Regular and Irregular Progressive Edge-Growth Tanner Graphs

Xiao–Yu Hu, Evangelos Eleftheriou, and Dieter–Michael Arnold

Abstract

We propose a general method for constructing Tanner graphs having a large girth by progressively establishing edges or connections between symbol and check nodes in an edge-by-edge manner, called progressive edge-growth (PEG) construction. Lower bounds on the girth of PEG Tanner graphs and on the minimum distance of the resulting low-density parity-check (LDPC) codes are derived in terms of parameters of the graphs. The PEG construction attains essentially the same girth as Gallager's explicit construction for regular graphs, both of which meet or exceed the Erdös–Sachs bound. Asymptotic analysis of a relaxed version of the PEG construction is presented. We describe an empirical approach using a variant of the "downhill simplex" search algorithm to design irregular PEG graphs for short codes with fewer than a thousand of bits, complementing the design approach of "density evolution" for larger codes. Encoding of LDPC codes based on the PEG construction is also investigated. We show how to exploit the PEG principle to obtain LDPC codes that allow linear time encoding. We also investigate regular and irregular LDPC codes using PEG Tanner graphs but allowing the symbol nodes to take values over GF(q), $q > 2$. Analysis and simulation demonstrate that one can obtain better performance with increasing field size, which contrasts with previous observations.

Keywords

LDPC codes, irregular and regular Tanner graphs, linear-time-encodable codes, girth

IBM Research, Zurich Research Laboratory, 8803 Rüschlikon, Switzerland. Email: {xhu, ele, arn}@zurich.ibm.com

I. Introduction

CODES on graphs [1-12][1] have attracted considerable attention owing to their capacity-approaching performance and low-complexity iterative decoding. The prime examples of such codes are the low-density parity-check (LDPC) codes. It is known that the belief-propagation (BP) or sum-product algorithm (SPA) over cycle-free Tanner graphs [1] provides optimum decoding, hence it is natural to try to minimize the influence of the cycles in the iterative decoding process. This approach has been adopted for both LDPC codes [13] and turbo codes [14] by using rather long block lengths. If the cycles are made long enough, the decoding algorithm may run several iterations without being affected by them while the error rate decreases exponentially with the number of independent iterations. Using the incidence matrix associated with a graph, Gallager proposed an explicit construction [13, pp. 81-89] that guarantees independent decoding iterations up to a lower bound for his LDPC codes. Unfortunately this construction is valid only for regular LDPC codes, and appears to be computationally infeasible for relatively large block lengths.

For most existing classes of LDPC codes, the Tanner graph is randomly constructed by avoiding cycles of length 4 [15-18]. To date, randomly constructed LDPC codes have largely relied on the sparsity of the parity-check matrix to avoid short cycles in the Tanner graph. Although random graphs have been used to construct LDPC codes with impressive performance [15, 19], there is no guarantee that any given random graph defines a good code having a suitable shortest cycle (girth) that facilitates iterative decoding as well as having a respectable minimum distance that enhances decoding performance in a high signal-to-noise (SNR) regime. In particular, for short block lengths, the probability of choosing an unfavorable random graph is surprisingly high. The minimum distance issue becomes critical if an irregular degree sequence is used.

Construction of LDPC codes based on finite geometries has been reported in [20]. Finite geometry LDPC codes have relatively good minimum distances and their Tanner graphs do not contain cycles of length 4. They can be put in either cyclic or quasi-cyclic form so that the encoding can be achieved in linear time by using simple feedback shift registers. With high-rate and very long block-length, these codes perform very

[1] See also the special issue on codes and graphs and iterative algorithms of IEEE Transactions on Information Theory, Feb. 2001, edited by B. J. Frey, R. Koetter, G. D. Forney, Jr., F. R. Kschischang, R. J. McEliece, and D. A. Spielman.

well under iterative decoding, only a few tenths of a decibel away from the Shannon limit. For further results, see [21-24].

Since the early work of Gallager, the first significant work in constructing LDPC codes based on *graph-theoretic* algebraic approach was reported in [25]. In [26, 27], explicit group-theoretic constructions of graphs were proposed. These graphs have girth exceeding the Erdös–Sachs bound [28], which is a non-constructive lower bound on the girth of random graphs and has the same significance as the Gilbert–Varshamov bound does in the context of minimum distance of linear codes. The notion of graph expansion was first introduced as an analysis tool in coding theory [3]. Recently, the algebraic approach has been pursued even further, with emphasis on constructing LDPC codes having almost the largest girth possible [29, 30]. However, having large girth is only one necessary condition to yield good codes; design of good irregular degree sequence turns out to be equally or more important. Furthermore, practical code-design aspects, such as flexible construction with arbitrary code rates and block sizes as well as linear-time encoding, often conflict with algebraic construction methods.

In this paper we present a simple but efficient method for constructing Tanner graphs having a large girth in a best-effort sense by progressively establishing edges between symbol and check nodes in an edge-by-edge manner, called PEG construction. Given the number of symbol nodes n, the number of check nodes m, and the symbol-node-degree sequence of the graph, an edge-selection procedure is started such that the placement of a new edge on the graph has as small an impact on the girth as possible. After a best-effort edge has been determined, the graph with this new edge is updated, and the procedure continues with the placement of a next edge. The PEG construction presented here is a general, non-algebraic method for constructing graphs with large girth that asymptotically guarantees a girth at least as large as the Erdös–Sachs bound. Simulation results show that the resulting LDPC codes of progressive PEG Tanner graphs significantly outperform randomly constructed ones at relatively short block lengths.

In general, LDPC codes based on randomly constructed Tanner graphs do not lend themselves to easy characterization in terms of girth distribution, minimum distance, and number of nearest neighbors. In contrast, a Tanner graph constructed using the PEG principle has elegant properties with respect to the girth of the graph and minimum distance of the induced binary LDPC code. In particular, lower bounds on the girth and on the minimum distance are derived in terms of parameters of the underlying PEG Tanner graph.

It was shown in [16, 31, 32] that carefully designed irregular LDPC codes can outperform regular ones. Using the density evolution technique, irregular graphs have been designed by optimizing the degree-distribution pairs via linear programming or differential evolution [18, 31]. This approach proven to be very successful for designing random long block-length LDPC codes; but it is not yet clear whether it is applicable to short block-length codes and the PEG construction. We describe an empirical Monte-Carlo approach using a variant of the "downhill simplex" search algorithm [40] to design irregular PEG graphs for short codes with fewer than a thousand of bits, serving as a complementary procedure to the asymptotic density evolution technique. In our approach we focus on irregular PEG graphs whose degree of parity-check nodes is made as uniform as possible, so-called right-concentrated sequences [34, 35]. The conventional downhill simplex algorithm [40, 41] is a nonlinear optimization procedure that operates on a multidimensional simplex. A similar optimization approach has been chosen in [33] to construct *random* irregular LDPC codes with two empirical cost functions: one based on empirical decoding trials, the other based on simulation decoding. In contrast, we design irregular LDPC codes based on the PEG construction and a nonlinear cost function that depends on the block-error rate at a certain E_b/N_0. It is demonstrated by simulations that our irregular PEG LDPC codes outperform significantly regular ones at short block lengths. Furthermore, it is shown that the irregular degree sequences obtained by the density-evolution approach, if properly chosen, work very well with the PEG construction too.

LDPC codes have been considered as a serious competitor of Turbo codes. One major drawback so far has been their apparent high encoding complexity that grows quadratically with the block length. The use of a cascade of subcodes to accomplish linear-time encoding complexity was proposed in [42, 31]. As each subcode is, in general, considerably smaller that the overall code, a performance loss is expected. Another approach, proposed in [43], is to force the parity-check matrix to have a lower or upper triangular form. Similar in spirit is the approach to incorporate a pattern, referred to as the zig-zag pattern, in the parity-check matrix [44, 45]. In [46] it is proved that optimized LDPC codes can be encoded in linear time. Specifically, one can choose a randomly-generated LDPC code with an optimized irregular degree sequence pair, and rearrange the parity-check matrix of this code to have an almost upper-triangular structure allowing linear-time encoding. The significance of [46] lies in the existence proof of optimal LDPC codes with almost upper-triangular structure. Note that the rearrangement of the parity-check matrix does not change the LDPC code itself and the complexity of rearrangement is quadratic in general. Thus it is preferable to construct such LDPC codes directly.

We investigate the construction of LDPC codes having (almost) triangular structure, good girth properties and optimum irregular degree sequence. It is shown that the linear-time encoding property can be combined with the PEG construction in a very natural way while maintaining equally good performance. To this end we demonstrate that our short block-length, linear-time encoding LDPC codes offer comparable performance to the standardized Turbo codes in third-generation high-speed wireless data services with essentially the same encoding/decoding complexity.

Finally, we investigate the regular and irregular LDPC codes by using the same PEG construction but allowing symbol nodes to assume values from a finite field with more than two elements. In particular, fields with 2^b elements denoted as GF(2^b), $b > 1$, have been considered. Such constructions were first proposed in [1], and performance results of randomly-constructed LDPC codes over GF(2^b) were reported in [47]. The fact that one can obtain better performance with increasing field size, is demonstrated by both analysis and simulation results, which contrasts with the observations of [47]. We report a short block-length (1008 bit) rate-1/2 irregular PEG LDPC code over $GF(2^6)$ with a block error rate $< 10^{-4}$ at $E_b/N_0 = 2$ dB, which appears to have the best performance at this short block length to date.

The remainder of this paper is organized as follows. Section II introduces the necessary definitions and notations on graphs. Section III describes the principle and the detailed algorithm of the PEG construction. In Section IV we deal with graph properties of PEG Tanner graphs; in particular, the lower bounds on the girth and on the minimum distance are derived. Section V deals with the asymptotic analysis of the PEG Tanner graphs based on a series of relaxation. We focus on the distance (weight) distribution function of the resulting LDPC codes. We prove the asymptotic optimality of the PEG construction by showing the asymptotic optimality of its weakened version. Section VI describes the variant of the "downhill simplex" search algorithm used to design near-optimum symbol-node-degree distributions for irregular PEG graphs. Section VII presents simulation results comparing the performance of regular and irregular LDPC codes defined on PEG Tanner graphs with that of randomly constructed ones. We address the linear-time encoding based on the PEG principle in Section VIII and compare the performance of linear-time encoding PEG LDPC codes with that of Turbo codes in IX. In Section X we extend the binary LDPC codes defined on PEG Tanner graphs to codes over a finite field with more than two elements. Finally, Section XI concludes the paper.

II. Definitions and Notations

An LDPC code is a linear code defined by a sparse parity check matrix H having dimension $m \times n$. A bipartite graph with m check nodes in one class and n symbol nodes in the other can be created using H as the integer-valued incidence matrix for the two classes. Such a graph is also called Tanner graph [1]. As a Tanner graph defines a unique parity check matrix and a parity check matrix corresponds to a unique Tanner graph, we shall use the terms Tanner graph and code interchangeably throughout the paper if no confusion arises. Formally, a Tanner graph is denoted as (V, E), with V the set of vertices (nodes), $V = V_c \cup V_s$, where $V_c = \{c_0, c_1, \ldots, c_{m-1}\}$ is the set of check nodes and $V_s = \{s_0, s_1, \ldots, s_{n-1}\}$ the set of symbol nodes. E is the set of edges such that $E = V_c \times V_s$, with edge $(c_i, s_j) \in E$ if and only if $h_{i,j} \neq 0$, $h_{i,j} \in H$, $0 \leq i \leq m-1, 0 \leq j \leq n-1$. A Tanner graph is called (d_s, d_c)-*regular* if every symbol node participates in d_s check nodes and every check node involves d_c symbol nodes; otherwise it is called *irregular*. Denote the symbol degree sequence by $D_s = \{d_{s_0}, d_{s_1}, \ldots, d_{s_{n-1}}\}$, in which d_{s_j} is the degree of symbol node s_j, $0 \leq j \leq n-1$, in nondecreasing order, i.e., $d_{s_0} \leq d_{s_1} \ldots \leq d_{s_{n-1}}$, and the parity-check degree sequence by $D_c = \{d_{c_0}, d_{c_1}, \ldots, d_{c_{m-1}}\}$, in which d_{c_j} is the degree of parity-check node c_j, $0 \leq j \leq m-1$, and $d_{c_0} \leq d_{c_1} \ldots \leq d_{c_{m-1}}$. Let also the set of edges E be partitioned in terms of V_s as $E = E_{s_0} \cup E_{s_1} \cup \cdots \cup E_{s_{n-1}}$, with E_{s_j} containing all edges incident on symbol node s_j. Further, denote the k-th edge incident on s_j by $E^k_{s_j}$, $0 \leq k \leq d_{s_j} - 1$. Fig. 1 shows an example of a $D_s = \{2, 2, 2, 2, 3, 3, 3, 3\}$ irregular Tanner graph, in which the check degree sequence is uniformly of degree 5, i.e., $D_c = \{5, 5, 5, 5\}$. Note that the definition of a Tanner graph is quite general in that the check node may be a simple parity check, a parity check over Galois field, or even a subcode such as BCH code.

A graph is called *simple* if 1) it does not have a self loop that is an edge joining a vertex to itself, 2) there is at most one edge between the same pair of vertices, and 3) all edges are non-directed. In a simple graph, we say that vertices x and y are *adjacent* if (x, y) is an edge. The set consisting of all vertices that are adjacent to x is called x's *neighbor*. A *subgraph* of a graph $G = (V, E)$ is a graph whose vertex and edge set are subsets of those of G. Note that, if $G' = (V', E')$ is a subgraph of G, then for every edge $e \in E'$, it must hold that both the vertices of e lie in V'. A sequence of distinct vertices, starting from x and ending with y, is called a *path* between x and y, if any two consecutive vertices in the sequence are adjacent. A closed path with edges starting from x and ending with itself is called a *cycle* of x. If there exists at least one path between x and y, then x and y is called *connected* or x is *reached* by y and vice versa. If two vertices x and y in the graph are connected, their distance $d(x, y)$ is then defined as the length (number of edges) of the shortest path joining them. A subgraph is *tree-like* if there is no cycle inside.

In general, an ensemble of bipartite or Tanner graphs is characterized by degree distribution pairs. In the case of symbol node, the degree distribution is defined as $\Lambda(x) = \sum_{i\geq 2}^{d_s^{\max}} \Lambda_i x^i$, where Λ_i is the fraction of symbol nodes connected to exactly i check nodes; $d_s^{\max}$ is the largest entry in $D_s = \{d_{s_0}, d_{s_1}, \ldots, d_{s_{n-1}}\}$, and $\sum_{i\geq 2}^{d_s^{\max}} \Lambda_i = 1$. Similarly, in the case of parity-check node, the degree distribution is defined as $\Phi(x) = \sum_{i\geq 2}^{d_c^{\max}} \Phi_i x^i$, where Φ_i is the fraction of parity-check nodes connected to exactly i symbol nodes; $d_c^{\max}$ is the largest entry in $D_c = \{d_{c_0}, d_{c_1}, \ldots, d_{c_{m-1}}\}$, and $\sum_{i\geq 2}^{d_c^{\max}} \Phi_i = 1$.

For a given symbol node s_j, define its *neighbor within depth l*, $\mathcal{N}_{s_j}^l$, as the set consisting of all check nodes reached by a subgraph (or a tree) spreading from symbol node s_j within depth l, as shown in the example in Fig. 2. Its complementary set, $\bar{\mathcal{N}}_{s_j}^l$, is defined as $V_c \backslash \mathcal{N}_{s_j}^l$, or equivalently $\bar{\mathcal{N}}_{s_j}^l \cup \mathcal{N}_{s_j}^l = V_c$. The subgraph rooted from s_j is generated by means of unfolding the Tanner graph in a breadth-first way. That is to say, we start from s_j, and traverse all edges incident on s_j; let these edges be $(s_j, c_{i_1}), (s_j, c_{i_2}), \ldots, (s_j, c_{i_{d_{s_j}}})$. Then explore all other edges incident on vertices $c_{i_1}, c_{i_2}, \ldots, c_{i_{d_{s_j}}}$ excluding $(s_j, c_{i_1}), (s_j, c_{i_2}), \ldots, (s_j, c_{i_{d_{s_j}}})$. This process continues until the depth can not increase further. Note that in the subgraph, duplicate vertices or edges may occur. Referring to Fig. 2, any symbol node residing at depth-l has a distance to s_j of $2l$, and any check node residing at depth-l has a distance to s_j of $2l+1$. Therefore $\mathcal{N}_{s_j}^l$ can be alternatively defined as the check node subset of distance (relative to s_j) smaller than or equal to $2l+1$.

Similarly, for a given parity-check node c_i, define its neighbor with depth l, $\mathcal{N}_{c_i}^l$, as the set consisting of all parity-check nodes reached by a subgraph (or a tree) spreading from c_i within depth-l, as shown in Fig. 3.

The set $\mathcal{N}_{s_j}^l$ and its counterpart $\bar{\mathcal{N}}_{s_j}^l$ can be efficiently evaluated in a recursive manner. Initially mark the symbol node s_j with integer 0 and set $d = 0$. While any vertex, either symbol node or parity-check node, was marked at the preceding stage, 1) look at all vertices marked d and mark all unmarked neighbors of such vertices with $d+1$; 2) replace d by $d+1$. At the termination of this algorithm, the mark of marked vertices is their distance to s_j, and the unmarked vertices, if exist, do not have a path connecting to s_j. Obviously $\mathcal{N}_{s_j}^l$ can be obtained by simply examining the check nodes having a mark less than or equal to $2l+1$.

In graph theory, girth g refers to the length of the shortest cycle in a graph. For each symbol node s_j, we define a local girth g_{s_j} as the length of the shortest cycle passing through that symbol node. The set of local girth $\{g_{s_j}\}$ is referred to as girth histogram. It follows, by definition, that $g = \min_j \{g_{s_j}\}$.

III. Progressive Edge-Growth (PEG) Construction

Constructing a Tanner graph with the largest possible girth is a rather difficult combinatorial problem. Nevertheless, a sub-optimum algorithm to construct a Tanner graph with a relatively large girth is feasible in practice. One such algorithm is the PEG algorithm that we present here, in which the local girth of a symbol node is maximized whenever a new edge is added to this symbol node. Suppose we have finished constructing edges of the first j symbol nodes on a Tanner graph, i.e., edges $E_{s_0} \cup E_{s_1} \cup \cdots \cup E_{s_{j-1}}$ have been established. Let g^t be the temporary girth under the current graph setting with edges $E_{s_0} \cup E_{s_1} \cup \cdots \cup E_{s_{j-1}}$. In other words, $g^t = \min\{g_{s_0}, g_{s_1}, \ldots, g_{s_{j-1}}\}$. The problem of constructing a graph having a large girth lies in how to select the edge set E_{s_j} of symbol node s_j such that adding these new edges to the current graph setting does not impair the current g^t excessively. This boils down to optimizing d_{s_j} edges of E_{s_j} to maximize the local girth g_{s_j} because, if adding E_{s_j} to the current graph results in a cycle shorter than g^t, this new short cycle must pass through symbol node s_j. Unfortunately the complexity of exhaustively searching for the optimum set E_{s_j} is governed by $\binom{m}{d_{s_j}}$, where $\binom{m}{d_{s_j}}$ is a binomial coefficient. Thus, we propose a best-effort algorithm known as progressive edge-growth, in which d_{s_j} edges of E_{s_j} are added to the current graph *on an edge-by-edge basis*, and the length of the shortest cycle passing through symbol node s_j is maximized whenever a new edge originating in s_j is being added. This can be accomplished simply by first expanding the tree originating in symbol node s_j up to depth l each time a new edge of s_j is being determined, such that $\bar{\mathcal{N}}_{s_j}^l \neq \varnothing$ but $\bar{\mathcal{N}}_{s_j}^{l+1} = \varnothing$ or the cardinality of $\mathcal{N}_{s_j}^l$ stops increasing but is smaller than m, and then placing an edge between s_j and a check node selected from the set $\bar{\mathcal{N}}_{s_j}^l$. In this way, the shortest cycle passing through this new edge under the current graph setting is guaranteed to be no shorter than $2(l+2)$. We summarize the proposed algorithm as follows.

Progressive Edge-Growth Algorithm:

for $j = 0$ to $n-1$ **do**

begin

for $k = 0$ to $d_{s_j} - 1$ **do**

begin

if $k = 0$

$E_{s_j}^0 \longleftarrow$ edge (c_i, s_j), where $E_{s_j}^0$ is the first edge incident to s_j and c_i is one check node such that it has the lowest check degree under the current graph setting $E_{s_0} \cup E_{s_1} \cup \cdots \cup E_{s_{j-1}}$.

else

expanding a tree from symbol node s_j up to depth l under the current graph setting such that $\bar{\mathcal{N}}_{s_j}^l \neq \varnothing$ but $\bar{\mathcal{N}}_{s_j}^{l+1} = \varnothing$, or the cardinality of $\mathcal{N}_{s_j}^l$ stops increasing but is less than m, then $E_{s_j}^k \longleftarrow$ edge (c_i, s_j), where $E_{s_j}^k$ is the k-th edge incident to s_j and c_i is one check node picked from the set $\bar{\mathcal{N}}_{s_j}^l$ having the lowest check-node degree.

end

end

There is a subtle point in the PEG algorithm that needs further comment. Whenever we encounter multiple choices for connecting to symbol node s_j, i.e., multiple check nodes exist in $\bar{\mathcal{N}}_{s_j}^l$, we select the one having the smallest number of incidence edges under the current graph setting. Such a selective strategy renders the resulting PEG Tanner graphs as check-node-degree uniform as possible and it tends to favor graphs with a non-zero degree in parity-check nodes (parity-check-node regular graphs), or concentrated graphs with two non-zero consecutive degrees otherwise. There is strong evidence indicating that such a degree sequence is optimum in some weak sense [34, 35].

