View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by DigitalCommons@University of Nebraska

University of Nebraska - Lincoln
DigitalCommons@University of Nebraska - Lincoln

Faculty Publications, Department of Mathematics | Mathematics, Department of

2009

Connections Between Computation Trees and Graph Covers

Deanna Dreher
University of Nebraska - Lincoln

Judy L. Walker
University of Nebraska - Lincoln, judy.walker@unl.edu

Follow this and additional works at: https://digitalcommons.unl.edu/mathfacpub

 Part of the Applied Mathematics Commons, and the Mathematics Commons

Dreher, Deanna and Walker, Judy L., "Connections Between Computation Trees and Graph Covers" (2009). *Faculty Publications, Department of Mathematics*. 170.
https://digitalcommons.unl.edu/mathfacpub/170

This Article is brought to you for free and open access by the Mathematics, Department of at DigitalCommons@University of Nebraska - Lincoln. It has been accepted for inclusion in Faculty Publications, Department of Mathematics by an authorized administrator of DigitalCommons@University of Nebraska - Lincoln.

Connections Between Computation Trees and Graph Covers

Deanna Dreher and Judy L. Walker
Department of Mathematics
University of Nebraska–Lincoln
Lincoln, NE 68588, USA.
Email: {s-dturk1, jwalker}@math.unl.edu

***Abstract*—Connections between graph cover pseudocodewords and computation tree pseudocodewords are investigated with the aim of bridging the gap between the theoretically attractive analysis of graph covers and the more intractable analysis of iterative message-passing algorithms that are intuitively linked to graph covers. Both theoretical results and numerous examples are presented.**

I. Introduction

Low density parity-check codes, along with their iterative message-passing decoders such as min-sum and sum-product, have been shown to achieve good bit error rates and perform close to capacity on channels of practical interest. While these iterative message-passing decoders are optimal on trees, iterative message-passing decoders are, in general, computationally efficient sub-optimal decoders for low density parity-check codes, and their efficiency makes them ideal for implementation. On the other hand, there is still little theoretical understanding of the effectiveness of these codes. One notable contribution in this direction is that of density evolution, which examines ensembles of codes [8], [9]. However, density evolution does not explain the non-codeword errors that arise in iterative message-passing decoding of a particular code.

The iterative message-passing decoders work, roughly speaking, as follows: initially, probabilities are assigned to each variable node based on the received word. The variable nodes send their information to their neighboring check nodes, which make calculations based on the information they receive and send some information back to their neighboring variable nodes. The variable nodes make some calculations with the information coming in from their neighboring check nodes and their original information, and send another message to their neighboring check nodes; this process continues until a stopping criteria is reached.

In particular, the decoders work locally: at each step of the algorithm, all that is needed for a particular vertex to make its calculation is data stored at that vertex and at its immediate neighbors. Intuitively, this leads one to consider *finite covers* of the Tanner graph, which look locally identical to the original Tanner graph but may be significantly larger.

Based on this intuition, many authors (see, e.g., [5], [6], [10]) have studied *graph cover pseudocodewords*, developing both a rich general theory and examining several specific examples in great detail. For example, zeta functions have been used to cleanly characterize pseudocodewords of cycle codes, and the fundamental cone gives a compact description of pseudocodewords for general codes. Additionally, there has been significant progress in characterizing problematic pseudocodewords for certain families of codes, such as codes constructed using finite geometries [7].

On the other hand, one thing we know for sure about both the min-sum and sum-product algorithms is that their behavior on Tanner graphs is precisely modeled by their behavior on computation trees. More precisely, Wiberg [11] showed that the output of the min-sum algorithm after m iterations is the vector whose i^{th} entry is the value assigned to the root node by a minimal cost configuration on a computation tree of depth m rooted at the i^{th} variable node.

The downside of Wiberg's characterization of these iterative message-passing algorithms is that computation trees are extremely difficult to study. Their size grows exponentially with each iteration, and the number of configurations on a tree is typically exponential with the size of the tree.

