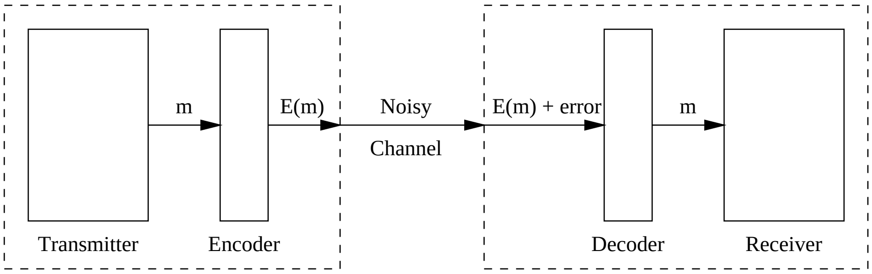
6.893 Approximability of Optimization Problems October 13, 1999

Lecture 10

Lecturer: Madhu Sudan *Scribe: Chandra Boyapati*

1 Error Correcting Code (ECC)

To deal with the problem of the inevitable noise that occurs during information transmission (or storage), one tends to encode information using an "error-correcting code", before transmitting it (or before storing it). This allows the receiver to recover the original information even if a few bits (more general, symbols) get flipped during the transmission.



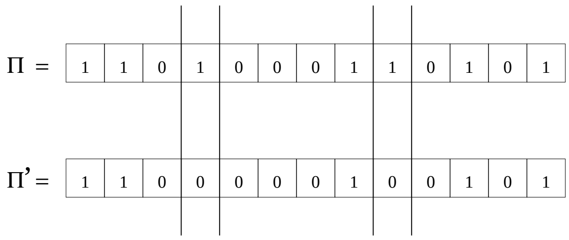
In this lecture, we will first formalize the notion of error correcting codes and list the various parameters of ECCs that have been considered in the literature. We will then discuss some particular error correcting codes, namely, the Reed-Solomon code, the Reed-Muller code and the Hadamard code [Lin82]. We will also discuss Forney's method of composing any two codes to get a new error correcting code.

1.1 PCPs and Error Correcting Codes

According to the PCP theorem [BLFS91, Hås97], for any language $L \in NP$, we have:

$$\forall x \in L,\ \exists\ proof\ \Pi\ s.t.\ \mathop{\mathbf{E}}_{R}[V^{\Pi}(x;R)] \geq 1-\epsilon$$

Now, consider a proof Π', which is obtained by flipping, say 1%, of the bits of Π.



1 % of the bits are flipped

Using the tight 3-query characterization of NP with PCPs [Hås97, GDLT98], we get:

$$\mathbf{Pr}[V^{\Pi'}(x;R) \text{ accepts}] \geq (1-\epsilon)(1-3\%) \geq .97-\epsilon$$

For any $x \notin L$, such a Π' does not exist.

1.2 Parameters of ECCs

Let, C = A collection of strings (or vectors) over some **q**-ary alphabet.

Let, **Block length of the code** = Length of the strings in C = **n**.

Thus, $C \subseteq \{0, ..., q-1\}^n$.

Let, **Information content of the code** = Length of the message = **k**.

Thus, $|C| = q^k$.

Let m_1 and m_2 be two distinct messages. Consider the quantities $E(m_1) + error_1$ and $E(m_2) + error_2$, where E is the encoding function. Let us assume that $error_1$ and $error_2$ flip at most t bits each. Now if $E(m_1) + error_1 = E(m_2) + error_2$, then we will not be able to successfully decode the messages. To protect from such errors, we have to design E such that $E(m_1)$ and $E(m_2)$ always differ in at least $2t+1$ places.

Let, **Distance of the code** = $\min_{m_1 \neq m_2} \{\text{Hamming distance}(E(m_1), E(m_2))\}$ = **d**.

Hamming distance between two strings (or vectors) is defined to be the number of co-ordinates where they differ.

Hamming distance$(x, y) = \#i \ s.t. \ x_i \neq y_i$.

