Low-Density Parity-Check (LDPC) Codes Constructed from Protographs

J. Thorpe[1]

We introduce a new class of low-density parity-check (LDPC) codes constructed from a template called a protograph. The protograph serves as a blueprint for constructing LDPC codes of arbitrary size whose performance can be predicted by analyzing the protograph. We apply standard density evolution techniques to predict the performance of large protograph codes. Finally, we use a randomized search algorithm to find good protographs.

I. Introduction

Low-density parity-check (LDPC) codes have emerged as one of the top contenders for near-channel-capacity error correction. Recently, more and more sophisticated classes of LDPC codes have been forwarded by members of the research community, each offering advances in one area or another.

An LDPC code is described by its (sparse) parity-check matrix. Such matrices can be efficiently represented by a bipartite (Tanner) graph. The standard iterative decoding algorithm, known as belief propagation (BP), passes messages along the edges of this graph. Much research has gone into understanding the properties required of a Tanner graph to produce an LDPC code that performs well under this decoding algorithm.

In this article, we introduce a new class of LDPC codes constructed from a template called a protograph. The protograph serves as a blueprint for constructing LDPC codes of arbitrary size whose performance can be predicted by analyzing the protograph. We apply standard density evolution techniques to predict the performance of large protograph codes. Finally, we use a randomized search algorithm to find good protographs.

II. Protographs and Protograph Codes

A protograph can be any Tanner graph, typically one with a relatively small number of nodes. A protograph $G = (V, C, E)$ consists of a set of variable nodes V, a set of check nodes C, and a set of edges E. Each edge $e \in E$ connects a variable node $v_e \in V$ to a check node $c_e \in C$. Parallel edges are permitted, so the mapping $e \rightarrow (v_e, c_e) \in V \times C$ is not necessarily 1:1.

[1] Communications Systems and Research Section.

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

As a simple example, we consider the protograph shown in Fig. 1. This graph consists of $|V| = 4$ variable nodes and $|C| = 3$ check nodes, connected by $|E| = 8$ edges. The four variable nodes in the protograph are denoted by "Type 1, 2, 3, 4," and the three check nodes by "Type A, B, C." By itself, this graph may be recognized as the Tanner graph of an $(n = 4, k = 1)$ LDPC code (in this case, a repetition code).

We can obtain a larger graph by a "copy-and-permute" operation, illustrated in Figs. 2 and 3. In Fig. 2, the protograph has been copied three times. Here the three copies are overlaid so that same-type vertices are in close proximity, but the overall graph consists of three disconnected subgraphs. In Fig. 3, the endpoints of the three copies of *each edge in the protograph* have been permuted among the three copies of the corresponding variable and check nodes. After this swapping of endpoints of edges, the three subgraphs are now interconnected. The graph in Fig. 3 is the Tanner graph of an $(n = 12, k = 3)$ LDPC code. We call this graph the *derived graph*, and the corresponding LDPC code a *protograph code*.

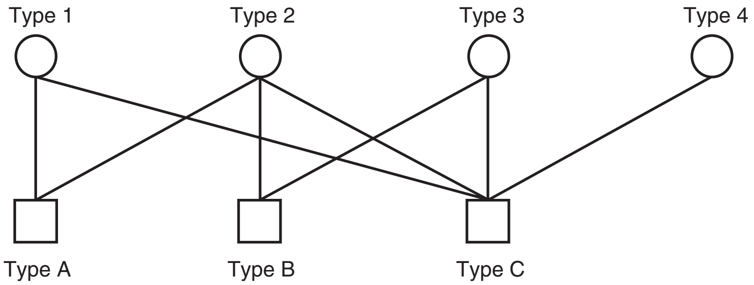

Fig. 1. A simple protograph.

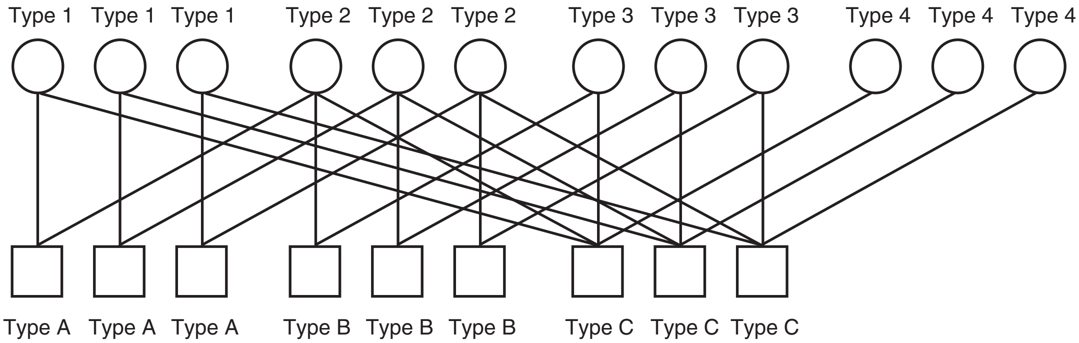

Fig. 2. A protograph copied three times.