Even so, we may still face a situation in which multiple choices exist because multiple check nodes in $\bar{\mathcal{N}}_{s_j}^l$ might have the same lowest degree, particularly during the start-up period. There are two main approaches to solve this problem. The first is to randomly select one of these check nodes. The other is to always select one according to its position in the order of $c_0, c_1, \ldots, c_{m-1}$. For instance, we can first sort the check nodes in $\bar{\mathcal{N}}_{s_j}^l$ that have the same lowest degree according to their subscripts, and then always pick the first one. Here we adopt the first approach; however, note that the second may also be of interest because of its deterministic nature.

We close this section by pointing out the following remarks.

1. *Complexity*— The computational load in evaluating the set $\bar{\mathcal{N}}_{s_j}^l$ depends primarily on the degree sequences D_s and D_c as well as the depth l. In a sparse graph the elements of D_s and D_c are small numbers compared with the block length n, and l grows at most logarithmically with the block length n (we will see this point later). This compares much favorably with other random constructions that aim to achieve large girth whose complexity generally grows exponentially with n.

2. *Non-greedy version* — The version presented above is greedy in the sense that the spreading subgraph of s_j proceeds as deep as possible, i.e., the depth l is maximized such that $\bar{\mathcal{N}}_{s_j}^l \neq \varnothing$ but $\bar{\mathcal{N}}_{s_j}^{l+1} = \varnothing$. This approach proves to be favorable if the minimum distance is at a premium, particularly for short block-length and/or high-rate codes [36-38]. However, for long block-length, low-rate codes in which the minimum distance is not a critical issue, it might be favorable to limit l to a certain value $l_{\max}$, 1) to make the check-node degree sequence concentrated in the strict sense, 2) possibly to reduce the *diameter* of the graph — the maximum

distance of distinct vertex pairs, such that fewer decoding iterations are required. This kind of variant is called *non-greedy* PEG algorithm. Note that if one set $l_{\max} = g_t/2 - 2$, where g_t is the target girth, then this variant bears some resemblance to the "bit-filling" algorithm appeared in the independent work [39].

3. *Look-ahead-enhanced version* — The PEG principle refers to constructing edges in stages, where at each stage we make the choice for an edge emanating from a symbol node which locally optimizes the shortest cycle passing through the assumed edge, by which we move closer to our final goal. Obviously, this short-sighted local optimization does not usually produce the best overall solution. One can enhance the greedy PEG algorithm by looking one-step ahead. In the look-ahead-enhanced version, the same procedure as in the greedy PEG algorithm is applied, except when multiple choices exist for placing the k edge of s_j. In this case, rather than randomly select one, additional test is introduced to distinguish multiple choices. That is, for each candidate in $\bar{\mathcal{N}}_{s_j}^{l}$, we evaluate the maximum depth l of s_j *assuming this candidate edge being put onto the graph*, and associate the maximum depth to each candidate, and finally select the one having the largest value as the parity-check node that the k edge of s_j joins.

4. *Flexibility and Scalability* — The idea underlying the PEG algorithm is quite flexible and scalable. The PEG algorithm can be used to construct good regular or irregular bipartite graphs. Later we will also show how to extend the PEG algorithm to construct linear-time encoding LDPC codes. The PEG algorithm is not necessarily limited to constructing bipartite graphs; it can be extended — although not treated here, in a straightforward manner to constructing arbitrary graphs with large girth by applying the PEG principle enunciated above.

IV. Graph Properties

A randomly-constructed Tanner graph does not guarantee a meaningful lower bound on the girth and the minimum distance. In contrast, a Tanner graph constructed with the PEG algorithm has elegant properties with respect to the girth and the minimum distance.

A. Girth bound

We need the following Lemmas to establish a lower bound on the girth of PEG Tanner graphs.

Lemma 1: Let (V, E) be a regular Tanner graph with d_c and d_s edges incident with each parity-check and symbol node, respectively. If $\mathcal{N}_{s_j}^{l}$ denotes the depth-l neighbor of a symbol node s_j such that $\mathcal{N}_{s_j}^{l} \subset V_c$ and $\mathcal{N}_{s_j}^{l+1} = V_c$, then l is lower bounded by

$$l \geq \lfloor t \rfloor \ , \tag{1}$$

where t satisfies

$$t = \frac{\log\left(md_c - \frac{md_c}{d_s} - m + 1\right)}{\log[(d_s - 1)(d_c - 1)]} - 1, \tag{2}$$

$\lfloor \cdot \rfloor$ indicates the floor of a floating-point number, and m denotes the cardinality of the set V_c of parity-check nodes.

Proof: Consider a depth-l subgraph spreading from any symbol node s_j, $s_j \in V_s$, such that $\mathcal{N}_{s_j}^{l} \subset V_c$ and $\mathcal{N}_{s_j}^{l+1} = V_c$. By definition the depth-0 subgraph contains d_s parity-check nodes, each giving rise to $(d_s - 1)(d_c - 1)$ parity-check nodes in the next round of spreading. Thus, there are $d_s(d_s - 1)(d_c - 1)$ check nodes at depth-1. Similarly, there are $d_s(d_s - 1)^l(d_c - 1)^l$ check nodes at depth l. In principle, duplicate parity-check nodes may occur in the subgraph during the spreading process. Let l' be the largest integer such that

$$d_s + d_s(d_s - 1)(d_c - 1) + \cdots + d_s(d_s - 1)^{l'}(d_c - 1)^{l'} \leq m, \tag{3}$$

which can be simplified to

$$\frac{d_s\left[(d_s - 1)^{l'+1}(d_c - 1)^{l'+1} - 1\right]}{(d_s - 1)(d_c - 1) - 1} \leq m. \tag{4}$$

Let t be the floating-point solution of the equation

$$\frac{d_s\left[(d_s - 1)^{t+1}(d_c - 1)^{t+1} - 1\right]}{(d_s - 1)(d_c - 1) - 1} = m. \tag{5}$$

Then $l \geq l' \geq \lfloor t \rfloor$, After simple manipulations (5) yields (2), which completes the proof. ■

Lemma 2: Let (V, E) be an irregular Tanner graph in which $d_c^{\max}$ and $d_s^{\max}$ are the largest degrees of the degree sequences D_c and D_s, respectively. If $\mathcal{N}_{s_j}^{l}$ denotes the depth-l neighbor of a symbol node s_j such that

$\mathcal{N}_{s_j}^{l} \subset V_c$ and $\mathcal{N}_{s_j}^{l+1} = V_c$, then l is lower bounded by

$$l \geq \lfloor t \rfloor \ , \tag{6}$$

where t satisfies

$$t = \frac{\log\left(md_c^{\max} - \frac{md_c^{\max}}{d_s^{max}} - m + 1\right)}{\log[(d_s^{\max}-1)(d_c^{\max}-1)]} - 1, \tag{7}$$

and m denotes the cardinality of the set V_c of parity-check nodes.

Proof: The proof is identical to that of Lemma 1, except that d_s and d_c are replaced by $d_s^{\max}$ and $d_c^{\max}$, respectively. ■

We now establish a lower bound on the girth of a PEG Tanner graph.

Theorem 1: Let (V, E) be an irregular PEG Tanner graph in which $d_c^{\max}$ and $d_s^{\max}$ are the largest degrees of the degree sequences D_c and D_s, respectively. The girth g of this graph is lower bounded by

$$g \geq 2(\lfloor t \rfloor + 2), \tag{8}$$

where t satisfies

$$t = \frac{\log\left(md_c^{\max} - \frac{md_c^{\max}}{d_s^{max}} - m + 1\right)}{\log[(d_s^{\max}-1)(d_c^{\max}-1)]} - 1, \tag{9}$$

and m denotes the cardinality of the set V_c of parity-check nodes. If (V, E) be a regular PEG Tanner graph with d_c and d_s edges incident with each parity-check and symbol node, $d_c^{\max}$, d_s^{max} in (9) are then replaced by d_c, d_s respectively.

Proof: Suppose that the closed path $(s_{j_0}, c_{i_0}), (c_{i_0}, s_{j_1}), (s_{j_1}, c_{i_1}), (c_{i_1}, s_{j_2}), \ldots, (s_{j_{g/2-1}}, c_{i_{g/2-1}}), (c_{i_{g/2-1}}, s_{j_0})$ is among the ones that provide the shortest cycle in a Tanner graph (V, E) constructed with the PEG algorithm where, without loss of generality, $j_{g/2-1}$ is the largest index among $j_0, j_1, \cdots, j_{g/2-1}$. Then the length of the shortest cycle in the graph, i.e., girth g, is equal to the local girth of symbol node $s_{j_{g/2-1}}$, i.e., $g = g_{j_{g/2-1}}$. As $j_{g/2-1}$ is the largest index, $g_{j_{g/2-1}}$ can be viewed as the girth of the symbol node $s_{j_{g/2-1}}$ in the intermediary graph with edges in the set $E_0 \cup E_1 \cup \cdots \cup E_{j_{g/2-1}}$. Clearly, the edges in the complementary set $E_{j_{g/2}} \cup \cdots \cup E_{n-1}$ have no impact on the local girth of $s_{j_{g/2-1}}$. Recall now the successive procedure in the PEG algorithm for placing edges in the set $E_{j_{g/2-1}}$. By construction the shortest possible cycle passing through symbol node $s_{j_{g/2-1}}$ has length $2(l+2)$ where l corresponds to the depth-l neighbor $\mathcal{N}^l_{s_{j_{g/2-1}}}$ such that $\bar{\mathcal{N}}^l_{s_{j_{g/2-1}}} \neq \varnothing$, but $\bar{\mathcal{N}}^{l+1}_{s_{j_{g/2-1}}} = \varnothing$. It can readily be seen that in our case $l+1 = g/2-1$. Therefore, by making use of Lemma 2 we obtain

$$g/2 - 2 \geq \lfloor t \rfloor \tag{10}$$

which implies $g \geq 2(\lfloor t \rfloor + 2)$, where t is subject to (9). ■

The bound on the girth provides a justification of the effort of the PEG algorithm to keep the check-node degree as uniform as possible. The more uniform the Tanner graph, the smaller the values of $d_s^{\max}$ and $d_c^{\max}$, thereby improving the lower bound.

An upper bound on the girth of a general Tanner graph can readily be derived based on the idea in [13, pp. 81–82].

Lemma 3: Let (V, E) be a regular Tanner graph with d_c and d_s edges incident with each parity-check and symbol node, respectively. The girth g of this graph is upper bounded by

$$g \leq 4\lfloor t \rfloor + 4, \tag{11}$$

where t is given by

$$t = \frac{\log\left[(m-1)(1-\frac{d_s}{d_c(d_s-1)})+1\right]}{\log[(d_c-1)(d_s-1)]}, \tag{12}$$

and m denotes the cardinality of the set V_c of parity-check nodes.

Proof: Consider a depth-l subgraph spreading from any parity-check node c_i, $c_i \in V_c$, such that $\mathcal{N}^l_{c_i} \subseteq V_c$ and $\mathcal{N}^{l+1}_{c_i} = V_c$. Furthermore, let's assume that there are no duplicate parity-check nodes within the subgraph of depth-l but there are duplicate parity-check nodes within depth-$(l+1)$. Under these conditions the girth g must be smaller than or equal to $4l+4$. The value l can be upper bounded based on the fact that the total number of parity-check nodes within the subgraph of depth-l must be smaller than or equal to the cardinality of V_c, namely m . Observe that there is only one parity-check node at depth-0, and there are $d_c(d_s-1)$

parity-check nodes at depth-1, $d_c(d_c-1)(d_s-1)^2$ at depth-2, and so forth. Thus, we have

$$1 + d_c(d_s-1) + d_c(d_c-1)(d_s-1)^2 + \cdots + d_c(d_c-1)^{l-1}(d_s-1)^l \leq m, \tag{13}$$

which reduces to

$$\frac{d_c(d_s-1)[(d_c-1)^l(d_s-1)^l-1]}{(d_c-1)(d_s-1)-1} + 1 \leq m. \tag{14}$$

If t is the solution of the equation

$$\frac{d_c(d_s-1)[(d_c-1)^t(d_s-1)^t-1]}{(d_c-1)(d_s-1)-1} + 1 = m. \tag{15}$$

then $t \geq \lfloor t \rfloor \geq l$. Therefore

$$g \leq 4l + 4 \leq 4\lfloor t \rfloor + 4 \tag{16}$$

where t is defined by (12). ■

Lemma 4: Let (V, E) be a PEG Tanner graph with d_c and d_s edges incident with each parity-check and symbol node, respectively. The girth g of this graph is always larger than or equal to the half of the upper bound.

Proof: To prove the argument, it suffices to show that the lower bound on the girth of a PEG Tanner graph is larger than or equal to the half of the upper bound of a general Tanner graph. The ratio of the lower bound to the upper bound is given by

$$\begin{aligned} \frac{2\left(\left\lfloor \frac{\log\left(md_c - \frac{md_c}{d_s} - m + 1\right)}{\log[(d_s-1)(d_c-1)]} - 1\right\rfloor + 2\right)}{4\left\lfloor \frac{\log\left[(m-1)(1-\frac{d_s}{d_c(d_s-1)})+1\right]}{\log[(d_c-1)(d_s-1)]}\right\rfloor + 4} &= \frac{2\left(\left\lfloor \frac{\log\left(\frac{md_cd_s - md_c - md_s}{d_s} + 1\right)}{\log[(d_s-1)(d_c-1)]}\right\rfloor + 1\right)}{4\left(\left\lfloor \frac{\log\left[\frac{md_cd_s - md_c - md_s}{d_c(d_s-1)} + 1 - (1 - \frac{d_s}{d_c(d_s-1)})\right]}{\log[(d_c-1)(d_s-1)]}\right\rfloor + 1\right)} \\ &\geq \frac{1}{2}, \end{aligned}$$

in which inequality holds because in a nontrivial parity-check code we have $d_c > d_s$ and $d_s > 1$. ■

The asymptotic Erdös-Sachs bound [28] states that a randomly generated regular graph with n vertices and of degree k has girth larger than or equal to $\log_{k-1} n$ with probability approaching 1 as $n \to \infty$. This lower bound is exactly half of an upper bound (asymptotically) that can be obtained in the same manner as we did in the proof of Lemma 3. Therefore, Lemma 4 implies that the girth of a PEG Tanner graph surpasses or meets the Erdös-Sachs bound.

We derive the subsequent two lemmas to tighten the upper bound on the girth of a general Tanner graph.

Lemma 5: Let (V, E) be a regular Tanner graph with d_c and d_s edges incident with each parity-check and symbol node, respectively. The girth g of the graph is upper bounded by

$$g \leq 4\lfloor t \rfloor + 4, \tag{17}$$

where t is given by

$$t = \frac{\log\left[(m-1)(1-\frac{d_s}{d_c(d_s-1)})+1\right]}{\log[(d_c-1)(d_s-1)]}, \tag{18}$$

and m denotes the cardinality of the set V_c of parity-check nodes. Furthermore, if

$$[(d_c-1)(d_s-1)]^{\lfloor t \rfloor} > m - 1 - \frac{d_c(d_s-1)\{[(d_c-1)(d_s-1)]^{\lfloor t \rfloor}-1\}}{(d_c-1)(d_s-1)-1}, \tag{19}$$

then

$$g \leq 4\lfloor t \rfloor + 2. \tag{20}$$

Proof: Consider a depth-l subgraph spreading from any parity-check node c_i, $c_i \in V_c$, such that $\mathcal{N}_{c_i}^l \subseteq V_c$ and $\mathcal{N}_{c_i}^{l+1} = V_c$. Furthermore, let's assume that there are no duplicate parity-check nodes within the subgraph of depth-l but there are duplicate parity-check nodes within depth-$(l+1)$. Under these conditions the girth g must be smaller than or equal to $4l+4$. The value l can be upper bounded $\lfloor t \rfloor$ where t is determined by (18).

Now let's check the total number of symbol nodes residing at depth-$\lfloor t \rfloor$. On the one hand, there are $d_c[(d_c-1)(d_s-1)]^{\lfloor t \rfloor}$ symbol nodes at depth-$\lfloor t \rfloor$, because there are $d_c(d_c-1)^{\lfloor t \rfloor-1}(d_s-1)^{\lfloor t \rfloor}$ distinct parity-check nodes residing at depth-$\lfloor t \rfloor$. On the other hand, since only $m-(1+d_c(d_s-1)+d_c(d_c-1)(d_s-1)^2+\cdots+d_c(d_c-1)^{\lfloor t \rfloor-1}(d_s-1)^{\lfloor t \rfloor})$ distinct check nodes are not present in the depth-$\lfloor t \rfloor$ subgraph and each may lead to d_c symbol nodes, there are at most $md_c-d_c(1+d_c(d_s-1)+d_c(d_c-1)(d_s-1)^2+\cdots+d_c(d_c-1)^{\lfloor t \rfloor-1}(d_s-1)^{\lfloor t \rfloor})$ distinct symbol nodes residing at depth-$\lfloor t \rfloor$. Therefore, if

$$d_c[(d_c-1)(d_s-1)]^{\lfloor t \rfloor} > md_c - d_c(1+d_c(d_s-1)+d_c(d_c-1)(d_s-1)^2+\cdots+d_c(d_c-1)^{\lfloor t \rfloor-1}(d_s-1)^{\lfloor t \rfloor}), \tag{21}$$

then there must be at least one duplicate symbol residing at depth-$\lfloor t \rfloor$, which implies

$$g \leq 4\lfloor t \rfloor + 2. \tag{22}$$

After simple algebra manipulations, (21) yields (19), which completes the proof. ■

Lemma 6: Let (V, E) be a regular Tanner graph with d_c and d_s edges incident with each parity-check and symbol node, respectively, m and n be the cardinality of the set V_c and V_s, respectively. The girth g of the graph is upper bounded by

$$g \leq \min\{g_1, g_2\}, \tag{23}$$

where

$$g_1 = \begin{cases} 4\lfloor t_1 \rfloor + 2 & \text{if } \mathcal{I}_1 = 0 \\ 4\lfloor t_1 \rfloor + 4 & \text{otherwise} \end{cases} \qquad g_2 = \begin{cases} 4\lfloor t_2 \rfloor + 2 & \text{if } \mathcal{I}_2 = 0 \\ 4\lfloor t_2 \rfloor + 4 & \text{otherwise} \end{cases} \tag{24}$$

in which

$$t_1 = \frac{\log\left[(m-1)(1-\frac{d_s}{d_c(d_s-1)})+1\right]}{\log[(d_c-1)(d_s-1)]}, \tag{25}$$

$$t_2 = \frac{\log\left[(n-1)(1-\frac{d_c}{d_s(d_c-1)})+1\right]}{\log[(d_c-1)(d_s-1)]}, \tag{26}$$

and $\mathcal{I}_1$ is equal to 0 if and only if

$$[(d_c-1)(d_s-1)]^{\lfloor t_1 \rfloor} > m - 1 - \frac{d_c(d_s-1)\{[(d_c-1)(d_s-1)]^{\lfloor t_1 \rfloor}-1\}}{(d_c-1)(d_s-1)-1}, \tag{27}$$

and $\mathcal{I}_2$ is equal to 0 if and only if

$$[(d_c-1)(d_s-1)]^{\lfloor t_2 \rfloor} > n - 1 - \frac{d_s(d_c-1)\{[(d_c-1)(d_s-1)]^{\lfloor t_2 \rfloor}-1\}}{(d_c-1)(d_s-1)-1}. \tag{28}$$

Proof: The new results in this lemma compared to Lemma 5 can be obtained by a duality of the proof of Lemma 5. The duality lies in the fact that one can exchange V_c and V_s and the same arguments apply. ■

Fig. 4 depicts both the lower bound on a PEG Tanner graph and the upper bound on a general Tanner graph for regular $d_s = 3, d_c = 6$ codes with varying m (in this case $n = 2m$). It is observed that the lower bound for the entire range of block lengths is above half the upper bound. Compared to Gallager's construction [13, pp. 81-89] for large girth, the PEG construction achieves essentially the same performance on the girth but with less complexity. Recall that for large block lengths the complexity of Gallager's construction becomes prohibitively large owing to the so-called emergency procedure contained therein. Also, the PEG construction can be applied to irregular graphs, whereas Gallager's construction only applies to regular graphs. Later it will be shown that the PEG algorithm can be extended to design linear-time-encoding LDPC codes in a very natural way. Fig. 5 compares the upper and lower bounds for regular $d_s = 4, d_c = 8$ codes. It is worthwhile to point out that the lower bound on the PEG graphs can be exceeded by the PEG algorithm. We observe, through experiments, that with some parameters d_s, d_c and m, the non-greedy PEG variant or look-ahead-enhanced variant can achieve larger girth than the lower bound, those cases of which were also shown in Fig. 4 as small circles.

