ACE SPECTRUM OF LDPC CODES

Dejan Vukobratović, Aleksandar Durendić, Vojin Šenk
Department of Communications and Signal Processing, University of Novi Sad

Abstract – Construction of short-length LDPC codes with good, both waterfall and error-floor, behavior is still an attractive research problem. Recently proposed construction algorithms in this field are based on remarkably simple ideas, but yet, their effectiveness can still be questioned. In this paper we investigate a novel measure of goodness of a given LDPC code, namely its ACE spectrum, based on a previously introduced ACE metrics associated with each cycle in LDPC code graph.

1. INTRODUCTION

Construction of short and medium length LDPC codes is currently an attractive research area. Many practical systems are constrained by delay, memory, processing or similar requirements that translates into a not too large maximum tolerable code length. Therefore, asymptotic behavior predicted by Density Evolution (DE) [1] analysis can not be fully taken into account, as it is known (and we will see in this paper) that short-length LDPC codes designed using only DE guidelines behave poorly. Instead, one has to analyze finite-length effects and construct LDPC code graphs that behave well under iterative decoding. In other words, the task is to construct LDPC code graphs that are free of subgraphs that are proved to present a problem for an iterative decoder.

Our LDPC code design goal and simulation results presented assume transmission over AWGN (Additive White Gaussian Noise) channel. Beside the fact that there is a lot of work done recently in the area of finite-length behavior of iterative decoding over AWGN channel, for the sake of simplicity we use results that address finite-length behavior of iterative decoding over BEC (Binary Erasure Channel) for our code design purposes. It turns out that designing good LDPC codes for BEC, produces good LDPC code performers over AWGN channel, making this design methodology well accepted.

Finite-length behavior of LDPC codes decoded iteratively is fully determined by the subgraphs of LDPC code graph called stopping sets [2]. Small stopping sets deteriorate the performance of LDPC codes, particularly in the error-floor region. There is no explicit construction that optimizes LDPC code graphs with respect to minimal stopping set size and this presents one of the most attractive open problems in LDPC code design. However, two recently proposed algorithms, PEG algorithm [3] and ACE constrained algorithm [4], heuristically produce LDPC code graphs with significantly larger minimal stopping sets compared with random construction. Both are essentially based on the fact that stopping set subgraphs are comprised of at least one, but usually more, interconnected cycles. The focus of our attention is the second one, namely ACE constrained LDPC code construction algorithm.

ACE constraining algorithm is very effective in the range of the pair \((d_{\text{max}}, q)\) of its input parameters (that will be defined in the next section). However, approaching larger values of its input parameters that give rise to expectably better LDPC codes, this algorithm runs into huge calculation complexity problems. By suggesting and analyzing novel measure of goodness of selected LDPC code, named ACE spectrum, we intend to approach to the LDPC code design in a different way. Since random construction of LDPC code and its ACE spectrum analysis are short-time consuming tasks, we ask ourself is it possible to find a comparably good LDPC code in shorter time by random search through an ensemble of random LDPC constructions, than by using ACE constrained algorithm. A similar approach, but with design goal of maximizing LDPC code graph girth by random search, has been taken in [5].

After introductory section, this paper continues with the description of ACE metrics and associated ACE constrained construction of LDPC codes in Section 2. In Section 3 we introduce ACE spectrum of a given LDPC code. Section 4 presents our simulation results and points out for a possible usefulness of ACE spectrum. Section 5 concludes this paper and points to a directions of our future work.

2. ACE CONSTRAINED LDPC CODES

Design of a good LDPC code graph assumes maximizing the size of a minimal stopping set. Tanner graph, or LDPC code graph \(G(H)\), is derived from the LDPC code parity check matrix \(H\) in a usual way, where variable or left nodes corresponds to columns, check or right nodes corresponds to rows, and edges corresponds to ones in the parity check matrix. We start with a definition of a stopping set.

Definition: The set of variable nodes such that every check node neighbor of this set is connected to this set at least twice is called stopping set. (Equivalently, the subgraph of original LDPC code graph that is induced from the stopping set has a check node degrees of at least two.)

It is pointed out in [4] that stopping set is comprised of one or several interconnected cycles. Since every stopping set possess cycle, we can drop down to a level of cycles and design good LDPC code by influencing cycles. Traditional approach to remove all short cycles is essentially good, since it removes all small stopping sets. PEG algorithm proceeds in this fashion. However, for a short-length codes this approach is not very effective since the obtained girths are still quite small.

