6.897 Algorithmic Introduction to Coding Theory November 13, 2002

Lecture 18

Lecturer: Madhu Sudan *Scribe: Johnny Chen*

In this lecture we discuss codes that:

- Transmit at rates close to Shannon capacity
- Decode from a fraction p of *random* errors (in the next lecture we will consider adversarial errors)
- Allow for linear time encoding/decoding

While the expander-based Spielman codes achieve these goals, there is more to be done. Specifically, practical issues such as speed need to be addressed.

The decoding algorithms we will see have simple descriptive complexity (they can be described in just a few lines) and are iterative, so without specific stopping criterion they may never terminate.

1 Turbo Codes

Turbo codes, introduced by Berrou, Glavieux and Thitimajshima in 1993 [1], are widely used codes that exhibit excellent empirical results, i.e., the decoding algorithm produces a codeword very quickly. While turbo codes are very popular, theoretical analysis of their performance has been unsupportive. Codewords have extremely small minimum distance, and no one has been able to prove that the decoding algorithm is *usually* fast or has small error probability. In contrast, Spielman codes have linear time encoding/decoding and provably good error probability. So why are turbo codes favored?

1.1 Practical Issues: Concrete Complexity

For probability of error p, Spielman codes with rate $R = 1 - H(p) - \epsilon$ and decoding error probability δ require blocklength n such that $\delta = e^{-\epsilon n}$. Even modest values, e.g. $p = 0.1$ and $\delta = 10^{-6}$, require that $n > 10^{2^6}$ be prohibitively large. In general, given p and desired δ, R we need to consider the smallest n satisfying the parameters.

Another issue is the dependence of the decoding algorithm's run-time on $1/\epsilon$. Recall Spielman codes can be decoded in time $2^{\epsilon^{-2}}n$; this exponential dependence is clearly undesirable. The best we can hope for is time linear in n and $\ln(1/\epsilon)$. The following codes achieve this bound.

2 Irregular Low-Density Parity-Check Codes

This presentation is drawn from two papers by Luby, Mitzenmacher, Shokrollahi, and Spielman [5] [6].

2.1 Description of Codes

The basic structure of LMSS codes uses low density generator graphs, like Spielman codes.

Definition 1 *The code $\mathcal{C}(B)$ corresponds to bipartite graph B with k left (message) nodes and τk right (check) nodes ($0 < \tau < 1$), where each check node is the $\oplus$ (XOR) of its neighbors.*

We will postpone the description of B, noting only that it is sparse random graph. (For linear time encoding/decoding with reasonable concrete complexity, sparse means the number of edges is linear in n and $\ln(1/\epsilon)$.) Remember that the Spielman construction works hard to protect check bits. The LMSS construction takes a different approach: it cascades codes of the form $\mathcal{C}(B)$ to correct errors in both message bits and check bits.

Definition 2 *The LMSS code is a family of codes $\mathcal{C}(B_0), \ldots, \mathcal{C}(B_t)$ constructed from graphs $B_0, \ldots, B_t$, where B_i has $\tau^i k$ left nodes and $\tau^{i+1} k$ right nodes. Here t is chosen so that $\tau^{t+1} \approx \sqrt{k}$;*

A conventional code, like Reed-Solomon or Spielman, is used to correct errors in $\mathcal{C}(B_t)$. By the choice of t, a slower (quadratic time) conventional code will not affect overall linear time encoding/decoding.

2.2 Encoding and Rate

Modulo the choices of the random bipartite graphs B_i, encoding LMSS codes is straightforward. Starting from k message bits, the check bits from $\mathcal{C}(B_0)$ are used as message bits for $\mathcal{C}(B_1)$. The check bits from $\mathcal{C}(B_1)$ are used as message bits for $\mathcal{C}(B_2)$, and so on. The conventional code encodes the check bits of $\mathcal{C}(B_t)$. Therefore the rate is roughly

$$k + \tau k + \tau^2 k + \ldots + \epsilon k \approx (1 - \tau)k.$$

2.3 Decoding

Encoding proceeds "left to right" - from $\mathcal{C}(B_0)$ to $\mathcal{C}(B_t)$. Decoding works in the opposite direction. It suffices to look at the decoding algorithm for $\mathcal{C}(B_0)$. For each edge (m, c), where m is a message node and c is a check node, the algorithm proceeds in *rounds*, where in each round one bit is passed from m to c and then one bit is passed from c to m. Let r_m and r_c be the initial received bit for message nodes m and check nodes c, respectively.

Definition 3 *For each edge (m, c), denote the bit passed from m to c at round i by $g^i_{m,c}$. Similarly, denote the bit passed from c to m at round i by $g^i_{c,m}$.*

The following hard-decision algorithm was originally proposed by Gallager [3].

Hard-Decision Decoding Algorithm
For all edges (m, c) do in parallel:

1. *Send* $g^i_{m,c}$. If $i = 0$, $g^0_{m,c} = r_m$. Else,
   - If $g^{i-1}_{c',m} = b$ for all $c' \neq c$, $g^i_{m,c} = b$.
   - Else $g^i_{m,c} = r_m$.
2. *Send* $g^i_{c,m}$, where $g^i_{c,m}$ is the XOR of r_c with message nodes adjacent to c other than m.