B. Minimum distance

Assume V_s takes on values from the binary alphabet $\{0, 1\}$ and V_c is a set of simple parity checks (SPC), the Tanner graph then translates into a Gallager's binary LDPC code. The randomly constructed (d_s, d_c)-regular code for $d_s \geq 3$ has a minimum distance that increases linearly with the block length n, for d_s and d_c constant [13]. This is only valid for relatively large block lengths, however, and a code with a low minimum distance will be impaired in its performance at high SNRs. Although finding the minimum distance turns out to be a difficult task, some bounds on the minimum distance of a general Tanner graph have been established in [48]. For a PEG Tanner graph, it is possible to derive a lower bound on the minimum distance in a similar way.

The lower bound on the minimum distance of a PEG Tanner-graph code is based on the properties of the subgraph induced by a minimum weight codeword in the code. Adopting the notation of [48], a symbol node whose associated value in the minimum weight codeword is nonzero will be called an *active* symbol node. The edges incident to active symbol nodes will be called *active* edges, and the check nodes with at least one active incident edge will be called *active* check nodes. Note that in a binary LDPC code, any edge incident to an active symbol node must be active, and any active check node must be incident to *even* active edges.

Lemma 7: Given a symbol-node regular Tanner graph with d_s $(d_s = 2)$ edges incident to each symbol node. Assume the girth of the graph is g. Then its minimum distance $d_{\min}$ satisfies

$$d_{\min} = g/2 \; . \tag{29}$$

Proof: This lemma follows directly from the fact that the shortest cycle in such a graph forms a complete active tree, i.e. a valid codeword, and the fact that it involves exactly $g/2$ active symbol nodes. ■

An application of Lemma 7 is to establish an upper bound on the minimum distance of an LDPC code with irregular Tanner graph. Consider a subgraph of the underlying Tanner graph of an LDPC code, which consists of all degree-2 symbol nodes and their associated edges and check nodes, and assume the girth of this subgraph is $g_{\text{sub}-2}$. It then follows that $g_{\text{sub}-2}/2$ is an effective upper bound of the LDPC code.

Lemma 8 [A Variation of Tanner'81]: Given a regular Tanner graph with d_c edges incident to each check node and d_s $(d_s \geq 3)$ to each symbol node. Assume the girth of the graph is g, $g > 4$. Then its minimum distance $d_{\min}$ satisfies

$$d_{\min} \geq 1 + \frac{d_s[(d_s-1)^{\lfloor (g-2)/4 \rfloor} - 1]}{d_s - 2}. \tag{30}$$

Furthermore, if $g/2$ is even, the lower bound on $d_{\min}$ can be made even tighter:

$$d_{\min} \geq 1 + \frac{d_s[(d_s-1)^{\lfloor (g-2)/4 \rfloor} - 1]}{d_s - 2} + (d_s-1)^{\lfloor (g-2)/4 \rfloor}. \tag{31}$$

Proof: Consider a subgraph induced by a minimum-weight codeword in the graph. The subgraph consists of n' active symbol nodes, m' active check nodes, and all (active) edges incident on n' active symbol nodes. Note that $d_{\min}$ is exactly equivalent to the number of active symbol nodes, n'. Now start a depth-l tree from any active symbol node as shown in Fig. 6, where $l = \lfloor (g-2)/4 \rfloor$. Any symbol node, check node, and edge resident in the tree must be active and mutually different, except that the last row of check nodes does not have to meet the condition of being mutually different if $g/2$ is odd. The number of active edges incident to an active check node can in principle be 2, 4, ..., etc.; however only a degree-2 (active) check node is considered to generate a minimum expansion for the number of active symbol nodes, which in turn gives rise to a lower bound on the minimum distance. There is only one active symbol node at depth 0, and there are d_s active symbol nodes at depth 1, $d_s(d_s-1)$ at depth 2, and similarly $d_s(d_s-1)^{l-1}$ at depth l. As the girth of the underlying graph is given by g, it is clear that all active symbol nodes within this depth-$\lfloor (g-2)/4 \rfloor$ tree must be mutually different, otherwise there always exists a closed path shorter than g, leading to a paradox. Thus $d_{\min}$ is lower bounded by the number of active symbol nodes within the depth-$\lfloor (g-2)/4 \rfloor$ tree, that is

$$\begin{aligned} d_{\min} &\geq 1 + d_s + d_s(d_s-1) + \cdots + d_s(d_s-1)^{l-1} \\ &= 1 + \frac{d_s[(d_s-1)^l - 1]}{d_s - 2} \\ &= 1 + \frac{d_s[(d_s-1)^{\lfloor (g-2)/4 \rfloor} - 1]}{d_s - 2}, \end{aligned} \tag{32}$$

which proves the first claim. If $g/2$ is even, the tree can proceed one step beyond depth $\lfloor (g-2)/4 \rfloor$, resulting in $d_s(d_s-1)^l$ active symbol nodes. These $d_s(d_s-1)^l$ active nodes are, however, not necessarily mutually different; a lower bound on the number of different active symbol nodes is $d_s(d_s-1)^l$ divided by d_s,[2] yielding the additive term $(d_s-1)^{\lfloor (g-2)/4 \rfloor}$. ■

Tanner derived a minimum-distance lower bound for general bipartite graphs [1], which, in the case that all subcodes are simple parity checks, turns out to be

$$d_{\min} \geq \begin{cases} \frac{2\left[(d_s-1)^{(g-2)/4} - 1\right]}{d_s-2} + \frac{2}{d_s}[d_s-1]^{(g-2)/4} & \text{for } g/2 \text{ odd} \quad , \\ \frac{2\left[(d_s-1)^{g/4} - 1\right]}{d_s-2} & \text{for } g/2 \text{ even} \ . \end{cases} \tag{33}$$

Comparing with (33), (30) and (31) are slightly stronger as in general $d_s \geq 3$.

Note that for the case of $g = 6$, (30) reduced to the conventional bound $d_{\min} \geq 1 + d_s$.[3] One might postulate that the PEG construction does not guarantee the uniformity of the check-node-degree sequence, although the resulting PEG Tanner graph is nearly check-node-degree uniform.[4] The above lower bound can easily be extended to those cases where the degree sequence of check nodes is not uniform.

Theorem 2: Given a symbol-node-uniform PEG Tanner graph with d_s ($d_s \geq 3$) edges incident to each symbol node, let $d_c^{\max}$ be the largest degree of check nodes. The minimum distance $d_{\min}$ satisfies

$$d_{\min} \geq \begin{cases} 1 + \frac{d_s[(d_s-1)^{\lfloor (g-2)/4 \rfloor} - 1]}{d_s-2} & \text{if } g/2 \text{ is odd} \\ 1 + \frac{d_s[(d_s-1)^{\lfloor (g-2)/4 \rfloor} - 1]}{d_s-2} + (d_s-1)^{\lfloor (g-2)/4 \rfloor} & \text{if } g/2 \text{ is even,} \end{cases} \tag{34}$$

[2] As each symbol node has at most d_s edges.

[3] This case was observed and pointed out to us by M. P. C. Fossorier.

[4] Our study of short codes indicates that uniformity of check nodes can actually be achieved through multiple trials.

where the girth g is lower bounded by

$$g \geq 2(\lfloor t \rfloor + 2), \tag{35}$$

in which

$$t = \frac{\log(md_c^{\max} - \frac{md_c^{\max}}{d_s} - m + 1)}{\log[(d_s - 1)(d_c^{\max} - 1)]} - 1. \tag{36}$$

Proof: The proof follows directly from Theorem 1 and Lemma 8. ■

Note that the above bound on the minimum distance is still a weak bound, although it always furnishes a meaningful bound on graphs having a large girth. There are two reasons for the weakness of this lower bound. The first is the assumption that all active check nodes are satisfied by exactly two symbol nodes, which weakens the estimate of the minimum distance. The second is that the condition that the last row of check nodes must be satisfied with additional active symbol nodes has not been taken into account.

The above theorem can readily be extended and applied to analyze the minimum distance of irregular PEG Tanner-graph codes. As a working example, suppose a simple symbol-node-degree distribution defined as $0.8x^2 + 0.2x^{20}$, and n is chosen as 3000 and m as 2000. As specified by the PEG construction, the edges of degree-2 symbol nodes are established first, and the resulting girth of the subgraph containing all degree-2 symbol nodes turns out to be 44, which means that the minimum distance involving only degree-2 symbol nodes is 22, a result derived directly from the argument that the shortest cycle involving only degree-2 symbol nodes leads to a minimum weight codeword. Next, the edges of degree-20 symbol nodes are constructed, and finally the global girth is found to be 6. By applying (32), an active tree stemming from a degree-20 symbol node will contain at least 21 symbol nodes, which means that a minimum codeword involving a degree-20 symbol node will have a weight of 21 at least. Therefore one can know that the minimum distance of the resultant PEG Tanner-graph code is lower-bounded by 21, or more specifically, in our example it can be either 21 or 22. This argument justifies the effort in the PEG construction to order the symbol degree sequence D_s in a *nondecreasing* order and hence construct the edges of lower-degree symbol nodes earlier, improving the minimum distance induced by the subgraph of lower-degree symbol nodes, particularly degree-2 symbol nodes.

V. Asymptotic Analysis of Ensemble PEG Codes

In the preceding section, we derived a lower bound on the minimum distance of binary LDPC codes underlying PEG Tanner graphs that is mainly based on the girth property. However, the analysis resting solely on the girth property seems insufficient to demonstrate that families of codes are asymptotically good. Next, we shall prove that the ensemble of PEG codes is indeed asymptotically good, i.e., its distance distribution asymptotically approaches that of the equiprobable ensemble of parity-check codes. In order to simplify the analysis, we will extend (to say exactly, relax) the PEG algorithm such that the PEG Tanner graphs degrade to a general sparse random ensemble and in turn all variants of PEG graphs can be visualized as expurgated random ensembles in which bad graphs with short cycles and/or small minimum distance codewords are avoided.

The equiprobable random ensemble of parity-check codes has been analyzed in [13]; it is a capacity-approaching family of codes for many practical channels, specifically binary-symmetric stationary ergodic channels [57]. The equiprobable random ensemble of parity-check codes of rate R and block length n is defined as the ensemble in which the $m \times n$, $m = n(1-R)$, parity-check matrix is filled with statistically independent equiprobable binary digits $\{0, 1\}$. Averaged over the equiprobable ensemble of parity-check codes, the minimum distance is a random variable whose distribution function can be represented by the following lemma.

Lemma 9 [Gallager'63]: Over the equiprobable ensemble of parity-check codes of length n and rate R, the minimum distance distribution function $\Pr(D \leq \delta n)$ is given by

$$\Pr(D \leq \delta n) = \sum_{j=1}^{\delta n} \binom{n}{j} 2^{-n(1-R)} \tag{37}$$

Proof: See [13, Eq. (2.6), p. 13]. ■

Now we describe the procedure to relax the PEG construction to facilitate simple analysis: Recalling that in the PEG algorithm, we partition the entire set of check nodes into two subsets whenever a new edge of a symbol node (s_j) is being added: one is the *prohibitive* subset, on which if the new edge is placed, there exits a short cycle — this check-node set corresponds to the depth-l neighbors of the symbol node $\mathcal{N}_{s_j}^l$; the other is the *feasible* subset from which each element is selected with *equal* probability to get connected with the symbol node, which is the complement set of $\mathcal{N}_{s_j}^{l}$. If we relax the greedy PEG algorithm such that the prohibitive subset only contains the existing connected parity checks, i.e., the expanding tree halts at depth-0 ($l = 0$), then the PEG construction reduced to a simple random construction,[5] each column in the parity-check matrix containing exactly d_s 1's. It is clear that the PEG codes are a particular subset of these random matrices, from which bad codes (graphs) containing small minimum distance (short cycles) are successfully avoided.

It still turns out to be a tedious task to analyze the distance distribution function of the random construction discussed previously. One can relax it further by means of removing the constraint that the 1's in each column of parity-check matrix be exactly d_s, instead, one can define a new ensemble of parity-check matrices, i.e. sparse random matrices H in which each entry in a column takes on 1 with probability p and 0 with $1 - p$, $p = d_s/m, 0 < p < 1/2$, p is called *density*. Such an ensemble is naturally a broader random construction and contains the previous random construction as its own subset. Note that this relaxed random construction with density p is asymptotically the same as the old one, and also it is a weaker extension since it includes some bad codes (graphs) again.

In summary, we obtain a series of relaxations of PEG constructions. These PEG Tanner graphs can naturally be viewed as *expurgated* ensembles of the sparse random graphs with density p, from which bad graphs having short cycles and/or small minimum distance are effectively precluded. Our asymptotic analysis of PEG constructions relies on an extreme degradation of PEG Tanner graphs, i.e, the sparse random matrix H with density p described previously.

Lemma 10: Let H be a sparse random $n(1 - R) \times n$ matrix over GF(2), with a density of p, $0 < p < 1/2$. The minimum distance distribution function Pr$(D \leq \delta n)$ is given by

$$\Pr(D \leq \delta n) = \sum_{j=1}^{\delta n} \binom{n}{j} \left[\frac{2}{1 + (1 - 2p)^j} \right]^{-n(1-R)} \tag{38}$$

Proof: Fix some nonzero vector $V = [v_0, v_1, \ldots, v_{n-1}] \in [\mathrm{GF}(2)]^n$, with exactly j nonzero coordinates, i.e., the Hamming weight of this vector is j. Without loss of generality, assume that the first j coordinates of V are nonzero. Let P_j be the probability that $\sum_{i=1}^{j} v_i h_{k,i} = 0$, where each $h_{k,i}$ is the entry of k-th row of the matrix

[5] This ensemble is equivalent to that of Section II-A.(3) in [15].

H and is thus chosen according to the uniform distribution of p. As there is a total of $n(1-R)$ independent check equations, the probability that V is a valid codeword is given by $P_j^{n(1-R)}$.

Given that the v_i's are fixed, the following recursion holds for P_j.

$$P_j = P_{j-1}(1-p) + (1-P_{j-1})p \ , \tag{39}$$

with the initial condition $P_0 = 1$. Denote $Q_j = P_j - 1/2$. It follows from (39) that

$$Q_j = Q_{j-1}(1-2p) \ , \tag{40}$$

with $Q_0 = 1/2$. Hence we obtain

$$Q_j = \frac{1}{2}(1-2p)^j \ , \tag{41}$$

and, consequently,

$$P_j = \frac{1}{2} + \frac{1}{2}(1-2p)^j \ . \tag{42}$$

The number of vectors V with exactly j nonzero coordinates is given by $\binom{n}{j}$; the minimum distance distribution function $\Pr(D \leq \delta n)$ can then be written as

$$\begin{aligned} \Pr(D \leq \delta n) &= \sum_{j=1}^{\delta n} \binom{n}{j} P_j^{n(1-R)} \\ &= \sum_{j=1}^{\delta n} \binom{n}{j} \left[\frac{1}{2} + \frac{1}{2}(1-2p)^j\right]^{n(1-R)} \\ &= \sum_{j=1}^{\delta n} \binom{n}{j} \left[\frac{2}{1+(1-2p)^j}\right]^{-n(1-R)} . \end{aligned}$$

This completes the proof. ■

Lemma 11: Let H be a sparse random $n(1-R) \times n$ matrix over GF(2), with a density of p, $0 < p < 1/2$. The minimum distance grows at least linearly with block length n.

Proof: It suffices to prove that, given any positive p, there exists a positive number ϵ such that $\Pr(d_{\min} \leq$

$\epsilon n) \xrightarrow{exp} 0$. Starting with (38), we have

$$\begin{aligned}\Pr(d_{\min} \leq \epsilon n) &= \sum_{j=1}^{\epsilon n} \binom{n}{j} \left[\frac{2}{1+(1-2p)^j}\right]^{-n(1-R)} \\ &\leq \sum_{j=1}^{\epsilon n} \binom{n}{j} \left[\frac{2}{1+(1-2p)}\right]^{-n(1-R)} .\end{aligned}$$

Recall the following inequality [13, (2.7), p. 13]

$$\sum_{j=1}^{\epsilon n} \binom{n}{j} \leq \binom{n}{\epsilon n} \frac{1-\epsilon}{1-2\epsilon} . \tag{43}$$

We obtain

$$\Pr(d_{\min} \leq \epsilon n) \leq \binom{n}{\epsilon n} \frac{1-\epsilon}{1-2\epsilon} \left[\frac{1}{1-p}\right]^{-n(1-R)} . \tag{44}$$

In view of the following bound on $\binom{n}{\epsilon n}$ [60, p. 530]

$$\frac{1}{\sqrt{8n\epsilon(1-\epsilon)}} 2^{nH_2(\epsilon)} \leq \binom{n}{\epsilon n} \leq \frac{1}{\sqrt{2\pi n\epsilon(1-\epsilon)}} 2^{nH_2(\epsilon)} . \tag{45}$$

We have

$$\Pr(d_{\min} \leq \epsilon n) \leq \frac{1}{1-2\epsilon} \sqrt{\frac{1-\epsilon}{2\pi n\epsilon}} 2^{n[H_2(\epsilon)-(1-R)\log_2 \frac{1}{1-p}]} . \tag{46}$$

As $H_2(\epsilon)$ is monotonically increasing within $(0, 1/2)$, there must be a small positive number ϵ, for any positive $p \in (0, 1/2)$, that satisfies

$$H_2(\epsilon) - (1-R)\log_2 \frac{1}{1-p} \leq 0 .$$

Thus,

$$\Pr(d_{\min} \leq \epsilon n) \longrightarrow 0 \qquad \text{exponentially with } n . \tag{47}$$

■

Theorem 3: Given a sparse random matrix with density p, i.e., a degradation of a PEG ensemble of uniform-symbol-degree d_s ($d_s \geq 3$), its minimum distance distribution function approaches closely that of the equiprobable ensemble of parity-check codes, for sufficiently large n, fixed R and fixed p, where

$0 < p < 1/2, p = d_s/[n(1-R)]$.

Proof: In view of (47), the minimum distance distribution function of the sparse random Tanner graph with fixed density p can be rewritten as

$$\begin{aligned} \Pr(D \le \delta n) &= \Pr(\epsilon n \le D \le \delta n) \\ &= \sum_{j=\epsilon n}^{\delta n} \binom{n}{j} \left[\frac{2}{1+(1-2p)^j}\right]^{-n(1-R)} \\ &\le \sum_{j=\epsilon n}^{\delta n} \binom{n}{j} \left[\frac{2}{1+(1-2p)^{\epsilon n}}\right]^{-n(1-R)} \\ &= \sum_{j=\epsilon n}^{\delta n} \binom{n}{j} 2^{-n(1-R)}[1+(1-2p)^{\epsilon n}]^{n(1-R)} \ . \end{aligned} \tag{48}$$

To prove (48) approaches asymptotically (37), it remains to show

$$\lim_{n\to\infty} [1+(1-2p)^{\epsilon n}]^{n(1-R)} = 1 \ . \tag{49}$$

Using $1 + x \le e^x$ twice we get

$$\begin{aligned} [1+(1-2p)^{\epsilon n}]^{n(1-R)} &\le \left[1+e^{-2p\epsilon n}\right]^{n(1-R)} \\ &\le e^{n(1-R)e^{-2p\epsilon n}} \ . \end{aligned} \tag{50}$$

Therefore

$$\begin{aligned} \lim_{n\to\infty} [1+(1-2p)^{\epsilon n}]^{n(1-R)} &\le \lim_{n\to\infty} e^{n(1-R)e^{-2p\epsilon n}} \\ &= e^{\lim_{n\to\infty} n(1-R)e^{-2p\epsilon n}} \\ &= e^0 = 1 \ . \end{aligned} \tag{51}$$

Combining the fact that $[1+(1-2p)^{\epsilon n}]^{n(1-R)} \ge 1$, we obtain (49). ■

Because the maximum-likelihood decoding performance of a linear code depends uniquely on the minimum distance distribution function, this theorem effectively states that the binary low-density parity codes underlying PEG Tanner graphs are asymptotically good, i.e., capable of achieving reliable communication at rates up to the capacity of the binary-symmetrical stationary ergodic channels, *when optimally decoded*.

We can prove that the binary PEG codes also attain the (asymptotic) Gilbert–Varshamov bound with high probability. A code with block length n and rate R satisfies the Gilbert–Varshamov minimum-distance bound if the relative minimum distance $\delta = d_{\min}/n$ satisfies

$$H_2(\delta) = 1 - R \,, \tag{52}$$

where $H_2(\delta)$ is the binary entropy function

$$H_2(\delta) = -\delta \log_2 \delta - (1-\delta)\log_2(1-\delta) \,. \tag{53}$$

Theorem 4: Given a sparse random matrix with density p, i.e., a degradation of the PEG ensemble of uniform-symbol-degree d_s ($d_s \geq 3$), the binary LDPC code underlying this graph attains the Gilbert–Varshamov minimum-distance bound with high probability.