Not all cycles of the same length are equally bad. This is noted in [4], and based on this observation all cycles of same length could be classified if we could evaluate them using some metric. The metric proposed in [4], named Extrinsic Message Degree (EMD) of the cycle, measures the level of connectivity of the cycle with the rest of the graph.
Definition: For a given cycle C in the LDPC code graph let \( V_C \) be the set of variable nodes in C and \( C(V_C) \) be the set of check node neighbors of \( V_C \). We can divide the set \( C(V_C) \) into three disjoint subsets:

- \( C^{\text{cycle}}(V_C) \): subset of \( C(V_C) \) belonging to the cycle C. Each node from \( C^{\text{cycle}}(V_C) \) is at least doubly connected to the set \( V_C \).
- \( C^{\text{cut}}(V_C) \): subset of \( C(V_C) \) that are not in the cycle C, but are at least doubly connected to the set \( V_C \), and
- \( C^{\text{ext}}(V_C) \): subset of \( C(V_C) \) singly connected to the set \( V_C \).

This definition leads to another, related to the edges:

Definition: For a given cycle C in the LDPC code graph and the corresponding set \( V_C \), let \( E(V_C) \) be the set of edges incident to \( V_C \). We can divide the set \( E(V_C) \) into three disjoint subsets:

- \( E^{\text{cycle}}(V_C) \): subset of cycle edges in \( E(V_C) \) incident to check nodes in \( C^{\text{cycle}}(V_C) \),
- \( E^{\text{cut}}(V_C) \): subset of cut edges in \( E(V_C) \) incident to check nodes in \( C^{\text{cut}}(V_C) \), and
- \( E^{\text{ext}}(V_C) \): subset of extrinsic edges in \( E(V_C) \) incident to check nodes in \( C^{\text{ext}}(V_C) \).

Definition: Extrinsic Message Degree of a given cycle C in the LDPC code graph, denoted EMD(C), is

\[
\text{EMD}(C) = |E^{\text{ext}}(V_C)|,
\]
where \( |E^{\text{ext}}(V_C)| \) is the cardinality of \( E^{\text{ext}}(V_C) \), i.e. the number of extrinsic edges of C. (See Fig.1 for an example.)

Fig.1. Cycle, cut and extrinsic nodes and edges

If a given cycle in LDPC code graph has low EMD, than its communication with the rest of the graph is limited. This limits the amount of new evidence about values of variable nodes in the cycle that could be collected from the rest of the graph. In the extreme case, when EMD of the cycle is zero, variable nodes in the cycle are isolated from the rest of the graph and the cycle is a stopping set.

It is not an easy task to find EMD of the cycle in the graph, since it takes additional steps to determine if the edge is extrinsic edge or cut edge. If we neglect this difference and account for both, extrinsic and cut edges, into the cycle metric, we get simplified version of the EMD metric.

Definition: Approximated Cycle EMD (ACE) of a given cycle C in the LDPC code graph, denoted ACE(C), is

\[
\text{ACE}(C) = |E^{\text{ext}}(V_C)| + |E^{\text{cut}}(V_C)|.
\]

It is easy to calculate ACE(C) as:

\[
\text{ACE}(C) = \sum_{v \in E(V_C)} (d(v) - 2),
\]
where \( d(v) \) is the degree of variable node \( v \).

ACE construction algorithm\(^1\) [4] is developed to design \((d_{\text{max}}, \eta)\) ACE constrained LDPC codes where, by construction, each cycle of length less than or equal to \( 2d_{\text{max}} \) has ACE value larger than \( \eta \). Simulation results [4] confirms that \((d_{\text{max}}, \eta)\) LDPC codes could reach very low error-floors for a selected values of \((d_{\text{max}}, \eta)\). However, increasing each of or simultaneously both of the parameters from pair \((d_{\text{max}}, \eta)\) leads this algorithm quickly into the computational problems, since finding \((d_{\text{max}}, \eta)\)-compliant graph becomes very hard task (for sufficiently large values in pair \((d_{\text{max}}, \eta)\) it could even become impossible.) This is particularly notable for high rate or short-length LDPC codes.

3. ACE SPECTRUM

In this Section we introduce an ACE spectrum. We prove its usefulness in the next Section by providing illustrative simulation results.

