CS225: Pseudorandomness Prof. Salil Vadhan

Lecture 14: Error-Correcting Codes

April 3, 2007

Based on scribe notes by Sasha Schwartz and Adi Akavia.

1 Basic Definitions

The field of *coding theory* is motivated by the problem of communicating reliably over noisy channels — where the data sent over the channel may come out corrupted on the other end, but we nevertheless want the receiver to be able to correct the errors and recover the original message. There is a vast literature studying aspects of this problem from the perspectives of electrical engineering (communications and information theory), computer science (algorithms and complexity), and mathematics (combinatorics and algebra). In this course, we are interested in codes as 'pseudorandom objects,' ones that are intimately related with the other objects we are studying. In particular, we will see how to use ideas from coding theory to construct the condensers and unbalanced expanders that we assumed in the previous lectures (for our construction of extractors).

The approach to communicating over a noisy channel is to restrict the data we send to be from a certain set of strings that can be easily disambiguated (even after being corrupted).

Definition 1 *A* q-ary code *is a set* $\mathcal{C} \subseteq \Sigma^{\hat{n}}$*, where* Σ *is an* alphabet *of size* q*. Elements of* $\mathcal{C}$ *are called* codewords*. Some key parameters:*

- $\hat{n}$ *is the* block length*.*
- $n = \log_2 |\mathcal{C}|$ *is the* message length*.*
- $\rho = n/(\hat{n} \cdot \log |\Sigma|)$ *is the* (relative) rate *of the code.*

An encoding function *for* $\mathcal{C}$ *is an injective mapping* Enc: $\{0,1\}^n \to \mathcal{C}$ *(for* n *a positive integer). Given such an encoding function, we view the strings in* $\{0,1\}^n$ *as* messages*. The code is* explicit *if* Enc *is computable in polynomial time.*

Note that every code $\mathcal{C}$ whose message length is an integer has an encoding function Enc. We view $\mathcal{C}$ and Enc as being essentially the same object (with Enc merely providing a 'labelling' of codewords), with the former being useful for studying the combinatorics of codes and the latter for algorithmic purposes. Our notation differs from the standard notation in coding theory in several ways. Typically in coding theory, the input alphabet is taken to be the same as the output alphabet (rather than $\{0,1\}$ and Σ, respectively), the blocklength is denoted n, and the message length (over Σ) is denoted k and is referred to as the rate.

So far, we haven't talked at all about the error-correcting properties of codes. Here we need to specify two things: the model of errors (as introduced by the noisy channel) and the notion of a successful recovery.

For the errors, the main distinction is between *random errors* and *worst-case errors*. For random errors, one needs to specify a stochastic model of the channel. The most basic one is the *binary symmetric channel* (over alphabet $\Sigma = \{0,1\}$), where each bit is flipped independently with probability δ. People also study more complex channel models, but as usual with stochastic models, there is always the question of how well the theoretical model correctly captures the real-life distribution of errors. We, however, will focus on *worst-case errors*, where we simply assume that at most some δ fraction of symbols have been changed. That is, when we send a codeword $c \in \Sigma^{\hat{n}}$ over the channel, the received word $r \in \Sigma^{\hat{n}}$ differs from c in at most $\delta\hat{n}$ places.

Definition 2 *For two strings $x, y \in \Sigma^{\hat{n}}$, their* (relative) Hamming distance $d_H(x,y)$ *equals* $\Pr_i[x_i \neq y_i]$. *For a string $x \in \Sigma^{\hat{n}}$ and $\delta \in [0,1]$, the* Hamming ball *of radius δ around x is the set $B(x,\delta)$ of strings $y \in \Sigma^{\hat{n}}$ such that $d_H(x,y) \leq \delta$. Define $H_q(\delta,\hat{n})$ to be such that $|B(x,\delta)| = q^{H_q(\delta,\hat{n})\cdot\hat{n}}$.*

For the notion of a successful recovery, the traditional model requires that we can uniquely decode the message from the received word (in the case of random errors, this need only hold with high probability). Our main focus will be on a more relaxed notion which allows us to produce a small list of candidate messages. As we will see, the advantage of such list-decoding is that it allows us to correct a larger fraction of errors.

Definition 3 *Let* Enc$\colon \{0,1\}^n \to \Sigma^{\hat{n}}$ *be an encoding algorithm for a code $\mathcal{C}$. A* δ-decoding *algorithm for* Enc *is a function* $\mathrm{Dec} : \Sigma^{\hat{n}} \to \{0,1\}^n$ *such that for every $m \in \{0,1\}^n$ and $r \in \Sigma^{\hat{n}}$ such that $d_H(\mathrm{Enc}(m), r) \leq \delta$, we have $\mathrm{Dec}(r) = m$. If such a function* Dec *exists, we call the code* δ-decodable.

