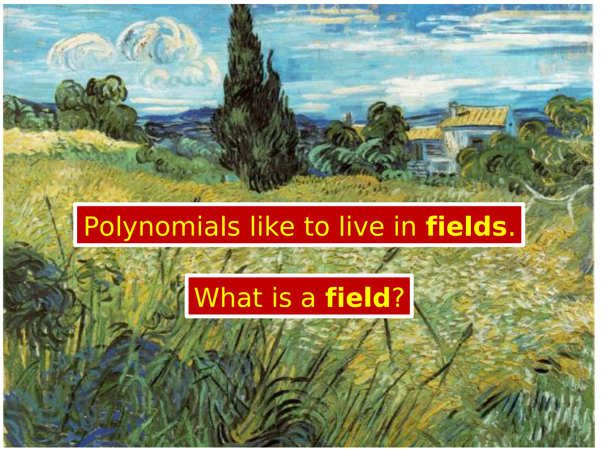
15-251: Great Theoretical Ideas in Computer Science
Lecture 22

Polynomials



Find out about the wonderful world of $\mathbb{F}_{2^k}$
where two equals zero, plus is minus,
and squaring is a linear operator!

– Rich Schroeppel

Fields

Informally, it's a number system where you can add, subtract, multiply, and divide-by-nonzero, with all the "usual laws of arithmetic" applying.

Examples:	Real numbers	$\mathbb{R}$
	Rational numbers	$\mathbb{Q}$
	Complex numbers	$\mathbb{C}$
	Integers mod ***prime***	$\mathbb{Z}_p$
NON-examples:	Integers $\mathbb{Z}$	division??
	Positive reals $\mathbb{R}^+$	subtraction??

Fields

A field can be specified by its addition and multiplication tables.

$\mathbb{F}_3 = \mathbb{Z}_3$

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

•	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

Must also check: Addition, multiplication are associative and commutative; field contains 0 such that 0+x=x ∀x; field contains 1 such that 1•x=x ∀x; for all x, exists −x s.t. x+(−x)=0; for all x≠0, exists x^{-1} s.t. x•x^{-1}=1; multiplication distributions over addition.

Finite fields

Some familiar *infinite* fields: $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$

Finite fields we know: $\mathbb{Z}_p$ aka $\mathbb{F}_p$, for p a prime

Is there a field with 2 elements? Yes
Is there a field with 3 elements? Yes
Is there a field with 4 elements? Yes

$\mathbb{F}_4$

+	0	1	a	b
0	0	1	a	b
1	1	0	b	a
a	a	b	0	1
b	b	a	1	0

•	0	1	a	b
0	0	0	0	0
1	0	1	a	b
a	0	a	b	1
b	0	b	1	a

Finite fields

Is there a field with 2 elements? Yes
Is there a field with 3 elements? Yes
Is there a field with 4 elements? Yes
Is there a field with 5 elements? Yes
Is there a field with 6 elements? No
Is there a field with 7 elements? Yes
Is there a field with 8 elements? Yes
Is there a field with 9 elements? Yes
Is there a field with 10 elements? No

Finite fields

Theorem:

There is a field with q elements
if and only if q is a power of a prime.

Up to *isomorphism*, it is unique.

I.e., all fields with q elements have the
same addition and multiplication tables,
after renaming elements.

This field is denoted $\mathbb{F}_q$.

Finite fields

Question:

If q is a prime power but not just a prime,
what ***are*** the addition and multiplication
tables of $\mathbb{F}_q$?

Answer:

It's a little hard to describe.

We'll tell you later, but for 251's purposes,
you only need to know about prime q.

Today's main topic:
Polynomials

Polynomials

Informally, a polynomial is an expression
that looks like this:

$$6x^3 - 2.3x^2 + 5x + 4.1$$

x is a symbol, called the *variable*

the 'numbers' standing next to
powers of x are called the *coefficients*

Polynomials

Informally, a polynomial is an expression
that looks like this:

$$6x^3 - 2.3x^2 + 5x + 4.1 \quad \in \mathbb{R}[\mathbf{x}]$$

Actually, coefficients can come from any *field*.