Proof: First, recall the following inequality [13, (2.7), p. 13]

$$\sum_{j=\epsilon n}^{\delta n} \binom{n}{j} \leq \sum_{j=1}^{\delta n} \binom{n}{j} \leq \binom{n}{\delta n} \frac{1-\delta}{1-2\delta} \,. \tag{54}$$

By inserting (54) in (48), we obtain

$$\Pr(D \leq \delta n) \leq \binom{n}{\delta n} \frac{1-\delta}{1-2\delta} 2^{-n(1-R)} \left[1 + (1-2p)^{\epsilon n}\right]^{n(1-R)} \,. \tag{55}$$

Recall the following bound on $\binom{n}{\delta n}$ [60, p. 530]

$$\frac{1}{\sqrt{8n\delta(1-\delta)}} 2^{nH_2(\delta)} \leq \binom{n}{\delta n} \leq \frac{1}{\sqrt{2\pi n\delta(1-\delta)}} 2^{nH_2(\delta)} \,. \tag{56}$$

Thus (55) is upper bounded by

$$\Pr(D \leq \delta n) \leq \frac{1}{1-2\delta} \sqrt{\frac{1-\delta}{2\pi n \delta}} 2^{nH_2(\delta)} 2^{-n(1-R)} \left[1 + (1-2p)^{\epsilon n}\right]^{n(1-R)} \,. \tag{57}$$

Using (49) for sufficiently large n, we obtain

$$\Pr(D \leq \delta n) \leq \frac{1}{1-2\delta} \sqrt{\frac{1-\delta}{2\pi n \delta}} 2^{n[H_2(\delta) - (1-R)]} \,. \tag{58}$$

For increasing n, the bound of $\Pr(D \leq \delta n)$ as a function of δ approaches a step function with the step at δ_0 such that $H_2(\delta_0) = (1-R)$. In particular if $\delta = \delta_0$, then

$$\Pr(D \leq \delta_0 n) \leq \begin{cases} \frac{1}{1-2\delta_0} \sqrt{\frac{1-\delta_0}{2\pi n \delta_0}} & \\ = 0 & \text{for } n \to \infty \end{cases} \tag{59}$$

Furthermore, recognizing that $H_2(\delta)$ is monotonically increasing with $\delta \in (0, 1/2)$, we conclude that for any $\epsilon > 0$, the probability for which $D < n(\delta_0 - \epsilon)$ exponentially approaches 0 with n. This completes the proof. ■

Next we shall take a broader view and investigate the average weight spectrum of a random ensemble of LDPC codes over Galois field GF(q). We are particularly interested in the average Hamming weight spectrum when LDPC codes over GF(q) are applied to binary channels [47]. Such codes are defined as follows: Given some p such that $0 < p < 1/2$, we choose the entries of each column in the matrix H independently, so that 0 is attained with probability $1 - p$, and each nonzero element in GF(q) with equal probability $p/(q-1)$. Note that if $q = 2$, this reduces to the binary random ensemble discussed earlier.

Define the symbol distance between two codewords in a linear code over GF(q) as the number of positions in which the codewords differ. The symbol weight of a codeword is the number of nonzero digits or the distance from the all-zero codeword. The symbol distance function $N_s(j)$ of a code is defined as the number of codewords of symbol weight j. It follows from the group properties of such a code that $N_s(j)$ is the number of codewords at distance j from any given codeword.

Theorem 5: Let $\overline{N_s(j)}$ be the average number of codewords of symbol weight j in a code averaged over the sparse random ensemble H based on some p and defined over GF(q). The parity-check codes are of symbol length $n/\log_2 q$ (totally n bits in binary length) and rate R. Then for j, $0 < j \le n/\log_2 q$,

$$\overline{N_s(j)} = \alpha \binom{\frac{n}{\log_2 q}}{j} (q-1)^j \left[\frac{q-1}{q} \left(1 - \frac{qp}{q-1}\right)^j + \frac{1}{q} \right]^{\frac{n(1-R)}{\log_2 q}}, \tag{60}$$

where α is the normalizing constant such that $\sum_j \overline{N_s(j)} = 2^{nR}$ and $\overline{N_s(0)} = 1$.

Proof: Fix some nonzero vector $V = [v_0, v_1, \ldots, v_{\frac{n}{\log_2 q}-1}] \in [\mathrm{GF}(q)]^{\frac{n}{\log_2 q}}$, with exactly j non-zero coordinates, i.e., a symbol weight of j. Without loss of generality, assume that the first j coordinates of V are nonzero. Let P_j be the probability that $\sum_{i=1}^{j} v_i h_{k,i} = 0$, where each $h_{k,i}$ is the entry of the k-th row of the matrix H and is thus chosen according to the uniform distribution of $p/(q-1)$ among nonzero elements of GF(q). As there is a total of $n(1-R)/\log_2 q$ independent check equations, the probability that V is a valid codeword is given by $P_j^{\frac{n(1-R)}{\log_2 q}}$.

Given that the v_i's are fixed, the following recursion [6] holds for P_j.

$$P_j = P_{j-1}(1-p) + \frac{(1-P_{j-1})p}{q-1} \ , \tag{61}$$

with the initial condition $P_0 = 1$. Let $Q_j = P_j - 1/q$. It follows from (61) that

$$Q_j = Q_{j-1}(1 - \frac{qp}{q-1}) \ , \tag{62}$$

with $Q_0 = (q-1)/q$. Hence we obtain

$$Q_j = \frac{q-1}{q}(1 - \frac{qp}{q-1})^j \ , \tag{63}$$

and, consequently,

$$P_j = \frac{q-1}{q}(1 - \frac{qp}{q-1})^j + \frac{1}{q} \ . \tag{64}$$

The number of vectors V with exactly j nonzero coordinates is given by

$$\binom{\frac{n}{\log_2 q}}{j}(q-1)^j \ .$$

The average number of codewords of symbol weight j is then given by

$$\overline{N_s(j)} = \alpha \binom{\frac{n}{\log_2 q}}{j}(q-1)^j \left[\frac{q-1}{q}(1 - \frac{qp}{q-1})^j + \frac{1}{q}\right]^{\frac{n(1-R)}{\log_2 q}} \ . \tag{65}$$

Finally note that there is only one all-zero codeword and that the number of all codewords should sum up to $q^{\frac{nR}{\log_2 q}} = 2^{nR}$. This completes the proof. ∎

Lemma 12: Let $\overline{N(k)}$ be the average number of codewords of Hamming weight k in a code averaged over the sparse random ensemble H based on some p and defined over GF(2). Then for $0 < k \le n$,

$$\overline{N(k)} = \alpha \binom{n}{k} \left[\frac{1}{2}(1-2p)^k + \frac{1}{2}\right]^{n(1-R)} \ , \tag{66}$$

where α is the normalizing constant such that $\sum_k \overline{N(k)} = 2^{nR}$ and $\overline{N(0)} = 1$.

Proof: This is obtained by inserting $q = 2$ into (60). ∎

[6]This recursion has been used to analyze the rank properties of a sparse random $n \times n$ matrix over GF(q) in [58].

Lemma 13: Let $\overline{N(k)}$ be the average number of codewords of Hamming weight k in a code averaged over the equiprobable random ensemble H defined over GF(2). Then for $0 < k \leq n$,

$$\overline{N(k)} = \binom{n}{k} 2^{-n(1-R)} . \tag{67}$$

Proof: This is obtained by inserting $q = 2, p = 1/2$ into (60). ■

Theorem 6: Let $\overline{N(k)}$ be the average number of codewords of Hamming weight k in a code averaged over the sparse random ensemble H based on some p and defined over GF(q), $q > 2$. The parity-check codes are of symbol length $n/\log_2 q$ (n in binary bits) and rate R. Then for $0 < k \leq n$,

$$\overline{N(k)} = \beta \sum_{j \geq k/\log_2 q}^{\min\{k, n/\log_2 q\}} \overline{N_s(j)} \cdot \frac{\binom{j \log_2 q}{k} \left[1 - \left(1 - \frac{k}{j \log_2 q}\right)^{\log_2 q}\right]^j}{(q-1)^j} , \tag{68}$$

where β is the normalizing factor such that $\sum_k \overline{N(k)} = 2^{nR}$, and $\overline{N(0)} = 1$.

Proof: The sparse random ensemble of parity-check codes over GF(q) of length $n/\log_2 q$ and rate R can be visualized as an ensemble of binary codes of length n and rate R when used over binary channels, thus there are 2^{nR} codewords in total, leading to $\sum_k \overline{N(k)} = 2^{nR}$. It is clear that the number of all-zero codewords is unique, i.e. $\overline{N(0)} = 1$.

Suppose a valid codeword has symbol weight of j, $0 < j \leq n/\log_2 q$. We need to evaluate the probability that this codeword contains k, $j \leq k \leq j \log_2 q$, 1's in its binary representation. Without loss of generality assume that each nonzero symbol in the codeword is uniformly distributed among the nonzero elements of GF(q). In total, there are $(q-1)^j$ patterns the j nonzero symbols can assume. On the other hand, there are $\binom{j \log_2 q}{k}$ possibilities that the 1's in their binary representation sum up to exactly k, of which only a fraction, i.e. $[1 - (1 - \frac{k}{j \log_2 q})^{\log_2 q}]^j$, satisfy the condition that all the j symbols be nonzero. That is to say, the probability that a codeword of symbol weight j contains k 1's in its binary representation is given by

$$\frac{\binom{j \log_2 q}{k}[1 - (1 - \frac{k}{j \log_2 q})^{\log_2 q}]^j}{(q-1)^j} .$$

From this, we obtain (68). ■

Figure 7 shows the Hamming weight spectra of binary random ensembles with different block lengths but keeping the same density and rate. As the block length increases, the Hamming weight spectrum of an LDPC code closely approaches that of an equiprobable random ensemble — the binomial distribution. This turns out to be a straightforward confirmation of Theorem 3.

Figure 8 shows the Hamming weight spectra of binary random ensembles with different densities, from $p = 0.01$ to $p = 0.5$. It can be seen that, as the density increases, an LDPC code closely approaches the equiprobable binary random code, which is the best code we can imagine but unfortunately is computationally infeasible to decode [59]. In contrast, an LDPC code can be decoded iteratively with the sum-product algorithm whose complexity is essentially linear to the block length. On the one hand, one would like to increase the density of an LDPC code such that it can approach the best code as closely as possible; on the other hand, iterative decoding requires the density to be small enough so that the sub-optimum decoding performance comes close to that of the optimum decoding. These two conflicting requirements must be well balanced, and pose significant design challenges, particularly in the case of finite block lengths.

Figure 9 shows the Hamming weight spectra of low-density random ensembles over GF(2), GF(2^2), GF(2^3), GF(2^4), GF(2^6), and GF(2^{10}) with the same block length of n, $n = 600$ bit, and the same density of p, $p = 0.01$. Also plotted is the spectrum of the equiprobable binary random ensemble. As the size of the Galois field increases, the Hamming weight spectrum closely approaches that of the equiprobable binary random ensemble, while keeping the same (binary) block length, the rate, and the density of the matrix (Tanner graph). Generally speaking, the sparser the parity-check matrix (Tanner graph), the smaller the deleterious effect of cycles in the graph and the closer the iterative decoding approaches maximum-likelihood performance. Figure 9 implies that it is possible to fix the density of the Tanner graphs, but improve the Hamming weight distribution by using higher-order codes, thereby achieving a better performance in the case of finite block lengths.

VI. Irregular PEG Tanner Graphs

For very long block lengths, the authors of [16, 17] prove a concentration theorem that states that in the limit large random graphs can be assumed to be effectively cycle-free. This allows the calculation of precise convergence thresholds under the sum–product decoding algorithm via the density evolution approach. An ensemble of random codes with optimum degree-distribution pairs can then be carefully chosen to optimize its threshold. This approach has been used to find good irregular LDPC codes based on the random construction that exhibit a performance extremely close to the Shannon limit on typical memoryless channels.

However, two issues complicate the analysis and design of codes having short block lengths. The first is that the concentration theorem applies only to the asymptotic case; for short block lengths, random graphs experience significant deviation as pointed out in [49]. The second is that the cycle-free assumption is no longer valid, which is a complication the density evolution cannot handle adequately.

In this section, we briefly describe an empirical Monte-Carlo approach using a variant of the "downhill simplex" optimization technique to design symbol-node-degree distributions. The resulting degree distribution is used to optimize a LDPC code with specific parameters n and m whose Tanner graph is constructed by the PEG algorithm.

The "downhill simplex" algorithm in general is a nonlinear unconstrained optimization procedure that operates on a multidimensional simplex consisting of several vertices [40, 41]. Here we adopt a simple variant of the "downhill simplex" method that can handle constraints. Each vertex represents a feasible symbol-node-degree distribution, and its cost function is defined as the block-error rate at a E_b/N_0 threshold. The worst vertex is replaced by a better vertex according to three basic operations: contraction, expansion, and reflection, resulting in a new simplex. This step is repeated until the diameter of the simplex is less than a preselected tolerance.

A. Constraints

The optimization parameters are the symbol-node-degree distribution that must be a probability vector. Assume there are l nonzero coefficients in the symbol-node-degree distribution $\Lambda(x)$, and define

$$\Lambda(x) = \sum_{i=1}^{l} \Lambda_i x^{d_i},$$

where d_i is the preselected degree of the i-th nonzero entry in the degree distribution. Then we have the following constraints:

$$\sum_{i=1}^{l} \Lambda_i = 1.0, \tag{69}$$

where $0.0 < \Lambda_i < 1.0$, for $i = 1, 2, \ldots, l$. To improve the searching efficiency, we can reduce this l-dimensional searching problem to a $(l-1)$-dimensional one, i.e., we form a constrained optimization problem with an $(l-1)$-dimensional parameter vector $\mathbf{\Lambda} = (\Lambda_1, \Lambda_2, \ldots, \Lambda_{l-1})$ under the following inequality constraints:

$$\begin{cases} 0.0 < \Lambda_i < 1.0 & \text{for } i = 1, 2, \cdots, l-1 \\ 0.0 < \sum_{i=1}^{l-1} \Lambda_i < 1.0. \end{cases} \tag{70}$$

The l-th coefficient of $\Lambda(x)$ is computed by

$$\Lambda_l = 1.0 - \sum_{i=1}^{l-1} \Lambda_i.$$

B. Cost function

We need a cost function to assess the quality of a valid symbol-node-degree distribution $\mathbf{\Lambda}$, and then minimize this function by manipulating the simplex. The most natural cost function is the block-error rate at a certain SNR per bit (E_b/N_0). In this approach we first construct a PEG graph in accordance with $\mathbf{\Lambda}$, and then evaluate the average block-error rate at a certain E_b/N_0 using Monte Carlo simulations with a prescribed number of blocks. The value of E_b/N_0 is initialized with a relatively large number, then gradually decreases in small steps whenever a zero block-error rate is encountered. The final E_b/N_0, at which there is no PEG graph with a symbol-node-degree distribution $\mathbf{\Lambda}$ having a zero block-error rate, is called the E_b/N_0 threshold.

C. Constrained "downhill simplex" search method

The method we have adopted in designing irregular PEG Tanner graphs is a constrained "downhill simplex" nonlinear optimization procedure that uses only cost function evaluations. A simplex in our setting is a geometrical manifold consisting of $2(l-1)$ vertices, with each vertex representing a valid symbol-node-degree distribution. We summarize the algorithm as follows:

The Constrained "Downhill Simplex"Algorithm:

Step 1: Initializing a simplex

1a. Find a vertex $\mathbf{\Lambda}^1$ that meets all constraints. This can easily be done manually.

1b. Determine the other $2l-3$ vertices according to:

1. **for** $j = 2$ to $2l-2$

    for $i = 1$ to $l-1$

    $\mathbf{\Lambda}_i^j \longleftarrow \mathrm{Random}[0,1]$

    end

end

where $\mathbf{\Lambda}_i^j$ is the i-th element of the j-th vertex $\mathbf{\Lambda}^j$ in the simplex, and $\mathrm{Random}[0,1]$ is a uniform [0, 1] random number generator.

2. Check whether $\sum_i^{l-1} \mathbf{\Lambda}_i^j < 1.0$ in the sequence of $j = 1, 2, \ldots, 2l-2$. Suppose the first j vertices all meet this constraint, and the $(j+1)$-th does not, then

$$\mathbf{\Lambda}^{j+1} = (\mathbf{\Lambda}^{j+1} + \mathbf{t})/2,$$

where

$$\mathbf{t} = \frac{1}{j}\sum_{k=1}^{j} \mathbf{\Lambda}^k.$$

Repeat this check procedure until all vertices $\mathbf{\Lambda}^j$, $j = 1, 2, \ldots, 2l-2$, satisfy the constraint.

1c. Evaluate cost functions of the vertices in the simplex, i.e., the block-error rates P_e^j of codes associated with irregular PEG graphs generated with $\mathbf{\Lambda}^j$, where $j = 1, 2, \ldots, 2l-2$.

Step 2: Forming a new simplex

2a. Determine the worst vertex P_e^W and the second worst vertex P_e^w,

$$P_e^W \stackrel{def}{=} \max_{1 \le j \le 2l-2} P_e^j$$

and

$$P_e^w \stackrel{def}{=} \max_{1 \le j \le 2l-2, j \ne W} P_e^j.$$

2b. Compute the reflection vertex $\mathbf{\Lambda}^r$ of the worst vertex $\mathbf{\Lambda}^W$ in the current simplex.

$$\mathbf{\Lambda}^r = (1+\alpha)\mathbf{\Lambda}^R - \alpha\mathbf{\Lambda}^W,$$

in which

$$\mathbf{\Lambda}^R = \frac{1}{2l-3} \sum_{j=1, j \neq W}^{2l-2} \mathbf{\Lambda}^j$$

2c. Search a new vertex $\mathbf{\Lambda}^n$ based on $\mathbf{\Lambda}^r$ to replace the worst vertex $\mathbf{\Lambda}^W$, forming a new simplex. The search procedure is as follows.

1. **for** i=1 to $l-1$

   if $\mathbf{\Lambda}_i^r < 0.0$ **then** $\mathbf{\Lambda}_i^r = \mathbf{\Lambda}_i^r + \delta$

   if $\mathbf{\Lambda}_i^r > 1.0$ **then** $\mathbf{\Lambda}_i^r = \mathbf{\Lambda}_i^r - \delta$

end

Repeat this "**for**" loop until $0.0 < \mathbf{\Lambda}_i^r < 1.0$ for all $i = 1, 2, \ldots, l-1$. δ is usually set to 10^{-5}.

2. Check whether $\sum_i^{l-1} \mathbf{\Lambda}_i^r < 1.0$. If not,

$$\mathbf{\Lambda}^r = (\mathbf{\Lambda}^R + \mathbf{\Lambda}^r)/2,$$

and then go back to **2c** in **Step 2**.

3. Evaluate the cost function P_e^r of the vertex $\mathbf{\Lambda}^r$. If $P_e^r < P_e^w$, then

$$\mathbf{\Lambda}^n = \mathbf{\Lambda}^r, \; P_e^n = P_e^r.$$

Otherwise,

$$\mathbf{\Lambda}^r = (\mathbf{\Lambda}^R + \mathbf{\Lambda}^r)/2,$$

and then go back to **2c** in **Step 2**.

2d. Compute the average distance of these vertices in the new simplex. If the average distance is smaller than a prescribed value, the optimum vertex is chosen as the one having the smallest block-error rate among the simplex, and the optimization algorithm terminates. Otherwise, go back to **2a** in **Step 2** and start a new round to evolve the simplex.

It can be seen that the computational complexity depends primarily on the evaluation of the cost function of a valid symbol-node-degree distribution. Experimental results show that in general approximately 100 evaluations of the block-error rate at the E_b/N_0 threshold will be sufficient for the constrained "downhill simplex” algorithm to yield a satisfactory symbol-node-degree distribution. Note that the computational load can be greatly reduced by means of efficient implementations of the sum–product algorithm for decoding LDPC codes [50, 51]. Further reductions may be possible by using other, easily computable cost functions, see for example [33].

Note that the Monte-Carlo scheme can be viewed as being complementary to the density evolution scheme in the sense that the former is an empirical approach and its accuracy depends primarily on the amount of computation power available, whereas the latter is a theoretical approach and is based on asymptotic analysis. The density evolution approach proves to be very successful for long block-length codes and for the random construction; but it is not clear yet whether it is still applicable to short block-length codes and to the PEG construction; The Monte-Carlo approach is also indispensable for non-binary LDPC codes where the density evolution approach is not yet available.

