Following the same argument in Section IV, $T = \sum_i \lambda_i$, we have the spectral entropy

$$H(S) = -\sum_i \frac{\lambda_i}{T} \log \frac{\lambda_i}{T}.$$

Then the Campbell bandwidth is $W_c = (1/2)e^{H(S)}$, and we can say the Campbell bandwidth is the minimum average bandwidth for encoding the process across all possible distortion levels.

IX. Conclusion

We have presented two new derivations of the coefficient rate introduced by Campbell. One derivation solidifies its interpretation as a coefficient rate, and shows that the spectral entropy of a random process is proportional to the logarithm of the equivalent bandwidth of the smallest frequency band that contains most of the energy. The second derivation implies that the number of samples of a particular component should be proportional to the variance of that component. We discussed the implications of the latter result for realization-adaptive source coding and provided a connection with the familiar reverse water-filling result from rate distortion theory. From the coefficient rate, we defined a quantity called the Campbell bandwidth of a random process, and we contrasted Fourier bandwidth, Shannon bandwidth, and Campbell bandwidth.

Acknowledgment

The authors are indebted to the referees for their constructive comments and insights.

References

[1] L. Campbell, "Minimum coefficient rate for stationary random processes," *Inf. Contr.*, vol. 3, no. 4, pp. 360–371, Dec. 1960.
[2] T. Berger, *Rate Distortion Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1968.
[3] N. Abramson, "Information theory and information storage," in *Proc. Symp. System Theory*, New York, Apr. 1965, pp. 207–213.
[4] J. D. Gibson, S. P. Stanners, and S. A. McClellan, "Spectral entropy and coefficient rate for speech coding," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Nov. 1993, pp. 925–929.
[5] S. McClellan and J. D. Gibson, "Variable-rate CELP based on subband flatness," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 3, pp. 120–130, Mar. 1997.
[6] J. D. Gibson, M. L. Moodie, and S. A. McClellan, "Variable rate techniques for CELP speech coding," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Pacific Grove, CA, Oct. 29 –Nov. 1, 1995.
[7] S. McClellan and J. D. Gibson, "Variable rate tree coding of speech," in *Proc. 1994 IEEE Wichita Conf. Communications, Networking, and Signal Processing*, Wichita, KS, Apr. 1994.
[8] R. Mester and U. Franke, "Spectral entropy-activity classification in adaptive transform coding," *IEEE J. Sel. Areas Commun.*, vol. 10, no. 5, pp. 913–917, Jun. 1992.
[9] R. R. Coifman and M. V. Wickerhauser, "Entropy-based algorithms for best basis selection," *IEEE Trans. Inf. Theory*, vol. 38, no. 2, pp. 713–718, Mar. 1992.
[10] E. Wesfreid and M. V. Wickerhauser, "Adapted local trigonometric transforms and speech processing," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3596–3600, Dec. 1993.
[11] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 243–250, Jun. 1996.
[12] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3445–3462, Dec. 1993.
[13] D. L. Donoho, M. Vetterli, R. A. DeVore, and I. Daubechies, "Data compression and harmonic analysis," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2435–2476, Oct. 1998.
[14] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 23–50, Nov. 1998.
[15] M. Effros, "Optimal modeling for complex system design," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 50–73, Nov. 1998.
[16] S. Mallat and F. Falzon, "Analysis of low bit rate image transform coding," *IEEE Trans. Signal Process.*, vol. 46, no. 4, pp. 1027–1042, Apr. 1998.
[17] J. L. Massey, "Toward an information theory of spread-spectrum systems," *Code Division Multiple Access Commun.*, pp. 29–46, 1995.
[18] U. Grenander and G. Szego, *Toeplitz Forms and Their Applications*. Berkeley, CA: Univ. Calif. Press, 1958.
[19] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.
[20] L. Rade, *Beta Mathematics Handbook*. Boca Raton, FL: CRC, 1992.
[21] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA: Kluwer Academic, 1991.
[22] C. E. Shannon, "Communication in the presence of noise," *Proc. IRE*, vol. 37, pp. 10–21, Jan. 1949.

Regular and Irregular Progressive Edge-Growth Tanner Graphs

Xiao-Yu Hu, *Member, IEEE*, Evangelos Eleftheriou, *Fellow, IEEE*, and Dieter M. Arnold, *Member, IEEE*

***Abstract*—We propose a general method for constructing Tanner graphs having a large girth by establishing edges or connections between symbol and check nodes in an edge-by-edge manner, called progressive edge-growth (PEG) algorithm. 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. Simple variations of the PEG algorithm can also be applied to generate linear-time encodeable LDPC codes. Regular and irregular LDPC codes using PEG Tanner graphs and allowing symbol nodes to take values over GF(q) ($q > 2$) are investigated. Simulation results show that the PEG algorithm is a powerful algorithm to generate good short-block-length LDPC codes.**

***Index Terms*—Girth, low-density parity-check (LDPC) codes, LDPC codes over GF(q), progressive edge growth (PEG), PEG Tanner graphs.**

I. Introduction

Codes on graphs [1]–[13] 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 [14] and turbo codes [15]

Manuscript received September 2, 2002; revised July 18, 2004. The material in this correspondence was presented in part at 2001 IEEE Global Telecommunications Conference, San Antonio, TX, November 2001.

X.-Y. Hu and E. Eleftheriou are with IBM Research, Zurich Research Laboratory, 8803 Rüschlikon, Switzerland (e-mail: xhu@zurich.ibm.com; ele@zurich.ibm.com).

D. M. Arnold was with IBM Research, Zurich Research Laboratory, Rüschlikon, Switzerland (e-mail: d.m.arnold@ieee.org).

Communicated by K. A. S. Abdel-Ghaffar, Associate Editor for Coding Theory.

Digital Object Identifier 10.1109/TIT.2004.839541

0018-9448/$20.00 © 2005 IEEE

by using rather long block lengths. In fact, for the binary-symmetric channel and sufficiently small crossover probability, it was shown that the decoding-error probability approaches zero with an increasing number of independent iterations [14]. Using the incidence matrix associated with a graph, Gallager proposed an explicit LDPC code construction [14, Appendix C] that guarantees independent decoding iterations up to a lower bound. Unfortunately, this construction is only valid for regular LDPC codes, and seems to be computationally infeasible for large block lengths.

For most existing LDPC codes, the Tanner graph is randomly constructed, avoiding cycles of length 4 [16]–[19]. 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 [16], [20], large girths facilitate iterative decoding and impose a respectable minimum distance bound that enhances decoding performance in a high-signal-to-noise (SNR) regime. Note that a large girth does not automatically imply a large minimum distance. Consider, for example, a code with only one parity check whose girth is infinity, but whose minimum distance is only two. For large block lengths, random graphs work very well, but for short block lengths, the probability of choosing an unfavorable random graph is surprisingly high. As observed in [21], the random ensemble average is to a large degree dominated by such "bad" graphs for short block lengths. The minimum distance issue becomes critical if an irregular degree sequence is used. This suggests that one needs to define an expurgated random ensemble to avoid graphs having short cycles.

Construction of LDPC codes based on finite geometries was reported in [22]. 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 a high rate and very long block length, these codes perform very well under iterative decoding, only a few tenths of a decibel away from the Shannon limit [22]. For more results obtained by various authors, the reader is referred to [23]–[26].

Since the early work of Gallager, the first significant work in constructing LDPC codes based on a *graph-theoretic* algebraic approach was reported in [27]. In [28], [29], explicit group-theoretic constructions of graphs were proposed. The girth of these graphs exceeds the Erdös–Sachs bound [30], which is a nonconstructive lower bound on the girth of random graphs and has the same significance as the Gilbert–Varshamov bound does in the context of the minimum distance of linear codes. The notion of graph expansion was first introduced as an analysis tool in coding theory [3]. Recently, this approach has been pursued even further, with emphasis on constructing LDPC codes having almost the largest girth possible [31], [32].

In this correspondence, 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 progressive edge-growth (PEG) algorithm. Given the number of symbol nodes, the number of check nodes, 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 the next edge. In addition, lower and upper bounds on the girth and a lower bound on the minimum distance are derived in terms of parameters of the underlying PEG Tanner graphs. Simulation results show that the PEG algorithm is a powerful algorithm for generating good regular and irregular LDPC codes of short and moderate block lengths.