Can allow multiple variables; we won't in this lecture.

The set of polynomials with variable x and
coefficients from field F is denoted **F[x]**.

Polynomials – formal definition

Let F be a field and let x be a variable symbol.

F[x] is the set of polynomials over F,

defined to be expressions of the form

$$c_d\, x^d + c_{d-1}\, x^{d-1} + \cdots + c_2\, x^2 + c_1\, x + c_0$$

where each c_i is in F, and $c_d \neq 0$.

We call d the degree of the polynomial.

Also, the expression 0 is a polynomial.

(By convention, we call its degree $-\infty$, but don't get too hung up on it.)

Adding and multiplying polynomials

You can add and multiply polynomials.

Example. Here are two polynomials in $\mathbb{F}_{11}[x]$:

$$P(x) = x^2 + 5x - 1$$
$$Q(x) = 3x^3 + 10x$$

$$\begin{aligned} P(x) + Q(x) &= 3x^3 + x^2 + 15x - 1 \\ &= 3x^3 + x^2 + \ \ 4x - 1 \\ &= 3x^3 + x^2 + \ \ 4x + 10 \end{aligned}$$

Adding and multiplying polynomials

You can add and multiply polynomials.

Example. Here are two polynomials in $\mathbb{F}_{11}[x]$:

$$P(x) = x^2 + 5x - 1$$
$$Q(x) = 3x^3 + 10x$$

$$\begin{aligned} P(x) \bullet Q(x) &= (x^2 + 5x - 1)(3x^3 + 10x) \\ &= 3x^5 + 15x^4 + 7x^3 + 50x^2 - 10x \\ &= 3x^5 + \ \ 4x^4 + 7x^3 + \ \ 6x^2 + \ \ x \end{aligned}$$

Adding and multiplying polynomials

Polynomial addition is associative and commutative.

$0 + P(x) = P(x) + 0 = P(x)$.

$P(x) + (-P(x)) = 0$.

Polynomial multiplication is associative and commutative.

$1 \bullet P(x) = P(x) \bullet 1 = P(x)$.

Multiplication distributes over addition:

$P(x) \bullet (Q(x) + R(x)) = P(x) \bullet Q(x) + P(x) \bullet R(x)$

If P(x) / Q(x) were always a polynomial,

then F[x] would be a field! Alas…

Dividing polynomials?

P(x) / Q(x) is not necessarily a polynomial.

So F[x] is not quite a field.

(It's just a "commutative ring with identity".)

Same with $\mathbb{Z}$, the integers:

it has everything except division.

Actually, there are many analogies between

F[x] and $\mathbb{Z}$.

Dividing polynomials?

$\mathbb{Z}$ has the concept of "division with remainder":

Given $a,b \in \mathbb{Z}$, $b \neq 0$, can write

$$a = q \bullet b + r,$$

where r is "smaller than" b.

F[x] has the same concept:

Given $A(x), B(x) \in F[x]$, $B(x) \neq 0$, can write

$$A(x) = Q(x) \bullet B(x) + R(x),$$

where $\deg(R(x)) < \deg(B(x))$.

“Division with remainder” for polynomials

Example: Divide $6x^4+8x+1$ by $2x^2+4$ in $\mathbb{F}_{11}[x]$

$$\begin{array}{r|l} & 3x^2+5 \\ \hline 2x^2+4 & 6x^4+8x+1 \\ & -\ 6x^4+x^2 \\ \hline & \quad -x^2+8x+1 \\ & -\ \ -x^2+9 \\ \hline & \qquad 8x+3 \end{array}$$

Check:

$6x^4+8x+1$
$= (3x^2+5)(2x^2+4)+(8x+3)$
(in $\mathbb{F}_{11}[x]$)

