CS 229r Essential Coding Theory Jan 27, 2020

Lecture 6

Instructor: Madhu Sudan *Scribe: Fabio Pruneri*

In this course, we construct codes using ideas and techniques from a wide variety of areas of Mathematics. Our primary goal is to obtain optimal codes in some sense, as opposite to deriving codes from a specific tools. Algebraic codes have remarkably good combinatorial parameters, in many cases matching the corresponding upper bound. Furthermore, many can be constructed efficiently in polynomial or even close to linear time using algebraic properties of underlying objects.

1 Review of basic concepts in algebra

1.1 Fields

A field over a set F is constructed by defining addition and multiplication $+, \cdot : F \times F \to F$ so they satisfy the usual properties:

1. Associativity: $(a+b)+c = a+(b+c)$ and $(ab)c = a(bc)$;
2. Commutativity: $a+b = b+a$ and $ab = ba$;
3. Identities: $\exists\ 0, 1 \in F$ such that $\forall a, a+0 = a, a \cdot 1 = a$;
4. Additive inverses: $\forall a, \exists -a$ such that $a + (-a) = 0$;
5. Multiplicative inverses: $\forall a \neq 0, \exists a^{-1}$ such that $a \cdot a^{-1} = 1$;
6. Distributivity: $a(b+c) = ab + ac$.

This is similar to how addition and multiplication are defined over the rationals $\mathbb{Q}$. Rings satisfy all properties of fields, except 5. (multiplicative inverses do not necessarily exist).

1.2 Finite fields

A field is finite if the underlying set F has a finite number of elements. We will generally deal with finite fields in this course, and denote them with $\mathbb{F}_q$ where $q = |F|$ is the size of the set/field.

As an example, it can be verified that for every prime p, the set $\mathbb{Z}_{/p}$ where addition and multiplication are defined modulo p is a field.

It turns out that finite fields $\mathbb{F}_q$ only exists for certain values of q, specifically only if $q = p^t$ for p prime and $t \geq 1$ is a prime power. Furthermore, for every valid size q, the field is unique up to isomorphism. We will learn how to construct fields of cardinality p^t for $t > 1$ later.

1.3 Polynomials over fields

We can define a ring of polynomials $\mathbb{F}[x]$ in a formal variable x over any field $\mathbb{F}$ by defining addition and multiplication in the traditional way

$$\mathbb{F}[x] = \{a_0 + a_1 x + ... + a_d x^d \mid a_0, ..., a_d \in \mathbb{F}, d \in \mathbb{N}, a_d \neq 0\}$$

$$(a_0 + ... + a_d x^d) + (b_0 + ... + b_d x^d) = (a_0 + b_0) + ... + (a_d + b_d) x^d$$

$$(a_0 + ... + a_d x^d) \cdot (b_0 + ... + b_d x^d) = \sum_{i=0}^{d} \sum_{j=0}^{d} a_j b_i x^{i+j}$$

padding polynomials with leading 0s if the degrees do not match. We usually denote polynomials with the letter $p \in \mathbb{F}[x]$, and where $\deg p = d$ is the degree.

$\mathbb{F}[x]$ is a ring, but not a field because multiplicative inverses are not necessarily defined (there does not exist any p such that $x^2 \cdot p = x$, for example). But we can still use the Euclidean algorithm to perform finite division as in the following statement:

Theorem 1. *If $f, g \in \mathbb{F}[x]$, there esist $q, r \in \mathbb{F}[x]$ such that $f = g \cdot q + r$ and $\deg r < \deg g$.*

It can also be proven that $\mathbb{F}[x]$ is an unique factorization domain (up to field elements).