Compared with other existing constructions, the significance of the PEG algorithm lies in 1) its simplicity, i.e., its complexity is such that it can easily be used for constructing codes of very large block lengths and good girth guaranteed by the lower bound, and 2) its flexibility, i.e., it successfully generates good codes for any given block length and any rate when using a density-evolution-optimized degree sequence. Moreover, with a slight modification, it can be used to generate linear-time-encodeable LDPC codes.

The remainder of this correspondence is organized as follows. Section II introduces the necessary definitions and notations on graphs. Section III describes the principle and the details of the PEG algorithm. In Section IV, we summarize the graph properties of PEG Tanner graphs; in particular, the lower bounds on the girth and on the minimum distance are derived. We briefly address linear-time encoding based on the PEG principle in Section V. Section VI presents simulation results comparing the performance of regular and irregular LDPC codes defined on PEG Tanner graphs with that of randomly constructed ones. In Section VII, we investigate the performance of PEG Tanner-graph codes over a finite field GF(q). Finally, Section VIII concludes this correspondence.

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 a Tanner graph [1]. As a Tanner graph defines a parity-check matrix and a parity-check matrix corresponds to a Tanner graph, we use the terms Tanner graph and parity-check matrix interchangeably. 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 $E \subseteq V_c \times V_s$, with edge $(c_i, s_j) \in E$ if and only if $h_{i,j} \neq 0$, where $h_{i,j}$ denotes the entry of H at ith row and jth column, $0 \leq i \leq m-1, 0 \leq j \leq n-1$. 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., $d_{s_0} \leq d_{s_1} \cdots \leq d_{s_{n-1}}$, 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 $d_{c_0} \leq d_{c_1} \cdots \leq d_{c_{m-1}}$. 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. Moreover, denote the kth edge incident on s_j by $E_{s_j}^k$, $0 \leq k \leq d_{s_j} - 1$. 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., $D_c = \{5, 5, 5, 5\}$.

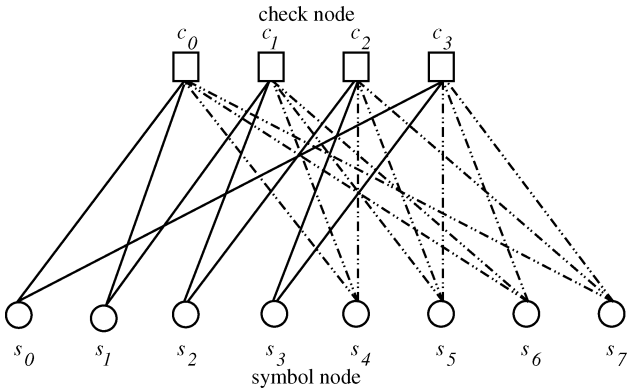

Fig. 1. An example of a symbol-node degree $D_s = \{2, 2, 2, 2, 3, 3, 3, 3\}$ irregular Tanner graph.

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 a pair of vertices, and 3) all edges are nondirected. 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 *neighbors*. 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 joined by an edge. If there exists at least one path between x and y, then x and y are 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 closed path with edges starting from x and ending at x is called a *cycle* of x. *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. By definition, if follows that $g = \min_j \{g_{s_j}\}$.

In general, an ensemble of bipartite or Tanner graphs is characterized by degree distribution pairs. In the case of the symbol nodes, 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 the parity-check nodes, 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 neighborhood 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; 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 we traverse 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 desired depth is reached. Note that in the subgraph duplicate vertices or edges may occur. Referring to Fig. 2, any symbol node residing for the first time at depth l has a distance of $2l$ to s_j, and any check node residing for the first time at depth l has a distance of $2l + 1$ to s_j. 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 neighborhood 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.

III. Progressive Edge-Growth (PEG) Construction

Constructing a Tanner graph with the largest possible girth is a rather difficult combinatorial problem. Nevertheless, a suboptimum algorithm to construct a Tanner graph with a relatively large girth is feasible. 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. Given the graph parameters, i.e., the number of symbol nodes n, the number of check nodes m, and the symbol–node–degree sequence D_s, 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. The underlying graph grows in an edge-by-edge manner, optimizing each local girth. Accordingly, the resulting Tanner graph is referred to as PEG Tanner graph. The fundamental idea is to find the most distant check node and then to place a new edge connecting the symbol node and this most distant check node.

Whenever a subgraph from symbol node s_j is expanded before an edge is established, two situations can occur: 1) the cardinality of $\mathcal{N}_{s_j}^l$ stops increasing but is smaller than m; 2) $\bar{\mathcal{N}}_{s_j}^l \neq \emptyset$, but $\bar{\mathcal{N}}_{s_j}^{l+1} = \emptyset$. In the first case, not all check nodes are reachable from s_j, so the PEG algorithm chooses the one that is not reachable, thus not creating any additional cycle. This often occurs in the initial phase of graph construction. In the second case, all check nodes are reachable from s_j, and the algorithm chooses the one that is at the largest distance from s_j, say at depth $l + 1$, so that the cycle created by establishing an edge is of the largest possible length $2(l + 2)$. 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 a check node such that it has the lowest check-node degree under the current graph setting $E_{s_0} \cup E_{s_1} \cup \cdots \cup E_{s_{j-1}}$.
  else
   expand a subgraph from symbol node s_j up to depth l under the current graph setting such that the cardinality of $\mathcal{N}_{s_j}^l$ stops increasing but is less than m, or $\bar{\mathcal{N}}_{s_j}^l \neq \emptyset$ but $\bar{\mathcal{N}}_{s_j}^{l+1} = \emptyset$, then $E_{s_j}^k \longleftarrow$ edge (c_i, s_j), where $E_{s_j}^k$ is the kth edge incident to s_j and c_i is a check node picked from the set $\bar{\mathcal{N}}_{s_j}^l$ having the lowest check-node degree.
 end
end

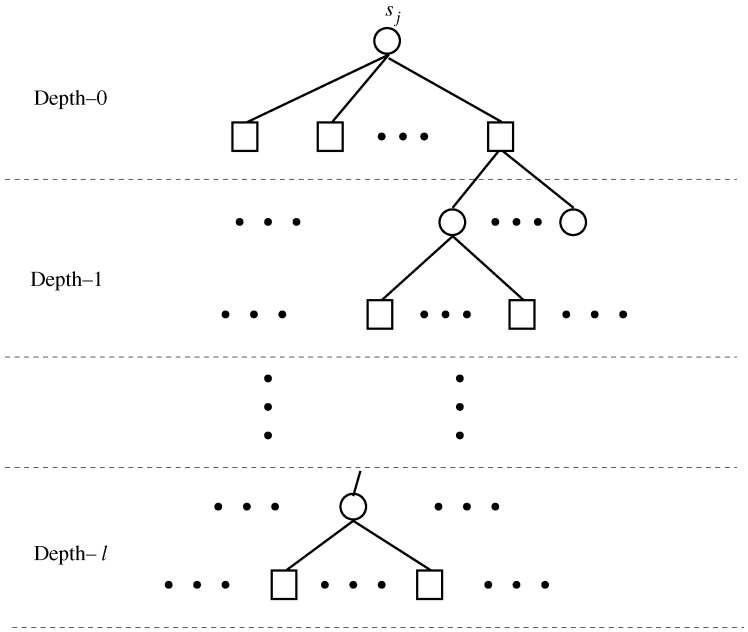

Fig. 2. A subgraph spreading from symbol node s_j.

The set $\mathcal{N}_{s_j}^l$ and its complement $\bar{\mathcal{N}}_{s_j}^l$ can be efficiently obtained in a recursive manner. One can set an indicator $\mathcal{I}_{c_i}$ for each check node c_i taking on values from the set $\{0, 1\}$. The indicator set $\mathcal{I}$ is initialized to 0. As the tree originating in s_j proceeds to depth l, the indicators of all check nodes included in the spanning tree are set to 1, indicating that these nodes belong to $\mathcal{N}_{s_j}^l$. Likewise, $\bar{\mathcal{N}}_{s_j}^l$ is obtained by checking whether the indicator $\mathcal{I}_{c_i}$ equals 0. Note that this simple version is definitely not the most efficient one; nevertheless, it proves to be good enough for generating practical PEG Tanner-graph codes.