Integers $\mathbb{Z}$	Polynomials F[x]
“size” = abs. value	“size” = degree
“division”: $a = qb+r$, $\lvert r\rvert < \lvert b\rvert$	“division”: $A(x) = Q(x)B(x)+R(x)$, $\deg(R) < \deg(B)$
can use Euclid’s Algorithm to find GCDs	can use Euclid’s Algorithm to find GCDs
p is “prime”: no nontrivial divisors	P(x) is “irreducible”: no nontrivial divisors
$\mathbb{Z}$ mod p: a field if p is prime	F[x] mod P(x): a field if P(x) is irreducible (with $\lvert F\rvert^{\deg(P)}$ elements)

Enough algebraic theory.

Let’s play with polynomials!

Evaluating polynomials

Given a polynomial $P(x) \in F[x]$,
$P(a)$ means its evaluation at element a.

E.g., if $P(x) = x^2+3x+5$ in $\mathbb{F}_{11}[x]$,

$P(6) = 6^2+3\cdot 6+5 = 36+18+5 = 59 = 4$

$P(4) = 4^2+3\cdot 4+5 = 16+12+5 = 33 = 0$

Definition: r is a **root** of P(x) if P(r) = 0.

Polynomial roots

Theorem:
Let $P(x) \in F[x]$ have degree 1.
Then P(x) has exactly 1 root.

Proof:
Write $P(x) = cx + d$ (where $c\neq 0$).
Then $P(r) = 0 \Leftrightarrow cr + d = 0$
$\Leftrightarrow cr = -d$
$\Leftrightarrow r = -d/c$.

Polynomial roots

Theorem:
Let $P(x) \in F[x]$ have degree 2.
Then P(x) has… how many roots??

E.g.: x^2+1…

# of roots over $\mathbb{F}_2[x]$: 1 (namely, 1)
# of roots over $\mathbb{F}_3[x]$: 0
# of roots over $\mathbb{F}_5[x]$: 2 (namely, 2 and 3)
# of roots over $\mathbb{R}[x]$: 0
# of roots over $\mathbb{C}[x]$: 2 (namely, i and −i)

The single most important theorem about polynomials over fields:

A nonzero degree-d polynomial has at most d roots.

Theorem: Over any field, a nonzero degree-d polynomial has at most d roots.

Proof by induction on $d \in \mathbb{N}$:

Base case: If P(x) is degree-0 then P(x) = a for some $a \neq 0$. This has 0 roots.

Induction:

Assume true for $d \geq 0$. Let P(x) have degree d+1.

If P(x) has 0 roots: we're done! Else let b be a root.

Divide with remainder: $P(x) = Q(x)(x-b) + R(x)$. $(*)$

$\deg(R) < \deg(x-b) = 1$, so R(x) is a constant. Say $R(x)=r$.

Plug $x = b$ into $(*)$: $0 = P(b) = Q(b)(b-b)+r = 0+r = r$.

So $P(x) = Q(x)(x-b)$. Now $\deg(Q) = d$. $\therefore$ Q has $\leq$ d roots.

$\therefore$ P(x) has $\leq$ d+1 roots, completing the induction.

Theorem: Over any field, a nonzero degree-d polynomial has at most d roots.

Reminder:

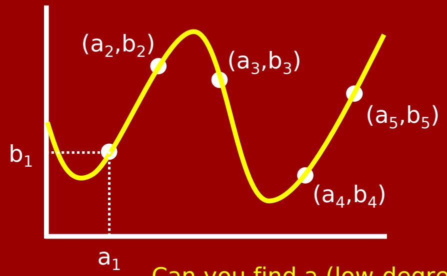
This is only true over a field.

E.g., consider $P(x) = 3x$ over $\mathbb{Z}_6$.

It has degree 1, but 3 roots: 0, 2, and 4.

Interpolation

Say you're given a bunch of "data points"



Can you find a (low-degree) polynomial which "fits the data"?