VII. Code Performance

A. Regular codes

In this section we study the performance of PEG Tanner graphs applied to binary LDPC codes by means of computer simulations. For comparison purposes, we use the rate-1/2 ($n = 504$, $m = 252$) code of MacKay in [52], which is based on a regular Tanner graph with $d_s = 3, d_c = 6$. Currently this code is one of the best codes with these parameters. A PEG Tanner graph of 504 symbol and 252 check nodes is generated with uniform degree 3 for each symbol node. The resulting graph is nearly check-node uniform with degree 6, except for 8 check nodes with a degree of 7, and 8 with a degree of 5. We also use a randomly constructed rate-1/2 (504, 252) code, in which the degree of symbol nodes is 3 and the positions of 1s in a column is determined by a random integer generator uniformly distributed among the set $\{0, 1, \ldots, m-1\}$. Additional tests are implemented to avoid 4 cycles in the graph representation.

Fig. 10 compares the girth histogram of the PEG, MacKay’s, and random graph codes. In the PEG Tanner graph, each symbol node has a local girth of 8, except for three symbol nodes with a local girth of 10. In MacKay’s code, 63% of the symbol nodes have a local girth of 6 and 37% one of 8. In the random graph, 79% of the symbol nodes have a local girth of 6 and 21% one of 8. The average local girth of these three graphs is 8.01, 6.74, and 6.42, respectively. Clearly, based on the girth histogram, the PEG Tanner graph has an advantage over its two counterparts.

Fig. 11 compares the bit- and block-error rates for the three codes after 80 iterations over an additive white Gaussian noise (AWGN) channel,[7] and reveals that the performance of the random graph is much worse than that of the other two codes, perhaps mainly because of its poor girth histogram. We observe that the LDPC code based on the PEG Tanner graph is always slightly better than MacKay's code. With 80 iterations and at a block-error rate of 5×10^{-5}, the LDPC code based on the PEG Tanner graph outperforms MacKay's code by 0.2 dB. The significance of this result should not be underestimated, considering that, to the best of our knowledge, MacKay's codes still are the best codes for short and medium block lengths.

Note that although both MacKay's code and the random graph have a global girth of 6, the performance of the latter degrades significantly. This suggests that in reality it is not only the girth but also the girth histogram that dominate the performance of the iterative decoding. In [49] the average of the girth histogram is used as a heuristic tool to select good codes from random graphs for short block lengths.

Fig. 12 compares the girth histograms of the PEG, MacKay's and random graphs with parameters $n = 1008, m = 504, d_s = 3, d_c = 6$. In the PEG Tanner graph, 17% of the symbol nodes have a local girth of 8 and 83% one of 10. In MacKay's code, 39.5% of the symbol nodes have a local girth of 6 and 60.3% one of 8. In the random graph, 55.6% of the symbol nodes have a local girth of 6 and 44.2% one of 8. The average local girth of these three graphs is 9.66, 7.214, and 6.892, respectively. Fig. 13 compares the bit and block-error rates for these three (1008, 504) codes after 80 iterations. Again, we observe that the LDPC code based on the PEG Tanner graph is much better than the LDPC code based on the random graph, and slightly better than MacKay's code.

We report that the look-ahead-enhanced PEG construction yields a Tanner graph of girth 10 for $n = 1008, m = 504, d_s = 3, d_c = 6$. This represents an improvement in terms of girth relative to the generic PEG construction, in which the girth is 8. Moreover, by applying Lemma 6 with these parameters we know that in

[7]Throughout the paper, the channel model is assumed to be binary-input AWGN channels.

this case the upper bound on the girth is 12, indicating that the look-ahead-enhanced PEG is close to globally optimum.

B. Irregular codes

In this subsection we report on the code performance of near-optimum irregular symbol-node-degree distributions for the PEG construction, obtained by the Monte-Carlo approach together with constrained "downhill simplex" algorithm described above. The symbol-node-degree distribution is $\Lambda(x) = 0.47532x^2 + 0.279537x^3 + 0.0348672x^4 + 0.108891x^5 + 0.101385x^{15}$, which is optimized for $n = 504, m = 252$. As required by the PEG construction, the check-node-degree distribution is automatically concentrated and thus uniquely determined by the symbol-node-degree distribution and the code rate [8]. The PEG LDPC code we considered is of rate-1/2, and its performance is depicted in Fig. 14. As can be seen this code outperforms MacKay's rate-1/2 (504, 252) code by about 0.4 to 0.5 dB in the entire range of E_b/N_0 after 80 iterations. Also plotted is the performance of a randomly constructed code using the same degree distribution,[9] exhibiting a very early "error floor" effect. This clearly shows that the PEG construction provides a significant advantage over the random construction, particularly for short block lengths and for cases in which the figure of merit is the block error rate.

We also investigate this degree distribution applied to the case $n = 1008, m = 504$. The performance of the resulting LDPC code constructed with the PEG algorithm is shown in Fig. 15. This figure indicates nearly 0.5 dB gain over MacKay's rate-1/2 (1008, 504) code. To the best of our knowledge, the irregular PEG LDPC codes reported here are the best ones to date in terms of block-error rate at short block lengths.

Density evolution has proven to be an efficient and effective approach to design good irregular-degree-distribution pairs with which LDPC codes based on random construction exhibit a performance extremely close to the Shannon limit for sufficiently long block lengths. It is thus tempting to combine the PEG construction with the symbol-node-degree distribution optimized by the density evolution approach to design

[8]Whenever a check-node-degree distribution is not explicitly shown here, it is assumed to be concentrated.

[9]The irregular random code is constructed column by column with appropriate degree, and the 1s in each column are determined by a uniform integer generator. Efforts have been taken to avoid 4-cycles.

LDPC codes. We investigate the performance of symbol-node-degree distributions[10] in [18, Tables I and II] under the PEG construction with parameters $n = 504, m = 252$. Among these symbol-node distributions with maximum symbol-node degrees 4, 5, 7, 9, 11, 15, 30, 50, the one with maximum degree 15 achieves the best performance, which is essentially the same performance as that of the one optimized by the empirical Monte-Carlo approach, see Fig. 16. This confirms that even for short block lengths, the degree distributions designed by the density evolution approach are still nearly optimum if the PEG construction is used. Note that the degree distribution with maximum degree 50 has a very good "threshold," which is only 0.06 dB away from the Shannon capacity in the limiting case, but suffers from a significant degradation due to the short block length. Also plotted is the performance of LDPC codes based on random construction with the same degree distributions. Once again, we observe that the PEG construction significantly outperforms the random construction, particularly in the high-SNR region.

The Monte-Carlo approach together with the constrained downhill simplex method to design near-optimum degree distributions is indispensable in two-fold meanings: first it confirmed that the irregular degree distributions optimized by the density evolution approach match well with the PEG construction for short block lengths; this contrasts the old belief that irregular degree distributions do not work well with the random construction for short block lengths. Secondly, it can also be applied to the cases of non-binary LDPC codes (in Section X) where a density-evolution approach is not available yet.

VIII. Low-Complexity Encoding

The complexity per bit of iterative decoding using BP or SPA on a Tanner graph has been shown to be independent of the block length n, but the encoding complexity generally scales as n^2. Several publications address this issue, see for example [31, 42-46]. The most common idea is to exploit the sparseness of the parity-check matrix H and its corresponding graph to obtain an efficient encoding format, namely, a triangular or almost triangular parity-check matrix. For example, in [44] the codeword w and the parity-check matrix

[10] We do not need the check-node distribution as the check-degree sequence is made as uniform as possible in the PEG algorithm.

H are partitioned into $w = [p, d]$ and $H = [H^p, H^d]$, respectively, such that

$$[H^p, H^d]w^T = 0. \tag{71}$$

It was found empirically that a good choice for H^p is the $m \times m$ square matrix

$$H^p = \begin{pmatrix} 1 & & & 0 \\ 1 & 1 & & \\ & \ddots & \ddots & \\ 0 & & 1 & 1 \end{pmatrix}_{m\times m} . \tag{72}$$

The matrix H^d is then created by constructing a random graph such that no 4-cycles are generated. Using (71) and (72), the parity-check bits $p = \{p_i\}$ can be computed by

$$p_1 = \sum_{j=1}^{n-m} h^d_{1,j} d_j \;\text{ and }\; p_i = p_{i-1} + \sum_{j=1}^{n-m} h^d_{i,j} d_j,\; i > 1, \quad (\text{mod2}), \tag{73}$$

where $d = \{d_i\}$ is the systematic part of the codeword, and $H^d = \{h^d_{i,j}\}$ is the $m \times (n - m)$ component of the partitioned parity-check matrix H. Clearly, the encoding process has become much simpler because the Gaussian elimination step is avoided. Moreover, computation and storage requirements in the encoder are also reduced because H^d is sparse by design. This approach clearly guarantees a linear-time encoding, but in general may result in some loss in performance. The structure of (72) gives rise to a special pattern on a Tanner graph that was called the "zigzag" pattern in irregular repeat accumulate (IRA) codes [45].

We show that this simple idea can also be applied in a natural way to a PEG Tanner graph, rendering linear-time encoding possible while maintaining a large girth and equally good performance. To facilitate linear-time encoding, we partition the symbol node set V_s into two disjoint subsets: a redundant subset V_s^p and an information subset V_s^d. The redundant subset contains the first m symbol nodes, which are redundant bits in a codeword and whose associated edges are prespecified as the simple "zigzag" pattern shown in Fig. 17 that corresponds to the square matrix H^p in (71). The information subset V_s^d contains the other $n - m$ symbol nodes, which correspond to information (systematic) bits in a codeword and whose associated edges are established by the PEG algorithm with the zigzag edges already in the graph. Note that the prescribed zigzag edges do not contain a single cycle before the PEG algorithm starts, thereby imposing no negative impact on the girth histogram.

One of the most important results on encoding of LDPC codes appears in [46] where it is proved that optimized LDPC codes can be encoded in linear time. In particular, one can choose a randomly-generated LDPC code with an optimized irregular degree sequence pair, and then the parity-check matrix of this code can be rearranged to have an (almost) upper-triangular structure that allows linear-time encoding. The significance of [46] lies in the existence proof of optimal LDPC codes with (almost) upper-triangular structure. Note that the rearrangement of a parity-check matrix does not change the LDPC code itself and the complexity of rearrangement is generally quadratic, it is thus very interesting to construct such LDPC codes directly.

The PEG algorithm can be easily tailored to construct LDPC codes having (almost) triangular structure, good girth properties and optimum irregular degree sequence. For the sake of clarity we focus on parity-check matrices with upper-triangular structure. In this case the $m \times m$ component $H^p = \{h^p_{i,j}\}$ of the parity-check matrix can be written as

$$H^p = \begin{pmatrix} 1 & h^p_{1,2} & \cdots & h^p_{1,m} \\ 0 & 1 & & \\ \vdots & \vdots & \ddots & \vdots \\ & & 1 & h^p_{m-1,m} \\ 0 & \cdots & 0 & 1 \end{pmatrix}_{m\times m}, \tag{74}$$

and the parity bits are computed according to

$$p_i = \sum_{j=i+1}^{m} h^p_{i,j} p_j + \sum_{j=1}^{n-m} h^d_{i,j} d_j, \quad (\mathrm{mod} 2), \tag{75}$$

where (75) is computed recursively from $i = m$ to $i = 1$. By combining the PEG construction method with the constraint that the resulting H matrix be of lower or upper triangular form, we obtain powerful, linear-time encodable, irregular LDPC codes with large girth and very good performance. As before, we partition the symbol node set V_s into the redundant subset V_s^p and the information subset V_s^d containing the first m symbol nodes (parity bits) and the other $n - m$ symbol nodes (systematic information bits), respectively. The edges of the symbol nodes are then established by means of the PEG algorithm while observing the special pattern in (74), so that a good girth histogram is obtained. As the procedure of establishing the edges of $n-m$ information bits follows the construction of edges of redundant subset V_s^p and is exactly the same as that described in Section III, we focus on the modified PEG algorithm for constructing edges of V_s^p.

PEG Algorithm for Establishing Edges of V_s^p:

for $j=0$ to $m-1$ **do**

begin

for $k=0$ to $d_{s_j}-1$ **do**

begin

if $k=0$

$E_{s_j}^0 \longleftarrow$ edge (c_j, s_j), where $E_{s_j}^0$ is the first edge incident to s_j. This edge corresponds to the '1' in the diagonal line of matrix H^p.

else

expanding a tree from symbol node s_j up to depth l under the current graph setting such that $\bar{\mathcal{N}}_{s_j}^l \cap \{c_0, c_1, \ldots, c_{j-1}\} \neq \varnothing$ but $\bar{\mathcal{N}}_{s_j}^{l+1} \cap \{c_0, c_1, \ldots, c_{j-1}\} = \varnothing$, or the cardinality of $\mathcal{N}_{s_j}^l$ stops increasing, then $E_{s_j}^k \longleftarrow$ edge (c_i, s_j), where $E_{s_j}^k$ is the k-th edge incident to s_j and c_i is one check node picked from the set $\bar{\mathcal{N}}_{s_j}^l \cap \{c_0, c_1, \ldots, c_{j-1}\}$ having the lowest check-node degree.

end

end

As an example we consider an irregular PEG Tanner-graph code whose parity-check matrix is forced into an upper triangular form. Fig. 18 compares the bit and block-error rates of the irregular PEG Tanner-graph code presented in Section VII-B, an irregular PEG Tanner-graph code with a parity-check matrix in upper diagonal form, and MacKay's code, all with parameters $n=1008, m=504$. The symbol-node-degree distribution for both irregular PEG Tanner-graph codes is $\Lambda(x)=0.47532x^2+0.279537x^3+0.0348672x^4+0.108891x^5+0.101385x^{15}$. When the parity check matrix is forced into an upper triangular form there is only one symbol node of degree 1. As can be seen, the two irregular codes designed according to the PEG algorithm have essentially the same performance and are about 0.5 dB better than MacKay's rate-1/2 ($n = 1008, m = 504$) code. Hence it is shown that linear-time encoding can be achieved without noticeable performance degradation under the PEG construction.

IX. Performance of PEG Codes versus Turbo Codes

Irregular LDPC codes with large block lengths have been shown to outperform Turbo codes [18]. However, it appears to be a common belief that for relatively short block lengths Turbo codes outperform LDPC codes. Here we take a closer look at the performance of linear-time-encodable PEG LDPC codes (of relatively short block lengths) as compared with that of Turbo codes in the CDMA2000 standard. The turbo encoder consists of two systematic, recursive, 8-state convolutional encoders concatenated in parallel, with an interleaver. The transfer function for the encoder is $G(D) = \left[1 \ \frac{n_0(D)}{d(D)} \ \frac{n_1(D)}{d(D)}\right]$, where $d(D) = 1+D^2+D^3$, $n_0(D) = 1+D+D^3$, and $n_1(D) = 1+D+D^2+D^3$. Depending on different puncturing patterns, rate-1/2, 1/3, and 1/4 codes can be obtained. Note that for each block a total of six bits are used to terminate the two convolutional encoders. For a detailed description of the standardized Turbo interleaver, we refer the reader to [53]. The LDPC codes and Turbo codes are compared at exactly the same rate and block length. For decoding Turbo codes, 12 iterations with the BCJR algorithm [54] in log-domain are employed; for decoding LDPC codes, 80 iterations using the low-complexity SPA of [51] are used so that the decoding complexity remains approximately the same. In designing linear-time-encodable binary LDPC codes, we take the irregular degree distribution as $\Lambda(x) = 0.477081x^2 + 0.280572x^3 + 0.0349963x^4 + 0.0963301x^5 + 0.0090884x^7 + 0.00137443x^{14} + 0.100558x^{15}$, chosen from [18].

Figs. 19 and 20 show the bit- and block-error rates over the AWGN channel for linear-time-encodable PEG LDPC codes and the CDMA2000 Turbo codes, with rate $\simeq 1/2$ and block lengths of 1024 and 2048, respectively. Both kinds of codes have comparable performance at exactly the same block length and rate; in the low-SNR region, the CDMA2000 Turbo codes are slightly better, whereas in the high-SNR region, the LDPC codes outperform the CDMA2000 Turbo codes. The advantage of the LDPC codes becomes more pronounced if the figure of merit is the block-error rate.

In conclusion, our linear-time-encodable PEG LDPC codes offer comparable performance to the standardized Turbo codes with essentially the same encoding/decoding complexity. It is anticipated that if better degree distributions are available from the analysis of finite block-length LDPC codes [55], the performance of such LDPC codes can be further improved. To close this section, we point out that, as the iterative decoding of LDPC codes is fully parallelizable and could be accomplished at significantly greater speeds, the linear-time-encodable PEG LDPC codes show promise for extremely-high-speed applications such as magnetic recording, optical communication, and broadband wireline/wireless access technologies.

X. PEG TANNER-GRAPH CODES OVER GF(q)

So far we have primarily considered binary LDPC codes represented by binary parity-check matrices or their corresponding bipartite graphs constructed using the PEG algorithm. These codes can easily be generalized to finite fields GF(q) in the same natural way as in [33, 47], i.e., by allowing the symbol nodes to assume values from the finite field. Specifically, fields with $q = 2^b$, $b > 1$, elements that are important in practical applications, and codes over GF(2^b) with linear constraints defined on non-binary sparse H matrices are considered in this section. In GF(2^b) each encoded symbol w_i consists of b binary bits. Since a symbol from the field GF(q), $q = 2^b$ for some integer b, may be represented as a binary string of b bits, we can use such codes with binary-input channels, transmitting one q-ary symbol for every b uses of the binary channel. The decoder interprets b bits $(y_0, y_1, \ldots, y_{b-1})$ from the channel as a single 2^b-ary symbol and sets the prior information of that symbol by assuming a product distribution for the values of each constituent bit. Namely

$$f^x := \prod_{i=0}^{b-1} f_{y_i}^{x_i}$$

where $f_{y_i}^{x_i}$ is the likelihood the ith constituent bit is equal to x_i, where $(x_0, x_1, \ldots, x_{b-1})$ is the binary representation of the transmitted symbol x.

The motivation that we use GF(q) LDPC codes with binary-input channels is based on the following arguments: Iterative decoding performance of an LDPC code is in general dominated by two competing factors. One is the Hamming weight spectrum which requires the density of parity-check matrix to be higher; the other is the performance loss due to iterative decoding for which lower density of parity-check matrix is favorable. The two conflicting requirements must be well balanced, particularly for short block lengths. It has been demonstrated in Section V that the Hamming weight spectrum can be much improved by a move from binary field to fields of higher order with the same density and the same binary block length.

We describe the construction of PEG GF(q) LDPC codes briefly: given the number of symbol nodes n, the number of parity-check nodes m, and the symbol-node-degree sequence of the graph, the PEG algorithm is started first, in exactly the same manner as the binary case, such that the placement of a new edge on the graph has as small an impact on the girth of the graph as possible. In this way a PEG Tanner graph is obtained that not only has a large girth but also a good girth histogram. To form a GF(q) parity-check matrix, the positions of nonzero entries are determined by the PEG Tanner graph, whereas the values of the nonzero entries of the parity-check matrix are selected randomly from a uniform distribution among nonzero elements of GF(q).

Table I shows optimized rate-1/2 irregular PEG Tanner-graph codes over GF(2^b) and their corresponding symbol-node-degree distributions. The optimization of the degree sequences was accomplished with the variant of the "downhill simplex"method described in Section VI. We compare codes having a block length of n symbols over GF(2^b) with binary codes of length nb bits.

Fig. 21 shows the performance of the irregular PEG Tanner-graph codes over a binary-input Gaussian channel. Five codes of rate 1/2 over GF(2), GF(8), GF(16), GF(32), and GF(64) are shown. All codes correspond to block lengths of 1008 bits (except the irregular PEG Tanner-graph code over GF(32), which has a block length of 202 symbols or 1010 bits). Also shown is the performance of the rate-1/2, $n = 1008$, $m = 504$ binary MacKay code as well as the sphere-packing bound for this block length. As can be seen, an improvement of 0.25 dB is obtained by moving from binary to GF(2^6) PEG construction. Furthermore, the overall gain of the GF(2^6) PEG construction compared with the binary MacKay code is approximately 0.75 dB. Finally, the rate-1/2 irregular PEG code over GF(2^6) shows a block-error rate $< 10^{-4}$ at E_b/N_0 2 dB, i.e., a performance that is only 0.4 dB from the Gallager's sphere-packing bound[11] of a binary-input AWGN channel [56, 60], which appears to be the best–known performance at this block length to date.

The performance results indicate that the PEG Tanner-graph codes over higher-order fields outperform the

[11] Bear in mind, however, that no undue significance should be attached to this 0.4 dB gap, as at this block length the formula for computing the Gallager bound might not be sufficiently exact.

binary ones. Furthermore, owing to the PEG construction that aims at large girth as well as good irregular degree sequence we have observed a monotonic improvement with increasing field order, which contrasts with the observations of [47]. Interestingly enough, the irregularity feature seems to be unnecessary if the higher-order field is sufficiently large, and the optimum graph tends to favor a regular one of degree-2 in all symbol nodes, termed cycle graph or cycle code, which is the lowest density graph in the context of iterative decoding.

