

## Structure of Pseudocodewords in Tanner Graphs

Christine Kelley<sup>†</sup> and Deepak Sridhara<sup>‡</sup>

<sup>†</sup> Department of Mathematics  
University of Notre Dame  
Notre Dame, IN 46556, U.S.A.  
email: [ckelley1@nd.edu](mailto:ckelley1@nd.edu)

<sup>‡</sup> Department of Mathematics  
Indian Institute of Science  
Bangalore 560012, India.  
email: [dsridhar@math.iisc.ernet.in](mailto:dsridhar@math.iisc.ernet.in)

### Abstract

We examine the structure of pseudocodewords in Tanner graphs and identify certain types of pseudocodewords that can be problematic with iterative decoding.

### 1. INTRODUCTION

Iterative decoders have gained widespread attention due to their remarkable performance in decoding LDPC codes. Analyzing their behaviour on finite-length LDPC codes has nevertheless remained a formidable task. Wiberg's [1] was among the earliest works in characterizing iterative decoder convergence on finite-length LDPC constraint graphs (or Tanner graphs). Both [1] and [2] examine the convergence behaviour of the min-sum iterative decoder on cycle codes, a special class of LDPC codes having only degree two variable nodes, and based on the structure of LDPC Tanner graphs, they provide some necessary conditions when the decoder fails to converge to a valid codeword.

Analogous works in [3] and [4] explain the behavior of iterative decoders using the terminology of covering graphs or lifts of the base Tanner graph. The common underlying idea is the role of *pseudocodewords* in determining decoder convergence. Pseudocodewords can be viewed as valid codeword configurations existing among all finite lifts of the base graph.

Pseudocodewords of a Tanner graph play analogous roles in determining convergence of an iterative decoder as *codewords* do for a maximum-likelihood (ML) decoder. In this paper, we study the structure of pseudocodewords of a Tanner graph assuming *min-sum* iterative decoding as in [1], and classify the different types of pseudocodewords that can arise depending on the graph structure. The main result is that a non-

codeword (nc) pseudocodeword that is *irreducible* can cause the min-sum decoder to fail to converge to a valid codeword. We also examine the minimal degree lifts needed to realize a pseudocodeword since this can potentially eliminate the need of examining all finite degree lifts of the base Tanner graph.

Definitions are introduced in Section 2. Section 3 analyzes the structure of pseudocodewords of general Tanner graphs, wherein the main results of the paper are presented. Some of the results are extensions of those in [1] and [2]. Section 4 presents some examples to illustrate the results derived in the paper.

### 2. PRELIMINARIES

Let  $G = (V, U; E)$  be a bipartite graph representing a binary LDPC code  $C$  with minimum distance  $d_{\min}$ , with  $|V| = n$  left (variable) nodes,  $|U| = m$  right (check) nodes, and edges  $E \subseteq \{(v, u) | v \in V, u \in U\}$ .

**Definition 2.1** The *support* of a vector  $\mathbf{x} = \{x_1, \dots, x_n\}$  is the set of indices  $i$  where  $x_i \neq 0$ .

**Definition 2.2** [6] A *stopping set* in  $G$  is a subset  $S$  of  $V$  where every neighbor of  $s \in S$  is connected to  $S$  at least twice. The size of the smallest stopping set in  $G$  is denoted by  $s_{\min}$ .

A degree  $\ell$  cover (lift)  $\hat{G}$  of  $G$  is defined in the following manner:

**Definition 2.3** A finite degree  $\ell$  cover of  $G = (V, U; E)$  is a bipartite graph  $\hat{G}$  where for each vertex  $x_i \in V \cup U$  there is a *cloud*  $\hat{X}_i = \{\hat{x}_{i_1}, \hat{x}_{i_2}, \dots, \hat{x}_{i_\ell}\}$  of vertices in  $\hat{G}$  with  $\deg(\hat{x}_{i_j}) = \deg(x_i)$  for all  $j \in [\ell]$ , and if  $(x_i, x_j) \in E$ , then there are  $\ell$  edges from  $\hat{X}_i$  to  $\hat{X}_j$  in  $\hat{G}$  connected in a 1 – 1 manner.

Figure 1 shows a base graph  $G$  and a degree four cover of  $G$ .