Definition: Let \( G(H) \) be an LDPC code graph with \( n \) variable nodes and let \( d_{\text{max}} \) be its largest left degree. Let \( l \) be an even integer \( 4 \leq l \leq 2d_{\text{max}} \). For each \( l \) let \( n_{\text{ACE}} = (n_{\text{ACE}}(0), ..., n_{\text{ACE}}(l - 1)) \) be \( k \)-tuple of values where

\[
n_{\text{ACE}}(i) = \text{number of variable nodes with a property that smallest ACE value of the cycle of length } l \text{ they belong to is equal to } i, 0 \leq i \leq (k - 1) = (l/2)*(d_{\text{max}} - 2).
\]

ACE spectrum of \( G(H) \), \( G(H) = (n_{\text{ACE}}, ..., n_{\text{ACE}}) \), is the \((d_{\text{max}} - 2)\)-tuple of \( n_{\text{ACE}} k \)-tuples, \( 4 \leq l \leq 2d_{\text{max}} \). (We will present \( G(H) \) conveniently in the form of histograms for each \( n_{\text{ACE}} \) in the following Section.)

It is possible to obtain a linear-time procedure that provides ACE spectrum of selected LDPC code. In fact, we developed this procedure as a modification of the Viterbi-like procedure used in [4] for examination if the newly added variable node is \((d_{\text{max}}, \eta)\) ACE compliant (ACE algorithm generates the \( G(H) \) LDPC code graph in a variable-node-by-node fashion). For the short-length LDPC codes this provides us with a very practical and fast ACE spectrum calculator.

The fact that ACE spectrum of an LDPC code graph can be easily calculated provides us with a useful and efficient tool for a quick evaluation of any LDPC code graph designed by any method. Also, we believe that by using ACE spectrum we can design LDPC code graphs that are in the region of parameters \((d_{\text{max}}, \eta)\) unattainable by ACE constraining algorithm. While search for a large girth LDPC code graphs was popular.

4. SIMULATION RESULTS

For the purpose of examination of the ACE spectrum usefulness in predicting the LDPC code behavior we created

\(^1\) For the details of ACE constrained construction algorithm see [4].
several different LDPC code graphs using various design rules and investigated both, its ACE spectrum and Bit Error Rate (BER) performance. In particular, we present results obtained for three LDPC code graphs. All codes are of length $n = 1000$, rate $r = \frac{1}{2}$, and with DE optimized left degree distribution for $d_{\text{max}} = 9$. First code is randomly generated without any further restrictions. Second code is free of cycles among degree 2 variable nodes. Third code is (3,8) compliant ACE constrained code. BER performance curves of these codes assuming transmission over AWGN channel are given in Fig. 3.

We can point out several things out of our simulation results. For the first code it is obvious that for short block lengths DE is not a sufficient design tool. Without avoiding short cycles among degree 2 variable nodes what we get is a large number of stopping sets of size 2 (note that $n^i_{\text{ACE}}(0) \sim 180$) that give rise to a high error-floor. Once we eliminate this problem in the second code, we have much better error-floor performance, but still significant number of dangerous, ACE equal to one and two, cycles at cycle lengths 4 and 6. By using (3,8) ACE constraining algorithm, we have eliminated all cycles of length less than or equal to 6 with ACE values smaller than 8 (which is clearly visible in the ACE spectrum of this code). Obtained error-floor performance is further improved.

We ran into serious computation problems when attempting to construct ACE constrained codes with the same parameters, but with $d_{\text{max}} = 4$ and small $\eta$ values such as 4. In [4], authors claim that they reached significantly larger region of $(d_{\text{max}}, \eta)$ pairs. Possible explanation for this is that they constructed rate $\frac{1}{2}$, length 10000 code, where larger code length enabled easier construction. For the shorter-length example they used (1268, 456) rate $\frac{1}{3}$ code, which is of considerably smaller rate and therefore easier to construct.
5. CONCLUSIONS

In this paper we introduced and experimented with a novel evaluation tool for LDPC code graphs named ACE spectrum. We demonstrated using simulation results its effectiveness in predicting LDPC code BER performance. In the following work, we will try to design LDPC code graphs with extremely good ACE spectrum characteristics by using random search through appropriately expurgated ensemble of irregular LDPC codes. We hope that we can, by using fast ACE spectrum calculating algorithm, find codes that belong to unattainable region of \((d_{\text{max}}, \eta)\) pairs for ACE constraining algorithm.

LITERATURE


