Probabilistic Decoding of Low-Density Cayley Codes

Adam L. Berger* **John D. Lafferty***
Computer Science Department
Carnegie Mellon University

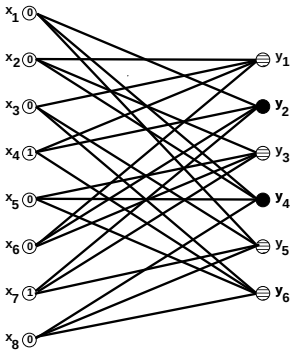
Abstract — We report on some investigations into the behavior of a class of low-density codes constructed using algebraic techniques. Recent work shows expansion to be an essential property of the graphs underlying the low-density parity-check codes first introduced by Gallager. In addition, it has recently been shown that certain spectral techniques similar to those based on Fourier analysis for classical cyclic codes can be applied to codes constructed from Cayley graphs. This motivates us to compare the behavior of algebraically constructed expanders and randomly generated bipartite graphs using a probabilistic decoding algorithm. Preliminary results indicate that the performance of the explicit, algebraic expanders is comparable to that of random graphs in the case where each variable is associated with only two parity checks, while such codes are inferior to randomly generated codes with three or more constraints for each variable.

1 Introduction

It has been known for over 30 years that random, sparse, regular bipartite graphs give rise to a powerful class of error-correcting codes. The idea is this: associate the nodes on one side of the graph with the variables of the code, and the nodes on the other side with linear constraints. An assignment of values to the variables is a codeword if and only if the neighbors of each constraint comprise a codeword in some subcode. It is conventional to take the subcode to be a simple parity check, and in fact some of the algorithms we will describe rely (although not in a truly fundamental way) on this fact. Figure 1 provides a simple illustration of this notion, showing both the graph and corresponding parity-check matrix for the associated code.

Codes constructed from sparse random bipartite graphs have two very attractive properties: they have good observed error-correcting ability, and they enjoy a simple decoding procedure whose complexity is linear in the code length. Two recent developments have motivated our investigations into this class of codes.

Expansion. In recent work, Sipser and Spielman [9] showed that graph expansion is essential for good asymptotic properties of these codes. The previous analysis using the girth of the graphs that Gallager [2] and Tanner [11] made was insufficient to show the existence of asymptotically good families of low-density codes.



$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Figure 1: The vector $(0,0,0,1,0,0,1,0)$ gives rise to four violated constraints (striped), and thus is not a codeword. The right-hand side shows the parity-check matrix corresponding to the graph.

Definition 1 *A graph $\Gamma = (V, E)$ with n vertices is said to be an ε-expander if for any vertex subset $S \subset V$ with $|S| \leq \frac{n}{2}$, $|\partial(S)| \geq \varepsilon |S|$, where $\partial(S) \equiv \{t : s \in S, t \notin S, \text{ and } (s,t) \in E\}$.*

Intuitively, a good expander graph yields a strong error-correcting code because a small group of erroneous variables will give rise to a large number of violated constraints. For instance, in Figure 1, the variables x_4 and x_7 give rise to four violated constraints. A decoding algorithm may, from the violated constraints, easily be able to discover which variables were received in error. Conversely, it may happen in a poor expander that an erroneous and a correct variable share the same set of neighboring constraints, in which case the constraint values do not reveal which of the two variables is in error.

The expansion of a graph is related to its spectrum in a surprising way. For a k-regular graph, the largest eigenvalue (the largest eigenvalue of its adjacency matrix) is $\lambda_0 = k$. Roughly speaking, if the gap $(k - \lambda_1)$ between the largest and second-largest eigenvalues is large, then the graph is a good expander. This spectral gap gives a handy means of testing for expansion. Numerically computing the spectrum of a graph on n vertices, however, has complexity $O(n^3)$, providing additional motivation

*Address: School of Computer Science, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA. Email: [aberger,lafferty]@cs.cmu.edu.

for devising expanders using algebraic techniques.