---

The first author was supported by a Center for Applied Mathematics fellowship at the University of Notre Dame, and the second author was supported by an Institute of Mathematics Initiative Research Associateship at the Indian Institute of Science and DRDO-India.

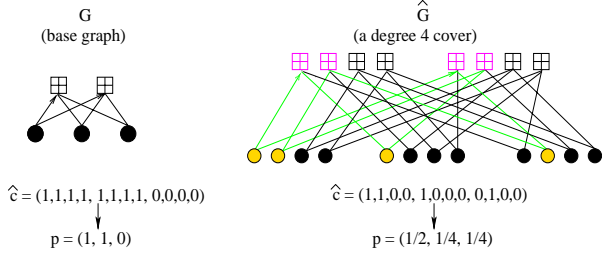


Figure 1: A pseudocodeword in the base graph (or a valid codeword in a lift).

**Definition 2.4** A *pseudocodeword*  $\mathbf{p}$  of  $G$  is any vector  $(p_1, p_2, \dots, p_n)$  that is obtained by reducing a valid codeword  $\hat{\mathbf{c}}$  of the code corresponding to a cover of  $G$  in the following way:

$$\hat{\mathbf{c}} = (\hat{c}_{1,1}, \dots, \hat{c}_{1,\ell}, \hat{c}_{2,1}, \dots, \hat{c}_{2,\ell}, \dots) \rightarrow (p_1, p_2, \dots, p_n) = \mathbf{p},$$

$$\text{where } p_i = \frac{\hat{c}_{i,1} + \hat{c}_{i,2} + \dots + \hat{c}_{i,\ell}}{\ell}.$$

Note that each component of the pseudocodeword is just the fraction of 1-valued variable nodes in the corresponding variable cloud. Observe that a codeword is trivially a pseudocodeword. Equivalently, a *pseudocodeword*  $\mathbf{p} = (p_1, p_2, \dots, p_n)$  is a vector of integer entries where  $p_i$  represents the number (rather than fraction) of variables nodes of value 1 in a lift  $\hat{G}$  that are lifts of the node  $v_i$  of the base graph  $G$ , where the values assigned to the variable nodes in the lift correspond to a valid codeword configuration [2], i.e.,

$$p_i = \hat{c}_{i,1} + \hat{c}_{i,2} + \dots + \hat{c}_{i,\ell}.$$

In this paper we use the latter definition.

An alternative formulation of pseudocodewords is in terms of the computation tree, as described in [1]. Let  $C(G)$  be the computation tree, corresponding to the *min-sum* iterative decoder, of the base LDPC constraint graph  $G$  [1]. (See Figure 2.) Let  $\hat{G}$  be a lift of  $G$ .

A *valid assignment* of variable nodes in a computation tree is one where all constraint nodes are satisfied. (Such an assignment is said to be *locally consistent* (LC).) We note here that a codeword  $\mathbf{c}$  corresponds to a valid assignment where for each  $i$ , all  $v_i$  nodes in computation tree are assigned the same value, whereas a pseudocodeword  $\mathbf{p}$  corresponds to a valid assignment where for each  $i$ , all  $v_i$  in the computation tree need not be assigned the same value.

We observe that the support of a pseudocodeword forms a stopping set in  $G$ .

**Lemma 2.1** *The support of a pseudocodeword  $\mathbf{p}$  of  $G$  is the incidence vector of a stopping set in  $G$ .*

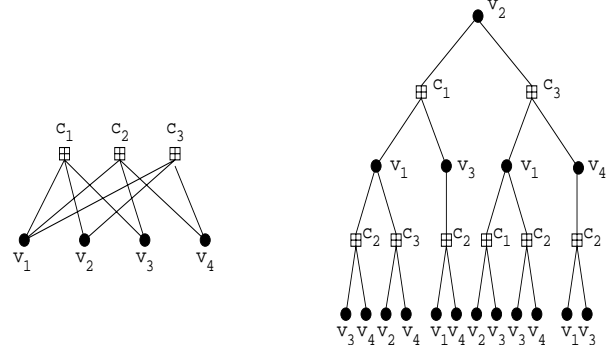


Figure 2: A graph and its computation tree after two iterations of message passing.