1.4 Evaluation of polynomials

We also define the evaluation map $Eval_{\mathbb{F}} : \mathbb{F}[x] \times \mathbb{F} \to \mathbb{F}$ over a field $\mathbb{F}$ as formalizing the intuitive definition of "substituting" a value in the field for x in the polynomial:

$$Eval_{\mathbb{F}}((a_0 + ... + a_d x^d), v) = a_0 + ... + a_d v^d$$

and we shorthand $p(v)$ for $Eval_{\mathbb{F}}(p, v)$. It behaves in the expected way, so that for example $(p+q)(v) = p(v) + q(v)$.

If you perform Euclidean division of an arbitrary $p \in \mathbb{F}[x]$ by $(x - \alpha)$ the result is a polynomial of degree 0, which is akin to a field element, which turns out to be the same as $p(\alpha)$. So if α is s.t. $p(\alpha) = 0$, $(x - \alpha)$ divides $p(x)$.

This, along with unique factorization as mentioned above, imply that a nonzero polynomial of degree d has at most d roots. This is a key fact for constructing Reed-Solomon codes.

2 Reed-Solomon codes

Reed-Solomon codes capture the idea of transmitting a polynomial by specifying its values when evaluated on n values of the underlying field. The code is a function of parameters $\mathbb{F}_q, n \leq q, k \leq n$ and a sequence of scalars $\langle \alpha_1, ..., \alpha_n \rangle$ in $\mathbb{F}_q$, where all α_is are distinct. Its messages are polynomial $m \in \mathbb{F}_q[x]$ of degree at most $\deg m \leq k - 1$ and the encoding function is the evaluation of the polynomial at the n given points:

$$E_{\mathbb{F}_q, n, k, \langle \alpha_1, ..., \alpha_n \rangle}(m) = (m(\alpha_1), ..., m(\alpha_n)) \in \mathbb{F}_q^n$$

so if $m = c_0 + ... + c_{k-1} x^{k-1}$, the first element of the encoded tuple is $m(\alpha_1) = c_0 + ... + c_{k-1} \alpha_1^{k-1}$. Please note that the points at which the polynomial is evaluated are fixed parameters of the code and are the same for all messages.

2.1 Parameters of a RS code

Here we claim that the code $C = \{E_{\mathbb{F}_q, n, k, \langle \alpha_1, ..., \alpha_n \rangle}(m) \mid m \in \mathbb{F}_q\}$ defined by the encoding function above is a linear $[n, k, n - k + 1]_q$ code.

Polynomials preserve degree under addition and multiplication by field elements, so this is a linear code. Clearly the set of valid messages has the same size as the set of polynomials in $\mathbb{F}_q[x]$ of degree at most $k - 1$, which is q^k by looking at the possible coefficients. The image of E is a set of n-tuples of $\mathbb{F}_q$ so has size at most q^n.

We are left to prove that this code has in fact distance $d = n - k + 1$. Assume by contradiction that two polynomials m_1, m_2 have smaller distance, and therefore the tuples $E(m_1), E(m_2)$ agree on at least k coordinates. This implies that $m_1(\alpha_i) = m_2(\alpha_i)$ for k different αs in the field (remember we picked αs to be all distinct), and this quickly leads to a contradiction.

Exercise 2. *Prove that if two polynomials m_1, m_2 of degree strictly smaller than k agree on k distinct values $\alpha_1, ..., \alpha_k$ (i.e. $m_1(\alpha_i) = m_2(\alpha_i)$), then $m_1 = m_2$. Hint.*[1]

Also note that $q \geq n$ is required, as the points at which the evaluation is sent need to be distinct, otherwise the additional information is not useful.