Spectral methods. Randomly generated low-density codes are most powerful for long block lengths. As a result, it may be impractical to use such codes since the only known encoding algorithm for them is the $O(n^2)$ method available for all linear block codes: systematize the parity check matrix and compute the check bits by matrix multiplication. (We note that Spielman [10] builds upon the results of [9] to obtain codes that are linear-time encodable as well, but we are here concerned only with codes that are within the original class of the low-density codes formulated by Gallager and Tanner.) Lafferty and Rockmore [5] have recently shown, however, that sub-quadratic encoding algorithms can be obtained when the underlying graphs have certain algebraic structure. In particular, when the graphs are Cayley graphs of finite groups, the methods of Fourier analysis on the group become available, and lead to spectral techniques that are analogous to those that use the fast Fourier transform for classical cyclic codes [1].

If G is a finite group and $\mathcal{A} = \{s_1, s_2, \ldots, s_d\} \subset G$ is a symmetric set of generators for G (meaning that if $s \in \mathcal{A}$ then $s^{-1} \in \mathcal{A}$), the *Cayley graph* $\Gamma(G, \mathcal{A})$ of G with respect to $\mathcal{A}$ is the d-regular graph having vertices indexed by G such that (g, h) is an edge if and only if $h = s^{-1}g$ for some $s \in \mathcal{A}$. This determines a canonical ordering on the neighbors of a vertex g: $x_i(g) = s_i^{-1}g$, for $i = 1, 2, \ldots, d$.

Fourier analysis on G is a very useful tool for working with codes constructed on Cayley graphs. A representation η of G over a field $\mathbf{F}$ is a map $\eta : G \longrightarrow GL_{d_\eta}(\mathbf{F})$ from G to the group of $d_\eta \times d_\eta$ matrices over $\mathbf{F}$ such that $\eta(xy) = \eta(x)\,\eta(y)$ for all $x, y \in G$. Given a representation η of G and a function $f : G \longrightarrow \mathbf{F}$, the *Fourier transform of f at η* is the $d_\eta \times d_\eta$ matrix $\hat{f}(\eta) = \sum_{x \in G} f(x)\,\eta(x)$. The adjacency matrix of a Cayley graph $\Gamma(G, \mathcal{A})$ is equal (up to a reordering of the groups elements) to the Fourier transform of the indicator function $\delta_{\mathcal{A}}$ evaluated at the right-regular representation. As a result, representation theory can be used to diagonalize the adjacency matrix, and efficiently compute its spectrum. It turns out that representation theory can also be used to obtain a fast encoding algorithm for the Cayley codes constructed on certain groups. We refer to [5] for the details of these ideas.

The remainder of this paper is organized as follows. In the following section we briefly discuss the Cayley graphs that we will consider. Section 3 discusses the probabilistic decoding algorithm that we have implemented. This decoder is essentially the same as the "belief propagation" decoder recently presented in [7]. In Section 4 we present the results of some preliminary experiments in which we apply the probabilistic decoder to low-density parity check codes built from the graphs discussed in Section 2.

2 Groups and Graphs

The Cayley graphs that we consider in this paper are the *Ramanujan graphs* of Lubotzky, Philips and Sarnak [6]. A k-regular graph $X_{n,k}$ is said to be Ramanujan in case $\lambda_1(X_{n,k}) \leq 2\sqrt{k-1}$. In terms of the second-largest eigenvalue, these graphs are asymptotically the best expanders possible, as a consequence of the inequality

$$\lim_{n \to \infty} \inf \lambda_1(X_{n,k}) \geq 2\sqrt{k-1}$$

where the infimum is taken over all k-regular graphs with more than n vertices.

The particular LPS graphs that we use here are constructed as Cayley graphs $\Gamma(G, \mathcal{A})$ of the group $G = PSL_2(\mathbf{F}_q)$. Recall that $SL_2(\mathbf{F}_q)$ is the *special linear group* of 2×2 matrices of determinant one having entries from the finite field $\mathbf{F}_q$ of q elements. The *projective special linear group* $PSL_2(\mathbf{F}_q)$ is obtained by dividing $SL_2(\mathbf{F}_q)$ by its center, $\{\pm I\}$ where I is the 2×2 identity matrix, and is a simple finite group of Lie type for $q \geq 5$. This group arises as the automorphism group of quadratic residue codes [8].