*Proof:* This can be seen by noting that every cloud of check nodes in the corresponding cover of  $G$  is connected to either none or at least two of the variable clouds in the support of  $\mathbf{p}$ . If this were not the case, then there would be a cloud of check nodes in the cover with at least one check node connected to exactly one variable node of bit value 1, leaving the check node constraint unsatisfied. Therefore, the corresponding variable nodes in the base graph  $G$  satisfy the condition of a stopping set.  $\square$

**Definition 2.5** A pseudocodeword  $\mathbf{p} = (p_1, \dots, p_n)$  is *irreducible* if it cannot be written as a sum of two or more codewords or pseudocodewords.

**Definition 2.6** The *pseudo-subgraph*, or *pseudograph*,  $G_S$  of a pseudocodeword  $\mathbf{p} = (p_1, \dots, p_n)$  with support  $S = \{i_1, \dots, i_k\}$  is the subgraph induced by  $\{v_{i_1}, \dots, v_{i_k}\}$  in  $G$ . The *pseudograph lift*  $\hat{G}_S$  is the subgraph of  $\hat{G}$  induced by the 1-valued variable nodes  $\{v_{i,j}, j = 1, \dots, m | v_i \in S, v_{i,j} = 1\}$ , where  $m$  is the degree of a lift graph  $\hat{G}$  that realizes<sup>1</sup>  $\mathbf{p}$ .

Note that  $\mathbf{p}$  may be realized in many lifts, and in addition, the subgraphs  $\hat{G}_S$  in these lifts may not be isomorphic.

**Definition 2.7** A *path* in a graph  $G$  is a sequence  $v_1 c_1 v_2 \dots$  of alternating variable and check nodes, not necessarily distinct, where  $v_i$  is a neighbor of  $c_{i-1}$  and  $c_i$  for all  $i$ . If the path starts and ends at the same point, it is said to be *closed*. A *walk* (respectively, *closed walk*) is a path (respectively, closed path) where all variable nodes are distinct.

The following definition characterizes the iterative decoder behavior, providing conditions when the *min-sum* decoder may fail to converge to a valid codeword.

<sup>1</sup>A lift-graph  $\hat{G}$  *realizes*  $\mathbf{p}$  if  $\mathbf{p}$  lifts to a valid codeword in  $\hat{G}$ .

**Definition 2.8** [2] A pseudocodeword  $\mathbf{p}=(p_1, p_2, \dots, p_n)$  is *good* if for all input weight vectors  $\mathbf{w} = (w_1, w_2, \dots, w_n)$  to the min-sum iterative decoder, there is a codeword  $\mathbf{c}$  that has lower overall cost than  $\mathbf{p}$ , i.e.,  $\mathbf{c}\mathbf{w}^T < \mathbf{p}\mathbf{w}^T$ .

Suppose the all-zeros codeword is the maximum-likelihood (ML) codeword for an input weight vector  $\mathbf{w}$ , then all non-zero codewords  $\mathbf{c}$  have a positive cost, i.e.,  $\mathbf{c}\mathbf{w}^T > 0$ . In this case, it is equivalent to say that pseudocodeword  $\mathbf{p}$  is *bad* if there is a weight vector  $\mathbf{w}$  such that for all non-zero codewords  $\mathbf{c}$ ,  $\mathbf{c}\mathbf{w}^T \geq 0$  and  $\mathbf{p}\mathbf{w}^T \leq 0$ .

The weight of a pseudocodeword is defined as follows [3]:

**Definition 2.9** Let  $\mathbf{p} = (p_1, p_2, \dots, p_n)$  be a pseudocodeword of the code  $C$  represented by the bipartite graph  $G$ . Then its *weight* is:

- $w_{BEC}(\mathbf{p}) = |\text{supp}(\mathbf{p})|$  for the binary erasure channel (BEC);
- $w_{BSC}(\mathbf{p})$  for the binary symmetric channel (BSC) is:
$$w_{BSC}(\mathbf{p}) = \begin{cases} 2e, & \text{if } \sum_e p_i = \frac{\sum_{i=1}^n p_i}{2} \\ 2e - 1, & \text{if } \sum_e p_i > \frac{\sum_{i=1}^n p_i}{2} \end{cases},$$

where  $e$  is the smallest number such that the sum of the  $e$  largest  $p_i$ 's is at least  $\frac{\sum_{i=1}^n p_i}{2}$ ;

- $w_{AWGN}(\mathbf{p}) = \frac{(p_1 + p_2 + \dots + p_n)^2}{(p_1^2 + p_2^2 + \dots + p_n^2)}$  for the additive white Gaussian noise channel (AWGN).