Interpolation

Let pairs $(a_1,b_1), (a_2,b_2), \ldots, (a_{d+1},b_{d+1})$ from a field F be given (with all a_i's distinct).

Theorem:

There is exactly one polynomial P(x) of degree at most d such that $P(a_i) = b_i$ for all $i = 1\ldots d+1$.

E.g., thru 2 points there is a unique linear polynomial.

Interpolation

There are two things to prove.

1. There is at *least* one polynomial of degree $\leq$ d passing through all d+1 data points.
2. There is at *most* one polynomial of degree $\leq$ d passing through all d+1 data points.

Let's prove #2 first.

Interpolation

Theorem:

Let pairs (a_1,b_1), (a_2,b_2), …, (a_{d+1},b_{d+1}) from a field F be given (with all a_i's distinct).
Then there is at most one polynomial $P(x)$ of degree at most d with $P(a_i) = b_i$ for all i.

Proof: Suppose P(x) and Q(x) both do the trick.

Let $R(x) = P(x)-Q(x)$.

Since $\deg(P), \deg(Q) \le d$ we must have $\deg(R) \le d$.

But $R(a_i) = b_i-b_i = 0$ for all $i = 1…d+1$.

$\therefore$ R(x) is the 0 polynomial; i.e., $P(x)=Q(x)$.

Interpolation

Now let's prove the other part, that there is at least one polynomial.

Theorem:

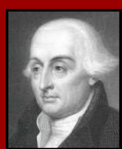
Let pairs (a_1,b_1), (a_2,b_2), …, (a_{d+1},b_{d+1}) from a field F be given (with all a_i's distinct).
Then there exists a polynomial $P(x)$ of degree at most d with $P(a_i) = b_i$ for all i.

Interpolation

The method for constructing the polynomial is called Lagrange Interpolation.

Discovered in 1779 by Edward Waring.

Rediscovered in 1795 by J.-L. Lagrange.



Lagrange Interpolation

a_1	b_1
a_2	b_2
a_3	b_3
…	…
a_d	b_d
a_{d+1}	b_{d+1}

Want $P(x)$
(with degree $\le d$)
such that $P(a_i) = b_i$ $\forall i$.

Lagrange Interpolation

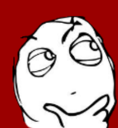
a_1	1
a_2	0
a_3	0
…	…
a_d	0
a_{d+1}	0

Can we do this special case?

Promise: once we solve this special case, the general case is very easy.

Lagrange Interpolation

a_1	1
a_2	0
a_3	0
…	…
a_d	0
a_{d+1}	0



Lagrange Interpolation

a_1	1
a_2	0
a_3	0
…	…
a_d	0
a_{d+1}	0

Just divide P(x) by this number.

Idea #1: $P(x) = (x-a_2)(x-a_3)\cdots(x-a_{d+1})$

Degree is d. ✔

$P(a_2) = P(a_3) = \cdots = P(a_{d+1}) = 0$. ✔

$P(a_1) = (a_1-a_2)(a_1-a_3)\cdots(a_1-a_{d+1})$. ??

Lagrange Interpolation

a_1	1
a_2	0
a_3	0
…	…
a_d	0
a_{d+1}	0

Numerator is a deg. d polynomial

Denominator is a nonzero field element

Idea #2:

$$S_1(x) = \frac{(x-a_2)(x-a_3)\cdots(x-a_{d+1})}{(a_1-a_2)(a_1-a_3)\cdots(a_1-a_{d+1})}$$

Call this the selector polynomial for a_1.