We suppose that $q \equiv 1 \ (mod\ 4)$. Let p be another prime which is a quadratic residue $(mod\ q)$. The LPS graphs $X^{p,q}$ are $(p+1)$-regular graphs on $PSL_2(\mathbf{F}_q)$ defined in the following manner. In the case where $p = 3$, they are given by the four generators

$$\frac{1}{\sqrt{3}} \begin{pmatrix} i & 1 \pm i \\ -1 \pm i & -i \end{pmatrix} \quad \frac{1}{\sqrt{3}} \begin{pmatrix} i & -1 \pm i \\ 1 \pm i & -i \end{pmatrix} \tag{1}$$

where $i = \sqrt{-1}\ (mod\ q)$. For $p > 3$, they are determined by the generating matrices

$$\frac{1}{\sqrt{p}} \begin{pmatrix} a_0 + i\,a_1 & a_2 + i\,a_3 \\ -a_2 + i\,a_3 & a_0 - i\,a_1 \end{pmatrix} \tag{2}$$

where (a_0, a_1, a_2, a_3) are the $p+1$ integral solutions to the equation $a_0^2 + a_1^2 + a_2^2 + a_3^2 = p$ having $a_0 > 0$ and odd and a_1, a_2, a_3 even.

Any Cayley graph on $PSL_2(\mathbf{F}_q)$ can be associated with a much smaller graph on the projective line $\mathbf{P}^1(\mathbf{F}_q) = \{0, 1, \ldots, q-1, \infty\}$ over $\mathbf{F}_q$. This is because $PSL_2(\mathbf{F}_q)$ acts on $\mathbf{P}^1(\mathbf{F}_q)$ by fractional linear transformations:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \omega = \frac{a\,\omega + b}{c\,\omega + d}\,. \tag{3}$$

In fact, the graph on the projective line is good expander if the graph on the full group is, because its spectral gap is at least as large. This can easily be shown using the representation theory of the underlying group [4].

This construction is useful for our purposes since the number of vertices in a Cayley graph grows as $O(q^3)$: $|PSL_2(\mathbf{F}_q)| = \frac{1}{2}q(q^2-1)$. It is important to observe that a graph on $\mathbf{P}^1(\mathbf{F}_q)$ will typically have edges that are self-loops. The number of these loops is small, however, and they do not pose a problem for our code constructions.

Not only are the LPS graphs good expanders, they also have large girth: $\text{girth}(X^{p,q}) \geq 2\log_p q = \Omega(\log n)$. This is relevant to the probabilistic decoder discussed in the following section; the decoder makes an independence assumption that breaks in the presence of cycles [2, 7].

2.1 Cayley codes

By a *Cayley code* we mean a code constructed in terms of a Cayley graph, where the symbols (variables) are placed on either the vertices or the edges of the graph. If the adjacency matrix of the graph is sparse (equivalently, the chosen set of generators of the group is small), then we say that the code is a *low-density Cayley code*. The bits labeling the neighbors of a given vertex are required to form a codeword in some *subcode*, the simplest example being a parity check. A Cayley code inherits symmetries from the group. All of the classical cyclic codes are Cayley codes, with the bits on the vertices, where the group G is $\mathbf{Z}/n\mathbf{Z}$, and the subcode is a parity check. In this case the fundamental theorem of algebra can be used to design codes having various rates and distances.

2.2 Random Graphs

It is a remarkable and convenient fact that almost all graphs are good expanders. In fact, one can establish upper and lower bounds on the expansion of random graphs which become tight as n gets large [9]. It is for this reason that random graphs provide a useful class on which to build recursive codes. Here we describe two methods for constructing random (c, d)-regular bipartite graphs.

Edge-collapsing construction. Begin by constructing a random matching between two sets of nd vertices. Collapse consecutive sets of d left nodes to form the n variables of Γ and consecutive sets of c right nodes to form the nc/d constraints of Γ. This generates a bipartite graph which may have multiedges. However, one can eliminate the multiedges by randomly swapping their endpoints. The 4-cycles can easily be removed by randomly swapping the endpoints of edges that participate in them.