The *minimal* pseudocodeword weight of  $G$  is the minimum weight over all pseudocodewords that occur over all possible finite-degree covers of  $G$ , and is denoted by  $w_{\min}^{BEC}$  for the BEC channel (and likewise, for other channels). The minimal pseudocodeword weight  $w_{\min}$  is of fundamental importance as it plays an analogous role in iterative decoding as the minimum distance  $d_{\min}$  in ML-decoding.

The following result relates the weight of a good pseudocodeword to the minimum distance of the code.

**Theorem 2.1** [7] For a good pseudocodeword  $\mathbf{p}$  of  $G$ ,  $w^{BSC/AWGN}(\mathbf{p}) \geq d_{\min}$ .

Note, however, that the above result does not indicate anything about the weight of a bad pseudocodeword. A bad pseudocodeword can have a weight that is either less than or larger than  $d_{\min}$ . In the context of iterative decoding it is the low weight bad pseudocodewords that are likely to cause the decoder to fail to converge.

### 3. STRUCTURE OF PSEUDOCODEWORDS

As in the previous section, let  $G$  be a bipartite graph representing a binary LDPC code  $\mathcal{C}$ , with  $|V| = n$  left (variable) nodes,  $|U| = m$  right (check) nodes, and edges  $E \subseteq \{(v, u) | v \in V, u \in U\}$ .

#### 3.1. Iterative decoding

The following result is an extension of the result in [2] to more general LDPC graphs:

**Theorem 3.1** A sufficient condition for the iterative min-sum decoder to converge to the all-zeros maximum-likelihood codeword after sufficiently many iterations, is that the weight of the all-ones assignment on any subtree  $S$ , in the computation tree  $C(G)$ , corresponding to a pseudocodeword  $\mathbf{p}$  is positive.

Wiberg gives a different formulation of the same result in [1].

#### 3.2. Irreducible pseudocodewords

We will now state the following preliminary results that will be useful in proving our main result on the structure of good and bad pseudocodewords of a base graph  $G$ .

**Lemma 3.1** Let  $\mathbf{p} = (p_1, p_2, \dots, p_n)$  be a pseudocodeword in the graph  $G$  that represents the LDPC code  $\mathcal{C}$ . Then the vector  $\mathbf{x} = \mathbf{p} \bmod 2$ , obtained by reducing the entries in  $\mathbf{p}$ , modulo 2, corresponds to a codeword in  $\mathcal{C}$ .

*Proof:* Let us first consider a single check node graph  $H$  which is connected to variable nodes  $v_1, \dots, v_k$ . Suppose  $\mathbf{b} = (b_1, \dots, b_k)$  is a pseudocodeword in this graph. Then,  $\mathbf{b}$  corresponds to a codeword in a lift  $\hat{H}$  of  $H$ . Every check node in  $\hat{H}$  is connected to an even number of variable nodes that are assigned value 1, and further, each variable node is connected to exactly one check node in the check cloud. Since the number of variable nodes that are assigned value 1 is equal to the sum of the  $b_i$ 's, we have  $\sum_i b_i \equiv 0 \pmod 2$ .

Now let  $\hat{G}$  be the corresponding lift of  $G$  wherein  $\mathbf{p}$  forms a valid codeword. Then each check node in  $\hat{G}$  is connected to an even number of variable nodes that are assigned value 1. From the above observation, if nodes  $v_{i_1}, \dots, v_{i_k}$  participate in the check node  $c_i$  in  $G$ , then  $p_{i_1} + \dots + p_{i_k} \equiv 0 \pmod 2$ . Let  $x_i = p_i \bmod 2$ , for  $i = 1, \dots, k$ . Then, at every check node  $c_i$ , we have  $x_{i_1} + \dots + x_{i_k} \equiv 0 \pmod 2$ . Since  $\mathbf{x} = (x_1, \dots, x_n) = \mathbf{p} \bmod 2$  is a binary vector satisfying all checks, it is a codeword in  $\mathcal{C}$ .  $\square$

The following remarks follow from the above lemma:

- If a pseudocodeword  $\mathbf{p}$  has at least one odd component, then it has at least  $d_{\min}$  odd components.
- If a pseudocodeword  $\mathbf{p}$  has a support size  $|\text{supp}(\mathbf{p})| < d_{\min}$ , then it has no odd components.
- If a pseudocodeword  $\mathbf{p}$  has no non-zero codeword contained in its support, then it has no odd components.