Lagrange Interpolation

a_1	0
a_2	1
a_3	0
…	…
a_d	0
a_{d+1}	0

Great! But what about this data?

$$S_2(x) = \frac{(x-a_1)(x-a_3)\cdots(x-a_{d+1})}{(a_2-a_1)(a_2-a_3)\cdots(a_2-a_{d+1})}$$

Lagrange Interpolation

a_1	0
a_2	0
a_3	0
…	…
a_d	0
a_{d+1}	1

Great! But what about this data?

$$S_{d+1}(x) = \frac{(x-a_1)(x-a_2)\cdots(x-a_d)}{(a_{d+1}-a_1)(a_{d+1}-a_2)\cdots(a_{d+1}-a_d)}$$

Lagrange Interpolation

a_1	b_1
a_2	b_2
a_3	b_3
…	…
a_d	b_d
a_{d+1}	b_{d+1}

Great! But what about this data?

$$\boxed{P(x) = b_1 \cdot S_1(x) + b_2 \cdot S_2(x) + \cdots + b_{d+1} \cdot S_{d+1}(x)}$$

Lagrange Interpolation – example

Over $\mathbb{F}_{11}$, find a polynomial P of degree ≤ 2 such that P(5) = 1, P(6) = 2, P(7) = 9.

$S_5(x) = \underbrace{6}(x-6)(x-7)$ $\qquad \frac{1}{(5-6)(5-7)}$

$S_6(x) = \underbrace{-}(x-5)(x-7)$

$S_7(x) = \underbrace{6}(x-5)(x-6)$

$P(x) = 1\ S_5(x) + 2\ S_6(x) + 9\ S_7(x)$

$= 6(x^2-13x+42) - 2(x^2-12x+35) + 54(x^2-11x+30)$

$= 3x^2+x+9$

Recall: Interpolation

Let pairs (a_1,b_1), (a_2,b_2), …, (a_{d+1},b_{d+1}) from a field F be given (with all a_i's distinct).

Theorem:

There is exactly one polynomial P(x) of degree at most d such that $P(a_i) = b_i$ for all i = 1…d+1.

Representing Polynomials

Let $P(x) \in F[x]$ be a degree-d polynomial.

Representing P(x) using d+1 field elements:

1. List the d+1 coefficients.
2. Give P's value at d+1 different elements.

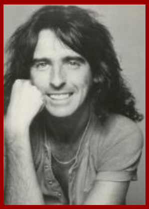
Rep 1 to Rep 2: Evaluate at d+1 elements

Rep 2 to Rep 1: Lagrange Interpolation

Application: Error-correcting codes

Sending messages on a noisy channel

Alice



" goo.gl/CXQ1OD " →

Bob

The channel may corrupt up to k symbols.

How can Alice still get the message across?

Sending messages on a noisy channel

Let's say messages are sequences from $\mathbb{F}_{257}$

CXQ1OD ↔ 118 114 120 85 66 78

noisy channel ↓

118 114 104 85 35 78

The channel may corrupt up to k symbols.

How can Alice still get the message across?

Sending messages on a noisy channel

Let's say messages are sequences from $\mathbb{F}_{257}$

CXQ1OD ↔ 118 114 120 85 66 78

noisy channel ↓

118 114 104 85 35 78

How to correct the errors?

How to even detect that there *are* errors?

Simpler case: "Erasures"

118 114 120 85 66 78

erasure channel ↓

118 114 ?? 85 ?? 78

What can you do to handle up to k erasures?

Repetition code

Have Alice repeat each symbol k+1 times.

118 114 120 85 66 78

becomes

118 118 118 114 114 114 120 120 120 85 85 85 66 66 66 78 78 78

erasure channel ↓

118 118 118 ?? ?? 114 120 120 120 85 85 85 66 66 66 78 78 78

If at most k erasures, Bob can figure out each symbol.

Repetition code – noisy channel

Have Alice repeat each symbol 2k+1 times.

118 114 120 85 66 78

becomes

118 118 118 114 114 114 120 120 120 85 85 85 66 66 66 78 78 78

noisy channel ↓

118 118 118 114 223 114 120 120 120 85 85 85 66 66 66 78 78 78

At most k corruptions: Bob can take maj. of each block.