Edge-vertex construction. This algorithm constructs a $(2, k)$-regular bipartite graph. Starting with an arbitrary k-regular graph $\Gamma^{(0)}_{n,k}$ on n nodes, this algorithm performs a random walk $\Gamma^{(0)}_{n,k}, \Gamma^{(1)}_{n,k}, \ldots$ on k-regular graphs as follows. In the i-th step of the walk, two edges $e, f \in E(\Gamma^{(i)}_{n,k})$ are chosen uniformly at random. The endpoints of e and f are swapped to construct $\Gamma^{(i+1)}_{n,k}$ if such a move would not create a multiedge, otherwise the graph remains unchanged. Thus, the graphs $\Gamma^{(i)}_{n,k}$ and $\Gamma^{(i+1)}_{n,k}$ differ by at most a 4-cycle. This procedure is known to be rapidly mixing [3]. As a practical consideration, however, in our experiments we simply performed 10^8 swaps, which required roughly an hour on a DEC Alpha workstation for each graph that was generated.

From a k-regular graph Γ one can construct a $(2, k)$-regular bipartite graph Γ' whose left vertices correspond to edges in Γ, and whose right vertices correspond to vertices in Γ. This is known as the "edge-vertex incidence construction."

3 Probabilistic Decoding

The probabilistic decoding scheme we have experimented with is due to Gallager [2]. It applies specifically to a recursive code where the subcode is a simple parity check. Although we have restricted attention to the binary symmetric channel, the algorithm extends to other channels.

As in Section 1, we denote the i'th variable by x_i, the j'th constraint by y_j, and the neighbors of x_i by $\partial(x_i)$. In addition, we introduce the following notation:

γ_i	value of the parity check associated with y_i. For example, in Figure 1, $\gamma_2 = \gamma_4 = 0$ and $\gamma_1 = \gamma_3 = \gamma_5 = \gamma_6 = 1$.
ρ_i	prior probability that $x_i = 0$
r^{ij}_k	probability that $x_i = k$, given the value of all constraints neighboring variable i except y_j.
s^{ij}_k	probability that the i'th parity check is satisfied (has value 0), given that variable j is k and the other variables have distributions given by r^{ij}.

Algorithm 1 *Probabilistic Decoding*

INITIALIZE*: set* $r^{ij}_0 \leftarrow \rho_i$ *and* $r^{ij}_1 \leftarrow 1 - \rho_i$.

UPDATE CONSTRAINT PROBS*: Update* s^{ij}_0 *and* s^{ij}_1 *via*

$$s^{ij}_k \leftarrow \sum_{\substack{x_m : m \in \partial(y_i)\backslash j \\ x_m \text{ and } x_j = k \text{ satisfy } y_i}} \prod_{m \in \partial(y_i)\backslash j} r^{im}_{x_m} \tag{4}$$

UPDATE VARIABLE PROBS*: Update* r^{ij}_0 *and* r^{ij}_1 *via*

$$\begin{aligned} r^{ij}_0 &\leftarrow \tfrac{1}{Z}\rho_i \prod_{m:m\in\partial(x_i)\backslash j} s^{mi}_0 \\ r^{ij}_1 &\leftarrow \tfrac{1}{Z}(1-\rho_i) \prod_{m:m\in\partial(x_i)\backslash j} s^{mi}_1 \end{aligned}$$

COMPUTE POSTERIOR PROBS*:*

$$\begin{aligned} \text{prob}(x_i = 0) &\leftarrow \tfrac{1}{Z}\rho_i \prod_{m:m\in\partial(x_i)} s^{mi}_0 \\ \text{prob}(x_i = 1) &\leftarrow \tfrac{1}{Z}(1-\rho_i) \prod_{m:m\in\partial(x_i)} s^{mi}_1 \end{aligned}$$

In all cases, Z is a normalization factor, chosen to ensure that probabilities sum to one. The algorithm terminates when the posterior probs converge (hopefully to a number close to 0 or 1) or when some maximum number of iterations is exceeded.

Among the 2^{d-1} possible assignments of the variables (excepting variable j) which neighbor constraint i, the sum in (4) is over just those assignments which satisfy constraint y_i when $x_j = 0$. Although s_0^{ij} would seem to require time exponential in d to compute, it can actually be computed in time linear in d using dynamic programming techniques such as the forward-backward algorithm [7].