Thus, we seek a way to connect the well-developed theory of graph covers and graph cover pseudocodewords to the much less well-developed, but more precisely related to decoding, theory of computation trees. One approach to this problem is via the *universal cover* of the Tanner graph. The universal cover is simultaneously the infinite computation tree of the Tanner graph (for any root node) and a cover of the Tanner graph that also covers every finite connected cover of the Tanner graph. Thus, if one can fully understand configurations on the universal cover and devise a decoder on the universal cover that simultaneously extends both graph cover decoding [10] and min-sum decoding, then one can possibly use the theory of graph covers and graph cover pseudocodewords to better understand computation trees and computation tree pseudocodewords, and, ultimately, the behavior of min-sum decoding. This is precisely what is attempted by the authors and their collaborators in [2]–[4].

In this paper, we take a different approach. Rather than lift computation trees and graph covers up to the universal cover and then come back down, we aim to relate computation trees and graph covers more directly. The remainder of this section introduces some definitions and relevant background. In Section II we give theoretical results on the relationship between the set of graph cover configurations and the set

Published in 2009 Information Theory and Applications Workshop (2009)
DOI: 10.1109/ITA.2009.5044971

of computation tree configurations, and we present examples illustrating these results in Section III.

We now formalize some of the above discussion with the following definitions.

Definition 1.1: An *unramified cover*, or simply a *cover*, of a finite graph G is a graph $\widetilde{G}$ along with a surjective graph homomorphism $\pi : \widetilde{G} \to G$, called a *covering map*, such that for each vertex v of T and each $\tilde{v} \in \pi^{-1}(v)$, the neighborhood of $\tilde{v}$ is mapped bijectively to the neighborhood of v. For a positive integer M, an *M-cover* of G is cover $\pi : \widetilde{G} \to G$ such that for each vertex v of G, $\pi^{-1}(v)$ contains exactly M vertices of $\widetilde{G}$. If $\widetilde{G}$ is an M-cover of G, we say the *degree* of $\widetilde{G}$ is M.

We say that a graph G is *connected* if, for any two vertices u, v of G, there is a path $u = v_0, v_1, \ldots, v_k = v$ from u to v in G. In the remainder of this paper, we will assume that the Tanner graphs for our codes are connected.

Definition 1.2: Let $G = (X \cup F, E)$ be a bipartite graph. A *configuration* on G is a an assignment $\boldsymbol{c} = (c_x)_{x \in X}$ of 0's and 1's to the vertices in X such that for each $f \in F$, an even number of the neighbors of f are assigned a 1 and the rest are assigned a 0.

Given a Tanner graph T with variable nodes $x_1, \ldots, x_n$ and an M-cover $\pi : \widetilde{T} \to T$ of T, we label the elements of $\pi^{-1}(x_i)$ as $x_{i,1}, \ldots, x_{i,M}$.

Definition 1.3: Let T be a Tanner graph for a binary linear code C and let $\widetilde{\boldsymbol{c}} = (c_{1,1}, \ldots, c_{1,M} : \cdots : c_{n,1}, \ldots, c_{n,M})$ be a configuration on some M-cover $\widetilde{T}$ of T. Two kinds of *graph cover pseudocodewords* are associated to $\widetilde{\boldsymbol{c}}$: The *unscaled* graph cover pseudocodeword corresponding to $\widetilde{\boldsymbol{c}}$ is the vector

$$\boldsymbol{p}(\widetilde{\boldsymbol{c}}) = (p_1, \ldots, p_n)$$

of nonnegative integers, where, for $1 \le i \le n$,

$$p_i = \#\{j \,|\, c_{i,j} = 1\}.$$

The *normalized* graph cover pseudocodeword corresponding to $\widetilde{\boldsymbol{c}}$ is the vector

$$\boldsymbol{\omega}(\widetilde{\boldsymbol{c}}) = \frac{1}{M}\boldsymbol{p}(\widetilde{\boldsymbol{c}}).$$

Definition 1.4 (Wiberg [11]): Let T be a Tanner graph, and assume an iterative message-passing algorithm has been run on T for a total of m iterations, where a single iteration consists of message-passing from the variable nodes to the check nodes and then back to the variable nodes. The *depth m computation tree* for T with *root node* v is the tree R obtained by tracing the computation of the final cost function of the algorithm at the variable node v of T recursively back through time, along with a surjective graph homomorphism $\pi : R \to T$, such that for each vertex v of T and each $\widetilde{v} \in \pi^{-1}(v)$ that is not on level m, the neighborhood of $\widetilde{v}$ is mapped bijectively to the neighborhood of v.

