo codes and to other codes with similar iterative decoding algorithms. Section V contains a discussion of what these results may imply for iterative decoding in a general context and Section VI contains the final conclusions.

II. Theoretical Analysis

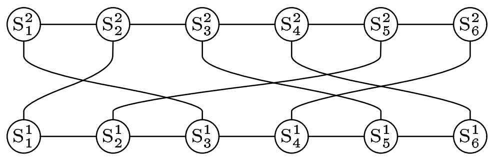
When characterizing the distribution of loop lengths in the ADG model we drop the X, Y nodes (since they are not on any loops) and the directionalities of the edges. Figure 2(a) shows the resulting *loop graph* of the turbo

The authors are with Information and Computer Science, University of California, Irvine, CA 92697-3425 USA. Email: {xge,eppstein,smyth}@ics.uci.edu. This paper was presented in part at ISIT 2000. This work was supported in part by AFOSR grant F49620-97-1-0313.



S_1^2 S_2^2 S_3^2 S_4^2 S_5^2 S_6^2
U_1 U_2 U_3 U_4 U_5 U_6
S_1^1 S_2^1 S_3^1 S_4^1 S_5^1 S_6^1

(a)



S_1^2 S_2^2 S_3^2 S_4^2 S_5^2 S_6^2
S_1^1 S_2^1 S_3^1 S_4^1 S_5^1 S_6^1

(b)

Fig. 2

The loop graph underlying the turbo code of Figure 1: (a) The X, Y nodes are dropped and the directionalities of the edges are also dropped, (b) The U nodes are also dropped.

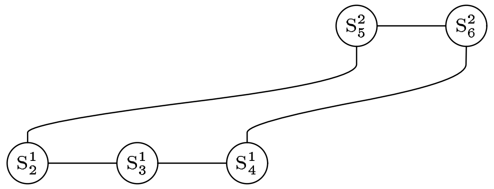


Fig. 3

A loop in Figure 2(b).

code in Figure 1. To simplify our analysis further, in Figure 2(b), the U nodes are also dropped. Results on this simplified loop graph can be easily extended to the original loop graph. In what follows, we will look at graphs such as Figure 2(b), in which

1. there are two parallel chains, each having K nodes,
2. each node is connected to one (and only one) node on the other chain, and
3. the 1-to-1 connections between the nodes on the two chains are chosen randomly, e.g., by a random permutation of the sequence $\{1, 2, \dots, K\}$.

To count the loops of various lengths in graphs like Figure 2(b), we define a loop of length c to be a sequence of c consecutive edges with no repeated node except that the starting node is also the ending node. We label the edges in a loop (see Figure 3 for an example) as follows:

1. $\rightarrow$: "Left-to-right on a chain" (e.g., $S_2^1S_3^1$ in Figure 3).
2. $\leftarrow$: "Right-to-left on a chain" (e.g., $S_6^2S_5^2$).
3. $|$: "Going from one chain to another" (e.g., $S_4^1S_6^2$).

For example, the loop S_2^1–S_3^1–S_4^1–S_6^2–S_5^2–S_2^1 will be labeled $\rightarrow\rightarrow|\leftarrow|$. In general, starting from a given node, a loop will have two label sequences, as it can be traversed in two directions. A label sequence for a loop must satisfy the following conditions:

1. The label "$\rightarrow$" cannot be adjacent to "$\leftarrow$". Note that the first and the last edges are adjacent to each other in a loop.
2. The label "$|$" cannot be adjacent to another "$|$".
3. The number of "$|$" labels must be even.

A label sequence satisfying the above conditions will be called a *candidate loop labeling*.

A. Number of candidate loop labelings

Let $NL(c, m)$ be the number of possible candidate loop labelings of length c and with m "$|$" labels.

First, we calculate the number of ways of putting m "$|$" labels on a loop of c edges so that no two "$|$" labels are adjacent to each other. This number is

$$\binom{(c-1)-(m-1)}{m} + \binom{(c-3)-(m-2)}{m-1} = \frac{c}{c-m}\binom{c-m}{m}. \qquad (1)$$

When the m edges are labeled "$|$", there will be m segments of unlabeled edges (delimited by the "$|$" labels). Each segment can be labeled either $\rightarrow$ or $\leftarrow$, so there will be 2^m ways of labeling these m segments.