Then, **C is said to be a $[\mathbf{n}, \mathbf{k}, \mathbf{d}]_\mathbf{q}$ code**.

2 Reed-Solomon Code

The Reed-Solomon encoding is based on *univariate polynomials over finite fields.* Recall that $\forall$ prime numbers p and $\forall$ positive integers q, $\exists$ a finite field $(F, +, *)$ s.t $|F| = p^q$.

Let $q = |F|$.
Let $n \leq q$. Let $x_1, ..., x_n$ be distinct elements of F.

Let message $m = m_0...m_{k-1}$, where $k \leq n$ and each $m_i \in F$.
Define a corresponding message polynomial $M(x) = \sum_{i=0}^{k-1} m_i x^i$.

Let the encoding of the message, $E(m) = \langle M(x_1), M(x_2), ...M(x_n) \rangle$.

Distance of Reed-Solomon code

$= $ Hamming distance between $\langle M_1(x_1), ...M_1(x_n) \rangle$ and $\langle M_2(x_1), ...M_2(x_n) \rangle$, where $M_1 \neq M_2$
$= n-$ The number of points at which $M_1 - M_2$ evaluates to 0
$\geq n - k + 1$, since degrees of M_1, $M_2 = k - 1$

But from the singleton bound in coding theory, we know that $\forall n, k, d$, if C is an $[n, k, d]_q$ code, then $d \leq n - k + 1$. *(Proof is left as an exercise.)*

Thus, distance of Reed-Solomon code $= n - k + 1$.

It follows from above that the Reed-Solomon code is a $[n, k, (n-k+1)]_q$ code.

2.1 Why it is a Non-Trivial Result

Even though the number of symbols in the alphabet q is greater than the message length k, this is a non-trivial result.

For example, let $k = \frac{n}{2}$, and let $q = n$. Then, the number of possible messages $= n^{\frac{n}{2}}$. Thus, the alphabet size is logarithmic in the information size.

3 Reed-Muller Code

The Reed-Muller encoding is based on *multivariate polynomials over finite fields.*

3.1 Multivariate Polynomial Distance Lemma

Let $P(x_1, ..., x_m) = \sum_{(i_1,...,i_m)} c_{i_1...i_m} x_1^{i_1}...x_m^{i_m}$

$Degree(x_1^{i_1}...x_m^{i_m})$ is defined to be $(i_1 + ... + i_m)$.

$Degree(P)$ is defined to be $\max_{(i_1,...,i_m)\ s.t.\ c_{i_1}...c_{i_m} \neq 0}\{degree(x_1^{i_1}...x_m^{i_m})\}$

The distance lemma states that two m-variate polynomials P_1 and P_2 with degrees $\leq d$ can agree on at most $\frac{d}{q}$ fraction of the inputs from F^m. *(Proof omitted.)*

3.2 Reed-Muller Code

As in Reed-Solomon code, let $(F, +, *)$ be a finite field and let $q = |F|$.

Think of a message as specifying a polynomial of degree d on m variables.

Then, the message size k

$= $ Number of distinct $x_1^{i_1}...x_m^{i_m}$, s.t. $(i_1 + ... + i_m) \leq d$

$= C_m^{m+d}$ *(Proof left as an exercise.)*

The message is encoded by evaluating it at all F^M points. Thus $n = q^m$.

From the multivariate polynomial distance lemma stated above, it follows that distance of Reed-Muller code $= q^m(1 - \frac{d}{q})$.

3.2.1 m=1, d=q/2: Reed-Solomon Code

When $m = 1$, the multivariate polynomial reduces to a univariate polynomial. The Reed-Muller code then becomes a $[n, \frac{n}{2}+1, \frac{n}{2}]_n \approx [n, \frac{n}{2}, \frac{n}{2}]_n$ code, which is also a Reed-Solomon code.

