Coding Theory
Lecture 12

Lecture by Dr. Elette Boyle
Typeset by Steven Karas

2018-01-18
Last edited 18:11:02 2018-01-18

Disclaimer These lecture notes are based on the lecture for the course Coding Theory, taught by Dr. Elette Boyle at IDC Herzliyah in the fall semester of 2017/2018. Sections may be based on the lecture notes written by Dr. Elette Boyle.

Agenda

- Recap for exam
- Free for all - any/many subjects

1 Exam

80% of the grade is based on the exam, 20% on the homeworks.

An sample exam and its solution have been published. The topics covered are listed in the syllabus.

The exam will have 6 questions worth a total of 112 points.

10 pages of notes will be allowed.

2 Basic Code Theory

Definition of block length n, dimension k, rate k/n, and distance d

3 Linear Codes

Finite Fields, linear algebra as background.

Defined by the codewords which form a linear subspace.

Generator matrix Parity check matrix

Code Families and asymptotics

Hamming Codes

Theorem: Distance of a linear code is the minimum weight of any codeword

Theorem: Distance of a linear code is the number of linearly dependent columns in the parity check matrix.

Dual code. Wasn't covered in the course. e.g. Hamming and Hadamard

4 Probability Theory

4.1 Chernoff Bound

$C^{\perp}$ provides a strong bound on the theoretical

4.2 Entropy

While we covered q-ary entropy, we are only expected to know binary entropy:

$$H_2(x) = -x \log x - (1-x) \log(1-x)$$

4.3 Hamming Balls

Approximate hamming balls:

$$\mathrm{Vol}_q(pn, n) \sim q^{H_q(p)n}$$

5 Bounds

5.1 Hamming bound

Based on sphere packing, exact bound is somewhat difficult to state. Asymptotically, this is based on the approximation of hamming ball volume.

5.2 GV Bound

Optimistic bound. If we double the size of the hamming balls, examines the worst case of removing balls. Gives us a bound on the rates, and constructs a linear code with the requested properties.

5.3 Singleton Bound

$$d \leq n - k + 1$$

Derived from removing prefixes of codewords.

5.4 Plotkin Bound

Pessimistic bound. The intuition is that the angles between neighboring codewords must be more than 90 degrees.

Derivation is done with a geometric lemma.

6 Channels

BSC_p channels have probability p of flipping bits.

$q\mathrm{SC}_p$ is a q-ary channel with $p/(q-1)$ probability of a specific error symbol.

Capacity of the channel is the highest rate that we can communicate over the channel:

$$Cap(\mathrm{BSC}_p) = 1 - H_2(p)$$

We should be able to compute the capacity for a simple channel.

7 Reed Solomon

Linear code over a finite field $\mathbb{F}_q$ where $q \geq n$.

7.1 Encoding

Let $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$.

First way is to use the message vector as the coefficients:

$$f_{\vec{m}} = \sum_{i=0}^{k-1} m_i x^i$$

Other approach is to interpolate the polynomial given the message vector as evaluation points.

In either case, we need for this to be a bijective mapping. Also, we'd like for the mapping to be efficiently computed.

The codeword $\vec{y}$ is:

$$\vec{y}_i = f_{\vec{m}}(\alpha_i)$$

7.2 Linear Code

From this, we can construct the generator matrix such that the columns are the evaluation points with coefficients of 1.

7.3 Singleton bound

consider the columns of the generator matrix

7.4 Unique Decoding

Berlekamp-Welch algorithm:

There must exist a polynomial with the same degree as the number of errors such that:

$$E(\alpha_i)P(\alpha_i) = y_i E(\alpha_i)$$

For example:

$$E(x) = \prod_{i \in \text{errors}} (x - \alpha_i)$$

Then use linearization to solve $N(\alpha_i) = E(\alpha_i)P(\alpha_i)$.

7.5 List decoding

Sudan algorithm (listed as algorithm 1 or 2 in the book. Algo 3 was not covered).

Basically, look at $Q(x, y)$ and choose the monomials of x and y to convert the received message into a system of linear equations. This can help us decode all the codewords in the voronoi cell, rather than just the hamming ball (which due to the curse of dimensionality, means there is a lot of leftover space).

8 Expander Codes

We can look at the parity check matrix as an adjacency matrix of a bipartite graph.

Each of the righthand nodes is a parity check on its neighbors to the left.

We proved that expansion happens.

8.1 Encoding & Decoding

For encoding, we can do matrix multiplication in linear time due to the sparsity of the generator matrix.

For decoding, we iteratively flip neighbors of failed parity checks. Proven to finish within linear time.

9 Almost Universal Hash Functions

Basically hash families that the probability of a collision is bound by ε.

Equivalent to codes with good rate, and we should know the equivalencies and be able to prove them.

10 Locally Decodable Codes

A (r, δ, ε)-LDC requires r samples from a received message with fewer than δ fraction of errors gives us a given message symbol with probability ε.

Locally decoding gives us a symbol from the original message. Locally correcting gives us a symbol from the codeword.

Systematic codes have the message appear as a prefix of the codeword. LCCs are LDCs for systematic codes.

10.1 Hadamard code

Dual of the hamming code, generator matrix $k \times 2^k$ is the columns of all the inner products. Provides $(2, \delta, 2\delta)$-LDC.

Local decoding the i-th position is selecting a random column $\vec{v}$, and the other at the position $\vec{v} + e_i$.

10.2 Reed-Muller

Multivariate extension of Reed Solomon.

Intuitively, oversampling of multivariate polynomial. So codewords are surfaces in the q-dimensional space.

$\left(q^m, \binom{m-\ell}{\ell}, \left(1 - \frac{\ell}{q}\right) q^m\right)$-code.

Gives us $(\ell + 1, \delta, (\ell + 1)\delta)$-LDC. Intuition of local decoding is taking a cross section of a line starting from the position we want: $L = \{\vec{w} + \vec{v}\lambda \mid \lambda \in \mathbb{F}_q\}$. This reduces the problem to Reed-Solomon.

11 Private Information Retrieval (PIR)

Given r servers that don't communicate, we want to request from each server such that no individual server learns which index in the database we want to read.

11.1 Smoothness

Smoothness says that when running LDC, the individual queries are uniformally distributed over the codeword.

12 Further Applications

We discussed secret sharing, secure computation, and regenerating codes.