So the total number of ways of labeling the edges of a loop of length c and with m "$|$" labels is $\frac{c}{c-m}\binom{c-m}{m} \times 2^m$. Every loop has two labeling sequences (by traversing in two directions), so we divide the total number by 2:

$$NL(c, m) = \frac{c}{c-m}\binom{c-m}{m} \times 2^m/2 = 2^{m-1}\frac{c}{c-m}\binom{c-m}{m}. \qquad (2)$$

B. Probability of finding a given candidate loop labeling at a randomly chosen node in a loop graph

Given a specific candidate loop labeling L of length c and with m "$|$" labels, and given a randomly chosen starting node A in a specific loop graph, we calculate the probability that the specific loop L exists at A.

Without loss of generality, suppose L begins with a label "$\rightarrow$" or "$\leftarrow$". (If L begins with the label "$|$", the last label will be "$\rightarrow$" or "$\leftarrow$", so we can start with the last label and traverse in the reverse direction.) We traverse a_1 edges (in the direction of the corresponding labels in L) on the same chain as the starting node, "chain 1", then we go to the other chain, "chain 2", via a "$|$" edge. Once on chain 2, we traverse b_1 edges on it (also in the direction of the labels), then come back to chain 1 via another "$|$" edge. This group of 4 steps is repeated $\frac{m}{2}$ times. Let a_t, b_t, $1 \le t \le \frac{m}{2}$ be the number of edges on chains 1 and 2 respectively for the i-th group, where $a_t, b_t \ge 1$.

After the last 4-step group, there are $a_{\frac{m}{2}+1} \ge 0$ labels at the end of L. If the last label of L is "$|$", $a_{\frac{m}{2}+1} = 0$, otherwise $a_{\frac{m}{2}+1} > 0$, and we have a stand-alone step to traverse $a_{\frac{m}{2}+1}$ edges on chain 1.

We find a loop if every step of the above process is successful. A step is unsuccessful if it arrives at a previously traversed node (except for the last step where arriving at the starting node A is required), or it attempts "$\rightarrow$" at the right boundary of a chain, or "$\leftarrow$" at the left boundary of a chain. When calculating the probabilities of success of the steps, the following facts are relevant:

• The starting node A is a randomly chosen node and can be anywhere on a chain.

• When going from node S^1_i on chain 1 to chain 2, because the connections between nodes on the two chains are chosen randomly, we can arrive at any node S^2_j on the other chain except the nodes that are known to be connected to chain-1 nodes other than S^1_i (and similarly for going from a node on chain 2 to chain 1).

The probabilities of success for the steps in the t-th 4-step group are as follows:

1. Traversing a_t edges on chain 1: The a_t+1 nodes on the a_t edges should not contain any node traversed by a previous group. The previous $t-1$ groups left behind between 1 and t segments of consecutive un-traversed nodes. So the probability of success $p^{(1)}_t$ is

$$1-\frac{(t-1)a_t}{K-\sum_{i=1}^{t-1}(a_i+1)} \le p^{(1)}_t \le 1-\frac{a_t}{K-\sum_{i=1}^{t-1}(a_i+1)}. \tag{3}$$

2. Going to chain 2: This step will be unsuccessful if we arrive at any previously traversed node on chain 2, although we need not worry about the $2(t-1)$ nodes on edges labeled "|" (these nodes are known to be connected to some other different nodes on chain 1).

$$p^{(2)}_t = 1-\frac{\sum_{i=1}^{t-1}(b_i-1)}{K-2(t-1)}. \tag{4}$$

3. Traversing b_t edges on chain 2: This is similar to step 1.

$$1-\frac{(t-1)b_t}{K-\sum_{i=1}^{t-1}(b_i+1)} \le p^{(3)}_t \le 1-\frac{b_t}{K-\sum_{i=1}^{t-1}(b_i+1)}. \tag{5}$$