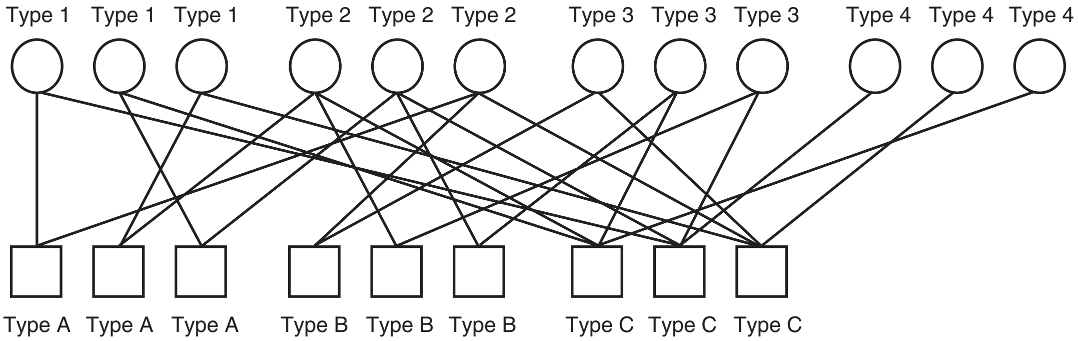

Fig. 3. A derived graph.

In general, we can apply the copy-and-permute operation to any protograph to obtain derived graphs of different sizes. This operation consists of first making T copies of the protograph, and then permuting the endpoints of each edge among the T variable and T check nodes connected to the set of T edges copied from the same edge in the protograph. Formally, we have the following definitions.

Definition 1. Let $G = (V, C, E)$ be a protograph. Let T be any positive integer and, for each $e \in E$, let π_e be an arbitrary permutation of $\{1, 2, \cdots T\}$. The derived graph is $G' = (V', C', E')$, where $V' = V \times \{1, 2, \cdots T\}$, $C' = C \times \{1, 2, \cdots T\}$, $E' = E \times \{1, 2, \cdots T\}$, and an edge $(e, t) \in E'$ connects the variable node $(v_e, t) \in V'$ to the check node $\big(c_e, \pi_e(t)\big) \in C'$.

Definition 2. A protograph code is an LDPC code whose Tanner graph is a derived graph.

The usual mapping of Tanner graphs to LDPC codes makes the implicit assumption that each variable node in the graph corresponds to a code symbol that will be transmitted over a channel. However, a useful refinement [1] is to allow the variable node set V to contain untransmitted variables. Under this refinement, each variable node $v \in V$ may be designated a *transmitted* node or an *untransmitted* node. The number of transmitted nodes is denoted n, and the number of untransmitted nodes is denoted u; thus $n + u = |V|$. The number of check nodes is denoted $r = |C|$. The dimension k of a code with untransmitted variables is $k = n + u - r$, and its rate is $R = (n + u - r)/n$.

A variable node (v, t) in G' has the same transmitted/untransmitted designation as v. The derived graph contains nT transmitted and uT untransmitted variable nodes, as well as rT check nodes. Thus, any derived graph has the same rate R as its protograph.

Untransmitted variables can improve the performance of protograph codes. These variables are decoded by the decoder in the same way as if the channel had yielded an “erasure.”

III. Deterministic Neighborhoods

A property of protograph codes not shared by other classes of irregular codes is that the local neighborhood of a node (v, t) in G' is completely determined by G. The local neighborhood to depth d consists of all nodes and edges connected to (v, t) by a path of length d or less. This neighborhood is a tree if there is at most one path of length d or less to any other node. In this tree, each node is adjacent to (different copies of) the same node types as in the protograph. To illustrate, we expand the neighborhood of a variable v of type 1 in Fig. 1 to depth $d = 3$ in Fig. 4.