A (δ, L)-list-decoding *algorithm for* Enc *is a function* $\mathrm{Dec} : \Sigma^{\hat{n}} \to (\{0,1\}^n)^L$ *such that for every $m \in \{0,1\}^n$ $r \in \Sigma^{\hat{n}}$, such that $d_H(\mathrm{Enc}(m), r) \leq \delta$, we have $m \in \mathrm{Dec}(r)$. If such a function Dec exists, we call the code* (δ, L)-list-decodable.

Note that a δ-decoding algorithm is the same as a $(\delta, 1)$-list-decoding algorithm. It is not hard to see that, if we do not care about computational efficiency, the existence of such decoding algorithms depends only on the combinatorics of the code.

Definition 4 *The* (relative) minimum distance *of a code $\mathcal{C} \subseteq \Sigma^{\hat{n}}$ equals* $\min_{x \neq y \in \mathcal{C}} d_H(x,y)$.

Proposition 5 *Let $\mathcal{C} \subseteq \Sigma^{\hat{n}}$ be a code with any encoding function.*

1. *$\mathcal{C}$ is δ-decodable iff its minimum distance is greater than 2δ.*
2. *$\mathcal{C}$ is (δ, L)-list-decodable iff for every $r \in \Sigma^{\hat{n}}$, we have $|B(r,\delta) \cap \mathcal{C}| \leq L$.*

The factor of 2 in Item 1 is the reason that list-decoding can provide an advantage over unique decoding in the fraction of errors corrected.

The main goals in constructing codes are to have infinite families of codes (e.g. for every message length n) in which we:

- Maximize the fraction δ of errors correctible (e.g. constant independent of n and $\hat{n}$).

- Maximize the rate ρ (e.g. a constant independent of n and $\hat{n}$).
- Minimize the alphabet size q (e.g. a constant, ideally $q = 2$).
- Keep the list size L relatively small (e.g. a constant or poly(n)).
- Have computationally efficient encoding and decoding algorithms.

In particular, coding theorists are very interested in obtaining the optimal tradeoff between the constants δ and ρ with efficiently encodable and decodable codes.

2 Existential Bounds

The existence of very good codes can be shown using the probabilistic method.

Theorem 6

1. *For all integers $\hat{n}, q$, and all $\delta \in (0, 1 - 1/q)$, there exists a q-ary code of block length $\hat{n}$, minimum distance at least δ, and rate $\rho \geq 1 - H_q(\delta, \hat{n})$.*
2. *For all all integers $\hat{n}, q, L \geq 2$, and all $\delta \in (0, 1-1/q)$, there exists a (δ, L)-list-decodable q-ary code of block length $\hat{n}$ and rate $\rho \geq 1 - H_q(\delta, \hat{n}) - 1/L$.*

Proof Sketch: We do not prove the first item; we prove the second item using the probabilistic method. Choose the $q^{\rho\hat{n}}$ elements of the code randomly and independently from $\Sigma^{\hat{n}}$. The probability that there is a Hamming Ball of radius δ containing $L + 1$ codewords is at most

$$q^{\hat{n}} \cdot \binom{q^{\rho\hat{n}}}{L+1} \cdot \left(\frac{q^{H_q(\delta,\hat{n})}}{q^{\hat{n}}} \right)^{L+1},$$

which is less than 1 by our setting of parameters. □

Note that while the rate bounds are essentially the same for achieving minimum distance and the list-decoding radius δ (as we take large list size), recall that minimum distance δ only corresponds to unique decoding up to radius roughly $\delta/2$.

Let's look at some special cases of the parameters in the above theorem. Over binary alphabets ($q = 2$), it turns out that $H_2(\delta, \hat{n})$ is at most the Shannon entropy $\mathrm{H}_{Sh}(\delta)$, so the rate is roughly $1 - \mathrm{H}_{Sh}(\delta)$ (as we take large a list size). (More generally, $H_q(\delta, \hat{n})$ is bounded by a q-ary analogue of Shannon entropy.) This is known to be tight for list-decoding, but it is unknown whether it is tight for unique decoding. We will be most interested in the case $\delta = 1/2 - \varepsilon$, in which case the rate is $1 - \mathrm{H}_{Sh}(1/2 - \varepsilon) = \Theta(\varepsilon^2)$, i.e. $\hat{n} = \Theta(n/\varepsilon^2)$ (for list size $L = \Theta(1/\varepsilon^2)$).