**Lemma 3.2** *A pseudocodeword  $\mathbf{p} = (p_1, \dots, p_n)$  can be written as  $\mathbf{p} = \mathbf{c}^1 + \mathbf{c}^2 + \dots + \mathbf{c}^k + \mathbf{r}$ , where  $\mathbf{c}^1, \dots, \mathbf{c}^k$ , are  $k$  (not necessarily distinct) codewords and  $\mathbf{r}$  is some residual vector, containing no codeword in its support, that remains after removing the codewords  $\mathbf{c}^1, \dots, \mathbf{c}^k$  from  $\mathbf{p}$ . Either  $\mathbf{r}$  is the all-zero vector, or  $\mathbf{r}$  is a vector comprising of 0 or even entries only.*

Note that we are not claiming that  $\mathbf{p}$  is reducible even though the vector  $\mathbf{p}$  can be written as a sum of codeword vectors  $\mathbf{c}^1, \dots, \mathbf{c}^k$ , and  $\mathbf{r}$ . Since  $\mathbf{r}$  need not be a pseudocodeword, it is not necessary that  $\mathbf{p}$  be reducible *structurally* as a sum of codewords and/or pseudocodewords (as in Definition 2.5).

*Proof Sketch:* Suppose  $\mathbf{c} \in \mathcal{C}$  is in the support of  $\mathbf{p}$ , then form  $\mathbf{p}' = \mathbf{p} - \mathbf{c}$ . If  $\mathbf{p}'$  contains a codeword in its support, then repeat the above step on  $\mathbf{p}'$ . Subtracting codewords from the pseudocodeword vector in this manner will lead to a decomposition of the vector  $\mathbf{p}$  as stated. We will now show that the residual vector  $\mathbf{r}$ , that contains no codeword in its support, is either the all-zero vector or a vector with 0 and/or even entries only.

From Lemma 3.1 we have  $\mathbf{x} = \mathbf{p} \bmod 2$  to be a codeword in  $\mathcal{C}$ . Since  $\mathbf{p} = \mathbf{c}^1 + \dots + \mathbf{c}^k + \mathbf{r}$ , we have  $\mathbf{x} = (\mathbf{c}^1 + \dots + \mathbf{c}^k) \bmod 2 + \mathbf{r} \bmod 2$ . But since  $\mathbf{x} \in \mathcal{C}$ , this implies  $\mathbf{r} \bmod 2 \in \mathcal{C}$ . However, since  $\mathbf{r}$  contains no codeword in its support,  $\mathbf{r} \bmod 2$  must be the all-zero codeword.  $\square$

The following two lemmas are used in proving the main theorem on the structure of irreducible pseudocodewords.

**Lemma 3.3** *Let  $\mathbf{p} = (p_1, \dots, p_n)$  be a pseudocodeword in  $G$  and suppose all non-zero  $p_i$  are even. If there is a codeword  $\mathbf{c} \in \mathcal{C}$  with  $\text{supp}(\mathbf{c}) \subseteq \text{supp}(\mathbf{p})$ , then  $\mathbf{p}$  is reducible as a sum of pseudocodewords.*

**Lemma 3.4** *Let  $\mathbf{p} = (p_1, \dots, p_n)$  be a pseudocodeword in  $G$ . Rewrite  $\mathbf{p} = \mathbf{x} + \mathbf{r}$ , where  $\mathbf{x} = \mathbf{p} \bmod 2$ . If there is a codeword  $\mathbf{c} \in \mathcal{C}$  with  $\text{supp}(\mathbf{c}) \subseteq \text{supp}(\mathbf{r})$ , then  $\mathbf{p}$  is reducible as a sum of pseudocodewords.*

**Theorem 3.2** *Let  $\mathbf{p} = (p_1, \dots, p_n)$  be an irreducible pseudocodeword in  $G$ . Then  $\mathbf{p}$  can be decomposed as*