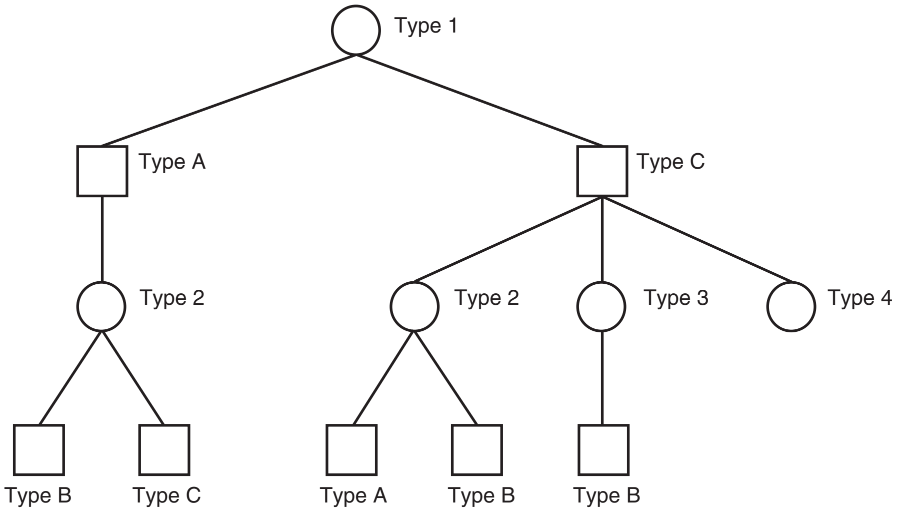

Fig. 4. An example of a neighborhood.

The local neighborhood tree can be constructed uniquely from the root downward. This leads to the following property of the local neighborhoods of derived graphs.

Property 1. If a neighborhood to depth d of a variable node (v, t) in the derived graph G' is tree-like, its structure is determined by the adjacencies in the protograph G. Note that v need not be tree-like within depth d in G.

We can expect that the performance of belief propagation on derived graphs should be relatively insensitive to the choice of $\{\pi_e\}$ because of the structure imposed by G. By contrast, in a Tanner graph generated from a given irregular degree profile, the structure of such a neighborhood is random, and the performance of belief propagation may depend more on the particular graph.

IV. Analysis of Protograph Code Ensembles

We have defined an arbitrary protograph code by applying a copy-and-permute operation to a protograph G. We now define an ensemble of protograph codes by specifying a probability distribution over the permutations $\{\pi_e, e \in E\}$. The ensemble that we shall consider is that obtained by generating π_e uniformly over all permutations of length T, and independently for each $e \in E$. For this ensemble, it can be shown that as $T \to \infty$, the probability that the neighborhood to fixed depth d of any node (v, t) is tree-like goes to 1, and thus the ensemble meets the criteria for analysis by density evolution.

Density evolution (DE) analysis [2] can be carried out on a protograph to determine whether or not decoding will yield arbitrarily small bit-error probability on a large derived graph. In this technique, the messages that are defined in the message-passing algorithm are treated as random variables, and their distributions are computed.

For the ensemble of codes constructed from a given protograph G, and for a given symmetric channel $\mathcal{C}$, we compute two distributions $\overrightarrow{q}_e^{\,i}$ and $\overleftarrow{q}_e^{\,i}$ on the messages passed in either direction along edge e at iteration i. Because of the deterministic neighborhood property, the message distributions $\overrightarrow{q}_e^{\,i}$ and $\overleftarrow{q}_e^{\,i}$ computed for an edge e in the protograph G are valid for *any* corresponding edge (e, t) in the derived graph G' as long as the neighborhood to depth $2i$ is tree-like.

Based on the message distributions $\overrightarrow{q}_e^{\,i}$, $\overleftarrow{q}_e^{\,i}$, and the channel $\mathcal{C}$, DE predicts a probability of decoder error $P_v^{(i)}$ for variable nodes of type v at iteration i. If message densities evolve such that $P_v^{(i)} \to 0$ as $i \to \infty$ for all v, then DE predicts that decoding will be successful, given large enough T. For a channel $\mathcal{C}_\theta$, whose fidelity increases with the parameter θ, the DE threshold θ^* for protograph G is the infimum of θ such that DE predicts successful decoding for G and $\mathcal{C}_\theta$.

V. Finding Good Protograph Codes

We devised an algorithm to search for protograph codes that perform well under BP on the additive white Gaussian noise (AWGN) channel. This search uses a randomized iterative algorithm called simulated annealing (SA) and is directed by our estimate of how the DE threshold changes as an underlying protograph is perturbed.