4. Coming back to chain 1: This is similar to step 2.

$$p^{(4)}_t = 1-\frac{1+\sum_{i=1}^{t}(a_i-1)}{K-(2t-1)}. \tag{6}$$

To be successful, the stand-alone step at the end must arrive at the starting node A after traversing the $a_{\frac{m}{2}+1}$ edges on chain 1. So its probability of success can be approximated as:

$$p^{(1)}_{\frac{m}{2}+1} \approx \frac{1}{K-\sum_{i=1}^{\frac{m}{2}}(a_i+1)}. \tag{7}$$

If $a_{\frac{m}{2}+1} > 0$, the probability of success of the whole process is

$$p(K,c,m) \approx \left(\prod_{t=1}^{\frac{m}{2}} p^{(1)}_t p^{(2)}_t p^{(3)}_t p^{(4)}_t\right) p_{\frac{m}{2}+1}(1). \tag{8}$$

If $a_{\frac{m}{2}+1} = 0$, the 4th step of the last 4-step group will end the whole process, so $p^{(4)}_t$ for $t=\frac{m}{2}$ will be different from Equation (6), and the term $p_{\frac{m}{2}+1}(1)$ is not needed. But the final result will be approximately the same as Equation 8 above.

Note that $\sum_{i=1}^{\frac{m}{2}+1} a_i + \sum_{i=1}^{\frac{m}{2}} b_i = c-m$, $a_i, b_i \ge 1$ for $1 \le i \le \frac{m}{2}$, and $a_{\frac{m}{2}+1} \ge 0$, Equation (8) can be bounded by

$$\begin{aligned} & \frac{1}{K-m}\prod_{t=1}^{\frac{m}{2}}\left[\left(\frac{K+2tm-tc-3t+3}{K+2m-c-2t+2}\right)^2\right. \\ & \left.\left(1-\frac{c-2m}{K-2t+2}\right)\left(1-\frac{c-2m+1}{K-2t+1}\right)\right] \\ \le\ & p(K,c,m) \qquad (9) \\ \le\ & \frac{1}{K-c+m}\prod_{t=1}^{\frac{m}{2}}\left[\left(1-\frac{1}{K-2t+2}\right)^2\right. \\ & \left.\left(1-\frac{1}{K-2t+1}\right)\right]. \end{aligned}$$

The ratio between the upper bound and the lower bound is close to 1. For example, when $K=64000, c=10, m=4$, the ratio is 1.0002. Given that the bounds are so close in the range of K, c, and m of interest, in the remainder of the paper we will simply approximate $p(K,c,m)$ by the arithmetic average of the upper and the lower bounds.

C. Probability of no loops of length c or less

Denote the candidate loop labelings of length c as L_1, L_2, ..., and let $\overline{L_i}$ mean that L_i is unsuccessful. The probability that none of the possible candidate loop labelings is successful can be approximated as follows:

$$\begin{aligned} & p(\text{no loops of length } c) \\ =\ & p(\overline{L_1}, \overline{L_2}, \ldots) \\ \approx\ & \prod_i p(\overline{L_i}) \\ =\ & \prod_{\substack{m:2\le m\le c/2 \\ m \text{ is even}}} (1-p(K,c,m))^{NL(c,m)}. \end{aligned} \tag{10}$$

In this independence approximation we are assuming that the successes of the candidate loop labelings are independent of one another. This is not strictly true, but appears to be a good approximation to first order.

Similarly, the probability that there are no loops of length c or less will be approximated as follows:

$$\begin{aligned} & p(\text{no loops of length } c \text{ or less}) \\ \approx\ & \prod_{i=4}^{c} p(\text{no loops of length } i), \end{aligned} \tag{11}$$

where we make another independence assumption that the events "no loops of length i", for $4 \le i \le c$, are independent of one another (again, not strictly true, but likely to be a good first order approximation).