XI. Conclusions

A general method for constructing Tanner graphs having large girth and minimum distance has been presented. The main principle in this construction is to optimize the placement of a new edge, connecting a particular symbol node to specific check node on the graph such that the largest possible local girth is achieved. In this way, the underlying graph grows in an edge-by-edge manner, optimizing each local girth, and is thus referred to as a progressive edge-growth (PEG) Tanner graph.

Upper and lower bounds on the girth of a PEG Tanner graph have been derived. These bounds depend on the number of symbol nodes and on that of check nodes, as well as on the maximum values of the symbol- and check-node degrees of the underlying graph. By comparing the lower bound and the upper bound on the girth, we claim that the girth of a PEG Tanner graph meets or surpasses the Erdös-Sachs bound. In addition, a lower bound on the minimum distance of binary LDPC codes defined on PEG Tanner graphs has also been derived.

The advantages of the PEG construction over a random one are twofold. First, it yields a much better girth distribution, thereby facilitating the task of the BP or SPA during the iterative decoding process. Second, it leads to (or guarantees) a meaningful lower bound on the minimum distance, providing insight into the performance of the code at high signal-to-noise ratios. Simulation results confirmed that using the PEG algorithm for constructing short-block-length LDPC codes results in a significant improvement compared to randomly constructed codes.

We have described an empirical Monte-Carlo approach using a variant of the “downhill simplex” search algorithm to design irregular PEG graphs for small codes with fewer than a thousand bits, which can be viewed as a complementary tool for the asymptotic analysis "density evolution". We found that even for small block lengths such as $n = 504$, there is a good degree distribution from the density evolution approach that works perfectly with the PEG construction. Simulation results showed that in conjunction with optimum symbol-node-degree distributions (obtained from either the empirical or the density evolution approach) our PEG construction yields the best binary LDPC codes at short block lengths to date. Linear-time-encodable LDPC codes have also been constructed by slightly modifying the PEG algorithm to yield a Tanner graph containing a zigzag pattern or (almost) triangular format. This easy encoding property is attained without noticeable performance degradation. We have also demonstrated that in third-generation high-speed wireless data services, the linear-time-encodable PEG LDPC codes offer comparable performance to standardized Turbo codes with essentially the same encoding/decoding complexity.

Finally, the regular and irregular LDPC codes have been generalized by using the same PEG construction but allowing the symbol nodes to take values over higher-order finite fields. This work confirms that by moving to higher-order fields short-block-length codes can be constructed that operate close to the Gallager's sphere-packing bound when decoded with the sum–product algorithm. We reported a short block-length (1008 bit) rate-1/2 irregular PEG LDPC code over $GF(2^6)$ with a block error rate $< 10^{-4}$ at $E_b/N_0 = 2$ dB, which appears to have the best–known performance at this short block length to date.

ACKNOWLEDGMENT

The authors are grateful to M. P. C. Fossorier, R. L. Urbanke, and M. Vetterli for insightful comments and suggestions, which have greatly improved this paper. We would also like to thank D. J. C. MacKay for providing the parity-check matrices of some LDPC codes simulated in this paper.

REFERENCES

[1] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 533–547, Sept. 1981.

[2] N. Wiberg, H.-A. Loeliger, and R. Kötter, "Codes and iterative decoding on general graphs," *European. Trans. Telecommun.*, vol. 6, pp. 513–526, Sept. 1995.

[3] M. Sipser and D. A. Spielman, "Expander codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1710–1722, Nov. 1996.

[4] N. Alon and M. Luby, "A linear-time erasure-resilient code with nearly optimal recovery," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1732–1736, Nov. 1996.

[5] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, "Turbo decoding as an instance of Pearl's belief propagation algorithm," *IEEE J. Selected Areas Commun.*, vol. 16, pp. 140–152, Feb. 1998.

[6] F. R. Kschischang and B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *IEEE J. Selected Areas Commun.*, vol. 16, pp. 219–230, Feb. 1998.

[7] R. Kötter and A. Vardy, "Factor graphs: Construction, classification and bounds," *Proc. 1998 IEEE Int'l. Symp. on Information Theory*, Cambridge, MA, Aug. 1998, p. 14.

[8] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum–product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, pp. 498–519, Feb. 2001.

[9] A. R. Calderbank, G. D. Forney, and A. Vardy, "Minimal tail-biting trellises: The Golay code and more," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1435–1455, July 1999.

[10] Y. Weiss, "Correctness of local probability propagation in graphical models with loops," *Neural Computation*, vol. 12, pp. 1–41, 2000.

[11] S. M. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Trans. Inform. Theory*, vol. 46, pp. 325–343, March 2000.

[12] G. D. Forney, "Codes on graphs: News and views," *Proc. of 2nd Int. Symp. on Turbo Codes & Related Topics*, Brest, France, Sept. 2000, pp. 9–16.

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

[14] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," *Proc. 1993 IEEE Int. Conf. on Communication*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[15] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, March 1999.

[16] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Analysis of low-density codes and improved designs using irregular graphs," *Proc. 30th ACM STOC*, Dallas, TX, 1998, pp. 249–258.

[17] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599–618, Feb. 2001.

[18] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of provably good low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619–637, Feb. 2001.

[19] S.-Y. Chung, G. D. Forney Jr., T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, no. 2, pp. 58–60, Feb. 2001.

[20] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries: a rediscovery and new results," *IEEE Trans. Inform. Theory*, vol. 47, pp. 2711–2736, Nov. 2001.

[21] R.M. Tanner, D. Srkdhara, and T. Fuja, "A class of group-structured LDPC codes," *Proc. of ISTA 2001,* Ambleside, England, 2001.

[22] J. L. Fan, "Array codes as low-density parity-check codes," *Proc. 2nd Intl. Symp. on Turbo Codes & Related Topics,* Brest, France, Sept. 2000.

[23] B. Vasic, "Combinatorial constructions of structured low-density parity-check codes for iterative decoding," submitted for publication, 2001.

[24] S. J. Johnson, and S. R. Weller, "Regular low-density parity-check codes from combinatorial designs," *Proc. of IEEE Inform. Theory Workshop,* Cairns, Australia, Sept. 2001.

[25] G. A. Margulis, "Explicit constructions of graphs without short cycles and low density codes," *Combinatorica*, vol. 2, no. 1, pp. 71–78,

1982.

[26] G. A. Margulis, “Explicit group-theoretical constructions of combinatorial schemes and their application to the design of expanders and concentrators,” *Problems of Information Transmission (USA)*, vol. 24, no. 1, pp. 39–46, Jan.-March 1988; translated from *Problemy Peredachi Informatsii (USSR)*, vol. 24, no. 1, pp. 51–60, Jan.-March 1988.

[27] A. Lubotzky, R. Phillips, and P. Sarnak, “Ramanujan graphs,” *Combinatorica*, vol. 8, no. 3, pp. 261–277, 1988.

[28] P. Erdös and H. Sachs, Reguläre Graphen gegebener Taillenweite mit minimaler Knotenzahl. *Wiss. Z. Univ. Hall Martin Luther Univ. Halle – Wittenberg Math. – Natur. Reine,* 12:251-257, 1963.

[29] J. Rosenthal and P. O. Vontobel, “Construction of LDPC codes based on Ramanujan graphs and ideas from Margulis,” *Proc. 38th Annual Allerton Conf. on Communication, Computing and Control*, Monticello, IL, Oct. 2000.

[30] P. O. Vontobel and R. M. Tanner, “Construction of codes based on finite generalized quadrangles for iterative decoding,” presented at the Int. Symp. on Information Theory, Washington, DC, June 24–29, 2001.

[31] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, “Practical loss-resilient codes,” *Proc. 29th ACM Symp. Theory of Computing*, El Paso, TX, May 1997, pp. 150-159.

[32] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, “Improved low-density parity-check codes and using irregular graphs and belief propagation,” *Proc. 1998 IEEE Int'l. Symp. on Information Theory*, Cambridge, MA, Aug. 1998, p. 117.

[33] M. C. Davey, *Error-Correction Using Low-Density Parity-Check codes*, Ph.D dissertation, University of Cambridge, Cambridge, UK, Dec. 1999.

[34] L. Bazzi, T. Richardson and R. Urbanke, “Exact thresholds and optimal codes for the binary symmetric channel and Gallager's decoding algorithm A,” *IEEE Trans. Inform. Theory,* to appear.

[35] M. A. Shokrollahi, “New sequence of linear time erasure codes approaching the channel capacity,” *Proc. 13th Int. Sym. on Applied Algebra, Algebraic Algorithm and Error-correcting Codes,* 1999.

[36] X.-Y. Hu, E. Eleftheriou, and D.-M. Arnold, “Progressive edge-growth Tanner graphs,” *Proc. of IEEE Global Telecommuni. Conf. (GLOBECOM),* San Antonio, Texas, USA, Nov. 2001.

[37] X.-Y. Hu, E. Eleftheriou, and D.-M. Arnold, “Irregular progressive edge-growth Tanner graphs,” *Proc. of 4th Intl. ITG Conf. on Source and Channel Coding,* Berlin, Jan. 2002.

[38] X.-Y. Hu, E. Eleftheriou, and D.-M. Arnold, “Irregular progressive edge-growth Tanner graphs,” *Proc. of IEEE Intl. Symp. Inform. Theory (ISIT),* Lausanne, Switzerland, July 2002.

[39] J. Campello, D. S. Modha, and S. Rajagopalan, “Designing LDPC codes using bit-filling,” *Proc. of Intl. Conf. Communi. (ICC),* Helsinki, Finland, June, 2001.

[40] S. L. S. Jacoby, J. S.Kowalik, J. T. Pizzo, and W. T. Veterling, “*Iterative Methods for Nonlinear Optimization Problems*,” Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1972.

[41] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Veterling, “*Numerical Recipes in C*,” Cambridge, 1988.

[42] D. Spielman, “Linear-time encodeable and decodable error-correcting codes,” *IEEE Trans. Inform. Theory,* vol. 42, pp. 1723-1731, Nov. 1996.

[43] D. J. C. MacKay, S. T. Wilson, and M. C. Davey, “Comparison of construction of irregular Gallager codes,” *IEEE Trans. Commun.*, vol. 47, pp. 1449–1454, Oct. 1999.

[44] L. Ping, W. K. Leung, and N. Phamdo, “Low density parity check codes with semi-random parity check matrix,” *IEE Elect. Lett.*, vol. 35, pp. 38–39, Jan. 1999.

[45] H. Jin, A. Khandekar, and R. J. McEliece, “Irregular repeat–accumulate codes,” *Proc. of 2nd Int. Symp. on Turbo Codes & Related Topics*, Brest, France, Sept. 2000, pp. 1–8.

[46] T. Richardson and R. Urbanke, “Efficient encoding of low-density parity-check codes,” *IEEE Trans. Inform. Theory*, vol. 47, pp. 638–656, Feb. 2001.

[47] M. C. Davey and D. MacKay, “Low-density parity-check codes over GF(q),” *IEEE Commun. Lett.*, vol. 2, no. 6, pp. 165-167, June 1998.

[48] R. M. Tanner, “Minimum distance bounds by graph analysis,” *IEEE Trans. Inform. Theory*, vol. 47, pp. 808–821, Feb. 2001.

[49] Y. Mao and A. Banihashemi, “Search of good low-density parity-check codes using girth distribution” (preprint, 2001).

[50] E. Eleftheriou, T. Mittelholzer, and A. Dholakia “Reduced-complexity decoding algorithm for low-density parity-check codes,” *IEE Elect. Lett.*, vol. 37, pp. 102–104, Jan. 2001.

[51] X.-Y. Hu, E. Eleftheriou, D.-M. Arnold, and A. Dholakia, “Efficient implementations of the sum–product algorithm for decoding LDPC codes,” *Proc. of IEEE Global Telecommuni. Conf. (GLOBECOM),* San Antonio, Texas, USA, Nov. 2001.

[52] D. J. C. MacKay, “Online database of low-density parity-check codes,” available at `http://wol.ra.phy.cam.uk/mackay/codes/data.html`.

[53] *Standards for CDMA2000 Spread Spectrum Systems,* EIA/TIA IS-2000, 1-6.

[54] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, “Optimal decoding of linear codes for minimizing symbol error rate,” *IEEE Trans. Inform. Theory,* vol. IT-20, pp. 284-287, Mar. 1974.

[55] C. Di, D. Proietti, E. Telatar, T. Richardson, and R. Urbanke, “Finite length analysis of low-density parity-check codes,” to appear at *IEEE Trans. Inform. Theory*, 2002.

[56] C. Schlegel and L. Perez, “On error bounds and Turbo-codes,” *IEEE Commun. Lett.,* vol. 3, pp. 205–207, July 1999.

[57] A. Aji, H. Jin, A. Khandekar, D. J. C. MacKay, and R. J. McEliece, “BSC thresholds for code ensembles based on ’Typical Pairs’ decoding,” *Codes, Systems, and Graphical Models,* the IMA volumes in mathematics and its applications, edited by B. Marcus and J. Rosenthal, Springer, 2001.

[58] J. Blomer, R. Karp, and E. Welzl, “The rank of sparse random matrices over finite fields,” Research Report, International Computer Science Institute, Berkeley, California, 1996.

[59] S. J. MacMullan, and O. M. Collins, “A comparison of known codes, random codes, and the best codes,” *IEEE Trans. Inform. Theory,* vol. 44, pp. 3009-3022, Nov. 1998.

[60] R. G. Gallager, *Information Theory and Reliable Communication*, John Wiley & Sons, New York, 1968.

TABLE AND CAPTION

TABLE I

OPTIMIZED SYMBOL-NODE-DEGREE DISTRIBUTIONS FOR RATE-1/2 PEG CODES OVER GF(2^b). THE BLOCK LENGTH IN BINARY BITS IS nb.

Galois field	(n, m)	Symbol-node-degree distribution	Ave. symbol degree
GF(2)	(1008, 504)	$0.47532x^2 + 0.279537x^3 + 0.0348672x^4 + 0.108891x^5 + 0.101385x^{15}$	3.994
GF(8)	(336, 168)	$0.643772x^2 + 0.149719x^3 + 0.193001x^4 + 0.013508x^5$	2.5762
GF(16)	(252, 126)	$0.772739x^2 + 0.102863x^3 + 0.113797x^4 + 0.010601x^5$	2.3623
GF(32)	(202, 101)	$0.84884x^2 + 0.142034x^3 + 0.009126x^4$	2.1603
GF(64)	(168, 84)	$0.94x^2 + 0.05x^3 + 0.01x^4$	2.07

Figure Captions

Fig. 1. An example of symbol node degree $D_s = \{2, 2, 2, 2, 3, 3, 3, 3\}$ irregular Tanner graph.

Fig. 2. A subgraph spreading from symbol node s_j. If there are no duplicate vertices inside the subgraph, then the subgraph is tree-like.

Fig. 3. A subgraph spreading from parity-check node c_i.

Fig. 4. Lower and upper bounds on a PEG regular Tanner graph with $d_s = 3, d_c = 6$.

Fig. 5. Lower and upper bounds on a PEG regular Tanner graph with $d_s = 4, d_c = 8$.

Fig. 6. An "active" tree induced by a minimum-weight codeword.

Fig. 7. Hamming weight spectra of binary random ensembles with different block lengths n.

Fig. 8. Hamming weight spectra of binary random ensembles with different densities p.

Fig. 9. Hamming weight spectra of random ensembles over GF(q) with the same block length of n (n=600 bit) and the same density of p, p=0.01.

Fig. 10. Girth histograms of a PEG Tanner graph, MacKay's code, and a random graph, with parameters $n = 504, m = 252, d_s = 3, d_c = 6$.

Fig. 11. Bit- and block-error rates of PEG Tanner-graph code, MacKay's code, and random graph code, with parameters $n = 504, m = 252, d_s = 3, d_c = 6$.

Fig. 12. Girth histograms of a PEG Tanner graph, MacKay's code, and a random graph, with parameters $n = 1008, m = 504, d_s = 3, d_c = 6$.

Fig. 13. Bit- and block-error rates of a PEG Tanner-graph code, MacKay's code, and random graph code, with parameters $n = 1008, m = 504, d_s = 3, d_c = 6$.

Fig. 14. Bit- and block-error rates of the irregular PEG Tanner-graph code, irregular random-graph code, and MacKay's code (regular) with parameters $n = 504, m = 252$. The symbol-node-degree distribution for the irregular PEG Tanner-graph code is $\Lambda(x) = 0.47532x^2 + 0.279537x^3 + 0.0348672x^4 + 0.108891x^5 + 0.101385x^{15}$.

Fig. 15. Bit- and block-error rates of the irregular PEG Tanner-graph code, irregular random-graph code, and MacKay's code (regular) with parameters $n = 1008, m = 504$. The symbol-node-degree distribution for the irregular PEG Tanner-graph code is $\Lambda(x) = 0.47532x^2 + 0.279537x^3 + 0.0348672x^4 + 0.108891x^5 + 0.101385x^{15}$.

Fig. 16. Block-error rates of irregular PEG (solid lines) and irregular random-graph codes (dashed lines) with density-evolution-optimized degree distributions; code parameters are $n = 504, m = 252$.

Fig. 17. Linear-time-encodable PEG Tanner graph with zigzag pattern. The solid edges correspond to the zigzag pattern, which is specified before the PEG algorithm starts. The dashed lines correspond to the edges established by the PEG algorithm.

Fig. 18. Bit- and block-error rates of an irregular PEG Tanner-graph code, an upper-triangular PEG Tanner-graph code, and MacKay's code.

Fig. 19. Bit- and block-error rates of a linear-time-encodable PEG LDPC code and a CDMA2000 turbo code, with block length 1024 and rate (512-6)/1024.

Fig. 20. Bit- and block-error rates of a linear-time-encodable PEG LDPC code and a CDMA2000 turbo code, with block length 2048 and rate (1024-6)/2048.

Figure 21. Bit- and block-error rates of irregular LDPC codes over GF(2), GF(8), GF(16), GF(32), and GF(64), based on PEG Tanner graph with the parameters given in Table I.

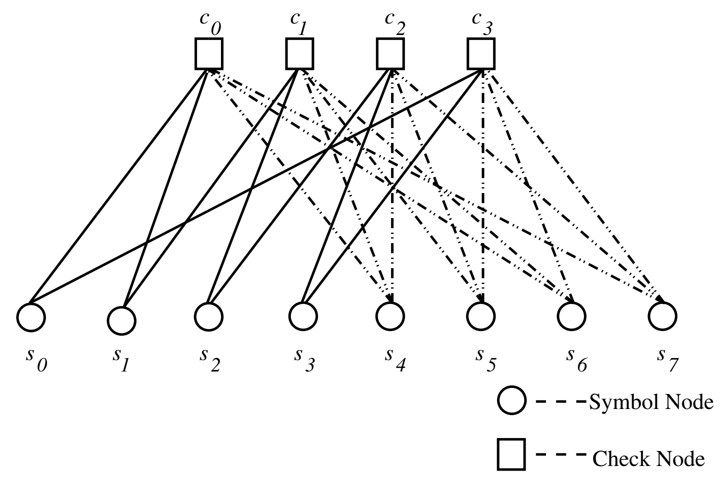

c_0
c_1
c_2
c_3
s_0
s_1
s_2
s_3
s_4
s_5
s_6
s_7
Symbol Node
Check Node

Fig. 1.

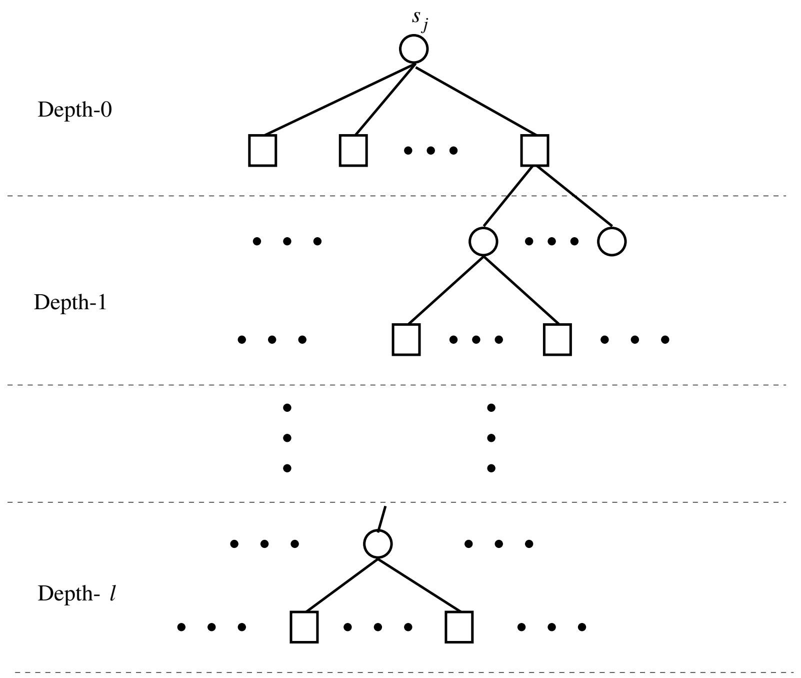

s_j
Depth-0
Depth-1
Depth- l

Fig. 2.