4 Preliminary Experiments

The analysis of [2] shows that the ensemble of low-density parity-check codes constructed from random $(2, d)$-regular bipartite graphs is asymptotically bad in the sense that the minimum distance δ of such codes satisfies $\delta \leq O(\frac{\log n}{\log d})$, where n is the block length. On the other hand, Sipser and Spielman [9] show that for sufficiently good subcodes, the edge-vertex graphs of the LPS graphs on $PSL_2(\mathbf{F}_q)$ give rise to an asymptotically good family of codes. Since random graphs provide better expanders than the explicit constructions, it is natural to ask how much error-correcting ability we sacrifice by using the algebraic graphs. To address this question, we carried out some preliminary experiments comparing the random and algebraic constructions of $(2, d)$-regular graphs with parity-check subcodes. Table 1 presents some representative results from these experiments.

n	*rate*	$PSL_2(q)$	$\mathbf{P}^1(q)$	*edge-vertex*	*random bipartite*
17136	0.85	0.0003	—	0.0003	0.0002
36540	0.66	0.0031	—	0.0026	0.0022
50616	0.50	0.0094	—	0.0079	0.0086
21300	0.80	—	0.0002	0.0003	0.0002
39012	0.66	—	0.0016	0.0028	0.0023
40064	0.50	—	0.0101	0.0090	0.0087

Table 1: Maximum fraction of bits correctable for each of 100 error patterns, using probabilistic decoding. In each experiment, the edge-vertex graph of an algebraic expander was compared against random edge-vertex and bipartite graphs of equivalent size.

As Figure 2 demonstrates (and as Gallager's analysis predicts), such codes are inferior to those obtained with three or more constraints for each variable. As discussed in [9], associating variables with paths of length $k \geq 2$ generalizes the edge-vertex construction to yield (c, d)-regular expander graphs. However, this construction should not be expected to yield useful codes, since large subsets of the variables will share several constraints in common. In the future we hope to evaluate low-density codes using more powerful subcodes, and to investigate Cayley codes for which both the rate and minimum distance can be analyzed using spectral properties of the underlying graphs.

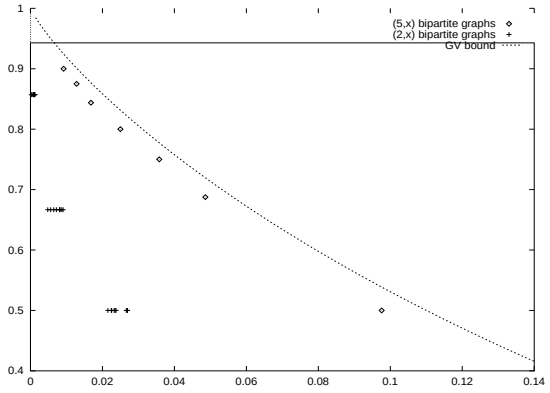


Figure 2: Comparison of $(5, d)$-regular random graphs with $(2, d)$-regular graphs for parity-check subcodes, in terms of fraction of bits correctable.

References

[1] R.E. Blahut. *Theory and Practice of Error-Control Codes*. Addison-Wesley, 1983.

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

[3] R. Kannan, P. Tetali, and S. Vempala. Simple Markov-chain algorithms for generating bipartite graphs and tournaments. In *ACM Symposium on Discrete Algorithms*, 1997.

[4] J. Lafferty and D. Rockmore. Fast Fourier analysis for SL_2 over a finite field and related numerical experiments. *Experimental Mathematics*, 1(2):115–139, 1992.

[5] J. Lafferty and D. Rockmore. Spectral techniques for expander codes. In *ACM Symposium on Theory of Computing (STOC'97)*, 1997. To appear.

[6] A. Lubotzky, R. Phillips, and P. Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

[7] D.J.C. MacKay. Good error-correcting codes based on very sparse matrices. In *IEEE International Symposium on Information Theory*, 1997.

[8] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland, Amsterdam, 1977.

[9] M. Sipser and D.A. Spielman. Expander codes. *IEEE Trans. on Information Theory*, 42(6):1710–1722, November, 1996.

[10] D.A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Trans. on Information Theory*, 42(6):1723–1731, November, 1996.

[11] R. Michael Tanner. A recursive approach to low complexity codes. *IEEE Trans. on Information Theory*, IT-27(5):533–547, September, 1981.