$\mathbf{p} = \mathbf{x} + \mathbf{r}$ , where  $\mathbf{x} = \mathbf{p} \bmod 2$  is a codeword (possibly  $\mathbf{0}$ ) and  $\mathbf{r}$  is a vector with no codewords in its support. Furthermore, for any other decomposition  $\mathbf{p} = \mathbf{c}^1 + \mathbf{c}^2 + \dots + \mathbf{c}^k + \mathbf{r}'$  (as in Lemma 3.2), we have  $\mathbf{x} = \mathbf{c}^1 + \mathbf{c}^2 + \dots + \mathbf{c}^k$  and  $\mathbf{r} = \mathbf{r}'$ .

*Proof:* From Lemma 3.1,  $\mathbf{p}$  can be written as  $\mathbf{p} = \mathbf{x} + \mathbf{r}$ , where  $\mathbf{x} = \mathbf{p} \bmod 2$ . From Lemma 3.3 and Lemma 3.4, we have that  $\mathbf{r}$  contains no codeword in its support. Suppose there is another decomposition  $\mathbf{p} = \mathbf{c}^1 + \mathbf{c}^2 + \dots + \mathbf{c}^k + \mathbf{r}'$  (as in Lemma 3.2), where  $\mathbf{r}'$  has only even (possibly zero) components and contains no codeword in its support. Then we have  $\mathbf{c}^1 + \mathbf{c}^2 + \dots + \mathbf{c}^k - \mathbf{x} = \mathbf{r} - \mathbf{r}'$ . Since  $\mathbf{x}$  has only 0 or 1 as components and since the right-hand side is a difference of two vectors having no odd components, the left-hand side must have all components non-negative and even, and in particular, it must be a sum of codewords. But since the right-hand side contains no codeword in its support, this implies that  $\mathbf{x} = \mathbf{c}^1 + \mathbf{c}^2 + \dots + \mathbf{c}^k$  and  $\mathbf{r} = \mathbf{r}'$ .  $\square$

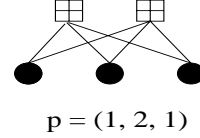


Figure 3: Example of a reducible pseudocodeword  $\mathbf{p}$ .

The “irreducible” condition in Theorem 3.2 is necessary, as illustrated by the following example. If  $\mathbf{p}$  is reducible, then  $\mathbf{p}$  may be decomposed, as in Lemma 3.2, in different ways: Consider the graph  $G$  in Figure 3. A pseudocodeword  $\mathbf{p} = (1, 2, 1)$  can be decomposed as: (a)  $\mathbf{p} = (1, 1, 0) + (0, 1, 1)$ , i.e., as a sum of two codewords, or (b)  $\mathbf{p} = (1, 0, 1) + (0, 2, 0)$ , i.e., as a sum of a codeword and a non-zero remainder  $\mathbf{r}$  that has no codeword in its support.

**Theorem 3.3** *All irreducible non-codeword (nc) pseudocodewords are bad.*

*Proof Sketch:* Suppose  $\mathbf{p}$  is an irreducible nc-pseudocodeword. Then  $\mathbf{p} = \mathbf{x} + \mathbf{r}$ , where  $\mathbf{x}$  and  $\mathbf{r}$  are as above. Suppose  $\mathbf{x} = \mathbf{c}^1 + \mathbf{c}^2 + \dots + \mathbf{c}^k$  is a disjoint union of  $k$  codewords. Suppose  $i^*$  is the index corresponding to a maximal component of  $\mathbf{p}$  in the support of  $\mathbf{r}$ . Then  $i^*$  belongs to the support of at most one of the codewords  $\mathbf{c}^1, \mathbf{c}^2, \dots, \mathbf{c}^k$ , say  $\mathbf{c}^j$ , since they all have disjoint support. Let  $i'$  be an index in  $(\text{supp}(\mathbf{c}^j) \setminus i^*)$  such that  $p_{i'} \leq p_{i^*}$ . We can then construct a weight vector  $\mathbf{w}$  as follows:

$$w_i = \begin{cases} -1 & i = i^* \\ +1 & i = i' \\ +\infty & i \in (\text{supp}(\mathbf{p})^c) \\ 0 & \text{otherwise} \end{cases}$$