The messages $g^i_{m,c}, g^i_{c,m}$ are guesses of the correct value of m. Note that $g^i_{m,c}$ makes no use of $g^i_{c,m}$ and vice-versa. The guess $g^i_{m,c}$ is set to r_m unless all adjacent check node other than c agree on the value of m. The guess $g^i_{c,m}$ is made assuming that all message nodes adjacent to c other than m are correct.

Theorem 4 *Fix a (random) $\mathcal{C}(B)$. For an edge (m, c) in B, let p_i be the probability that $g^i_{m,c}$ is incorrect. If the sequence p_i converges to 0, then for any $\eta > 0$ there is a sufficiently large blocklength n such that the decoding algorithm decodes all but at most ηn bits in some constant number r_η of rounds with high probability.*

The analysis of the decoding algorithm involves martingales and is beyond the scope of the course. Note that the theorem does not show that decoding *completes* successfully. Once the number of errors is sufficiently small, another algorithm is used to finish decoding with high probability. It turns out that the random bipartite graphs B are sufficiently good expanders (although we still have not described how they are constructed!) so that the FLIP algorithm from the Sipser-Spielman construction works.

Before revealing the structure of the random bipartite graphs, we will consider the motivation for these codes, hopefully providing some intuition.

2.4 Motivation: Tornado Codes [5]

We begin with a new channel model.

Definition 5 (Elias [2]) *In the binary erasure channel (BEC), each bit is transmitted correctly independently with fixed probability p. If a bit is lost, the symbol "?" (not in the input alphabet) is received.*

Intuitively, the BEC should have higher capacity than the binary symmetric channel, since the corrupted bits are known. Indeed, Elias [2] showed that the capacity of the erasure channel is $1-p$. Tornado codes, introduced by Luby, Mitzenmacher, Shokrollahi and Spielman [5], are codes from the erasure channel resembling the construction from the previous section. The encoding and rate will be the same, but the decoding algorithm is simpler. This is because only the "?" need to be changed. Again, we only need to consider one level $\mathcal{C}(B)$ of the cascading graphs. At the last level is a conventional code, and we assume that all erasures have been corrected there.

Erasure Decoding Algorithm

Choose a random check node c with one unknown neighbor m and set m to the XOR of c with all $m' \neq m$ adjacent to c. Repeat until all message nodes are determined.

We can view the decoding process of $\mathcal{C}(B)$ in terms of a subgraph of B consisting of unknown message nodes on the left and their corresponding check nodes on the right. The algorithm selects a degree one check node in the subgraph and sets its unknown neighbor to its value. The check node, its neighbor, and all edges adjacent to its neighbor are then removed from the subgraph. The process continues until all message nodes are removed.

Again, the analysis of the decoding algorithm is beyond the scope of the course.

2.5 Irregular Generator Graphs

We are now ready to reveal the structure of the random bipartite graphs. Until now we have been dealing with regular bipartite graphs. It turns out that the irregular LDPC and Tornado codes use random bipartite graphs with carefully chosen *irregular* degree sequences. The intuition for this is clearer for Tornado codes. Note that message nodes with higher degree are more likely to be decoded. At the same time, check nodes with low degree are more likely to determine all their neighbors and lead to faster decoding. Since the sum of the degrees of left nodes and right nodes must be equal, there is a tension between left degrees and right degrees. Keep in mind that if we view decoding in terms of the subgraph of B having vertices and edges removed, then degree sequences are changing at each step.

Definition 6 *For a bipartite graph B, let λ_i be the probability that a left node has degree i and ρ_i be the probability that a right node has degree i. Then the vectors $(\lambda_1, \ldots, \lambda_m)$ and $(\rho_1, \ldots, \rho_m)$ are degree distributions for left and right nodes.*

Finding optimal distributions is a non-linear optimization problem and still open. Through analysis and simulation, degree distributions with $\lambda_i \propto \frac{1}{i^2}$ and $\rho_i \propto \frac{\alpha^i}{i!}$ (where α depends on p and the message length k) were found.

Irregular graphs, the intuition from Tornado codes, turned out to work for the binary symmetric channel as well. For insight about the relation of degree sequences to decoding (beyond the original papers [5], [6]) we refer the reader to the simple analysis given in [4]. For a deeper treatment of message passing in regular and irregular LDPC codes we refer the reader to [8] [7].

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima. Near shannon limit error-correcting coding and decoding. *Proc. Int. Conf. Communication*, pages 1064–1070, May 1993.

[2] P. Elias. Coding for two noisy channels. *Information Theory, 3rd London Symp.*, pages 61–76, 1955.

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

[4] M. Luby, M. Mitzenmacher, and M. A. Shokrollahi. Analysis of random processes via and-or tree evaluation. *SODA*, pages 364–373, 1998.

[5] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. Spielman. Efficient erasure correcting codes. *IEEE Transactions on Information Theory*, 47(2):569–584, February 2001.

[6] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. Spielman. Improved low-density parity-check codes using irregular graphs. *IEEE Transactions on Information Theory*, 47(2):585–598, February 2001.

[7] T. Richardson, M. A. Shokrollahi, and R. Urbanke. The capacity of low-density parity check codes under message-passing decoding. *IEEE Transactions on Information Theory*, 47(1):599–618, February 2000.

[8] T. Richardson, M. A. Shokrollahi, and R. Urbanke. Design of capacity-approaching irregular low-density parity-check codes. *IEEE Transactions on Information Theory*, 47(1):619–637, February 2000.