3.2.2 m=2, d=q/2

$n = q^2 \Longrightarrow q = \sqrt{n}$

$k = C_2^{2+d} \approx \frac{d^2}{2} = \frac{q^2}{8} = \frac{n}{8}$

$Distance = q^2(1 - \frac{d}{q}) = \frac{q^2}{2} = \frac{n}{2}$

Thus, we get a $[n, \frac{n}{8}, \frac{n}{2}]_{\sqrt{n}}$ code.

3.2.3 Any small m, d=q/2

$n = q^m \implies q = \sqrt{n}$

$k = C_m^{m+d} \approx \frac{d^m}{m!} = \frac{q^m}{m!\ 2^m} = \frac{n}{m!\ 2^m}$

$Distance = q^m(1 - \frac{d}{q}) = \frac{q^m}{2} = \frac{n}{2}$

Thus, we get a $[n, \frac{n}{m!\ 2^m}, \frac{n}{2}]_{n^{\frac{1}{m}}}$ code.

3.2.4 d=1, q>d $\implies$ q$\geq$2

When $d = 1$, we get Reed-Muller code of order 1.

We get a $[n = q^m, (m+1), n(1 - \frac{1}{q})]_q$ code.

When $q = 2$, we get a $[n = 2^m, (m+1), \frac{n}{2}]_q$ code, which is the binary code.

4 Hadamard Code

4.1 Hadamard Matrix

An $m \times n$ matrix H over $\{+1, -1\}$ is a Hadamard matrix if $HH^T = nI$.

The diagonal entries in HH^T are obviously n. The other entries in HH^T will all be 0 iff the vectors defined by the rows of H are all orthogonal to each other, that is, if their dot products are zero.

The dot product of the vectors defined by i^{th} and j^{th} rows of H, $<H_i, H_j> = \sum_k H_{ik}H_{jk}$
$= (\#k \text{ where } H_{ik}H_{jk} = 1) - (\#k \text{ where } H_{ik}H_{jk} = -1)$

$<H_i, H_j> = 0 \quad \forall i \neq j$
$\implies (\#k \text{ where } H_{ik}H_{jk} = 1) = (\#k \text{ where } H_{ik}H_{jk} = -1)$
$\implies (\#k \text{ where } H_{ik} = H_{jk}) = (\#k \text{ where } H_{ik} \neq H_{jk})$
$\implies n = 2(\#k \text{ where } H_{ik} \neq H_{jk})$
$\implies (\#k \text{ where } H_{ik} \neq H_{jk}) = \frac{n}{2}$

4.2 Hadamard Code

The Hadamard code is defined over a binary alphabet. Let H be a $m \times n$ Hadamard matrix. (For such a matrix to exist, n should be even and $m \leq n$.)

Let, the length of the messages, k, be equal to $\log_2 m$. Let $m_1 ... m_k$ be such a message.

The encoding of the message is given as follows. $< m_1 ... m_k >$ can be treated as a binary number b in

$[0, m-1]$. The encoding of the message would then be the b^{th} row in the $m \times n$ Hadamard matrix H. The block length of the code is thus equal to n.

The distance of the code is the maximum Hamming distance between any two rows of the matrix, which, as we derived in section 4.1, is equal to $\frac{n}{2}$.

The Hadamard code is therefore a $[n, \log_2 m, \frac{n}{2}]_2$ code.

When $m = n$, we get a $[n, \log_2 n, \frac{n}{2}]_2$ code.

5 Forney's Concatenation of Codes

In the previous sections, we have seen error correcting codes with various properties.