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 check-node selection strategy renders the resulting PEG Tanner graphs as check-node-degree uniform as possible. In particular, it tends to produce graphs with uniform degree in parity-check nodes (parity-check-node-regular graphs), or concentrated graphs with two consecutive nonzero degrees. One can easily apply additional constraints on the check-node degree distribution, e.g., by setting appropriate maximum degrees on individual check nodes; this however might not be necessary as there is strong evidence that a concentrated degree sequence on the check nodes is optimum [33], [34].

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 in the initial phase of PEG construction. There are two main approaches to solve this problem. The first is to randomly select one of these check nodes. The second 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. In this correspondence, we adopt the first approach. Note, however, that the second may also be of interest because of its deterministic nature.

We close this section with the following remarks.

1) *Complexity*—The computational load in obtaining the set $\mathcal{N}_{s_j}^l$ or $\bar{\mathcal{N}}_{s_j}^l$ primarily depends on the degree sequences D_s and D_c as well as on the depth l. In a sparse graph, the elements of D_s and D_c are small numbers irrespective of n, and l grows at most logarithmically with m. In the worst case, the computational complexity and the storage requirements of the PEG algorithm scale as $O(n\,m)$ and $O(n)$, respectively, whereas the complexity and storage requirements of Gallager's explicit construction [14, Appendix C] for large girth, in the best case, are both $O(n^2)$.

2) *Nongreedy version*—The version presented above is greedy in the sense that the subgraph spreading from 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 appears to be favorable if the minimum distance is at a premium, particularly for short-block-length and/or high-rate codes [36], [37]. However, for long-block-length, low-rate codes, in which the minimum distance is in principle large, 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, and 2) possibly to reduce the *diameter* of the graph, i.e., the maximum distance of distinct vertex pairs, such that fewer decoding iterations are required. This variant is called the *nongreedy* PEG algorithm. Note that if one sets $l_{\max} = g_t/2 - 2$, where g_t is the target girth, then this variant bears some resemblance to the "bit-filling" algorithm described independently in [38].

3) *Look-ahead-enhanced version*—The PEG principle refers to constructing graphs by attaching edges in stages, where at each stage we choose an edge emanating from a symbol node such that the shortest cycle passing through the assumed edge is locally optimized. Clearly, this local optimization usually does not produce the best possible 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 several choices exist for placing the kth edge of s_j. In this case, additional testing is introduced to ensure that a better choice is used. Specifically, for each candidate parity-check node in $\bar{\mathcal{N}}_{s_j}^l$, we evaluate the maximum possible depth l the subgraph expanded from s_j would have if an edge connecting the candidate parity-check node with s_j had been put onto the graph. Then we select the parity-check node having the largest l as the parity-check node that the kth edge of s_j joins.

4) *Flexibility and scalability*—The PEG algorithm can be used to construct regular and irregular bipartite graphs with arbitrary size. It generates good codes for any given block length and rate, provided a good degree sequence is supplied. Its low complexity makes it suitable for constructing codes of very large lengths and, with a slight modification, for constructing linear-time-encoding LDPC codes. Any other algorithm known, e.g., Gallager's construction, does not have this degree of flexibility. The underlying PEG principle is flexible and broadly applicable; for example, with only a minor modification it can be used to generate graphs that are strictly regular [35]. By incorporating an extra criterion called "approximate cycle-extrinsic message degree (ACE)" [40] at the stage of selecting one check node from the set $\bar{\mathcal{N}}_{s_j}^{l}$, the error floor at high SNRs for irregular PEG codes can be further improved [41]. Rate-compatible LDPC codes based on the PEG principle have recently been published in [42].

IV. Graph Properties

A randomly constructed Tanner graph guarantees neither a meaningful lower bound on the girth nor the minimum distance. In contrast, a PEG Tanner graph exhibits some rather elegant properties in terms of girth and minimum distance.

A. Girth Bounds

The number of independent iterations has been analyzed in [14]. In particular, an upper bound on the maximum number of independent iterations i is derived. This bound is general and applies to any (d_s, d_c)-regular Tanner graph. More importantly, an explicit construction procedure is described by which it is always possible to find a (d_s, d_c)-regular graph for which the maximum number of independent iterations is bounded by $t - 1 < i \leq t$, where the real number t depends on d_s, d_c, and the code block length n. In [14], it was also shown that the girth g and i are related by the following inequality:

$$i < g/4 \leq i + 1.$$

The PEG construction procedure described here also guarantees the existence of a regular or irregular graph whose girth satisfies a lower bound. We use the following lemma to establish a lower bound on the girth of the PEG Tanner graphs.

Lemma 1: 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. Let $\mathcal{N}_{s_j}^{l}$ denote the depth-l neighborhood of any 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

$$l \geq \lfloor t_{\text{low}}^{\text{irr}} \rfloor \tag{1}$$

where $t_{\text{low}}^{\text{irr}}$ is given by

$$t_{\text{low}}^{\text{irr}} = \frac{\log\left(m d_c^{\max} - \frac{m d_c^{\max}}{d_s^{\max}} - m + 1\right)}{\log[(d_s^{\max} - 1)(d_c^{\max} - 1)]} - 1 \tag{2}$$

$\lfloor \cdot \rfloor$ 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 of an irregular Tanner graph which spreads from any symbol node s_j, $s_j \in V_s$, such that $\mathcal{N}_{s_j}^{l} \subset V_c$ and $\mathcal{N}_{s_j}^{l+1} = V_c$. Let also $d_c^{\max}$ and $d_s^{\max}$ be the largest degrees of D_c and D_s, respectively. By definition the depth-0 subgraph contains at most $d_s^{\max}$ parity-check nodes, each giving rise to at most $(d_s^{\max}-1)(d_c^{\max}-1)$ parity-check nodes in the next round of spreading. Thus, there are at most $d_s^{\max}(d_s^{\max}-1)(d_c^{\max}-1)$ check nodes at depth 1. Similarly, there are at most $d_s^{\max}(d_s^{\max}-1)^{l}(d_c^{\max}-1)^{l}$ 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

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

which can be simplified to

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

Let $t_{\text{low}}^{\text{irr}}$ be the solution of the equation

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

that is,

$$t_{\text{low}}^{\text{irr}} = \frac{\log\left(m d_c^{\max} - \frac{m d_c^{\max}}{d_s^{\max}} - m + 1\right)}{\log[(d_s^{\max}-1)(d_c^{\max}-1)]} - 1. \tag{6}$$

Then $l \geq l' = \lfloor t_{\text{low}}^{\text{irr}} \rfloor$. □

Note that the above lemma also holds for a (d_c, d_s)-regular Tanner graph, with $d_c^{\max} = d_c$ and $d_s^{\max} = d_s$. 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

$$g \geq 2(\lfloor t_{\text{low}}^{\text{irr}} \rfloor + 2) \tag{7}$$

where $t_{\text{low}}^{\text{irr}}$ is given by (2).

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 those that provide the shortest cycle in a PEG Tanner graph (V, E), where, without loss of generality, $j_{g/2-1}$ is the largest index among $j_0, j_1, \ldots, j_{g/2-1}$. 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 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 procedure in the PEG algorithm for placing edges successively in the set $E_{j_{g/2-1}}$. Whenever a subgraph from symbol node $s_{j_{g/2-1}}$ is expanded before an edge is established, two cases may occur: 1) the cardinality of $\mathcal{N}_{s_{j_{g/2-1}}}^{l}$ stops increasing but is smaller than m; 2) $\bar{\mathcal{N}}_{s_{j_{g/2-1}}}^{l} \neq \emptyset$, but $\bar{\mathcal{N}}_{s_{j_{g/2-1}}}^{l+1} = \emptyset$. In case 1), not all check nodes are reachable from s_j, so the PEG algorithm chooses the one that is not reachable, thus avoiding the creation of an additional cycle. In case 2), 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 neighborhood $\mathcal{N}_{s_{j_{g/2-1}}}^{l}$ such that

$\bar{\mathcal{N}}^{l}_{s_{j_{g/2-1}}} \neq \varnothing$, but $\bar{\mathcal{N}}^{l+1}_{s_{j_{g/2-1}}} = \varnothing$. Therefore, by making use of Lemma 1 we obtain $g \geq 2(\lfloor t^{\mathrm{irr}}_{\mathrm{low}} \rfloor + 2)$, where $t^{\mathrm{irr}}_{\mathrm{low}}$ is given by (2). □

The bound on the girth justifies 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. Moreover, it can readily be seen that this lower bound is always better than the lower bound guaranteed by Gallager's explicit construction [14, Appendix C].