Since computation trees are necessarily connected, any computation tree configuration that is induced by a graph cover configuration is induced by a connected graph cover configuration. Thus, it is important to determine which graph cover pseudocodewords have connected realizations.

The authors and their collaborators have shown [1] that with a very basic restriction on our Tanner graph, every normalized graph cover pseudocodeword has a connected realization. The restriction on the Tanner graph is given by

$$a_c\left(1 - \frac{1}{a_v}\right) \ge 2,$$

where a_c and a_v are the average check node degree and average variable node degree, respectively. Thus, for the majority of practical codes and a significant portion of cycle codes, every normalized graph cover pseudocodeword has a connected realization, and so every normalized graph cover pseudocodeword induces computation tree configurations. The inverse question of which computation tree configurations are induced by graph cover configurations is investigated in the next section.

II. Realizations of computation tree pseudocodewords

A different way to ask how computation tree configurations and graph cover pseudocodewords relate is to ask whether, given a computation tree configuration, there is a graph cover configuration that induces it. As a first step in this direction, we show that every computation tree is contained in a finite cover of the Tanner graph.

Proposition 2.1: Let T be a Tanner graph and let (R, π_R) be a computation tree for T. Then there exists a finite cover $(\widetilde{T}, \pi_{\widetilde{T}})$ that contains a subgraph isomorphic to (R, π_R). More precisely, let T' be any spanning tree of T and let M be the number of connected components of $\pi_R^{-1}(T')$ in R. Then there is an M-cover $(\widetilde{T}, \pi_{\widetilde{T}})$ of T that contains a subgraph S such that there is a graph isomorphism $\phi : S \to R$ satisfying $\pi_R \circ \phi = \pi_{\widetilde{T}}|_S$.

Proof: Note that the final condition of the proposition says that $\widetilde{T}$ contains a subgraph S that is isomorphic to R in a way that respects the labels on the nodes, i.e. if x is a node in R that is a copy of the vertex v of T, then the corresponding node in S is also a copy of v in $\widetilde{T}$.

Let T' be a spanning tree of T and let M be the number of connected components of $\pi_R^{-1}(T')$ in R. Since there are at most M copies of each vertex v of T in R, we may first form a forest R^* by taking the disjoint union of R with sufficiently many appropriately labeled isolated vertices so that there are exactly M copies of each vertex v of T in R^*. We now add edges to R^* as follows: For each edge $e = xu$ of T, there are M copies of x in R^*, M copies of u in R^*, and $m_e := |\pi_R^{-1}(e)|$ copies of e in R^*. Thus there are exactly $M - m_e$ copies of x in R^* that are not adjacent to any copy of u in R^*, and there are exactly $M - m_e$ copies of u in R^* that are not adjacent to any copy of x in R^*. This means that there is a matching between these copies of x and of u, and we form an edge between each matched pair. Repeating this procedure for each edge e of T creates an M-cover $\widetilde{T}$ of T that contains R as a subgraph so that the respective vertex labels (projection maps) agree. ∎

Proposition 2.1 shows that every computation tree is contained in a graph cover, but what we really want to know is

whether every computation tree configuration is induced by a graph cover configuration. In other words, given a computation tree configuration, is there a graph cover configuration that is, in this sense, compatible with it? To answer this question, we need a definition.

Definition 2.2: Let $S = (X \cup F, E)$ be a bipartite graph and let $(\boldsymbol{c}, S)$ be a configuration on S. A *configured subgraph* of $(\boldsymbol{c}, S)$ is a configuration $(\boldsymbol{c}', S')$, where S' is a subgraph of S, and $\boldsymbol{c}'$ is the restriction of $\boldsymbol{c}$ to S'.

Let T be a Tanner graph. For $i = 1, 2$, let (S_i, π_i) be a bipartite graph along with a bijective graph homomorphism $\pi_i : S_i = (X_i \cup F_i, E_i) \to T$, and let $(\boldsymbol{c}_i, S_i)$ be a configuration on S_i. Then $(\boldsymbol{c}_1, S_1)$ and $(\boldsymbol{c}_2, S_2)$ are *isomorphic* if there is a graph isomorphism $\phi : S_1 \to S_2$ such that $\pi_2 \circ \phi = \pi_1$ and $\boldsymbol{c}_1$ assigns a value of 1 to $x \in X_1$ if and only if $\boldsymbol{c}_2$ assigns a value of 1 to $\phi(x) \in X_2$.