2.2 Generator matrix

As for every linear code, we can find the associated matrix G such that $E(m) = mG$. If $m = c_0 + ... + c_{k-1}x^{k-1}$ this rewrites as

$$\begin{bmatrix} c_0 & c_1 & ... & c_{k-1} \end{bmatrix} \cdot G = \begin{bmatrix} m(\alpha_1) & m(\alpha_2) & ... & m(\alpha_n) \end{bmatrix}$$

which one can see by expanding the polynomials is solved by the matrix

$$G = \begin{bmatrix} 1 & 1 & ... & 1 \\ \alpha_1 & \alpha_2 & ... & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & ... & \alpha_n^2 \\ ... & ... & ... & ... \\ \alpha_1^{k-1} & \alpha_2^{k-1} & ... & \alpha_n^{k-1} \end{bmatrix}$$

which reminds of a Vandermonde matrix (except with maybe the wrong dimensions). Of course, dealing with this matrix explicitly is computationally messy (for instance, to check that the distance is the one claimed above), but it can sometimes be useful as in the following exercise.

Exercise 3. *Prove that you can recover m from the tuple $E(m)$ in deterministic polynomial time. Hint.*[2]

Remember that RS codes are optimal, in the sense that they match the upper bound $d \leq n - k + 1$ proved two lectures ago. We call codes C of optimal parameters $(n, k, d)_q, d = n - k + 1$ "Maximum Distance Separable" codes.

Exercise 4. *Prove that if C is a linear MDS code with generator matrix G, the code generated by $G^{\perp}$ is also MDS.*

2.3 Decoding with an adversary

Suppose now I send $E(m) = (m(\alpha_1), ..., m(\alpha_n))$ but you receive a tuple $(y_1, ..., y_n)$ tampered . by an adversary, so that some values y_i might be different from the "correct" value $m(\alpha_i)$ and be replaced with other field elements. Of course, a simple algorithm to dealing with this is to enumerate all possible tuples that differ from the received one in few positions, until we are able to decode one into a polynomial of degree strictly smaller than k. But this generally takes exponential time, so it is computationally infeasible.

In future lectures we will see a polynomial-time solution to this problem exists, using elementary properties of the underlying fields.

3 From small to big fields - Wozencraft codes

Finite fields also allow us to construct a family of codes that probabilistically match the Gilbert-Varshamov bound, but first we must learn to construct finite fields of size $q = p^t$ for $t > 1$.

[1]Apply the last statement of section 1.4 to the polynomial $m_1 - m_2$, which also has degree strictly smaller than k.
[2]Invert the matrix G

3.1 Constructing all finite fields

Unfortunately, the set of polynomials $\mathbb{F}[x]$ is not a field but a ring, because multiplicative inverses do not necessarily exist. One direction could be extending it to fractions of the form $\frac{p}{q}$ for $p, q \in \mathbb{F}[x]$, similarly to how rationals extend integers.

But a more promising approach is akin to the transformation the brings $\mathbb{Z} \to \mathbb{Z}_{/p}$: select a polynomial g of degree t that is monic and irreducible, and map elements $m \in \mathbb{F}_p[x]$ to the remainders obtained by dividing m by t. This set, $\mathbb{F}_p[x]_{/g}$ contains all polynomials of degree strictly smaller than t and coefficients in $\mathbb{F}_p$, so it has the desired p^t elements.

Exercise 5. *Prove that by defining addition and multiplication over $\mathbb{F}_p[x]$ modulo an irreducible g, all field axioms are satisfied.*

One can see how building a field of size, say, $2^8 = 256$ could be useful when trying to transmit bytes using linear codes. But how to find a suitable g without checking all p^t possibilities? One approach is to factor random polynomials until one works, which works because of the following exercise.

Exercise 6. *Prove that a random polynomial in $\mathbb{F}_p[x]$ of degree k is reducible with probability $\frac{1}{p}$.*

There also exists a deterministic algorithm for finding such a polynomial, but this is beyond the scope of the course. In any case, assume we can construct fields of any size, in time logarithmic to that size.

3.2 Transposing the map to polynomial fields

Suppose for now we wish to create a code over the binary alphabet of rate $R = \frac{1}{2}$. This is essentially a mapping $\mathbb{F}_2^k \to \mathbb{F}_2^{2k}$. The key insight is that we can instead define a mapping $\mathbb{F}_{2^k} \to \mathbb{F}_{2^k}^2$ to the same effect, and then relate this back to the original problem.

