Comparison of Constructions of Irregular Gallager Codes

David J. C. MacKay, Simon T. Wilson and Matthew C. Davey

Department of Physics, University of Cambridge

Cavendish Laboratory, Madingley Road,
Cambridge, CB3 0HE, United Kingdom.

mackay|stw11|mcdavey@mrao.cam.ac.uk

To appear in *IEEE Transactions on Communications.* Submitted 30 July 1998.

Abstract

The low density parity check codes whose performance is closest to the Shannon limit are 'Gallager codes' based on irregular graphs. We compare alternative methods for constructing these graphs and present two results.

First, we find a 'super–Poisson' construction which gives a small improvement in empirical performance over a random construction.

Second, whereas Gallager codes normally take N^2 time to encode, we investigate constructions of regular and irregular Gallager codes which allow more rapid encoding and have smaller memory requirements in the encoder. We find that these 'fast–encoding' Gallager codes have equally good performance.

1 Introduction

Gallager codes [3, 4] are low density parity check codes constructed at random subject to constraints on the weight of each row and of each column. The original *regular* Gallager codes have very sparse random parity check matrices with uniform weight t per column and t_r per row. [We will also use the term 'regular' for codes which have nearly uniform weight columns and rows — for example, codes which have some weight 2 columns and some weight 3 columns.] These codes are asymptotically good, and can be practically decoded with Gallager's sum–product algorithm giving near Shannon limit performance when large block lengths are used [7, 8, 6]. Regular Gallager codes have also been found to be competitive codes for short block-length CDMA applications [10].

Recent advances in the performance of Gallager codes are summarised in figure 1. The rightmost curve shows the performance of a regular binary Gallager code with rate 1/4. The best known binary Gallager codes are *irregular* codes whose parity check matrices have *nonuniform* weight per column [5]; the performance of one such code is shown by the second curve from the right. The best known Gallager codes of all are Gallager codes defined over finite fields $GF(q)$ [2, 1]. The remaining two solid curves in figure 1 show the performance of a regular Gallager code over $GF(16)$ [2] and an irregular code over $GF(8)$ with bit error probability of 10^{-4} at $E_b/N_0 = -0.05$dB [1]. In comparing this code with the rate 1/4 Turbo code shown by the dotted line, the following points

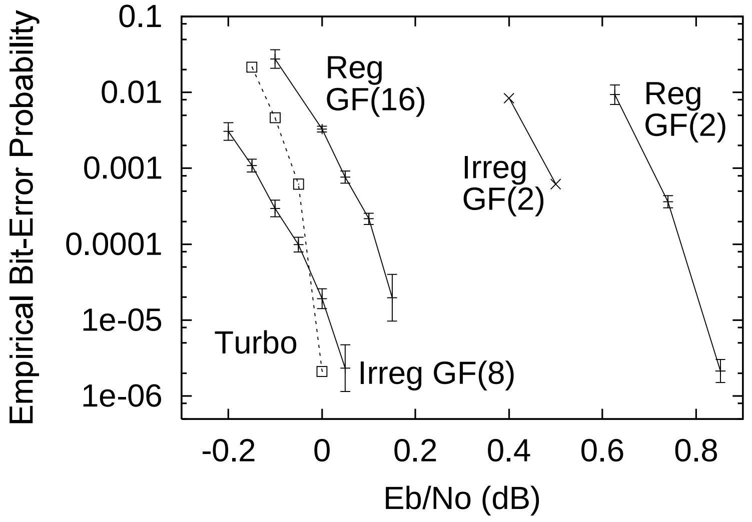

Figure 1: Empirical results for Gaussian Channel, Rate 1/4 Left–Right : Irregular LDPC, $GF(8)$ blocklength 24000 bits; JPL Turbo, blocklength 65536 bits; Regular LDPC, $GF(16)$, blocklength 24448 bits; Irregular LDPC , $GF(2)$, blocklength 64000 bits; Regular LDPC, $GF(2)$, blocklength 40000 bits. (Reproduced from [1].)

should be noted. (1) The transmitted blocklength of the irregular Gallager code is only 24000 bits, whereas that of the Turbo code is 65536 bits. (2) The errors made by the Gallager codes were all detected errors, whereas Turbo codes make undetected errors at high signal to noise ratio. This difference is not caused by a difference in the decoding algorithm: both codes are decoded by the sum–product algorithm [9]. Turbo Codes make undetected errors because *they have low–weight codewords*. For Gallager codes, the rate of occurrence of undetected errors is extremely small because they have good distance properties (the minimum distance scales linearly with the blocklength) [4]. In all our experiments with Gallager codes of block length greater than 1000 and column weight at least 3, undetected errors have never occurred.

The excellent performance of irregular Gallager codes is the motivation for this paper, in which we explore ways of further enhancing these codes.