The following results show that the connection between computation tree configurations and graph cover configurations is extremely strong: given a computation tree R for a Tanner graph T and a configuration $\boldsymbol{c}$ on R, there is a graph cover configuration $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ that not only induces $(\boldsymbol{c}, R)$, but also contains a configured subgraph isomorphic to $(\boldsymbol{c}, R)$. In the case of cycle codes, the result is very clean:

Theorem 2.3: Let T be the Tanner graph of a cycle code with minimum degree at least two, let R be a computation tree for T, and let $\boldsymbol{c}$ be a configuration on R. Then there is a finite cover $\widetilde{T}$ of T and a configuration $\widetilde{\boldsymbol{c}}$ on $\widetilde{T}$ such that $(\boldsymbol{c}, R)$ is isomorphic to a configured subgraph of $(\widetilde{\boldsymbol{c}}, \widetilde{T})$. Furthermore, $\widetilde{T}$ can be taken to be a cover of degree either M or $2M$, where M is as in Proposition 2.1.

The proof of this theorem goes, roughly speaking, as follows: Let $\widetilde{T}$ be an M-cover of T that contains a subgraph isomorphic to R, as guaranteed by Proposition 2.1, and then copy the configuration $\boldsymbol{c}$ onto that subgraph. One then proves that $\boldsymbol{c}$ can be extended to a configuration on all of either $\widetilde{T}$ or $\widetilde{T}'$, where $\widetilde{T}'$ is a $2M$-cover of T formed by twisting together two copies of $\widetilde{T}$ in a manner determined by $\boldsymbol{c}$.

In the general case, the result is not as clean. In particular, the degree of the cover is not so simply described. Theorem 2.4 below gives an upper bound on the necessary degree of a cover that has a configuration that induces the computation tree configuration we started with, but that cover need not contain the computation tree as a configured subgraph. Corollary 2.5 asserts that there is a cover that contains the computation tree as a configured subgraph, but does not give us information on the degree of that cover. As a curious side-result, it follows from the proof of Theorem 2.4 that the rational point $(\frac{1}{2}, \ldots, \frac{1}{2})$ is a normalized graph cover pseudocodeword for any parity-check matrix with minimum row degree at least two.

Theorem 2.4: Let H be a parity-check matrix with minimum row weight at least two, let $T = T(H) = (X \cup F, E)$ be its Tanner graph, and let R be a computation tree for T of some finite depth d. For each $f \in F$, let M_f be the number of copies of f on R. Set $M = \max_f M_f$.

Then, for any configuration $\boldsymbol{c}$ on R, there is a $4M$-cover $\widetilde{T}$ of T and a configuration $\widetilde{\boldsymbol{c}}$ on $\widetilde{T}$ such that

- $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ induces the configuration $(\boldsymbol{c}, R)$,
- $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ contains a configured subgraph that is isomorphic to $(\boldsymbol{c}', R')$, where $(\boldsymbol{c}', R')$ is the configured computation tree subgraph of $(\boldsymbol{c}, R)$ of depth $d - 1$, and
- the normalized graph cover pseudocodeword corresponding to $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ is $(\frac{1}{2}, \ldots, \frac{1}{2})$.

Although the $4M$-cover $\widetilde{T}$ ensured by Theorem 2.4 need not be connected, the configured subgraph of it isomorphic to $(\boldsymbol{c}', R')$ is contained in some connected component $\widetilde{T}'$ of $\widetilde{T}$. The first two bullets in the theorem clearly still apply to $(\widetilde{\boldsymbol{c}}', \widetilde{T}')$, where $\widetilde{\boldsymbol{c}}'$ is the configuration on $\widetilde{T}'$ ensured by the theorem. However, if $\widetilde{T}' \neq \widetilde{T}$, then the third bullet will almost certainly not be valid for $\widetilde{\boldsymbol{c}}'$; see Example 3.3. Thus, although the theorem implies that every computation tree configuration is related to the graph cover pseudocodeword $(\frac{1}{2}, \ldots, \frac{1}{2})$, the corresponding realization $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ of this graph cover pseudocodeword need not be connected. Moreover, it is the graph cover pseudocodeword corresponding to the connected graph cover configuration $(\widetilde{\boldsymbol{c}}', \widetilde{T}')$ that has any hope of sharing properties under an appropriate cost function [10], [11] with $(\boldsymbol{c}, R)$.