Using Equations (9), (10), and (11), we can now analytically estimate $p(\text{no loops of length } c \text{ or less})$.

c	$p_{\text{simulation}}$ (a)	$p_{\text{theoretical}}$ (b)	$a-b$
4	0.99994900	0.99993750	0.00001150
5	0.99979300	0.99978127	0.00001173
6	0.99950100	0.99950013	0.00000087
7	0.99904100	0.99906295	-0.00002195
8	0.99814200	0.99818917	-0.00004717
9	0.99618000	0.99622599	-0.00004599
10	0.99201000	0.99203233	-0.00002233
11	0.98391600	0.98388264	0.00003336
12	0.96872700	0.96844956	0.00027744
13	0.93864800	0.93862939	0.00001861
14	0.88064100	0.88075430	-0.00011330
15	0.77345100	0.77414022	-0.00068922
16	0.59722500	0.59830123	-0.00107623
17	0.35751600	0.35878319	-0.00126719
18	0.12877900	0.12942941	-0.00065041
19	0.01655100	0.01676785	-0.00021685
20	0.00025700	0.00027848	-0.00002148

TABLE I
SIMULATION VS. THEORETICAL ESTIMATES OF THE PROBABILITY OF NO LOOPS OF LENGTH c OR LESS, AT A RANDOMLY CHOSEN NODE IN THE ADG MODEL OF TURBO CODES WITH K=64000, AS A FUNCTION OF c. THE U NODES ARE IGNORED.

III. NUMERICAL AND SIMULATION RESULTS

For each value of chain length $K = 1000$, 2000, 10000, 64000, we constructed 10000 different graphs (i.e., each graph has a different random permutation) of the corresponding length, and for each graph, we counted the loops of length $c = 4, 5, \ldots, 20$, at 100 randomly chosen nodes. In total, the loop counts at $1,000,000$ nodes are collected to generate an empirical estimate of the true p(no loops of length c or less) for each value of K.

The simulation estimates, together with the theoretical estimates and their differences, for $K = 64000$, are shown in Table I. The difference in error is never greater than about 0.005 in probability. Note that neither the simulation estimates nor the theoretical estimates are exact. Thus, differences between the two may be due to either sampling variation or error introduced by the approximations.

Figure 4 shows the plot and log plot of the simulation and theoretical estimates for K=1000, 2000, 10000, and 64000. There appears to be a "soft threshold effect" in the sense that beyond a certain value of c, it rapidly becomes much more likely that there are loops of length c or less at a randomly chosen node. The location of this threshold increases as K increases (i.e., as the length of the chain gets longer).

To characterize the curves in the plot, we look at the limiting case when K is very large, i.e., $K \gg c$. In this case, the total number of candidate loop labelings of length