An upper bound on the girth of a general Tanner graph can be derived based on the approach in [14] in a straightforward way. We state the result without a proof. The reader is referred to [14, Appendix C] for details.

Lemma 2: Let (V, E) be a (d_s, d_c)-regular Tanner graph. The girth g of this graph is upper-bounded by

$$g \leq 4\lfloor t^{\mathrm{reg}}_{\mathrm{upp}} \rfloor + 4 \tag{8}$$

where $t^{\mathrm{reg}}_{\mathrm{upp}}$ is given by

$$t^{\mathrm{reg}}_{\mathrm{upp}} = \frac{\log\left[(m-1)(1-\frac{d_s}{d_c(d_s-1)})+1\right]}{\log[(d_c-1)(d_s-1)]}. \tag{9}$$

Comparing the lower and upper bounds in (7) and (8), respectively, one can easily see that the girth of a regular PEG Tanner graph is always larger than or equal to half of the upper bound. This result is analogous to the asymptotic bounds on the girth of regular graphs having only one type of nodes. Specifically, the asymptotic Erdös–Sachs bound [30] states that a randomly generated regular graph with n vertices and degree r has a girth that is larger than or equal to $(1+o(1))\log_{r-1} n$, with probability approaching 1 as $n \to \infty$, and is half of the asymptotic upper bound $(2+o(1))\log_{r-1} n$ [28]. Using similar arguments as in [14, Appendix C], one can readily derive an analogous Erdös–Sachs asymptotic bound for bipartite (Tanner) graphs and show that the lower bound on the girth of PEG Tanner graphs in (7) always meets this analogous Erdös–Sachs bound. The Erdös–Sachs bound may be regarded as an analog of the Varshamov–Gilbert bound in coding theory, as both are derived from similar arguments. Note that, similarly to the Varshamov–Gilbert bound, the Erdös–Sachs bound is nonconstructive in the sense that the corresponding sequence of graphs is defined nonconstructively. It is not clear from the proof of the Erdös–Sachs bound how to construct a sequence $\{G_j\}$ of regular graphs of degree r explicitly such that $n(\{G_j\}) \to \infty$ and $g(\{G_j\}) \geq (1+o(1))\log_{r-1} n(\{G_j\})$ for $j \to \infty$.

The following lemma provides an even tighter upper bound on the girth of a general Tanner graph.

Lemma 3: Let (V, E) be a (d_s, d_c)-regular Tanner graph. The girth g of the graph is upper-bounded by

$$g \leq \min\{g_1, g_2\} \tag{10}$$

where

$$g_1 = \begin{cases} 4\lfloor t_1 \rfloor + 2, & \text{if } \mathcal{I}_1 = 0 \\ 4\lfloor t_1 \rfloor + 4, & \text{otherwise} \end{cases}$$
$$g_2 = \begin{cases} 4\lfloor t_2 \rfloor + 2, & \text{if } \mathcal{I}_2 = 0 \\ 4\lfloor t_2 \rfloor + 4, & \text{otherwise} \end{cases} \tag{11}$$

in which

$$t_1 = \frac{\log\left[(m-1)(1-\frac{d_s}{d_c(d_s-1)})+1\right]}{\log[(d_c-1)(d_s-1)]}, \tag{12}$$

$$t_2 = \frac{\log\left[(n-1)(1-\frac{d_c}{d_s(d_c-1)})+1\right]}{\log[(d_c-1)(d_s-1)]} \tag{13}$$

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

$$[(d_c-1)(d_s-1)]^{\lfloor t_1 \rfloor} > m - 1 - \frac{d_c(d_s-1)\{[(d_c-1)(d_s-1)]^{\lfloor t_1 \rfloor} - 1\}}{(d_c-1)(d_s-1)-1} \tag{14}$$

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

$$[(d_c-1)(d_s-1)]^{\lfloor t_2 \rfloor} > n - 1 - \frac{d_s(d_c-1)\{[(d_c-1)(d_s-1)]^{\lfloor t_2 \rfloor} - 1\}}{(d_c-1)(d_s-1)-1}. \tag{15}$$

The details of the proof of Lemma 3 can be found in [39].

Fig. 3 depicts both the lower bound on a PEG Tanner graph and the two upper bounds on a general Tanner graph for regular $d_s = 3, d_c = 6$ codes with varying m (in this case, $n = 2m$). It can be seen that the upper bound of Lemma 3 is tighter than that of Lemma 2. Moreover, the lower bound of a PEG Tanner graph is higher than half of the two upper bounds for the entire range of block lengths. The lower bound corresponding to Gallager's construction is also shown in Fig. 3, illustrating that the lower bound of a PEG Tanner graph is better than that of Gallager's explicit construction.

Compared with Gallager's explicit construction, the PEG construction in general achieves a better girth with much less complexity. The PEG algorithm is quite simple, whereas the complexity of Gallager's explicit construction remains prohibitively large for medium and large block lengths. So far, we are not aware of practical LDPC codes based on Gallager's explicit construction. More importantly, the PEG algorithm can also be applied to generate irregular graphs, whereas Gallager's construction only applies to regular graphs. The flexibility of the PEG algorithm even allows us to design linear-time-encoding LDPC codes without sacrificing decoding performance. It is worthwhile to point out that in practice the lower bound on PEG Tanner graphs can be exceeded. We have designed good LDPC codes based on the nongreedy PEG variant or the look-ahead-enhanced variant that achieve a girth that is larger than the lower bound. In Fig. 3 these codes are indicated by circles, corresponding to $m = 20, 75, 430, 3000, 30\,000$, and $300\,000$.

B. Minimum-Distance Bound

Assume that 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 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 block length n, for d_s and d_c constant [14]. 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 of a generic linear code is an NP-hard problem, some bounds on the minimum distance of a general Tanner graph have been established in [1], [43]. For a PEG Tanner graph, it is possible to derive a lower bound on the minimum distance in a similar way. In fact, using Tanner's approach [1] and arranging the graph in tree form with the symbol node as root, it can readily be shown that for a (d_s, d_c)-regular graph with $(d_s \geq 3)$, the minimum distance $d_{\min}$ satisfies

$$d_{\min} \geq 1 + \frac{d_s[(d_s-1)^{\lfloor (g-2)/4 \rfloor} - 1]}{d_s - 2}. \tag{16}$$

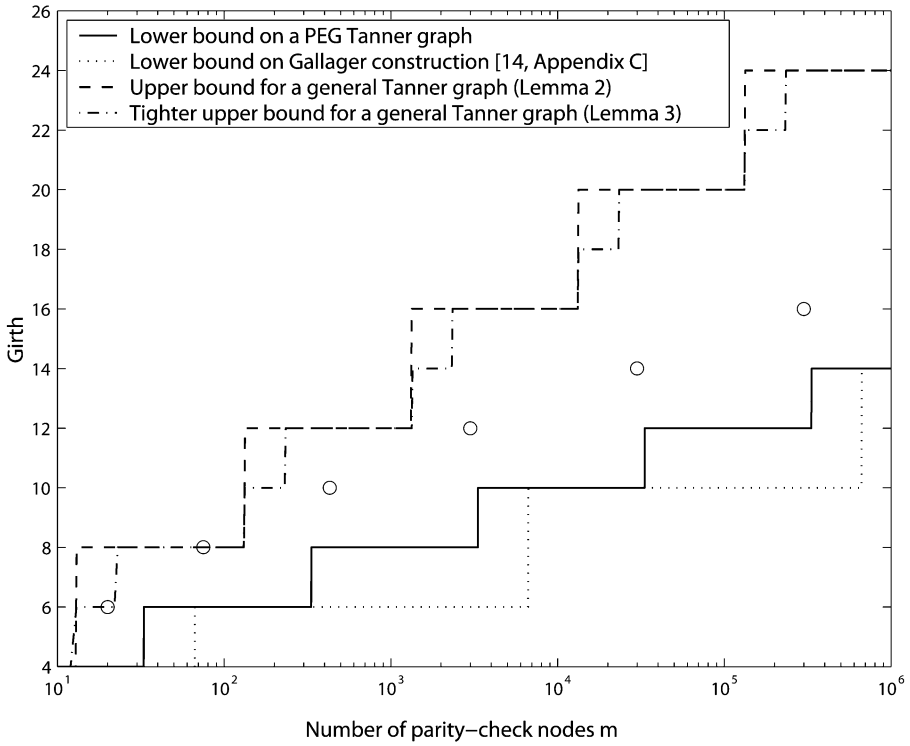

Fig. 3. Lower and upper bounds on a PEG regular Tanner graph with $d_s = 3$, $d_c = 6$.