As a corollary to Theorem 2.4, we obtain a version of Theorem 2.3 for the general case.

Corollary 2.5: Let H be a parity-check matrix with minimum row weight at least two, let $T = T(H)$ be its Tanner graph, and let $(\boldsymbol{c}, R)$ be a computation tree configuration for T. Then there exists a finite cover $\widetilde{T}$ of T and a configuration $\widetilde{\boldsymbol{c}}$ on $\widetilde{T}$ such that $(\boldsymbol{c}, R)$ is isomorphic to a configured subgraph of $(\widetilde{\boldsymbol{c}}, \widetilde{T})$.

Proof: Let R'' be the computation tree for T obtained by extending R for an additional iteration. Then there is a configuration $\boldsymbol{c}''$ on R'' that restricts to $(\boldsymbol{c}, R)$. Applying Theorem 2.4 to $(\boldsymbol{c}'', R'')$ gives the result. ∎

As we will see in Section III below, there are often graph cover configurations $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ that satisfy the conclusions of Theorem 2.4 but with the degree of $\widetilde{T}$ significantly smaller than $4M$. By taking into account the specific computation tree configuration under consideration, a tighter bound can be obtained:

Theorem 2.6: Let H be a parity-check matrix with minimum row weight at least two, let $T = T(H) = (X \cup F, E)$ be its Tanner graph, and let $(\boldsymbol{c}, R)$ be a computation tree configuration for T of some finite depth d. For each $f \in F$, let $M_f(\boldsymbol{c})$ be the number of copies of f on R such that at least one adjacent vertex (which is necessarily a copy of some $x \in X$) is assigned a "1" by $\boldsymbol{c}$, and let $N_f(\boldsymbol{c})$ be the number of copies of f on R such that every adjacent vertex is assigned a "0" by $\boldsymbol{c}$. Set $M(\boldsymbol{c}) = \max_f M_f(\boldsymbol{c})$ and $N(\boldsymbol{c}) = \max_f N_f(\boldsymbol{c})$.

Then there is a $(3M(\boldsymbol{c}) + N(\boldsymbol{c}))$-cover $\widetilde{T}$ of T and a configuration $\widetilde{\boldsymbol{c}}$ on $\widetilde{T}$ such that

- $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ induces the configuration $(\boldsymbol{c}, R)$,
- $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ contains a configured subgraph that is isomorphic to $(\boldsymbol{c}', R')$, where $(\boldsymbol{c}', R')$ is the configured computation tree subgraph of $(\boldsymbol{c}, R)$ of depth $d - 1$, and

- the normalized graph cover pseudocodeword corresponding to $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ is $(\frac{2M(\boldsymbol{c})}{3M(\boldsymbol{c})+N(\boldsymbol{c})}, \ldots, \frac{2M(\boldsymbol{c})}{3M(\boldsymbol{c})+N(\boldsymbol{c})})$.

Theorem 2.6 again has this curious side-result about the fundamental polytope containing a certain constant vector of rational numbers. One might ask how far that result can be pushed. The next proposition gives the answer.

Proposition 2.7: Let H be a parity-check matrix with minimum row weight at least two, let $T = (X \cup F, E)$ be the corresponding Tanner graph and let c be a nonnegative real number. If at least one $f \in F$ has odd degree, set $\delta = \frac{M-1}{M}$, where M is the smallest odd integer such that some $f \in F$ has degree M. Otherwise, set $\delta = 1$. Then $(c, \ldots, c)$ is a normalized graph cover pseudocodeword if and only if $c \leq \delta$.

III. Examples

In this section, we present a series of examples that illustrate the results of Section II.

Our first example shows that, with M as in Theorem 2.4, the minimal degree of a graph cover configuration satisfying the conditions in Theorem 2.4 may be smaller than $4M$.