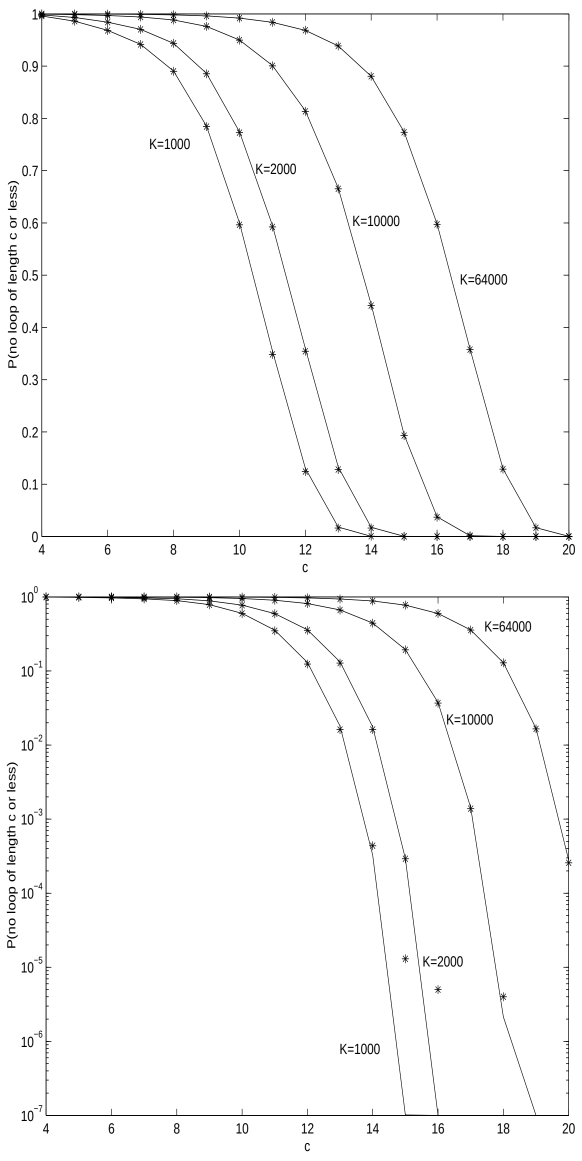


Fig. 4
SIMULATION (* SYMBOLS) VS. THEORETICAL ESTIMATES (SOLID LINES) OF THE PROBABILITY OF NO LOOPS OF LENGTH c OR LESS, AT A RANDOMLY CHOSEN NODE, AS A FUNCTION OF c, FOR K=1000, 2000, 10000, AND 64000.

c is

$$\begin{aligned} NL(c) &= \sum_{\substack{m:2\leq m\leq c/2\\ m \text{ is even}}} NL(c,m) \\ &= \sum_{\substack{m:2\leq m\leq c/2\\ m \text{ is even}}} 2^{m-1}\frac{c}{c-m}\binom{c-m}{m} \\ &\approx 2^{c-2}, \end{aligned} \tag{12}$$

and the probability of successfully finding a loop labeled with a given candidate loop labeling is approximately $\frac{1}{K}$, so the probability of no loops of length c or less at a randomly chosen node is

$$\begin{aligned}
& p(\text{no loops of length } c \text{ or less}) \\
\approx & \prod_{i=4}^{c} \left(1 - \frac{1}{K}\right)^{2^{i-2}} \\
= & \left(1 - \frac{1}{K}\right)^{2^{c-1}-4} \\
\approx & e^{-\frac{2^{c-1}-4}{K}}. \qquad (13)
\end{aligned}$$

This probability equals 0.5 at $c_{0.5} = \log_2(K \log 2 + 4) + 1$, which provides an indication of how the curve will shift to the right as K increases. Roughly speaking, to double $c_{0.5}$, one would have to square the chain length from K to K^2.

IV. Other Results

Up to this point we have been ignoring the U nodes when counting the loops. The results can readily be extended to include these U nodes by counting each edge labeled with "|" (that connects nodes from different chains) as two edges.

Various extensions of turbo codes are also amenable to this form of analysis. For example, for the case of a turbo code with more than two constituent encoders, one can generalize the notion of candidate loop labeling and count accordingly. As another example, we applied the same techniques of counting loops to turbo codes with S-random permutation [6]. In both simulation and analysis the S-random construction is shown to eliminate very short loops and for larger loops results in only a small systematic decrease in the probability of such loops [7].

For other codes with similar iterative decoding algorithms to turbo codes, the same techniques of analysis can be applied. For example, for low density parity check (LDPC) codes [8], [9], we find that the loop length distribution shows qualitatively similar behavior to that of turbo codes, although the analytic approximations are less accurate than for the turbo codes (when compared to simulation results) [7].

V. Connections to Iterative Decoding

For turbo codes we have shown that randomly chosen nodes are relatively rarely on a loop of length 10 or less, but are highly likely to be on a loop of length 20 or less (for a block length of 64000). It is interesting to conjecture about what this may tell us about the accuracy of the iterative message-passing algorithm in this context.