The irregular codes of Luby, Mitzenmacher, Shokrollahi and Spielman [5] have parity check matrices with both nonuniform weight per row and nonuniform weight per column. It has not yet been established whether both of these non-uniformities are desirable. In our experience with codes for noisy channels, performance is more sensitive to the distribution of column weights. In this paper we concentrate on irregular codes with the weight per row as uniform as possible.

We can define an irregular Gallager code in two steps. First, we select a *profile* that describes the desired number of columns of each weight and the desired number of rows of each weight. The parity check matrix of a code can be viewed as defining a bipartite graph with 'bit' vertices corresponding to the columns and 'check' vertices corresponding to the rows. Each non–zero entry in the matrix corresponds to an edge connecting a bit to a check. The profile specifies the degrees of the vertices in this graph.

Second, we choose a *construction method*, that is, a pseudo–random algorithm for putting edges between the vertices in a way that satisfies the constraints. [In the case of non-binary Gallager codes we also need to choose an algorithm for assigning values to

This paper has two parts. In the first part (section 3) we compare alternative construction methods for a fixed profile in order to find out whether the construction method matters. In the second part (section 4) we examine regular and irregular constructions which lend themselves to rapid encoding. One motivation for this second study is that the only drawback of regular Gallager codes compared to Turbo codes for CDMA applications appears to be their greater encoding complexity [10].

In the experiments presented here, we study binary codes with rate 1/2 and blocklength about $N = 10,000$. We simulate an additive white Gaussian noise channel in the usual way [2] and examine the block error probability as a function of the signal to noise ratio. The error bars we show are one standard deviation error bars on the estimate of the logarithm of the block error probability $\hat{p}$, defined thus: when we observe r failures out of n trials, $p_{\pm} = \hat{p}\exp(\pm\sigma_{\log p})$ where $\sigma_{\log p} = \sqrt{(n-r)/(rn)}$.

2 Constructions

We compare the following methods.

Poisson: The edges are placed 'completely at random' subject to the profile constraints, and the rule that you can't put two edges between one pair of vertices, which would correspond to a double entry in the parity check matrix. One way to implement a Poisson construction is to make a list of all the columns in the matrix, with each column appearing in the list a number of times equal to its weight, then make a similar list of all the rows in the matrix, each row appearing with multiplicity equal to its weight, and then map one list onto the other by a random permutation, taking care not to create duplicate entries [5].

A variation of this construction is to require that no two columns in the parity check matrix have an overlap greater than one, *i.e.,* forbid cycles of length 4 in the graph. [Similar to construction 1A in [8].] A second variation requires that the graph to have no cycles of length less than some l. [Similar to construction 1B in [8].] This constraint can be quite hard to enforce if the profile includes high weight rows or columns.

Permutations. We can build parity check matrices by superposing random permutation matrices [4]. The convenience of this method depends on the profile. There are many ways of laying out these permutation matrices to satisfy a given profile. We will distinguish *'super Poisson'* and *'sub Poisson'* constructions.

- In a super–Poisson construction, the distribution of high weight columns per row has greater variance than a Poisson distribution;
- In a sub Poisson construction, the distribution of high weight columns per row has smaller variance than a Poisson distribution.

	Column weight	Fraction of columns	Row weight	Fraction
Profile 3	3	1	6	1
Profile 93	Column weight	Fraction of columns	Row weight	Fraction
	3	11/12	7	1
	9	1/12		

Table 1: The two profiles studied in this paper

3 Comparing Poisson, 'super–Poisson' and 'sub–Poisson' constructions

3.1 Profiles and constructions studied in this paper

3.1.1 Regular codes: 3 and 33

As our baseline we study regular Gallager codes with weight per column exactly $t = 3$ and weight per row exactly $t_r = 6$. We construct parity check matrices satisfying this profile from permutation matrices in two ways, labelled '3' and '33', shown diagrammatically in the upper panels of figure 2. In the figures, a square containing an integer (for example, '3') denotes the superposition inside that square of that number of random permutation matrices. The matrices are generated at random subject to the constraint that no two non–zero entries coincide.

3.1.2 Irregular codes: 93p, 93a, 93x and 93y

We chose the profile '93' shown in table 1. It has columns of weight 9 and of weight 3; all rows have weight 7. Note that this profile only differs from the regular profile '3' in that some extra 1s are added to $\frac{1}{12}$ of the columns. We emphasize that this profile has not been carefully optimized, so the results of this paper should not be taken as describing the best that can be done with irregular binary Gallager codes. We chose this profile because it lends itself to interesting experiments.

We will refer to the bits that connect to 9 checks as 'elite' bits. We use four different constructions that match this profile, named as follows. These constructions are depicted diagrammatically in the upper panels of figure 2.

Poisson: 93p. In this construction, while most checks will connect to one or two elite bits, a fraction of them will connect to more than two elite bits, and some will connect to none.

Sub–Poisson: 93a. This construction allocates exactly one or two elite bits to each check.

Super–Poisson: 93x and 93y are respectively moderately and very super–Poisson. In 93y, one third of the checks are connected to *four* elite bits, one third are connected to *one*, and one third are connected to *none*.