Example 3.1: Figure 1 shows a Tanner graph T and a computation tree configuration $(\boldsymbol{c}, R)$ for T.

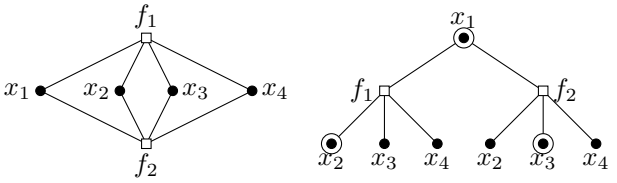

Fig. 1. The Tanner graph T and a computation tree configuration $(\boldsymbol{c}, R)$ for the Tanner graph T of Example 3.1. Vertices are assigned a value of 1 by $\boldsymbol{c}$ if they are circled and 0 otherwise.

Since every check node of T appears in R once, we have $M = 1$ in Theorem 2.4. Further, the computation tree configuration $(\boldsymbol{c}', R')$ of that theorem is simply the single vertex x_1, assigned a value of 1. Therefore, we know by Theorem 2.4 that there is a 4-cover $\widetilde{T}$ of T and a configuration $\widetilde{\boldsymbol{c}}$ on $\widetilde{T}$ such that $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ induces $(\boldsymbol{c}, R)$, assigns a value of 1 to some copy of x_1 in $\widetilde{T}$, and has corresponding normalized graph cover pseudocodeword $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})$.

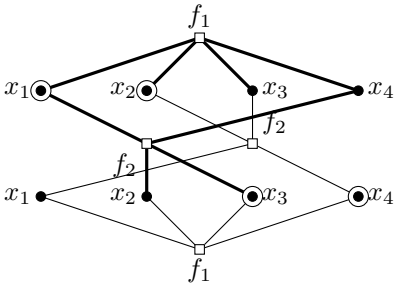

Fig. 2. A configuration $\widetilde{\mathbf{c}}$ on a 2-cover $\widetilde{T}$ of the Tanner graph T of Example 3.1.

Figure 2 shows a 2-cover $\widetilde{T}$ of T and a configuration $\widetilde{\boldsymbol{c}}$ on $\widetilde{T}$. The bold edges show a configured subgraph of $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ that induces $(\boldsymbol{c}, R)$ and contains a subgraph that is isomorphic to the depth one truncation $(\boldsymbol{c}', R')$ of $(\boldsymbol{c}, R)$, and such that the graph cover pseudocodeword corresponding to $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ is $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})$. Thus $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ satisfies all of the conclusions of Theorem 2.4, except that we needed only a 2-cover rather than a 4-cover to do so. Of course, two disjoint copies of $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ is a 4-cover of T that satisfies the theorem as well, thus showing that the $4M$-cover guaranteed by Theorem 2.4 need not be connected. On the other hand, Figure 3 gives an example of a connected 4-cover of T that contains the entire configured computation tree $(\boldsymbol{c}, R)$ of Figure 1. □

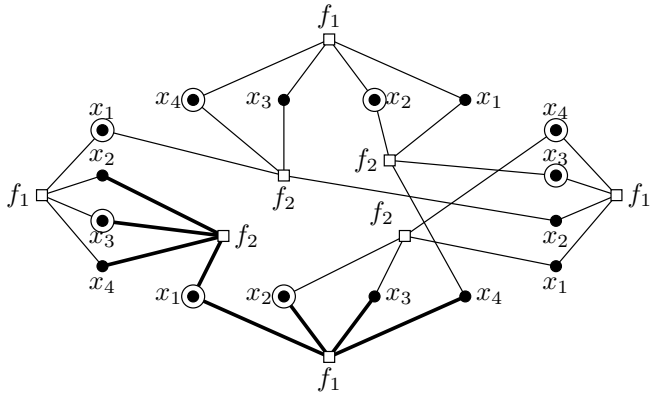

Fig. 3. A configuration $\widetilde{\boldsymbol{c}}$ on a 4-cover $\widetilde{T}$ of the Tanner graph T of Example 3.1.

The next example illustrates that sometimes only the smaller computation tree configuration $(\boldsymbol{c}', R')$ can be obtained.

Example 3.2: Consider the Tanner graph T in Figure 4 and the computation tree R for T shown in Figure 5.