For large alphabets q, we have $H_q(\delta, \hat{n}) \leq \delta + 1/\log q$, in which case the rate is approaches $1 - \delta$. We will be most interested in the case $\delta \approx 1 - \varepsilon$, where the above bounds imply codes where we can correct a $1 - \varepsilon$ fraction of errors with a list size of $O(1/\varepsilon)$ and an alphabet of size $2^{O(1/\varepsilon)}$.

While we are primarily interested in list-decodable codes, minimum distance is often easier to bound. The following allows us to translate bounds on minimum distance into bounds on list-decodability.

Proposition 7 (Johnson Bound)

1. *If $\mathcal{C}$ has minimum distance $1-\varepsilon$, then it is $(1-O(\sqrt{\varepsilon}), O(1/\sqrt{\varepsilon}))$-list-decodable.*
2. *If a* binary *code $\mathcal{C}$ has minimum distance $1/2-\varepsilon$, then it is $(1/2-O(\sqrt{\varepsilon}), O(1/\varepsilon))$- list-decodable.*

Proof: We prove Item 1. The proof is by inclusion-and-exclusion. Let $r \in \Sigma^{\hat{n}}$, say $C_1, ..., C_s$ are codewords at distance at most $1-\varepsilon'$ from r, for $\varepsilon' = \sqrt{2\varepsilon}$ and $s = 2/\varepsilon'$.

$$\begin{aligned} 1 &\geq \text{fraction of positions where } r \text{ agrees with some } C_i \\ &\geq \sum_i agreement(r, C_i) - \sum_{1 \leq i < j \leq s} agreement(C_i, C_j) \\ &\geq s\varepsilon' - \binom{s}{2} \cdot \varepsilon \\ &> 2 - 1 = 1 \end{aligned}$$

where the last inequality is by our setting of parameters. We reached a contradiction, implying $s < \frac{2}{\varepsilon'}$. ■

Note that the quadratic loss in the distance parameter. This means that optimal codes with respect to minimum distance are not necessarily optimal with respect to list-decoding. Nevertheless, if we do not care about the exact tradeoff between the rate and the decoding radius, in both cases, the above can yield codes where the decoding radius is as large as possible (approaching 1 and 1/2, respectively).

3 Explicit Codes

As usual, most applications of error-correcting codes (in particular the original motivating one) require computationally efficient encoding and decoding. For now, we focus on only the efficiency of encoding.

3.1 Hadamard Code

- $\Sigma = \{0, 1\}$.
- Codewords: truth tables of all linear functions $L : \mathbb{Z}_2^n \to \mathbb{Z}_2$.
- $\hat{n} = 2^n$
- For $m \in \{0,1\}^n$, Enc(m) is the linear function $L_m(x) = \sum_i m_i x_i \bmod 2$.
- Minimum distance $\delta = 1/2$
- List-decodable up to radius $1/2 - \varepsilon$ with lists of size $O(1/\varepsilon^2)$.

3.2 Reed-Solomon Codes

- $\Sigma = \mathbb{F}_q$ for the finite field $\mathbb{F}_q$ of size q.
- Codewords: truth tables of all polynomials $p : \mathbb{F}_q \to \mathbb{F}_q$ of degree at most d.
- $n = (d+1) \cdot \log q$, $\hat{n} = q$.
- Minimum distance $\delta = 1 - d/q$
- List-decodable up to radius $1 - O(\sqrt{d/q})$ with lists of size $O(\sqrt{q/d})$.
- Common settings: $|\mathbb{F}| = O(d)$ (constant rate and distance), $|\mathbb{F}| = \text{poly}(d)$ ($\hat{n} = \text{poly}(n)$, $\delta = 1 - 1/n^{\Omega(1)}$).

3.3 Reed-Muller Code

- $\Sigma = \mathbb{F}_q$ for the finite field $\mathbb{F}_q$ of size q.
- Codewords: truth tables of all multivariate polynomials $p : \mathbb{F}_q^m \to \mathbb{F}_q$ of total degree at most d.
- $n = \binom{m+d}{d} \cdot \log q$, $\hat{n} = q^m$.
- Minimum distance $\delta = 1 - d/q$
- List-decodable up to radius $1 - O(\sqrt{d/q})$ with lists of size $O(\sqrt{q/d})$.
- Generalization of both Reed-Solomon ($m = 1$) and Hadamard (essentially $d = 1$ and $q = 2$, except also have complements of the linear functions).