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

$$d_{\min} \geq 1 + \frac{d_s[(d_s-1)^{\lfloor (g-2)/4 \rfloor} - 1]}{d_s - 2} + (d_s-1)^{\lfloor (g-2)/4 \rfloor}. \quad (17)$$

The proof of this bound is based on counting symbol nodes in an active subgraph induced by a minimum-weight codeword, which is a straightforward variation of the minimum distance result in [1]. Compared with the original Tanner minimum-distance lower bound, (16) and (17) are slightly stronger as in general $d_s \geq 3$. For example, in the case of $g = 6$, (16) reduces to the conventional bound $d_{\min} \geq 1 + d_s$, whereas the lower bound in [1] yields the weaker result $d_{\min} \geq 4 - 2/d_s$, because in general $d_s \geq 3$. In the case of $g = 8$, both bounds yield the same result, namely, $d_{\min} \geq 2d_s$. On the other hand, for $g = 10$, the bound in (16) yields $1 + d_s^2$, whereas the lower bound in [1] leads to the weaker result $4(d_s - 1) + 2/d_s$ for $d_s \geq 3$.

The above bound can readily be extended to the symbol-node-uniform case where the degree sequence of check nodes is not necessarily uniform. Thus, we obtain the following general result.

Lemma 4: 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 $d_{\min}$ of the resulting LDPC code satisfies (18) at the bottom of the page, in which

$$t_{\text{low}}^{\text{irr}} = \frac{\log(m d_c^{\max} - \frac{m d_c^{\max}}{d_s} - m + 1)}{\log[(d_s-1)(d_c^{\max}-1)]} - 1. \quad (19)$$

The proof follows directly from (16), (17), and the lower bound on the girth of PEG Tanner graphs in Theorem 1.

Note that the above bound on the minimum distance still is a weak bound for two reasons. 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 in the active subgraph must be satisfied with additional active symbol nodes has not been taken into account. Nevertheless, (18) always furnishes a meaningful bound on graphs having a large girth.

V. Linear-Time Encoding

The computational complexity per block of iterative decoding using BP or SPA on a Tanner graph has been shown to be essentially linear with block length n, but the encoding complexity per block increases quadratically by n^2. Several publications address this issue, with the aim of obtaining linear-time-encoding complexity, see for example [44]–[49]. The most common approach 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.

The PEG algorithm can easily be tailored to construct LDPC codes having (almost) triangular structure, good girth properties, and an

$$d_{\min} \geq \begin{cases} 1 + \frac{d_s[(d_s-1)^{\lfloor \frac{t_{\text{low}}^{\text{irr}}+1}{2} \rfloor} - 1]}{d_s - 2}, & \text{if } \lfloor t_{\text{low}}^{\text{irr}} \rfloor \text{ is odd} \\ 1 + \frac{d_s[(d_s-1)^{\lfloor \frac{t_{\text{low}}^{\text{irr}}+1}{2} \rfloor} - 1]}{d_s - 2} + (d_s-1)^{\lfloor \frac{t_{\text{low}}^{\text{irr}}+1}{2} \rfloor}, & \text{if } \lfloor t_{\text{low}}^{\text{irr}} \rfloor \text{ is even} \end{cases} \quad (18)$$

optimum irregular degree sequence. According to the linear-time-encoding principle, the codeword w and the parity-check matrix H are partitioned into $w = [p, d]$ and $H = [H^p, H^d]$, respectively, such that

$$[H^p, H^d]w^T = 0 \tag{20}$$

where the $m \times m$ component $H^p = \{h^p_{i,j}\}$ of the parity-check matrix is forced (constructed) to have the special form

$$H^p = \begin{pmatrix} 1 & h^p_{1,2} & \cdots & \cdots & h^p_{1,m} \\ 0 & 1 & & & \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ & & 0 & 1 & h^p_{m-1,m} \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix}_{m \times m} \tag{21}$$

in which $h_{i,j} = 1$ for $i = j$ and $h_{i,j} = 0$ for $i > j$. Hence, the parity-check bits $p = \{p_i\}$ are computed according to

$$p_i = \left(\sum_{j=i+1}^{m} h^p_{i,j} p_j + \sum_{j=1}^{n-m} h^d_{i,j} d_j \right) \mod 2 \tag{22}$$

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. Equation (22) is computed recursively from $i = m$ to $i = 1$. 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 is sparse by design.

Accordingly, we partition the symbol node set V_s in the Tanner-graph representation into the redundant subset V^p_s and the information subset V^d_s, which contain 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 (21), so that the linear-time-encoding property results. As the procedure of establishing the edges of $n-m$ information bits follows the construction of edges of V^p_s and is exactly the same as that described in Section III, we focus on the modified PEG algorithm for constructing edges of V^p_s.

PEG Algorithm for Establishing Edges of V^p_s:
for $j = 0$ to $m-1$ **do**
begin
 for $k = 0$ to $d_{s_j} - 1$ **do**
 begin
  if $k = 0$
   $E^0_{s_j} \longleftarrow$ edge (c_j, s_j), where $E^0_{s_j}$ is the first edge incident to s_j. This edge corresponds to the "1" in the diagonal line of matrix H^p.
  else
   expand a subgraph from symbol node s_j up to depth l under the current graph setting such that $\bar{\mathcal{N}}^l_{s_j} \cap \{c_0, c_1, \ldots, c_{j-1}\} \neq \varnothing$ but $\bar{\mathcal{N}}^{l+1}_{s_j} \cap \{c_0, c_1, \ldots, c_{j-1}\} = \varnothing$, or the cardinality of $\mathcal{N}^l_{s_j}$ stops increasing, then $E^k_{s_j} \longleftarrow$ edge (c_i, s_j), where $E^k_{s_j}$ is the kth edge incident to s_j and c_i is a check node picked from the set $\bar{\mathcal{N}}^l_{s_j} \cap \{c_0, c_1, \ldots, c_{j-1}\}$ having the lowest check-node degree.
 end
end

VI. Code Performance

In this section, we first 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 [50], which is based on a regular Tanner graph with $d_s = 3$, $d_c = 6$. This code was randomly constructed followed by *ad hoc* optimization procedures, and has been widely used as a benchmark. 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 eight check nodes with a degree of 7, and eight 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 1's 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 guarantee that no four cycles occur in the graph representation.

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. Fig. 4 shows a perspective of the girth properties of the various graphs. It depicts the girth of the left-hand subgraph of symbol node s_j as a function of j. The left-hand subgraph of s_j consists of the symbol nodes $\{s_0, s_1, \ldots, s_{j-1}\}, 0 \leq j \leq n-1$, the edges that emanate from them, and the parity-check nodes they are connected to. It is desirable, in particular for irregular LDPC codes, that the girth of the left-hand subgraph of s_j decreases slowly as a function of j such that the possibility that lower degree nodes together form a small cycle decrease. Furthermore, for irregular Tanner graphs, lower degree symbol nodes intuitively require more iterations during the decoding process than higher degree symbol nodes do, and they are also more likely to lead to low-weight codewords. Therefore, having a large girth on the left-hand subgraph of lower degree symbol nodes is a nice property inherent in the PEG construction. In addition, optimizing the girth of the left-hand subgraph can also facilitate the design of LDPC codes having as high a rate as possible while satisfying the requirement of a good global girth. One can easily think of the following (minor) improvement to the generic PEG algorithm obtained by adding an extra procedure to make the girth of the left-hand subgraph decrease as slowly as possible: If the local girth of the current symbol node is less than the girth of its left-hand subgraph (before the current-working symbol node), which indicates a decrease in the girth of the left-hand subgraph, we simply discard all the edges for the the current symbol node and redo the PEG algorithm for it (with random seeds) until a maximum number of trials has been made or the local girth of the symbol node is no longer less than the girth of its left-hand subgraph. It is empirically observed that this modification will not always yield a noticeable improvement, but on some occasions—down to very low bit- or block-error rates—it can improve the error floor due to near- (pseudo-) codewords.