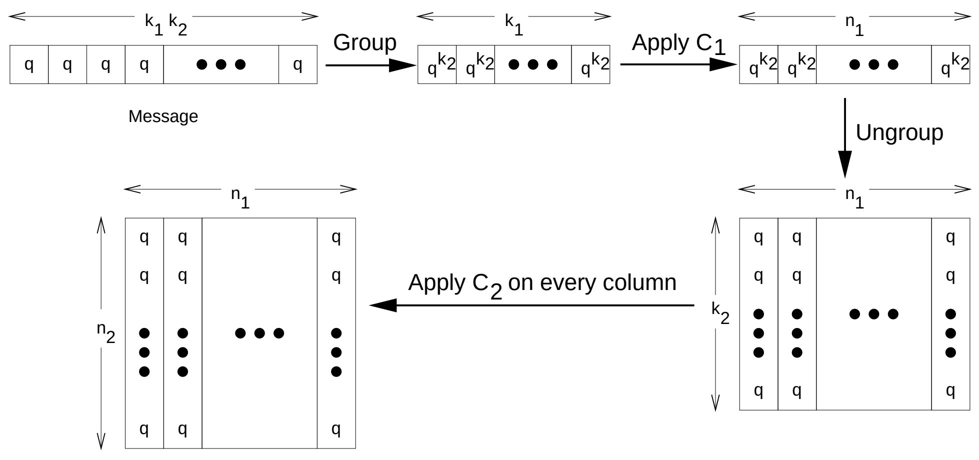
Some ECCs have a message length which is linear in the block length of the code n, though the alphabet size in such codes is very large ($O(poly(n))$). For example, see sections 3.2.1, 3.2.2 and 3.2.3.

On the other hand, other ECCs have a constant sized alphabet, but they have a message size which is only $O(\log n)$. For example, see sections 3.2.4 and 4.2.

Thus, there seems to be a trade-off between message size and alphabet size. Ideally, we would want to combine codes that have large message sizes with codes that have small alphabet sizes to give new codes that give us better trade-offs. Forney's concatenation allows us to do exactly that. (We will see very similar constructions when we study PCPs in depth later in the course.)

Let C_1 be a $[n_1, k_1, d_1]_{q^{k_2}}$ code.
Let C_2 be a $[n_2, k_2, d_2]_q$ code.

Using Forney's concatenation, $C_1 \odot C_2$ will be a $[n_1 n_2, k_1 k_2, d_1 d_2]_q$ code. Forney's method is pictorially depicted below.



5.1 Some Applications of Forney's concatenation

5.1.1 Example 1

Let $C_1 = [n, \frac{n}{2}, \frac{n}{2}]_n$ (Reed-Solomon code)
Let $C_2 = [n, \log n, \frac{n}{2}]_2$ (Hadamard code)

Then, $C_1 \odot C_2 = [n^2, \frac{n}{2}\log n, \frac{n^2}{4}]_2 \approx [N, \frac{\sqrt{N}}{2}, \frac{N}{4}]_2$

5.1.2 Example 2

Let $C_1 = [n_1, \frac{n_1}{2}, \frac{n_1}{2}]_{n_1}$ (Reed-Solomon code)
Let $C_2 = [n, \frac{n}{2}, \frac{n}{2}]_n$ (Reed-Solomon code)
Let $n_1 = 2^{\frac{n}{2}}$

Let $C_3 = C_1 \odot C_2$
Let $C_4 = [n, \log n, \frac{n}{2}]_2$ (Hadamard code)
Let $C_5 = C_3 \odot C_4$

Then $C_5 = [N = n_1 n^2, \frac{1}{4} n_1 n \log n, \frac{N}{8}]_2 \approx [N = k\log^2 k, \frac{1}{8} k \log k \log\log k, \frac{N}{8}]_2$

References

[BLFS91] L. Babai, L. Levin L. Fortnow, and M. Szegedy. Checking computations in polylogarithmic time. In *23rd Annual ACM Symposium on Theory of Computing*, 1991.

[GDLT98] V. Guruswami, M. Sudan D. Lewin, and L. Trevisan. A tight characterization of NP with 3-query PCPs. In *39th IEEE Symposium on Foundations of Computer Science*, 1998. Also available as ECCC Technical Report TR97-038.

[Hås97] J. Håstad. Some optimal inapproximability results. Technical Report TR97-037, Electronic Colloquium on Computational Complexity, 1997. Preliminary version in *Proceedings of STOC*, 1997.

[Lin82] J H Van Lint. *Introduction to Coding Theory.* Springer Verlag, 1982.