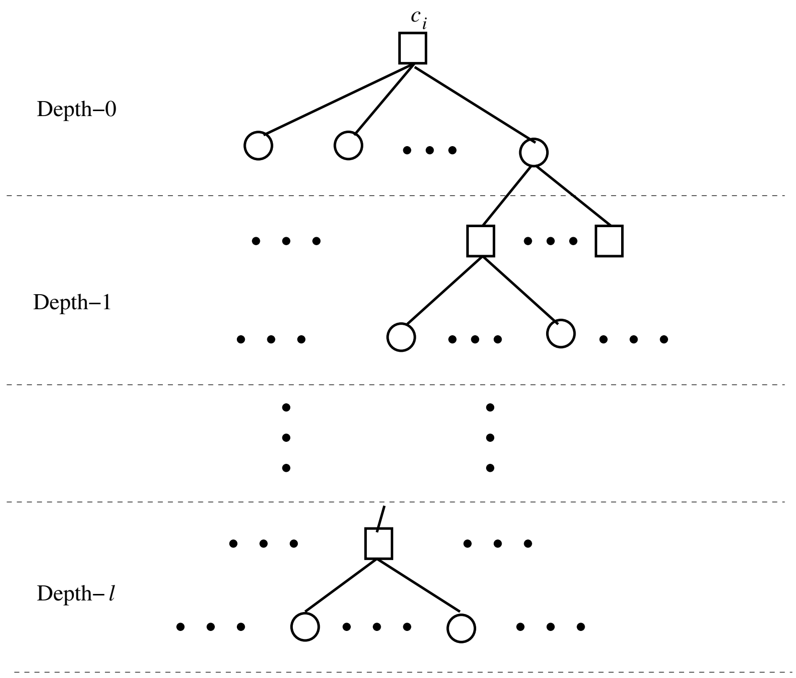


Fig. 3.

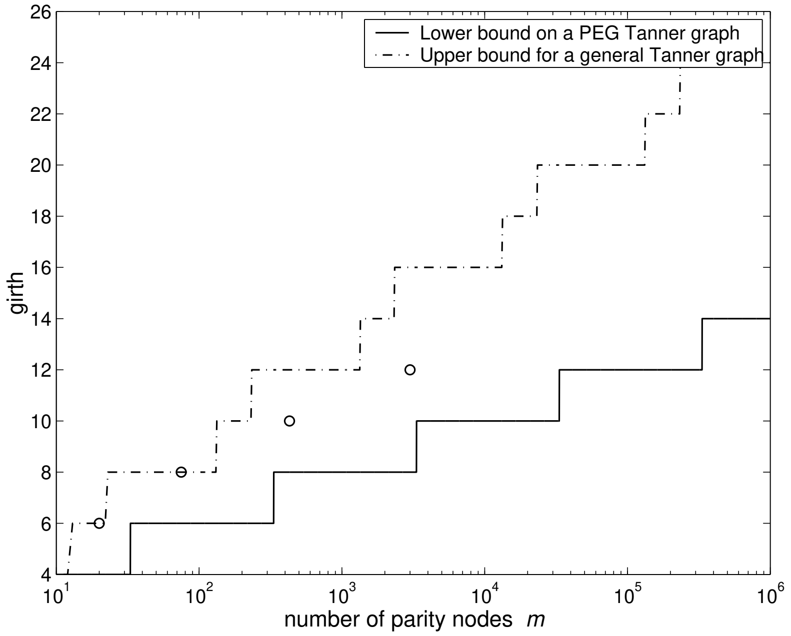

Lower bound on a PEG Tanner graph
Upper bound for a general Tanner graph
girth
number of parity nodes m

Fig. 4.

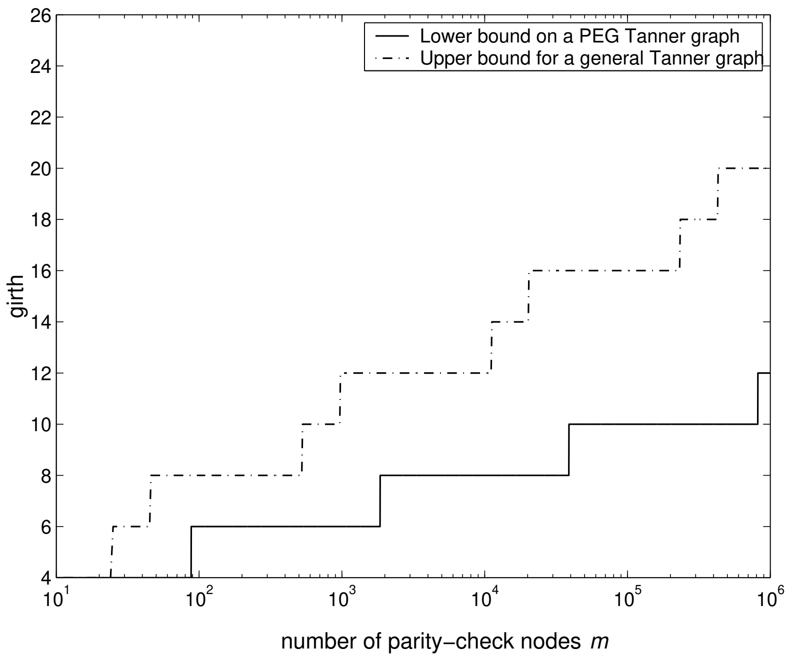

Lower bound on a PEG Tanner graph
Upper bound for a general Tanner graph
girth
number of parity−check nodes m

Fig. 5.

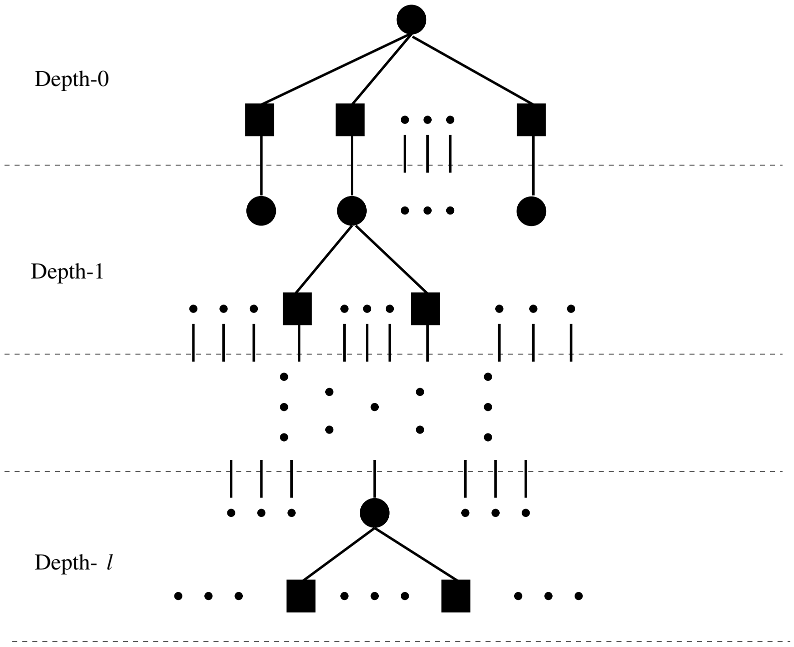

Depth-0
Depth-1
Depth- l

Fig. 6.

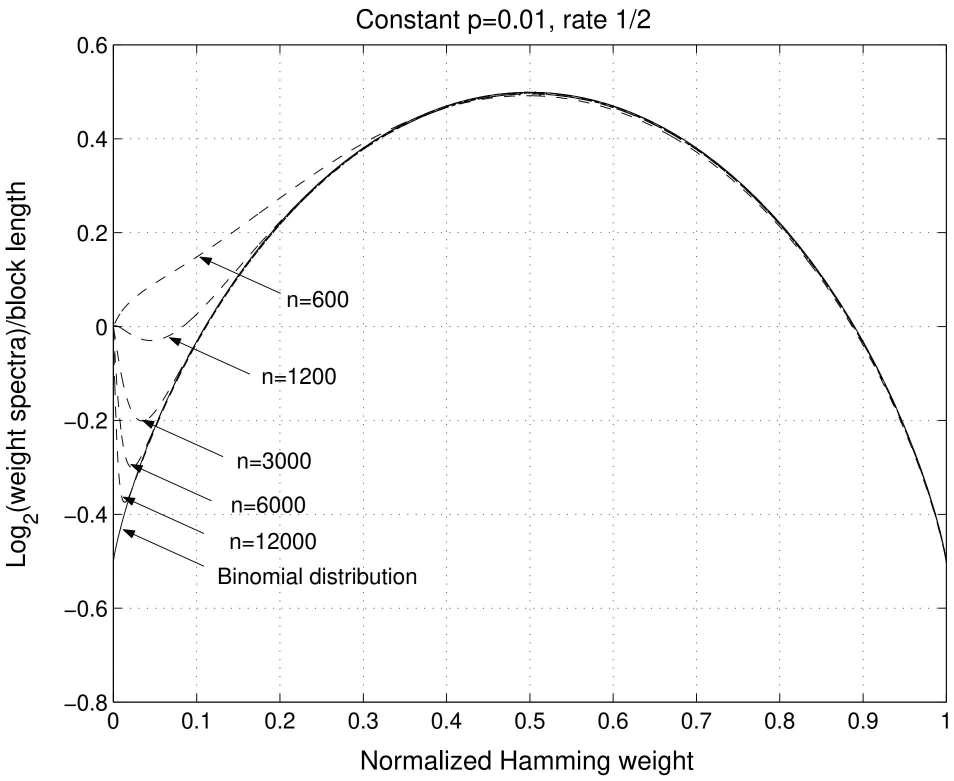

Constant p=0.01, rate 1/2
Log2(weight spectra)/block length
Normalized Hamming weight
n=600
n=1200
n=3000
n=6000
n=12000
Binomial distribution
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Fig. 7.

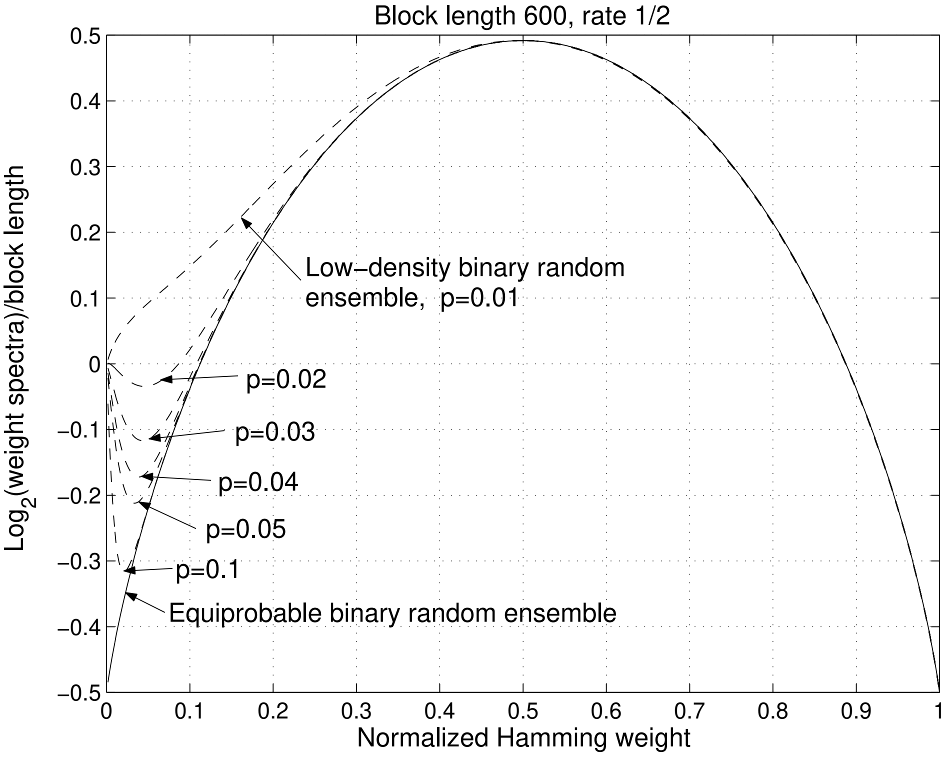

Block length 600, rate 1/2
Log2(weight spectra)/block length
Normalized Hamming weight
Low−density binary random ensemble, p=0.01
p=0.02
p=0.03
p=0.04
p=0.05
p=0.1
Equiprobable binary random ensemble

Fig. 8.

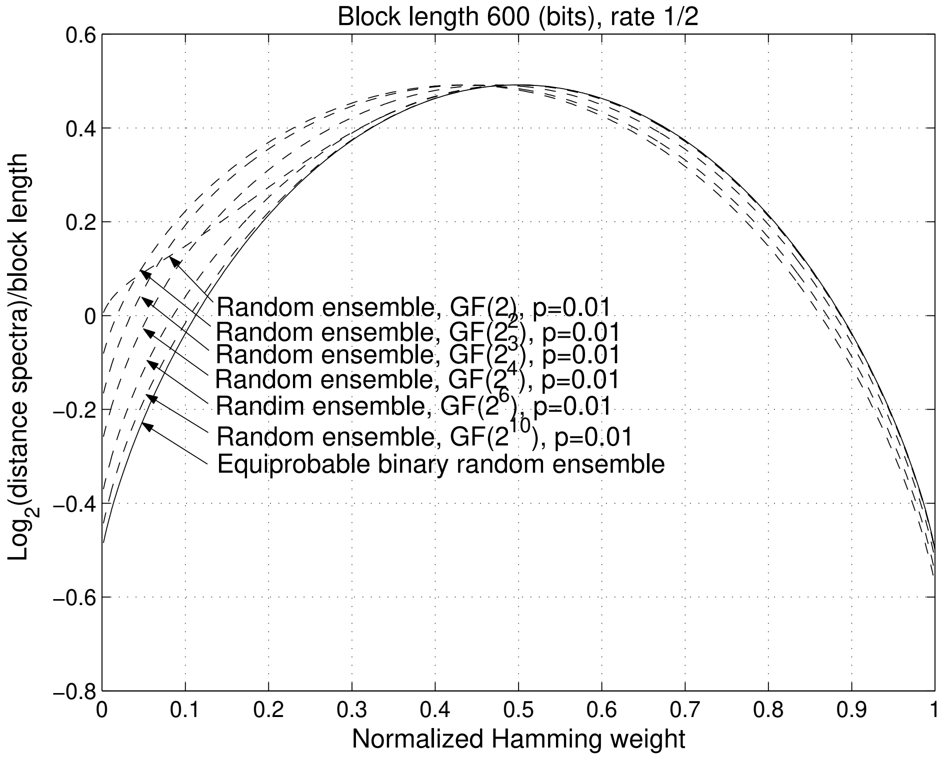

Block length 600 (bits), rate 1/2
Log2(distance spectra)/block length
Normalized Hamming weight
Random ensemble, GF(2), p=0.01
Random ensemble, GF(2^2), p=0.01
Random ensemble, GF(2^3), p=0.01
Random ensemble, GF(2^4), p=0.01
Randim ensemble, GF(2^6), p=0.01
Random ensemble, GF(2^10), p=0.01
Equiprobable binary random ensemble

Fig. 9.

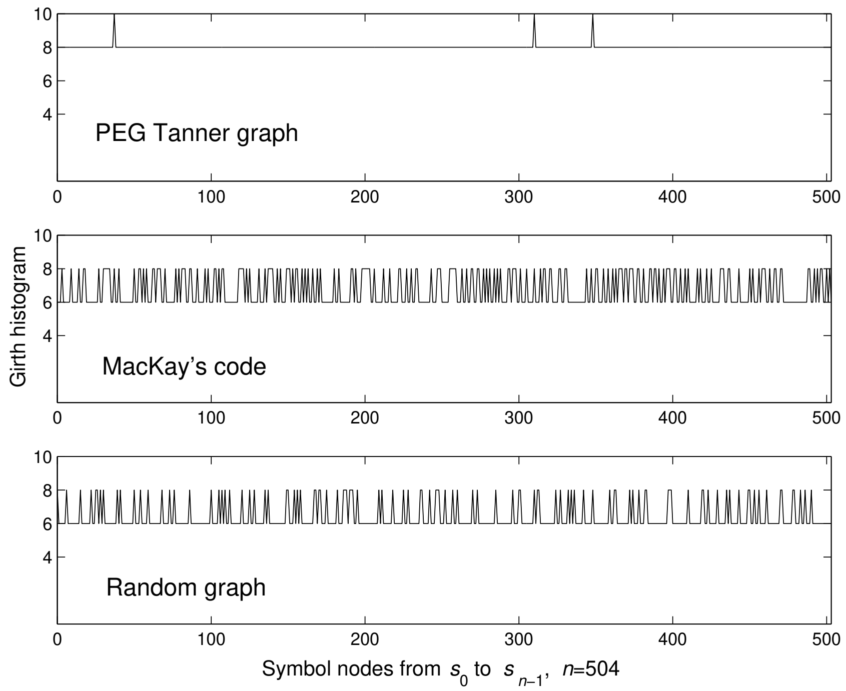

PEG Tanner graph
MacKay's code
Random graph
Girth histogram
Symbol nodes from s_0 to s_{n-1}, n=504

Fig. 10.

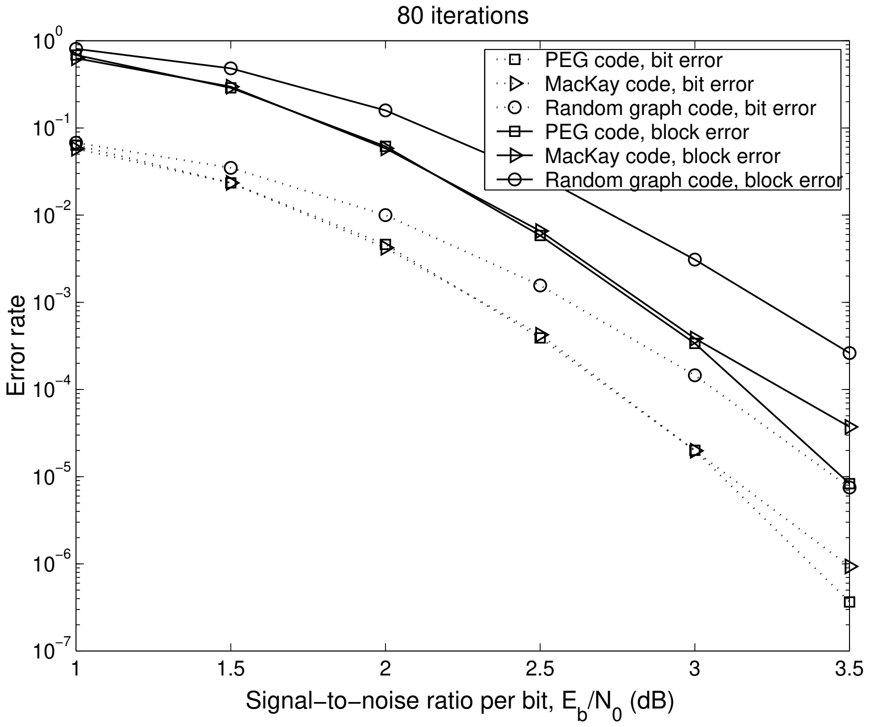
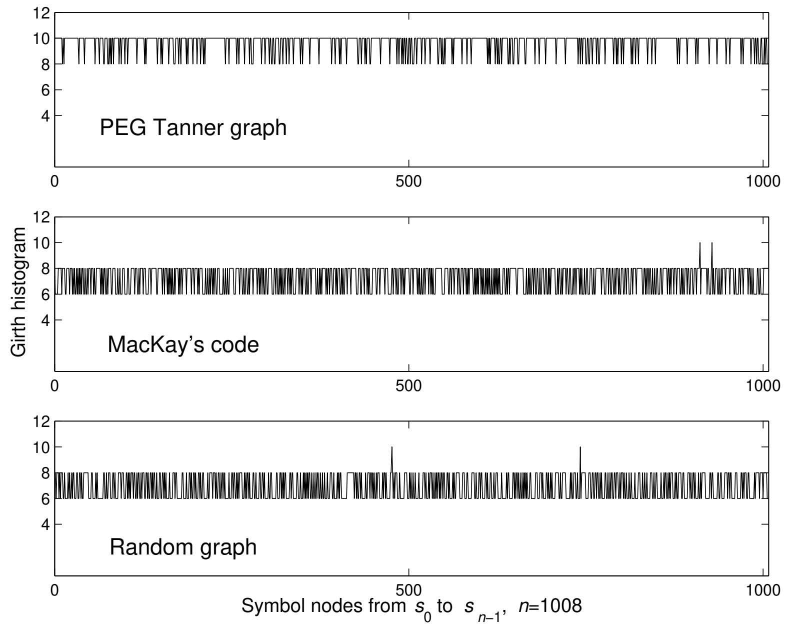
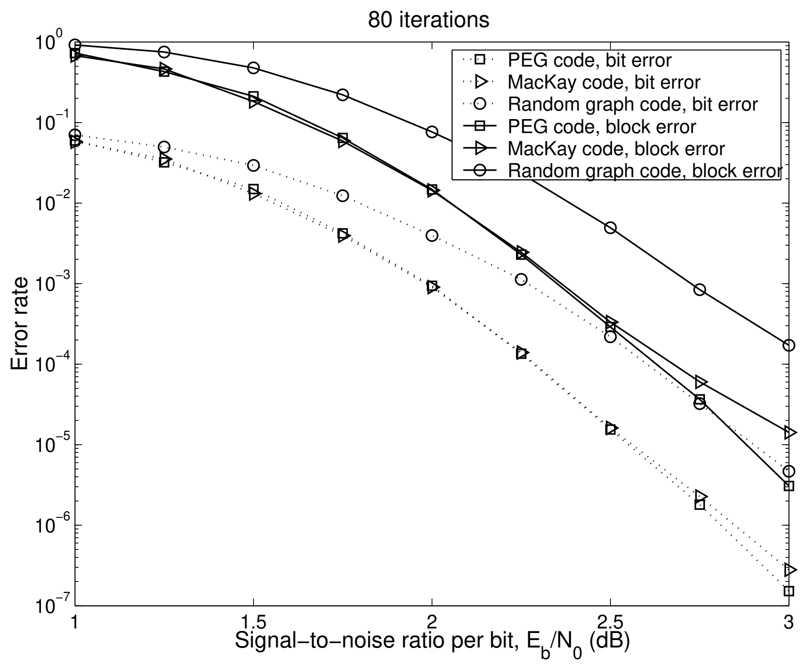

80 iterations
PEG code, bit error
MacKay code, bit error
Random graph code, bit error
PEG code, block error
MacKay code, block error
Random graph code, block error
Error rate
Signal−to−noise ratio per bit, E_b/N_0 (dB)

Fig. 11.