Fig. 5 compares the bit- and block-error rates for the three codes after 80 iterations over a binary-input additive white Gaussian noise (AWGN) channel, and shows 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 collect at least 100 block errors per simulation point. We observe that the LDPC code based on the PEG Tanner graph is 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 are among 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 the girth histogram may be of greater importance than the girth for the performance of iterative decoding. For instance, in [51], the average of the girth histogram is used as a heuristic tool to select good codes from random graphs for short block lengths. Of course, one can also apply the approach of [51] to select good codes among PEG Tanner graphs, anticipating further performance improvements.

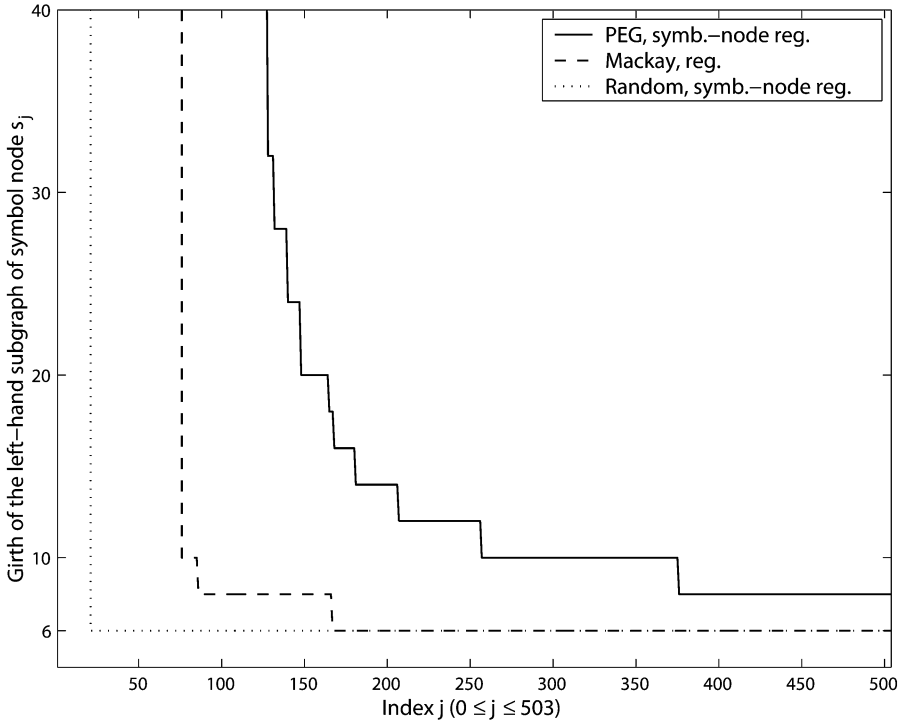

Fig. 4. Girth of the left-hand subgraphs of symbol node s_j in a PEG Tanner graph, MacKay's code, and a random graph, with $n = 504$, $m = 252$, $d_s = 3$, $d_c = 6$.

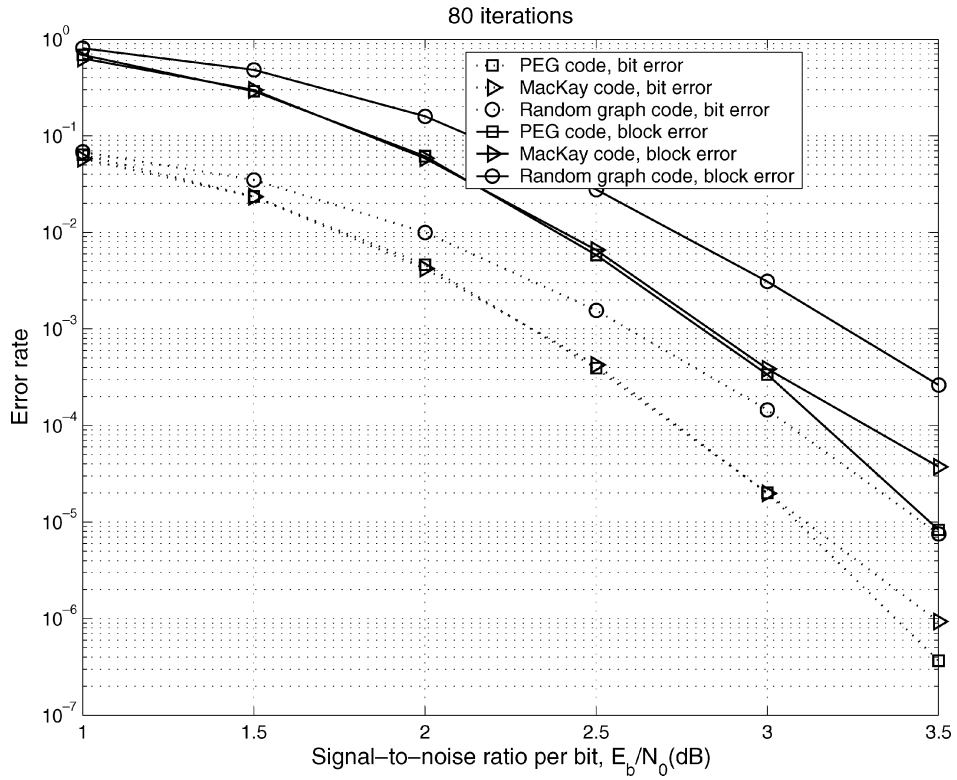

Fig. 5. Bit- and block-error rates of a PEG Tanner-graph code, MacKay's code, and a random graph code, with $n = 504$, $m = 252$, $d_s = 3$, $d_c = 6$.

Density evolution [52], [18] has proved to be an efficient and effective approach to design of 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 algorithm with the symbol-node-degree distribution optimized by density evolution to design LDPC codes. We investigate the performance of symbol-node-degree distributions as given in [19, Tables I and II] using the PEG construction with $n = 504$, $m = 252$. Note that the check-node distribution is not needed as the check-degree sequence is made as uniform as possible by the PEG algorithm. Among these symbol-node distributions with maximum symbol-node degrees $4, 7, 11, 15, 30$, the one

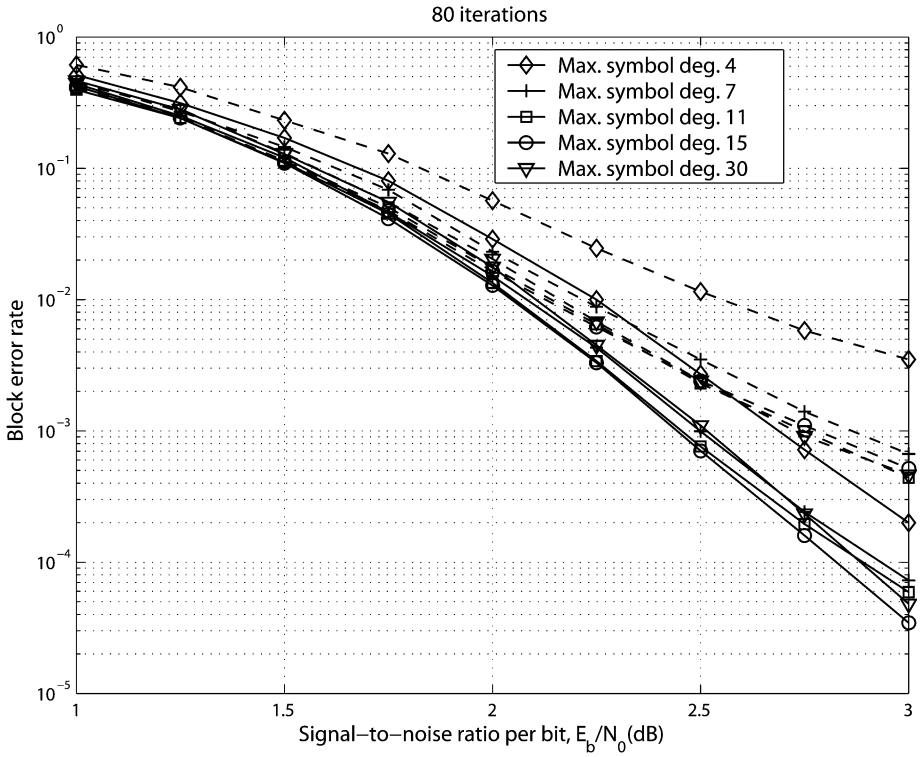

Fig. 6. Block-error rates of irregular PEG (solid lines) and irregular random-graph codes (dashed lines) with density-evolution-optimized degree distributions; code parameters are $n = 504$, $m = 252$.