This is pretty wasteful!

To send message of d+1 symbols and guard against k erasures, we had to send (d+1)(k+1) total symbols.

Can we do better?

This is pretty wasteful!

To send message of d+1 symbols and guard against k erasures, we had to send (d+1)(k+1) total symbols.

To send even 1 message symbol with k erasures, ***need*** to send k+1 total symbols.

Maybe for d+1 message symbols with k erasures, d+k+1 total symbols can suffice??

Enter polynomials

Say Alice's message is d+1 elements from $\mathbb{F}_{257}$

118 114 120 85 66 78

Alice thinks of it as the coefficients of a degree-d polynomial $P(x) \in \mathbb{F}_{257}[x]$

$$P(x) = 118x^5 + 114x^4 + 120x^3 + 85x^2 + 66x + 78$$

Now wants to send the degree-d polynomial P(x).

Send it in the Values Representation!

$P(x) = 118x^5 + 114x^4 + 120x^3 + 85x^2 + 66x + 78$

Alice sends P(x)'s values on d+k+1 inputs:
P(1), P(2), P(3), …, P(d+k+1)

This is called the **Reed–Solomon encoding**.

Send it in the Values Representation!

$P(x) = 118x^5 + 114x^4 + 120x^3 + 85x^2 + 66x + 78$

Alice sends P(x)'s values on d+k+1 inputs:
P(1), P(2), P(3), …, P(d+k+1)

If there are at most k erasures, then
Bob still knows P's value on d+1 points.

Bob recovers P(x) using Lagrange Interpolation!

Example

What about corruptions, not erasures?

Trickier. So let Alice now send P(x)'s value on
d + 2k + 1 inputs.

Assuming at most k corruptions,
Bob will have at least d+k+1 'correct' values.

P(1), P(2), bogus, P(4), bogus, P(6), …, P(d+2k+1)

Trouble: Bob does not know which values are bogus.

Corruptions under Reed–Solomon

Assuming at most k corruptions,
Bob will have at least d+k+1 'correct' values.

P(1), P(2), bogus, P(4), bogus, P(6), …, P(d+2k+1)

P(x) is a poly of degree ≤ d
which disagrees with the received
data on at most k positions.

Theorem: It is the **only** such polynomial.

Corruptions under Reed–Solomon

Theorem: P(x) is the **only** polynomial of degree ≤ d which disagrees with the data on ≤ k positions.

Proof:

Suppose Q(x) is another such poly.
P(x) and Q(x) disagree with each other on at most 2k positions.
∴ they agree with each other on at least (d+2k+1)−2k = d+1 positions.
∴ P(x) = Q(x) since they are degree ≤ d.

Corruptions under Reed–Solomon

Theorem: P(x) is the **only** polynomial of degree ≤ d which disagrees with the data on ≤ k positions.

Therefore Bob can determine P(x)!

Brute force algorithm:

Take each set of d+1 out of d+2k+1 values.
Interpolate to get a polynomial Q(x) of deg ≤ d.
Check if it agrees with ≥ d+k+1 values.

Efficient Reed–Solomon

Brute-force ‘decoding’ takes $2^{O(d)}$ time. ☹

Peterson 1960: a $O(d^3)$ decoding alg.

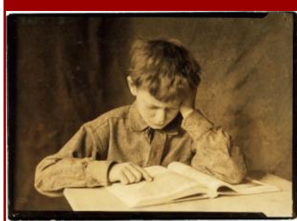
Berlekamp & Massey, late ‘60s:
key practical improvements

CMU’s Prof. Guruswami:
efficient algorithms to meaningfully correct more than k corruptions

Reed−Solomon codes are used in practice!

These are all RS codes.

Study Guide



Definitions:
Fields, polynomials

Theorem/proof:
Degree-d polys have at most d roots.

Algorithms:
Polynomial division with remainder
Lagrange Interpolation
Error correction and detection with Reed−Solomon