It is possible to show that there is a well-defined "distance effect" in message propagation for typical ADG models. Consider a simple model where there is a hidden Markov chain consisting of binary-valued S_i state nodes, $1 \leq i \leq K$. In addition there are observed Y_i's, one for each state S_i and which only depend directly on each state S_i. $p(\mathrm{Y}_i|\mathrm{S}_i)$ is a conditional Gaussian with mean S_i and standard deviation σ. One can calculate the effect of any observed Y_i on any hidden node S_j, $j < i$, in terms of the expected difference between $p(\mathrm{S}_j|\mathrm{Y}_j, \ldots, \mathrm{Y}_{i+1})$ and $p(\mathrm{S}_j|\mathrm{Y}_j, \ldots, \mathrm{Y}_i)$, averaged across many observations of the Y's. This average change in probability, from knowing Y_i, can be shown to die off exponentially as a function of distance along the chain. Furthermore, one can show that as the channel becomes more reliable (σ decreases), the dominance of local information over information further away becomes stronger, i.e., Y_i has less effect on the posterior probability of S_j, on average.

The exponential decay of information during message propagation suggests that there may exist graphs with loops where the information being propagated by a message-passing algorithm (using the completely parallel, or concurrent, version of the algorithm) can effectively "die out" before causing the algorithm to over-count. Of course, as we have seen in this paper, there is a non-zero probability of loops of length $c \geq 4$ for realistic turbo graphs, so that this line of argument is insufficient on its own to explain the apparent accuracy of iterative decoding algorithms.

It is also of interest to note that that iterative decoding has been empirically observed to converge to stable bit decisions within 10 iterations or so. As shown experimentally in [10], even beyond 10 iterations of message-passing there are still a small fraction of nodes which typically change bit decisions. Combined with the results on loop length distributions in this paper, this would suggest that it is certainly possible that over-counting is occurring at such nodes. It may be possible to show, however, that any such over-counting has relatively minimal effect on the overall quality of the posterior bit decisions.

VI. Conclusions

The distributions of loop lengths in the graphical models for turbo decoding were analyzed and simulated. Short loops (e.g., of length $c \leq 8$) occur with relatively low probability at any randomly chosen node. As the loop length increases, there is a threshold effect and the probability of finding a loop of length c or less approaches 1 (e.g., for $c > 20$). For turbo codes, as K, the number of information bits, becomes large, the probability that a loop of length c or less exists at any randomly chosen node behaves approximately as $e^{-\frac{2^{c-1}-4}{K}}, c \geq 4$. In summary, the results in this paper demonstrate that the loop lengths in the graphical models of turbo codes and LDPC codes have a specific distributional character. We hope that this information can be used to further understand the workings of iterative decoding.

Acknowledgments

The authors are grateful to R. J. McEliece and the coding group at Caltech for many useful discussions and feedback.

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo codes," in *Proceedings of the IEEE International Conference on Communications*, 1993, pp. 1064–1070.

[2] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann Publishers, Inc., San Mateo, CA., 1988.

[3] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, “Turbo decoding as an instance of Pearl’s ‘belief propagation’ algorithm.,” *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 140–152, Feb 1998.

[4] F. R. Kschischang and B. J. Frey, “Iterative decoding of compound codes by probability propagation in graphical models.,” *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 219–230, 1998.

[5] R. J. McEliece, E. R. Rodemich, and J.-F. Cheng, “The turbo-decision algorithm,” in *Proceedings of 33rd Annual Allerton Conference on Communication, Control, and Computing*, Oct. 4-6, 1995, pp. 366–379.

[6] S. Dolinar and D. Divsalar, “Weight distributions for turbo codes using random and nonrandom permutations,” Tech. Rep., Jet Propulsion Laboratory, Pasadena, CA, Aug 1995.

[7] X. Ge, D. Eppstein, and P. Smyth, “The distribution of cycle lengths in graphical models for iterative decoding,” Technical Report UCI-ICS 99-10, `http://www.datalab.uci.edu/papers/trcycle.pdf`, Mar 1999.

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

[9] D. J. C. MacKay and R. M. Neal, “Near Shannon limit performance of low density parity check codes,” *Electronics Letters*, vol. 32, no. 18, pp. 1645–1646, August 1996, Reprinted *Electronics Letters*, vol 33, no 6, 13th March 1997, p.457–458.

[10] B. J. Frey, *Graphical Models for Machine Learning and Digital Communication*, MIT Press, Cambridge, MA, 1998.