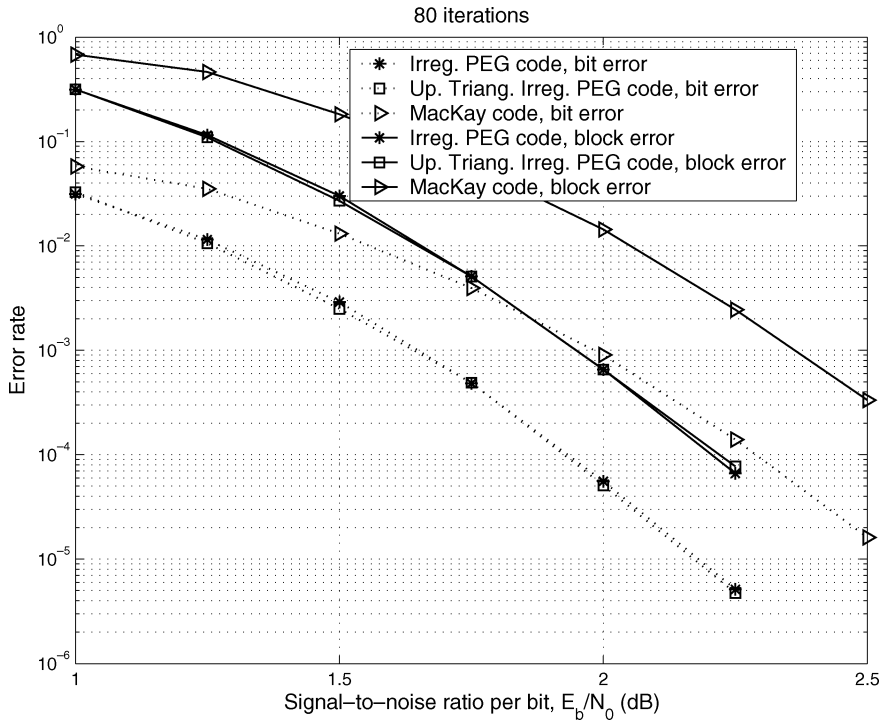

Fig. 7. Bit- and block-error rates of an irregular PEG Tanner-graph code, an upper-triangular PEG Tanner-graph code, and MacKay's code, all with $n = 1008$, $m = 504$.

with maximum degree 15 achieves the best performance, see Fig. 6. This reveals the interesting finding that even for short block lengths, the degree distributions designed by the density-evolution approach still are very good if the PEG algorithm is used. We observe that the PEG construction significantly outperforms the irregular random construction, particularly in the high-SNR region. This irregular random code is constructed column by column with appropriate symbol-node degree. The position of 1's in each column is determined according to a uniformly distributed integer random variable and there are no four cycles in the code's graph representation

We investigate the performance of an irregular PEG Tanner-graph LDPC code whose parity-check matrix is forced into an upper triangular form, thus having linear-time-encoding property. Fig. 7 compares the bit- and block-error rates of the irregular PEG Tanner-graph code,

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

an irregular PEG Tanner-graph code with a parity-check matrix in upper diagonal form, and MacKay's regular code, all with $n = 1008$, $m = 504$. The symbol-node-degree distribution for both irregular PEG Tanner-graph codes chosen is that from [19, Table II] with maximum symbol-node degree 15. When the parity-check matrix is forced into an upper triangular form there is 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, which suggests that with the PEG construction linear-time encoding can be achieved without noticeable performance degradation.

VII. 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 way as in [53], [54], i.e., by allowing the symbol nodes to assume values from the finite field. As a symbol from the field GF(q), $q = 2^b$ for some integer $b > 1$, 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 consecutive 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, i.e.,

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

Here $f_{y_i}^{x_i}$ is the likelihood that 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.

Let us briefly recall the construction of PEG-based LDPC codes over GF(q). 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 initially started in exactly the same manner as in the binary case, i.e., 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 a variant of the "downhill simplex" method described in [39]. We compare codes having a block length of n symbols over GF(2^b) with binary codes of length nb bits.

The performance results indicate that PEG Tanner-graph codes over higher order fields significantly outperform the binary ones. Furthermore, thanks to the PEG algorithm, which aims at large girth as well as an optimized (by downhill search) irregular degree sequence, we have observed a *monotonic* improvement with increasing field order. It is also observed that the optimum degree sequence favors a lower average column weight as the field order increases. 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. This new insight complements the findings in [53], [54].

VIII. CONCLUSION

A general method for constructing Tanner graphs with large girth has been presented. Its main principle is to optimize the placement of a new edge connecting a particular symbol node to a 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 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 and check nodes, as well as on the maximum values of the symbol- and check-node degrees of the underlying graph. In addition, a lower bound on the minimum distance of binary LDPC codes defined on PEG Tanner graphs has also been derived.

Simulation results demonstrated that using the PEG algorithm for constructing short-block-length LDPC codes results in a significant improvement compared with randomly constructed codes. We empirically found that even for small block lengths, such as $n = 504$, there is a good degree distribution from density evolution that works perfectly with the PEG construction. Linear-time encodeable LDPC codes have also been constructed by slightly modifying the PEG algorithm to yield a Tanner graph with triangular format. This easy encoding property can be achieved without noticeable performance degradation.

Fig. 8 shows the performance of the irregular PEG Tanner-graph codes over a binary-input AWGN 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). Furthermore, the overall gain of the GF(2^6) PEG code compared with the binary MacKay code is approximately 0.75 dB. Finally, the rate-$1/2$ irregular PEG code over

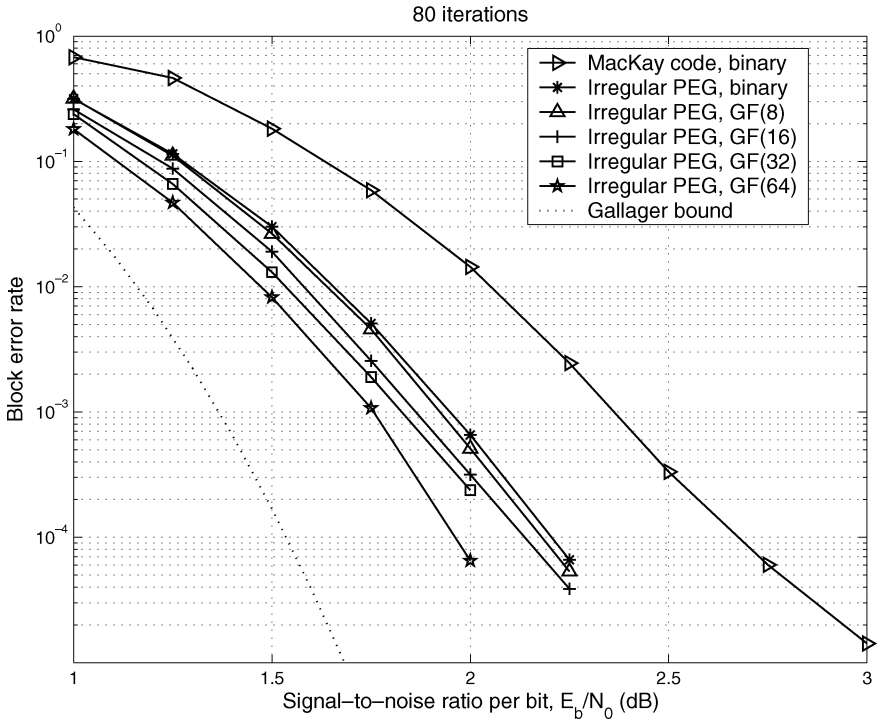

Fig. 8. Block-error rates of irregular LDPC codes over GF(2), GF(8), GF(16), GF(32), and GF(64), based on a PEG Tanner graph with the parameters given in Table I.