SA approximately minimizes over a search space S an energy function $\mathcal{E} : S \to \mathbb{R}$. It requires a random perturbation function $\rho : S \to S$ and a decreasing temperature profile $\mathcal{T}_j$. It begins with an arbitrary solution $s_0 \in S$, and at each iteration j applies the random perturbation ρ to generate a new solution $s'_j = \rho\left(s_j\right)$. If $\mathcal{E}\left(s'_j\right) < \mathcal{E}\left(s_j\right)$, then $s_{j+1} = s'_j$. If $\mathcal{E}\left(s'_j\right) \geq \mathcal{E}\left(s_j\right)$, then $s_{j+1} = s'_j$ with probability $\exp\left(-\left[\mathcal{E}(s'_j) - \mathcal{E}(s_j)\right]/\mathcal{T}_j\right)$ and s_j otherwise. The algorithm returns the solution s_j at the final iteration.

In our application, we search over protographs G with fixed n, u, and r. Since the code rate R is determined by these parameters, it is constant throughout the optimization space. Thus, we can design good codes for any desired rate by a judicious choice of n, u, and r.

We define three types of perturbations on a protograph G in the search space: removing an edge in G, adding an edge to G, and swapping the endpoints of two edges in G. The random perturbation ρ chooses one of these types at random, and chooses uniformly randomly among all possibilities for that type.

We would ideally like to use the DE threshold as our energy function $\mathcal{E}$. However, it is quite impractical to run full DE at each iteration. Instead, we use the reciprocal-channel approximation introduced in [3], a single-parameter approximation to DE for the AWGN channel. Further, instead of running approximate DE several times per SA iteration (j) to determine the threshold (under approximate DE) accurately, we run approximate DE just once at an operating point above the threshold for the current solution G_j. We let our energy function $\mathcal{E}$ be the number of decoding iterations (i) required to drive the output error probability below some small constant ε. Empirically, we have observed that this quantity varies nearly monotonically with the threshold.

VI. Protograph Optimization Results

Figure 5 shows an example of a simple rate 1/2 protograph found by the SA search method. This optimized protograph has $n = 8$ transmitted variables, $u = 1$ untransmitted variable, $r = 5$ checks, and a total of $|E| = 29$ edges, including 5 pairs of parallel edges. The untransmitted variable node has degree 9, the transmitted variable nodes have degrees $\{2, 2, 2, 2, 3, 3, 3, 3\}$, and the check nodes have degrees $\{4, 5, 6, 6, 8\}$. The (approximate) DE threshold on the AWGN channel for long codes constructed from this protograph is $E_b/N_0 = 0.283$ dB, which is about 0.1 dB better than the (exact) DE threshold of 0.4090 dB found by Richardson and Urbanke [4] for optimized irregular LDPC codes with maximum variable node degree 9. The irregular LDPC codes in [4] are connected according to a random ensemble constrained only by the distribution of node degrees, not by the additional structure built into the protograph.

We constructed a large ($n = 8192, k = 4096$) protograph code by interconnecting $T = 1024$ copies of the protograph in Fig. 5. The interconnections are determined by 29 separate permutations of $\{1, \cdots, 1024\}$, selected using a variation of the progressive edge growth (PEG) algorithm [5] to avoid short loops in the derived graph. The decoding performance of this code is shown in Fig. 6 and compared to that of

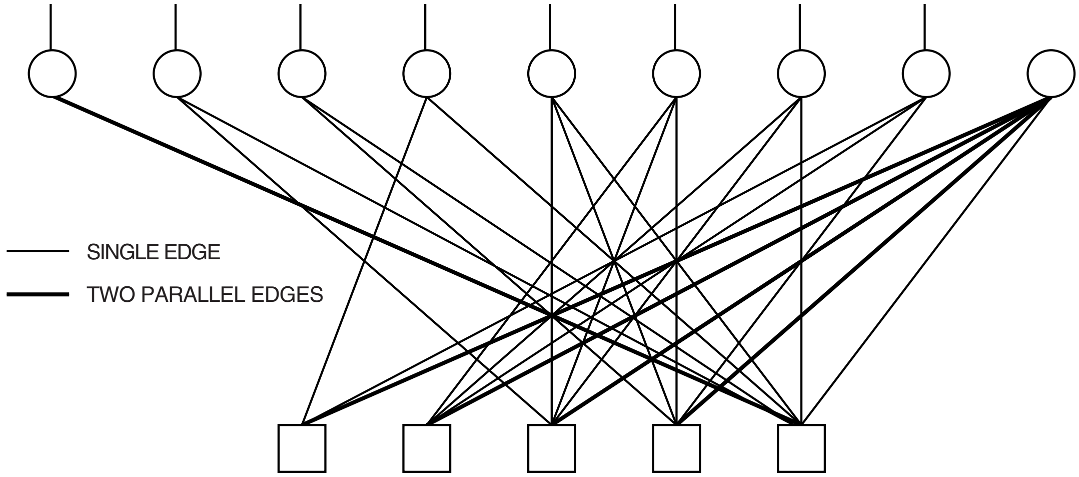

Fig. 5. An optimized protograph for a rate 1/2 protograph code.