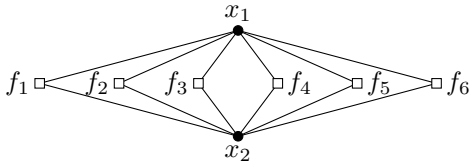

Fig. 4. The Tanner graph T for Example 3.2.

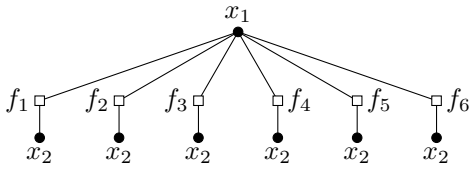

Fig. 5. A computation tree R for the Tanner graph T of Example 3.2.

In this example, we again have $M = 1$ in Theorem 2.4. Thus Theorem 2.4 asserts the existence of a 4-cover $\widetilde{T}$ of T with certain properties. However, since there are six copies of x_2 in R, no computation tree configuration $(\boldsymbol{c}, R)$ could be isomorphic to a configured subgraph of $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ for any configuration $\widetilde{\boldsymbol{c}}$ on $\widetilde{T}$. □

As mentioned previously, if $(\boldsymbol{c}, R)$ is a computation tree configuration, then the realization $(\widetilde{\boldsymbol{c}}, \widetilde{T})$ of $(\frac{1}{2}, \ldots, \frac{1}{2})$ that induces $(\boldsymbol{c}, R)$ in Theorem 2.4 need not be connected. However, there is a connected component of $\widetilde{T}$ that also induces $(\boldsymbol{c}, R)$, since R is connected; call this connected component $(\widetilde{\boldsymbol{c}}', \widetilde{T}')$, where $\widetilde{\boldsymbol{c}}'$ is the configuration $\widetilde{T}'$ inherits from $\widetilde{T}$. Then $(\widetilde{\boldsymbol{c}}', \widetilde{T}')$ is connected and induces $(\boldsymbol{c}, R)$, but its normalized graph cover pseudocodeword may no longer be $(\frac{1}{2}, \ldots, \frac{1}{2})$. Example 3.3 shows an occurrence of this.

Example 3.3: Figure 6 shows a Tanner graph T and a computation tree configuration $(\boldsymbol{c}, R)$ for T.

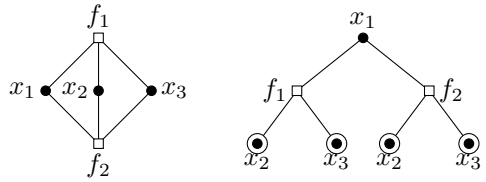

Fig. 6. The Tanner graph T and a computation tree configuration $(\boldsymbol{c}, R)$ for the Tanner graph T of Example 3.3.

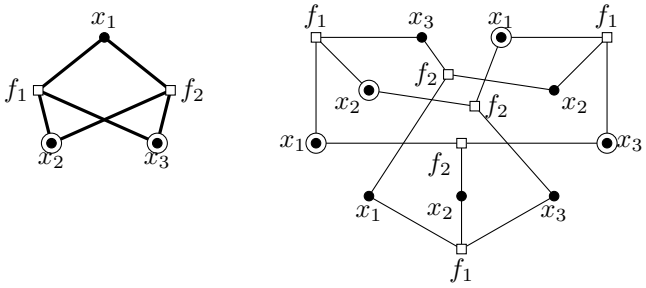

Fig. 7. A disconnected 4-cover $\widetilde{T}$ and a configuration $\widetilde{\boldsymbol{c}}$ on $\widetilde{T}$ that induces the configuration in Figure 6.