Observe that since  $p_{i^*} \geq p_{i'}$ , we have  $\mathbf{p}\mathbf{w}^T \leq 0$  and  $\mathbf{c}\mathbf{w}^T \geq 0$  for any codeword  $\mathbf{c}$ . This shows that  $\mathbf{p}$  is a bad pseudocodeword.  $\square$

### 3.3. Pseudocodewords and lift-degrees

It is of interest to relate a pseudocodeword with the smallest lift degree of the graph which realizes it. In general, characterizing irreducible pseudocodewords suffices, since any pseudocodeword can be expressed as a sum of irreducible pseudocodewords. Further, the weight of any pseudocodeword is lower bounded by the smallest weight of its constituent pseudocodewords. Therefore, given a graph  $G$ , it is useful to find the smallest lift degree needed to realize all irreducible pseudocodewords.

One parameter of interest is the maximum component  $t$  which can occur in any irreducible pseudocodeword of  $G$ . In [7], the weight of a pseudocodeword  $\mathbf{p}$  is shown to be lower bounded in terms of  $t$  and the support size of  $\mathbf{p}$  as follows:

**Lemma 3.5** [7] *Suppose every irreducible pseudocodeword  $\mathbf{p} = (p_1, p_2, \dots, p_n)$  in  $G$  has components  $0 \leq p_i \leq t$ , for  $1 \leq i \leq n$ , then:*

$$(a) \quad w^{AWGN}(\mathbf{p}) \geq \frac{2t^2}{(1+t^2)(t-1)+2t} |\text{supp}(\mathbf{p})| \geq \frac{2t^2}{(1+t^2)(t-1)+2t} s_{\min},$$

$$(b) \quad w^{BSC}(\mathbf{p}) \geq \frac{1}{t} |\text{supp}(\mathbf{p})| \geq \frac{1}{t} s_{\min}.$$

In the case of cycle codes, i.e., LDPC codes having only degree 2 variable nodes, [2] shows that  $t = 2$ . In general, for any given graph, there is such a (finite)  $t$ . The smallest lift degree needed to realize a pseudocodeword is given by the following lemma.

**Lemma 3.6** *Let  $\mathbf{p} = (p_1, \dots, p_n)$  be an irreducible pseudocodeword of a graph  $G$  having largest right degree  $d_r^+$ , and let  $0 \leq p_i \leq t$ , for  $i = 1, \dots, n$ . Then the smallest lift degree  $m_{\min}$  needed to realize  $\mathbf{p}$  satisfies*

$$\max_i p_i \leq m_{\min} \leq \max_{h_j} \frac{\sum_{v_i \in N(h_j)} p_i}{2} \leq \frac{td_r^+}{2},$$

where the maximum is over all check nodes  $h_j$  in the graph and  $N(h_j)$  denotes the variable node neighbors of  $h_j$ .

**Remark:** If  $\mathbf{p}$  is any pseudocodeword and  $b$  is the maximum component, then the smallest lift degree needed to realize  $\mathbf{p}$  is at most  $\frac{bd_r^+}{2}$ .

## 4. EXAMPLES

In this section we present three different examples of Tanner graphs which give rise to different types of pseudocodewords.

Example 1 in Figure 4 shows a graph with all pseudocodewords having weight at least  $d_{\min}$ .

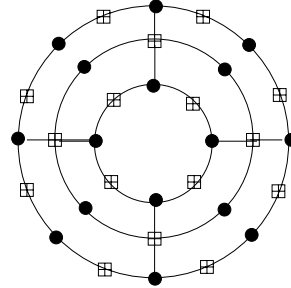


Figure 4: A graph with all pseudocodewords having weight at least  $d_{\min}$ .

Example 2 in Figure 5 shows a graph that has both good and bad pseudocodewords. Consider  $\mathbf{p} = [1, 0, 1, 1, 1, 1, 3, 0, 0, 1, 1, 1, 1, 0]$ . Letting  $\mathbf{w} = [1, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0]$ , we obtain  $\mathbf{p}\mathbf{w}^T = -2$  and  $\mathbf{c}\mathbf{w}^T \geq 0$  for all codewords  $\mathbf{c}$ . Therefore,  $\mathbf{p}$  is a bad pseudocodeword. In particular, we note that  $w^{BSC/AWGN}(\mathbf{p}) = 8$ , while  $d_{\min} = 10$ .