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 Gallager's sphere-packing bound[1] of a binary-input AWGN channel [55], [56], which appears to be the best iterative-decoding performance at this block length and rate known today.

Finally, the regular and irregular binary 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, to our knowledge, appears to exhibit the best iterative-decoding performance at this short block length achieved to date.

ACKNOWLEDGMENT

The authors are grateful to M. P. C. Fossorier, T. Mittelholzer, R. L. Urbanke, and M. Vetterli for insightful comments and suggestions. They also thank D. J. C. MacKay for valuable comments and for providing the parity-check matrices of some LDPC codes simulated herein. They thank the Associate Editor and two anonymous reviewers for constructive comments, which have greatly helped improve the exposition of the correspondence.

REFERENCES

[1] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. IT-27, no. 6, pp. 533–547, Sep. 1981.

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

[1]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.

[3] M. Sipser and D. A. Spielman, "Expander codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pp. 1710–1722, Nov. 1996.

[4] N. Alon and M. Luby, "A linear-time erasure-resilient code with nearly optimal recovery," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, 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. Sel. 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, no. 2, pp. 219–230, Feb. 1998.

[7] R. Kötter and A. Vardy, "Factor graphs: Construction, classification, and bounds," in *Proc. 1998 IEEE Int. Symp 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. Inf. Theory*, vol. 47, no. 2, 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. Inf. Theory*, vol. 45, no. 5, pp. 1435–1455, Jul. 1999.

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

[11] S. M. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 325–343, Mar. 2000.

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

[13] "Special issue on codes and graphs and iterative algorithms," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, Feb. 2001.

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

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

[16] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

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

[18] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[19] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of provably good low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[20] 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.

[21] C. Di, D. Proietti, İ. E. Telatar, T. Richardson, and R. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 48, no. 6, pp. 1570–1579, Jun. 2002.

[22] 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. Inf. Theory*, vol. 47, no. 7, pp. 2711–2736, Nov. 2001.

[23] R. M. Tanner, D. Sridhara, and T. Fuja, "A class of group-structured LDPC codes," in *Proc. ICSTA 2001,* Ambleside, UK, 2001.

[24] J. L. Fan, "Array codes as low-density parity-check codes," in *Proc. 2nd Int. Symp. Turbo Codes & Related Topics*, Brest, France, Sep. 2000, pp. 543–546.

[25] B. Vasić, "Combinatorial constructions of structured low-density parity-check codes for iterative decoding," manuscript, submitted for publication.

[26] S. J. Johnson and S. R. Weller, "Regular low-density parity-check codes from combinatorial designs," in *Proc. IEEE Information Theory Workshop*, Cairns, Australia, Sep. 2001, pp. 90–92.

[27] G. A. Margulis, "Explicit constructions of graphs without short cycles and low density codes," *Combinatorica*, vol. 2, no. 1, pp. 71–78, 1982.

[28] ——, "Explicit group-theoretical constructions of combinatorial schemes and their application to the design of expanders and concentrators," *Probl. Inf. Transm.*, vol. 24, no. 1, pp. 39–46, Jan.-March 1988. Translated from *Probl. Pered. Inf.*, vol. 24, no. 1, pp. 51-60, Jan.-Mar. 1988.

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

[30] P. Erdös and H. Sachs, "Reguläre Graphen gegebener Taillenweite mit minimaler Knotenzahl," *Wiss. Z. Martin-Luther-Univ. Halle-Wittenberg Math.-Naturwiss. Reine*, vol. 12, pp. 251–257, 1963.

[31] J. Rosenthal and P. O. Vontobel, "Construction of LDPC codes using Ramanujan graphs and ideas from Margulis," in *Proc. 38th Annu. Allerton Conf. Communication, Computing and Control*, Monticello, IL, Oct. 2000, pp. 248–257.

[32] P. O. Vontobel and R. M. Tanner, "Construction of codes based on finite generalized quadrangles for iterative decoding," in *Proc. IEEE Intl. Symp. Information Theory*, Washington, DC, Jun. 2001, p. 223.

[33] 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. Inf. Theory*, vol. 50, no. 9, pp. 2010–2021, Sep. 2004.

[34] M. A. Shokrollahi, "New sequence of linear time erasure codes approaching the channel capacity," in *Proc. 13th Int. Symp. Applied Algebra, Algebraic Algorithm and Error-Correcting Codes*, 1999, pp. 65–76.

[35] G. Richter, "Construction of completely regular LDPC codes with large girth," in *Proc. ETH Zurich 2003 Winter School on Coding and Information Theory*, Ascona, Switzerland, Feb. 2003. Available [Online] at http://www.isi.ee.ethz.ch/winterschool/docs/richter.pdf.

[36] X.-Y. Hu, E. Eleftheriou, and D.-M. Arnold, "Progressive edge-growth Tanner graphs," in *Proc. IEEE Global Telecommunications Conf. (GLOBECOM)*, San Antonio, TX, Nov. 2001. *CD Proceedings*, paper no. 0-7803-7208-5/01.

[37] ——, "Irregular progressive edge-growth Tanner graphs," in *Proc. IEEE Intl. Symp. Information Theory*, Lausanne, Switzerland, Jun./Jul. 2002, p. 480.

[38] J. Campello, D. S. Modha, and S. Rajagopalan, "Designing LDPC codes using bit-filling," in *Proc. IEEE Int. Conf. Communications*, Helsinki, Finland, Jun. 2001, pp. 55–59.

[39] X.-Y. Hu, "Low-delay low-complexity error-correcting codes on sparse graphs," Ph.D. dissertation, Swiss Federal Institute of Technology Lausanne (EPFL), Lausanne, Switzerland, 2002.

[40] T. Tian, C. Jones, J. D. Vilasenor, and R. D. Wesel, "Construction of irregular LDPC codes with low error floors," in *Proc. IEEE Int. Conf. Communications* , vol. 5, Anchorage, AK, May 2003, pp. 3125–3129.

[41] H. Xiao and A. H. Banihashemi, "Improved progressive-edge-growth (PEG) construction of irregular LDPC codes," *IEEE Commun. Lett.*, to be published.

[42] M. R. Yazdani and A. H. Banihashemi, "On construction of rate-compatible low-density parity-check codes," *IEEE Commn. Lett.*, vol. 8, no. 3, pp. 159–161, Mar. 2004.

[43] R. M. Tanner, "Minimum distance bounds by graph analysis," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 808–821, Feb. 2001.

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

[45] D. Spielman, "Linear-time encodeable and decodable error-correcting codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pp. 1723–1731, Nov. 1996.

[46] D. J. C. MacKay, S. T. Wilson, and M. C. Davey, "Comparison of construction of irregular Gallager codes," *IEEE Trans. Commun.*, vol. 47, no. 10, pp. 1449–1454, Oct. 1999.

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

[48] H. Jin, A. Khandekar, and R. J. McEliece, "Irregular repeat-accumulate codes," in *Proc. 2nd Int. Symp. Turbo Codes & Related Topics*, Brest, France, Sep. 2000, pp. 1–8.

[49] T. Richardson and R. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 638–656, Feb. 2001.

[50] D. J. C. MacKay. Online Database of Low-Density Parity-Check Codes. [Online]. Available: http://wol.ra.phy.cam.uk/mackay/codes/data.html

[51] Y. Mao and A. Banihashemi, "A heuristic search for good low-density parity-check codes at short block lengths," in *Proc. Int. Conf. Communications*, vol. 1, Helsinki, Finland, Jun. 2001, pp. 41–44.

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

[53] M. C. Davey, "Error-correction using low-density parity-check codes," Ph.D. dissertation, Univ. Cambridge, Cambridge, U.K., 1999.

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

[55] C. Schlegel and L. Perez, "On error bounds and turbo-codes," *IEEE Commun. Lett.*, vol. 3, pp. 205–207, Jul. 1999.

[56] R. G. Gallager, *Information Theory and Reliable Communication*. New York: Wiley, 1968.