The configuration $\widetilde{\boldsymbol{c}}$ on the 4-cover $\widetilde{T}$ of T in Figure 7 satisfies all of the requirements of Theorem 2.4. However, the configuration on the component on the left, which is isomorphic to the original graph T, induces $(\boldsymbol{c}, R)$ and has normalized graph cover pseudocodeword $(0, 1, 1)$. □

Our final example concerns the relationship between Theorem 2.4 and Theorem 2.6. It was asserted in Section II that, given a specific computation tree configuration $(\boldsymbol{c}, R)$ for the Tanner graph T, Theorem 2.6 gives a better upper bound on the degree of the cover $\widetilde{T}$ of T needed to admit a configuration $\widetilde{\boldsymbol{c}}$ that induces $(\boldsymbol{c}, R)$ and that contains a configured subgraph isomorphic to $(\boldsymbol{c}', R')$, the configuration obtained by restricting $\boldsymbol{c}$ to the computation tree R' of depth one less than the depth of R. To see this, recall that Theorem 2.4 guarantees the existence of a $4M$-cover that admits the desired $\widetilde{\boldsymbol{c}}$, where M is the maximum number of copies of any $f \in F$ in R. On the other hand, Theorem 2.6 guarantees the existence of a $(3M(\boldsymbol{c}) + N(\boldsymbol{c}))$-cover that does the trick, where $M(\boldsymbol{c}) = \max_{f \in F} M_f(\boldsymbol{c})$, $N(\boldsymbol{c}) = \max_{f \in F} N_f(\boldsymbol{c})$ and, for any $f \in F$, we have that $M_f(\boldsymbol{c}) + N_f(\boldsymbol{c})$ is the total number of copies of f in R. Thus $M(\boldsymbol{c}) \leq M$ and $N(\boldsymbol{c}) \leq M$, and so $3M(\boldsymbol{c}) + N(\boldsymbol{c}) \leq 4M$. In fact, the bound in Theorem 2.6 can yield a significant improvement over that of Theorem 2.4. Our final example illustrates this.

Example 3.4: Let T be the Tanner graph given in Figure 8, and let $(\boldsymbol{c}, R)$ be the computation tree configuration for T given in Figure 9.

Since there are three copies of f_5 in the computation tree R of Figure 9, Theorem 2.4 asserts the existence of a 12-cover that induces the configuration $(\boldsymbol{c}, R)$ of Figure 9. However,

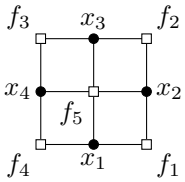

Fig. 8. The Tanner graph T of Example 3.4.

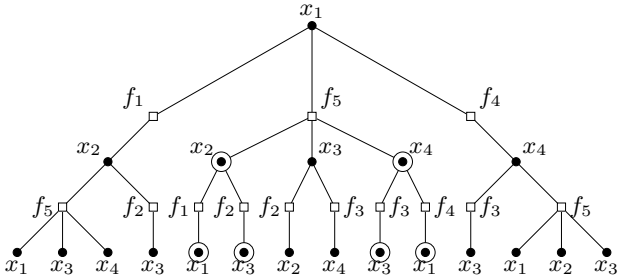

Fig. 9. A computation tree configuration $(\boldsymbol{c}, R)$ for the Tanner graph of Example 3.4.

since $M(\boldsymbol{c}) = 1$ and $N(\boldsymbol{c}) = 2$, Theorem 2.6 tells us that there is a graph cover configuration on a 5-cover that does the trick. □

ACKNOWLEDGEMENTS

This work was supported in part by National Science Foundation 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, K. Morrison, E. Psota, D. Turk, L. C. Pérez, and J .L. Walker, *Universal cover decoding*, In preparation, 2008.

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

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

[6] R Koetter, W.-C. W. Li, P. O. Vontobel, and J. L. Walker, *Pseudocodewords of cycle codes via zeta functions*, Proc. IEEE Inform. Theory Workshop (San Antonio, TX, USA), 2004, pp. 7–12.

[7] P. O. Vontobel R. Smarandache, *Pseudo-codeword analysis of Tanner graphs from projective and euclidean planes*, IEEE Transactions on Information Theory **IT-53** (2007), no. 7, 2376–2393.

[8] T. Richardson, A. Shokrollahi, and R. Urbanke, *Design of capacity-approaching irregular low-density parity check codes*, IEEE Transactions on Information Theory **47** (2001), no. 2, 619–637.

[9] T. Richardson and R. Urbanke, *The capacity of low-density parity check codes under message-passing decoding*, IEEE Transactions on Information Theory **47** (2001), no. 2, 599–618.

[10] 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*.

[11] N. Wiberg, *Codes and decoding on general graphs*, Ph.D. thesis, Linköping University, Linköping, Sweden, 1996.