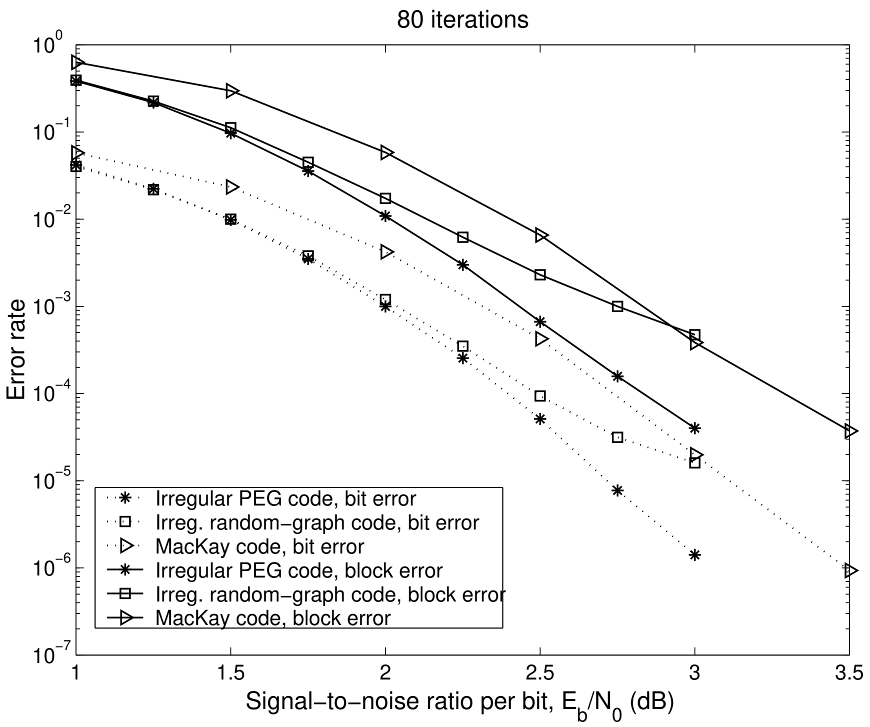
PEG Tanner graph
MacKay's code
Random graph
Girth histogram
Symbol nodes from s_0 to s_{n-1}, n=1008

Fig. 12.



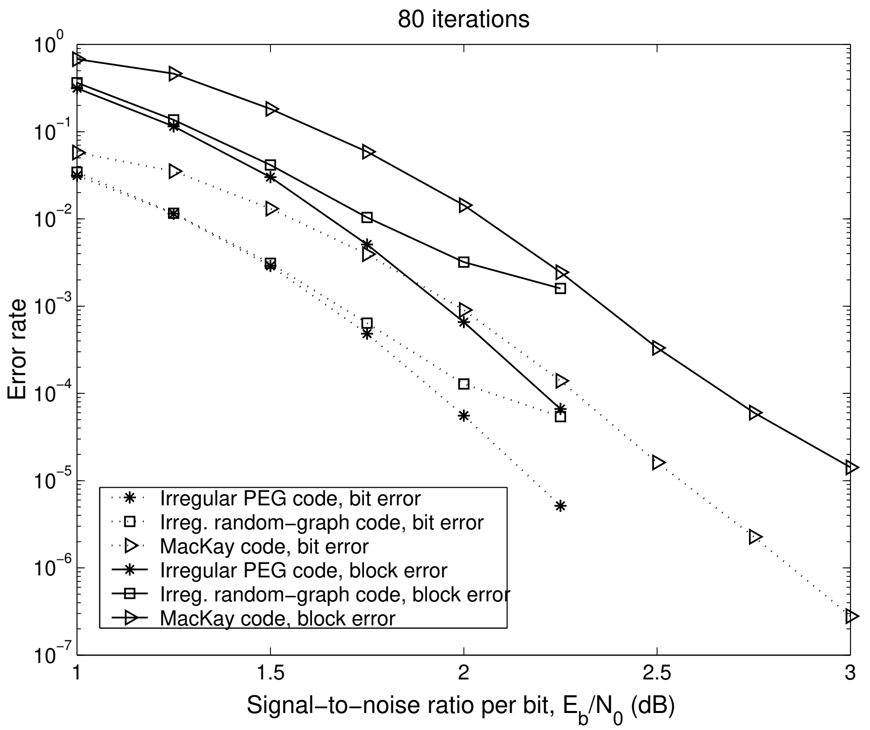
80 iterations
PEG code, bit error
MacKay code, bit error
Random graph code, bit error
PEG code, block error
MacKay code, block error
Random graph code, block error
Error rate
Signal−to−noise ratio per bit, E_b/N_0 (dB)

Fig. 13.



80 iterations
Error rate
Signal−to−noise ratio per bit, E_b/N_0 (dB)
Irregular PEG code, bit error
Irreg. random−graph code, bit error
MacKay code, bit error
Irregular PEG code, block error
Irreg. random−graph code, block error
MacKay code, block error

Fig. 14.



80 iterations
Error rate
Signal−to−noise ratio per bit, E_b/N_0 (dB)
Irregular PEG code, bit error
Irreg. random−graph code, bit error
MacKay code, bit error
Irregular PEG code, block error
Irreg. random−graph code, block error
MacKay code, block error

Fig. 15.

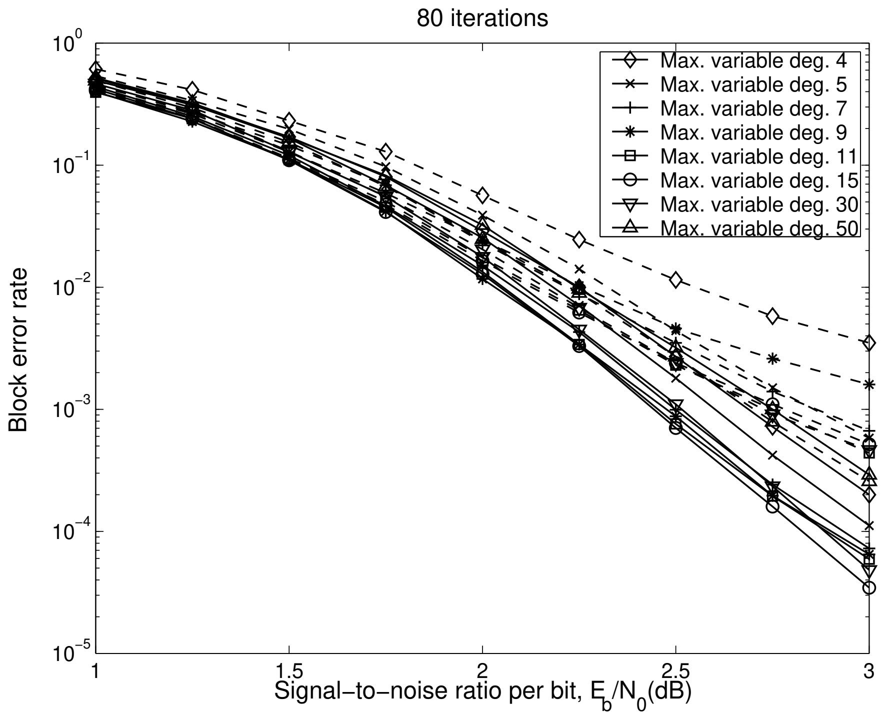

80 iterations
Block error rate
Signal−to−noise ratio per bit, E_b/N_0(dB)
Max. variable deg. 4
Max. variable deg. 5
Max. variable deg. 7
Max. variable deg. 9
Max. variable deg. 11
Max. variable deg. 15
Max. variable deg. 30
Max. variable deg. 50

Fig. 16.

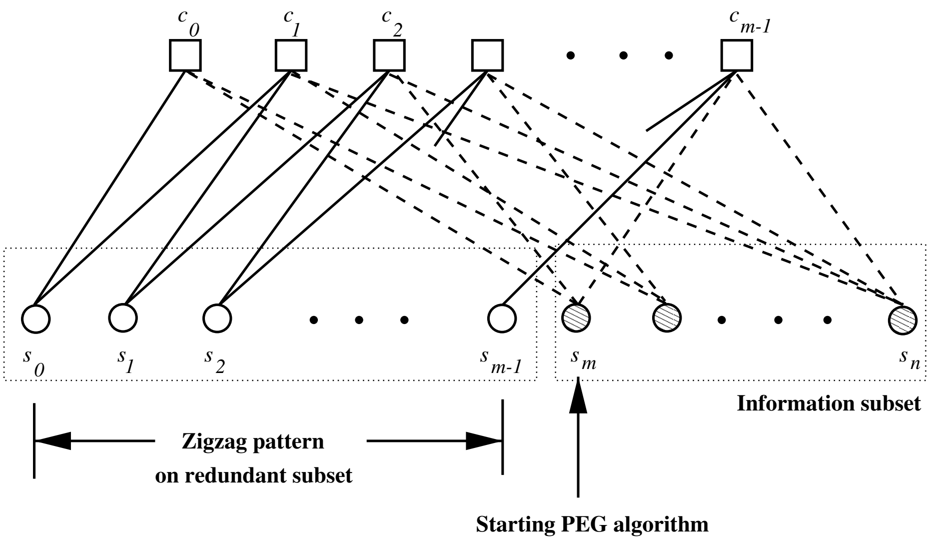

c_0 c_1 c_2 c_{m-1}
s_0 s_1 s_2 s_{m-1} s_m s_n
Information subset
Zigzag pattern on redundant subset
Starting PEG algorithm

Fig. 17.

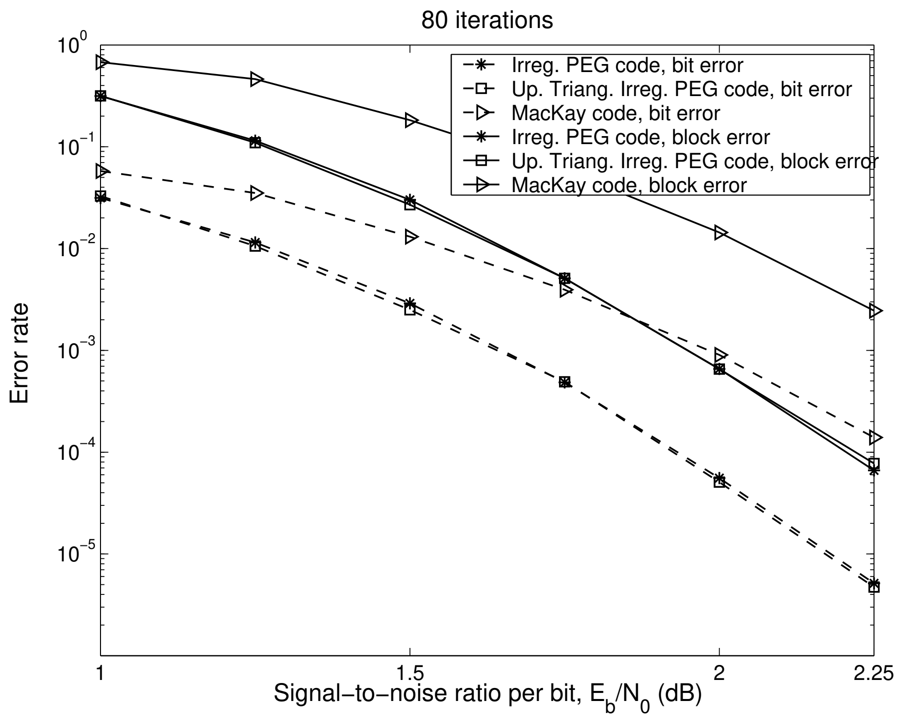

80 iterations
Irreg. PEG code, bit error
Up. Triang. Irreg. PEG code, bit error
MacKay code, bit error
Irreg. PEG code, block error
Up. Triang. Irreg. PEG code, block error
MacKay code, block error
Error rate
Signal−to−noise ratio per bit, E_b/N_0 (dB)

Fig. 18.

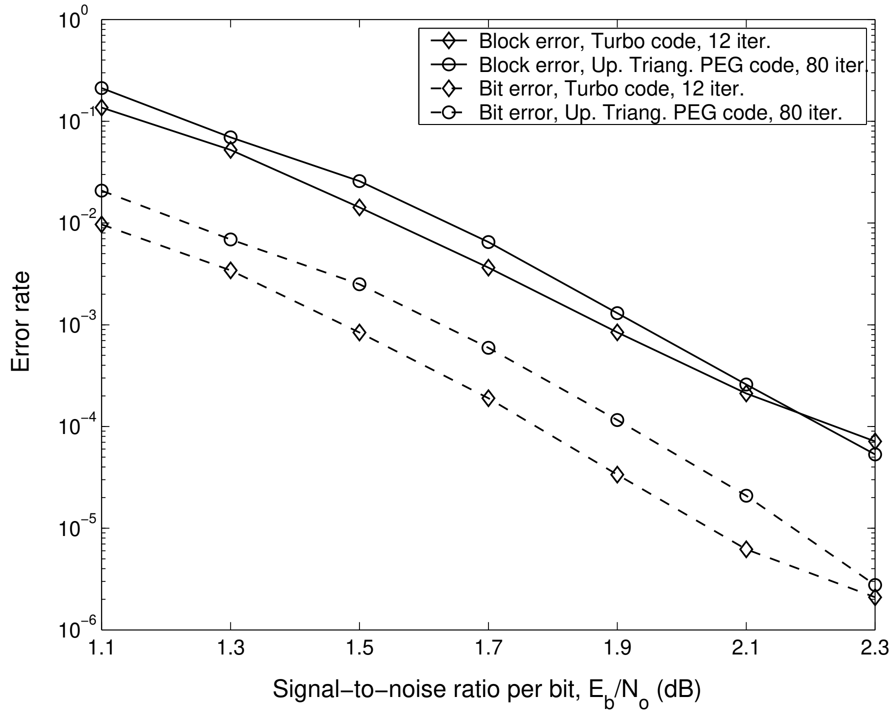

Block error, Turbo code, 12 iter.
Block error, Up. Triang. PEG code, 80 iter.
Bit error, Turbo code, 12 iter.
Bit error, Up. Triang. PEG code, 80 iter.
Error rate
Signal−to−noise ratio per bit, E_b/N_o (dB)

Fig. 19.

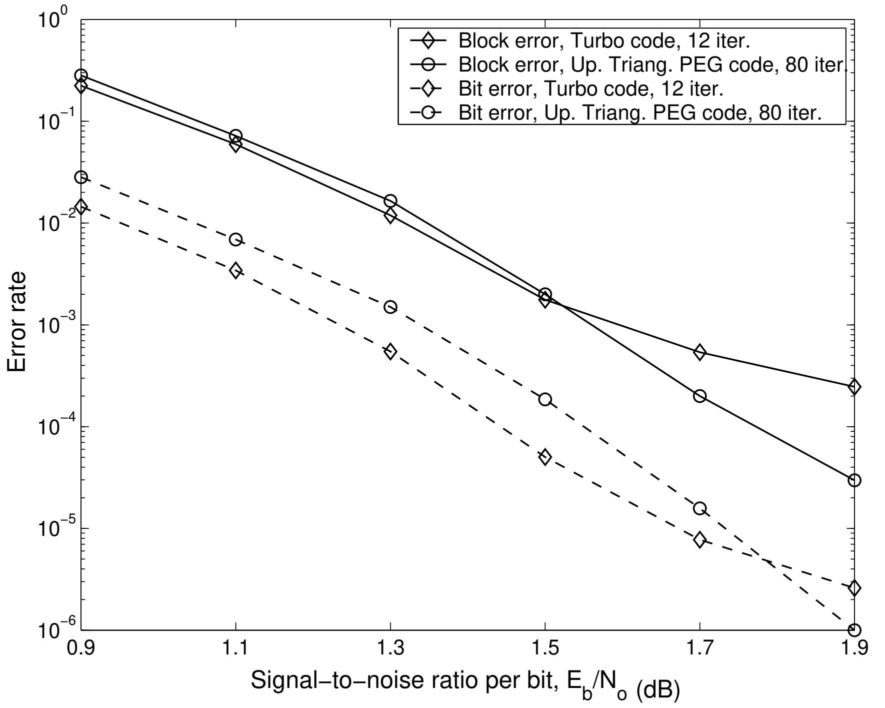

Block error, Turbo code, 12 iter.
Block error, Up. Triang. PEG code, 80 iter.
Bit error, Turbo code, 12 iter.
Bit error, Up. Triang. PEG code, 80 iter.
Error rate
Signal−to−noise ratio per bit, E_b/N_o (dB)

Fig. 20.

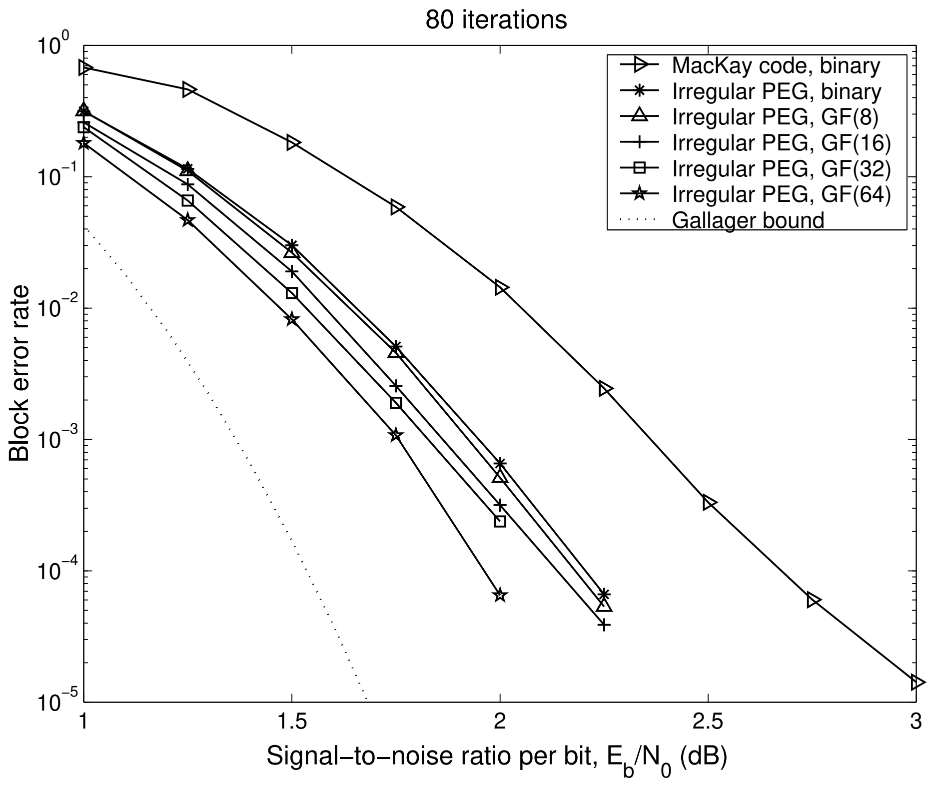

80 iterations
MacKay code, binary
Irregular PEG, binary
Irregular PEG, GF(8)
Irregular PEG, GF(16)
Irregular PEG, GF(32)
Irregular PEG, GF(64)
Gallager bound
Block error rate
Signal−to−noise ratio per bit, E_b/N_0 (dB)

Fig. 21.