Example 3 in Figure 6 shows a graph on  $m+1$  variable nodes, where all but the left-most variable node form a minimal stopping set of size  $m$ . When  $m$  is even, the only irreducible pseudocodewords are of the form  $(k, 1, 1, \dots, 1)$ , where  $0 \leq k \leq m$  and  $k$  is even. When  $m$  is odd, the only irreducible pseudocodewords have the form  $(k, 1, 1, \dots, 1)$ , where  $1 \leq k \leq m$  and  $k$  is odd. In general, any pseudocodeword of this graph is a linear combination of these irreducible pseudocodewords. When  $k$  is not 0 or 1, then these are nc-irreducible pseudocodewords, and are therefore bad. When  $m$  is odd,

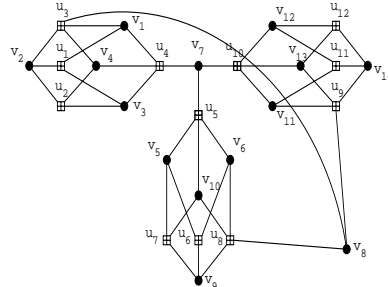


Figure 5: A graph with good and bad pseudocodewords.

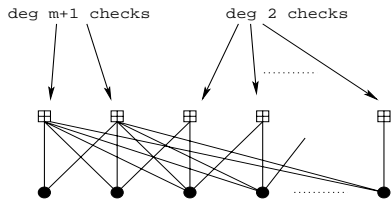


Figure 6: A graph with only bad nc-pseudocodewords.

any pseudocodeword that reduces as a sum of pseudocodewords and includes at least one of these nc irreducible pseudocodewords in the sum, is also bad. However, for even  $m$  the above is not true. As an example, for  $m = 4$ , the pseudocodeword  $\mathbf{p} = (2, 3, 3, 3, 3)$  is a good pseudocodeword, where  $\mathbf{p}$  can be written as  $\mathbf{p} = (2, 1, 1, 1, 1) + 2(0, 1, 1, 1, 1)$ . This example shows that a necessary (but not sufficient) condition for a pseudocodeword to be bad is that it contains at least one nc irreducible pseudocodeword in its structural decomposition as a sum of irreducible pseudocodewords. We also observe that for both the BSC and AWGN channels, all of the irreducible pseudocodewords have weight at most  $d_{\min} = m$  or  $m + 1$ , depending on whether  $m$  is even or odd.

## 5. CONCLUSIONS

We examined the structure of pseudocodewords in Tanner graphs and identified a class of pseudocodewords that can be problematic with iterative decoding. We also established bounds on the smallest lift degree needed to realize a given pseudocodeword. It would be useful to classify other patterns of pseudocodewords and see how they affect convergence of the iterative decoder.

## References

- [1] N. Wiberg, *Codes and Decoding on General Graphs*. PhD thesis, University of Linköping, Linköping, Sweden, 1996.
- [2] G. Horn, *Iterative Decoding and Pseudocodewords*. PhD thesis, California Institute of Technology, Pasadena, CA, 1999.
- [3] G. D. Forney, Jr., R. Koetter, F. Kschischang, and A. Reznik, *On the effective weights of pseudocodewords for codes defined on graphs with cycles*, vol. 123 of *Codes, systems, and graphical models, IMA Vol. Math. Appl.*, ch. 5, pp. 101–112. Springer, 2001.
- [4] R. Koetter and P. O. Vontobel, “Graph-covers and iterative decoding of finite length codes,” in *proc. of the IEEE International Symposium on Turbo Codes and Applications*, (Brest), Sept. 2003.
- [5] J. Feldman, *Decoding Error-Correcting Codes via Linear Programming*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 2003.
- [6] C. Di, D. Proietti, I. Teletar, T. Richardson, and R. Urbanke, “Finite-length analysis of low-density parity-check codes on the binary erasure channel,” *IEEE Transactions on Information Theory*, vol. 48, pp. 1570–1579, June 2002.
- [7] C. Kelley, D. Sridhara, J. Xu, and J. Rosenthal, “Pseudocodewords and Stopping Sets”, in *Proceedings of the IEEE International Symposium on Information Theory*, June 27 - July 3, 2004, Chicago, USA.