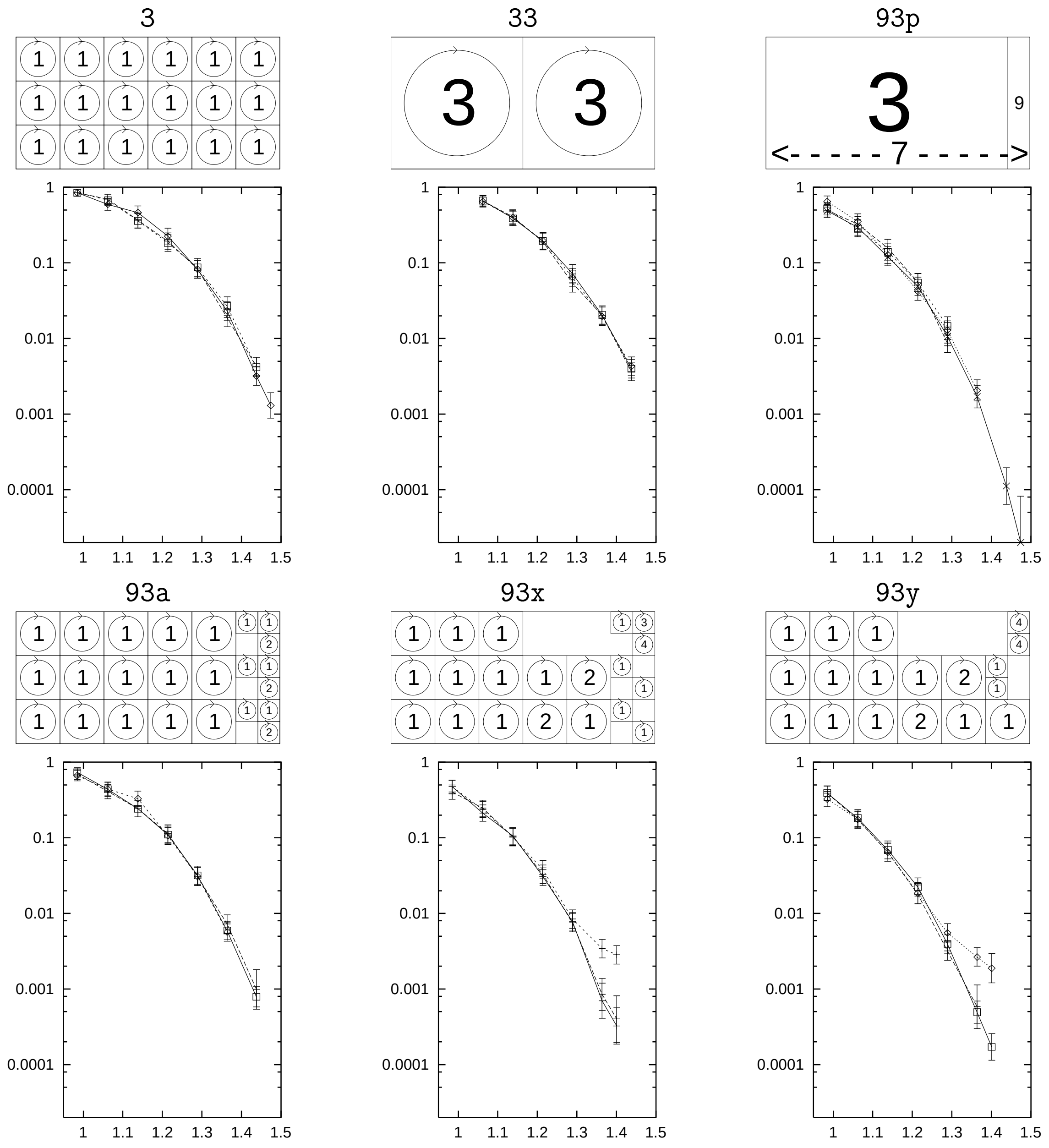

Figure 2: Upper panels: constructions of regular and irregular codes. Lower panels: performance of these codes. The construction types shown are regular, (`3`, `33`), Poisson (`93p`), sub–Poisson (`93a`), super–Poisson (`93x`), and super–Poisson (`93y`).
Notation for upper panels for all constructions except `93p`: an integer represents a number of permutation matrices superposed on the surrounding square. Horizontal and vertical lines indicate the boundaries of the permutation blocks. Notation for the Poisson construction `93p`: integers '3' and '9' represent column weights. The integer '7' represents the row weight.
Lower panels show the performance of several random codes of each construction. Vertical axis: block error probability. Horizontal axis: E_b/N_0 in dB. All codes have $N = 9972$, and $K = M = 4986$.
All errors were detected errors, as is usual with Gallager codes.

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | sales@docketalarm.com | 1-866-77-FASTCASE

fastcase® Smarter legal research.