In fact, we actually create a collection of codes depending on a parameter $\alpha \in \mathbb{F}_{2^k}$ as given by the following encoding function:

$$E_\alpha(m) = (m, \alpha m), \forall m \in \mathbb{F}_{2^k}$$

this is a function $\mathbb{F}_{2^k} \to \mathbb{F}_{2^k}^2$ and its associated code $C_\alpha = \{(m, \alpha m) \mid m \in \mathbb{F}_{2^k}\}$ has size 2^k as desired. But how do we map this back to the original problem and specifically, how do we relate the distance of this code to the Hamming distance over tuples?

3.3 Existence of good C_α by probablistic method

First, construct a linear mapping L that sends elements from $\mathbb{F}_{2^k}$ back to elements of $\mathbb{F}_2^k$. For instance, it's easy to see that the mapping of a polynomial to its coefficients $c_0 + ... + c_{k_1} x^{k-1} \to (c_0, ..., c_{k-1})$ is linear. Compose this mapping with E_α to get its projection $\tilde{E_\alpha} : \mathbb{F}_2^k \to \mathbb{F}_2^{2k}$:

$$\tilde{E_\alpha}(x) = (L \circ E_\alpha \circ L^{-1})(x)$$

define $\tilde{C}_\alpha$ similarly. Now notice this lemma

Lemma 7. *If $\alpha \neq \beta$, $C_\alpha \cap C_\beta = \overline{0}$.*

Proof. Suppose instead there is an element $(x, y) \in C_\alpha \cap C_\beta$. This means $(m_1, m_1\alpha) = (x, y) = (m_2, m_2\beta)$ for some m_1, m_2. But $m_1 = x = m_2$ implies $m_1 = m_2$ and $m_1\alpha = y = m_2\beta$ implies $m_1\alpha - m_2\beta = x(\alpha - \beta) = 0$. But since $\alpha \neq \beta$, $m_1 = m_2$, and $(x, y) = \overline{0}$, as desired. □

Now we are ready to complete the proof

Theorem 8. *Let d be the largest integer such that $Vol(2k, d-1) < 2^k$. Then there exists $\alpha \in \mathbb{F}_{2^k}$ such that $\Delta(\tilde{C_\alpha}) \geq d$.*

Proof. Assume by contradiction that for every α we have $\Delta(\tilde{C_\alpha}) < d$. Using the fact that our codes $\tilde{C_\alpha}$ are linear, this condition rewrites as

$$\forall \alpha, \exists\; m \neq \overline{0} \mid \Delta(m, \overline{0}) < d$$

But notice that, as a consequence of the Lemma proved above, the same $m' \in \mathbb{F}_{2^k} - \overline{0}$ cannot be in both $C_{\alpha_1}, C_{\alpha_2}$, so the same $m \in \mathbb{F}_2^k - \overline{0}$ cannot be in both $\tilde{C_{\alpha_1}}, \tilde{C_[}\alpha_2$, for $\alpha_1 \neq \alpha 2$. Since there are at most $2^k - 1$ valid values of m, but 2^k values of α and corresponding $\tilde{C_\alpha}$, at least one $\tilde{C_\alpha}$ must not have any m for which the above condition holds, contradiction. $\square$

In fact, with very slight tweaks to the above proof we can prove a similar statement that allows us to efficiently find a good value of α

Exercise 9. *Prove most αs give a code of distance $d - \sqrt{k}$.*

We will use this fact constructively in the following lectures to get good binary codes explicitly.

4 Future directions

The codes we explored today perform well only if the underlying alphabet (field) is large; in future lectures we will try to make it smaller in different ways:

1. Using polynomials in more than one variable;
2. Representing elements of a large alphabet as strings in a smaller alphabet;
3. Taking the subset of a code in a large alphabet by selecting only codes with digits 0 and 1.

All of these will lead to codes that are optimal in some cases.