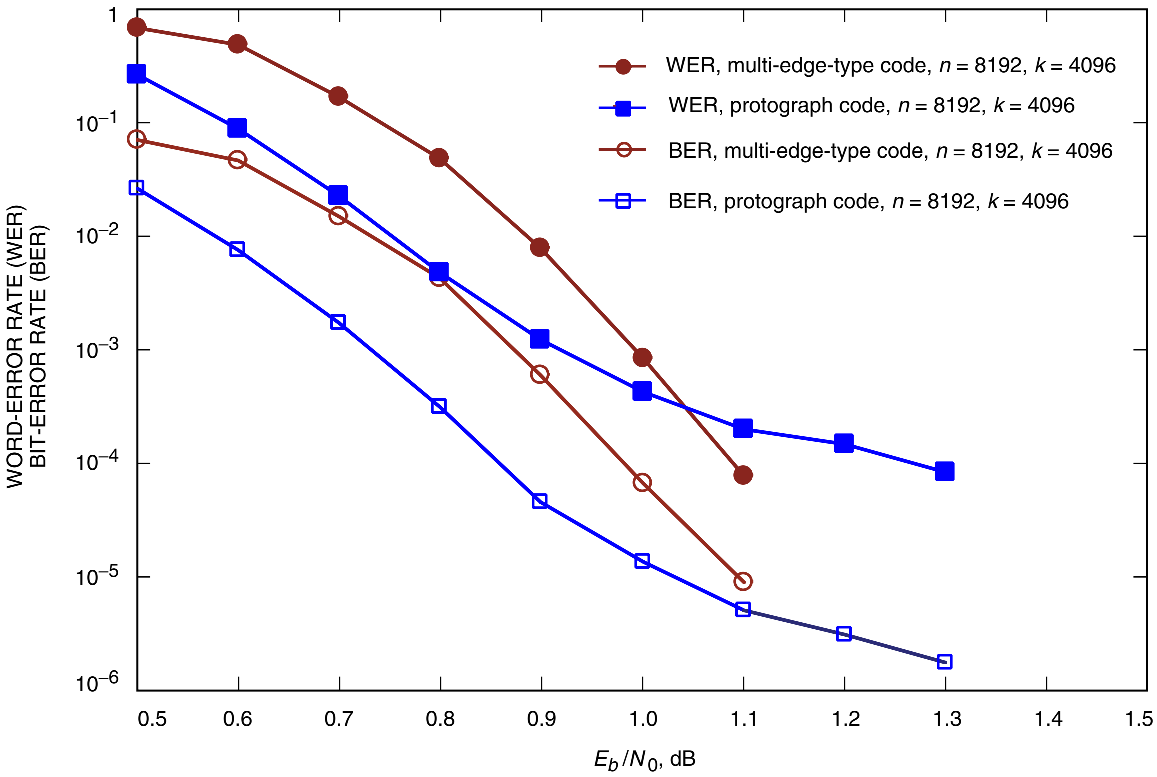

Fig. 6. Decoding performance of a large protograph code constructed from the protograph in Fig. 5.

a multi-edge-type LDPC code that we designed using the principles discussed in [1]. The performance of the protograph code is perhaps 0.1 dB better than that of the multi-edge-type code in the threshold region, but it suffers from the appearance of a true error floor due to low-weight codewords above a word-error rate of 10^{-4}. Our current criterion for selecting a good protograph minimizes the protograph code's (asymptotic) threshold but does not penalize code structures that might yield high error floors. Future work will investigate how to eliminate the error floor or to trade it off versus a small sacrifice in the optimized threshold.

References

[1] T. Richardson, "Multi-Edge Type LDPC Codes," presented at the Workshop honoring Prof. Bob McEliece on his 60th birthday (but not included in the proceedings), California Institute of Technology, Pasadena, California, May 24–25, 2002.

[2] T. J. Richardson and R. L. Urbanke, "The Capacity of Low-Density Parity-Check Codes under Message-Passing Decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599–618, February 2001.

[3] S.-Y. Chung, *On the Construction of Some Capacity-Approaching Coding Schemes,* Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, Massachusetts, September 2000.

[4] T. J. Richardson and R. L. Urbanke, "Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619–637, February 2001.

[5] X.-Y. Hu, E. Eleftheriou, and D.-M. Arnold, "Progressive Edge-Growth Tanner Graphs," *IEEE Global Telecommunications Conference 2001,* vol. 2, pp. 995–1001, November 25–29, 2001.