

Lecture 1 (30-07-2021)

30 July 2021 17:31

Polynomials - $x^2 + 2x + 1$
 $x^3 + 7x^2 + 9x + 3y + 2.1$
 $x^3 + \sqrt{3}y$
 $x^4 + 2.1i$ ($i^2 = -1$)

There are formal variables above and coefficients
from some other "objects".

Fields - Reals \mathbb{R} ,
Rationals \mathbb{Q} ,
Complex \mathbb{C} . } set, with operations $(+, \cdot)$,
operations have certain properties.

In general: $F \rightarrow$ a set

binary operations : $+$, \cdot

Properties:

① Associativity: $a + (b + c) = (a + b) + c$ } $\forall a, b, c \in F$
 $=: a + b + c$
 $(a \cdot b) \cdot c = a \cdot (b \cdot c)$

② Commutativity: $a + b = b + a$, $a \cdot b = b \cdot a$ $\forall a, b \in F$

③ Distributivity: $a \cdot (b + c) = a \cdot b + a \cdot c$ $\forall a, b, c \in F$

④ Existence of:
identity $\exists 0 \in F$ s.t. $0 + a = a \quad \forall a \in F$
 $\exists 1 \in F$ s.t. $1 \cdot a = a \quad \forall a \in F$
 ↴ multiplicative identity

So far, $\mathbb{Z}, \mathbb{Q}, \mathbb{C}, \mathbb{R}$ has the above properties.

(If such an identity exists, it is unique)

⑤ Existence of: $\forall a \in F \exists b \in F$ s.t. $a + b = 0$.

(5) Existence of inverse: $\forall a \in F \exists b \in F$ s.t. $a + b = 0$.
 inverse: $\forall a \in F \setminus \{0\} \exists c \in F$ s.t. $ac = 1$.

(Denoted $-a$ and a^{-1} , resp.)

Example: \mathbb{Z} is not a field. 2 has no mult. inverse.

$\mathbb{Q}, \mathbb{R}, \mathbb{C}$ are fields, these are infinite.

There are finite fields as well: $\{0, 1\} \rightarrow$ set
 define + and modulo 2.

This field is denoted \mathbb{F}_2 .

Similarly, there is $\mathbb{F}_3 = \{0, 1, 2\} \rightarrow$ sums and products
 modulo 3.

In general, given a prime p , we have

$$\mathbb{F}_p = \{0, 1, \dots, p-1\} \rightarrow \text{modulo } p.$$

Moreover, this is "the unique" field with p elements.

But there are not all. For p prime and $k \in \mathbb{N}$,

\exists a field with p^k elements, denoted \mathbb{F}_{p^k} .

(Again unique.)

Remark: \mathbb{F}_4 is NOT $\{0, 1, 2, 3\}$ with + and \cdot modulo 4.
 (2^{10} has no mult. inverse.)

[Alternate notation: $\mathbb{F}_{p^k} = GF(p^k)$.]

Non-examples of fields (1) \mathbb{Z} is not a field.

(2) $GL_n(F) = \{n \times n \text{ inv. matrices over } F\}$
 ↳ not even closed under (usual) addition

(3) $M_n(F) = \{n \times n \text{ matrices over } F\}$
 ↳ non-commutative for $n \geq 2$.

- Subsequent lectures:
- More about finite fields
 - What \mathbb{F}_{p^k} looks like.
 - Other properties. - useful later.
-

Commutative Ring.

Field properties apart from multiplicative inverse.

E.g. \mathbb{Z} is a commutative ring.

$F[x]$ — where F is a field

$F[x_1, \dots, x_n]$ — ring of polynomials in variables x_1, \dots, x_n .

• We shall use "ring" to mean "commutative ring".

→ Now we define something in between rings and fields.



Def. A ring R is an integral domain if

$$\forall a, b \in R : ab = 0 \Rightarrow a = 0 \text{ or } b = 0.$$

(There are no nonzero zero divisors.)

Example. \mathbb{Z} is an int domain, even $\mathbb{Z}[x]$ and $\mathbb{F}[x]$.

$\{0, 1, 2, 3\}$ with $+$ and \cdot mod 4 is NOT.

Def. $a \in R$ is a unit if it has a multiplicative inverse (in R).

That is, $\exists b \in R$ s.t. $ab = 1$.

Example. $\mathbb{Z} \rightarrow$ only two units $\{ \pm 1 \}$.

$\mathbb{F}[x] \rightarrow$ the units are precisely $\mathbb{F}^\times := \mathbb{F} \setminus \{0\}$.

Defn.: $a \in R$ is said to be **irreducible** if $\forall b, c \in R$:

$$a = bc \Rightarrow b \text{ is a unit} \\ \text{or} \\ c \text{ is a unit.}$$

Example. ① In \mathbb{Z} , the irreducibles are precisely the primes.

② $R = F[x]$. Irreducible polynomials
= poly. which cannot be written
as $b(x) c(x)$ where
 $\deg(b(x))$ and $\deg(c(x)) \geq 1$.
(since the units are precisely nonzero
field constants)

Defn.: An integral domain R is said to be a UFD if

- ① every element can be written as a product of irreducible elements,
- ② decomposition in ① is unique up to multiplication by units of R .

Example. ① \mathbb{Z} is a UFD (Fundamental Theorem of Arithmetic)

② $F[x]$, $F[x_1, \dots, x_n]$ are also UFDs.

Aim: Design efficient algorithms for coming up with a polynomial decomposition into irreducibles.

Getting from an integral domain to a field.

Given a problem over $\mathbb{Z} \rightarrow$ would be useful to view it over a field.

Example. $f(x) \in \mathbb{Z}[x] \rightarrow$ want to factorise will be helpful to think of it as $f \in \mathbb{Q}[x]$.

- Work over $\mathbb{Q}[x]$.
- Recover solution over $\mathbb{Z}[x]$

In general : $R \rightarrow$ integral domain
 $\tilde{R} \rightarrow$ field of fractions of R .

Defining \tilde{R} : $\tilde{R} \equiv R \times (R \setminus \{0\}) / \sim$

where $(a, b) \sim (a', b')$ iff $ab' = a'b$.

\sim is indeed an equivalence relation.

The following are well-defined:

$$[(a, b)] + [(c, d)] = [(ad + bc, bd)],$$

$$[(a, b)] \cdot [(c, d)] = [(ac, bd)].$$

Example. $R = \mathbb{F}[x]$. $\tilde{R} = \left\{ \frac{p(x)}{q(x)} : q(x) \neq 0 \right\}$
↓
field of rational functions.

Lecture 2 (03-08-2021)

03 August 2021 17:33

Recall : polynomial ring $R[x]$. Assume that R is an integral domain.
Given $f(x) = \sum a_i x^i$, define

$$\deg(f(x)) := \max \{i : a_i \neq 0\}$$

for $f(x) \neq 0$.

or $f(x) \in R[x]$ and $\deg(f(x)) = d \Rightarrow f(x) = f_0 + \dots + f_d x^d$
with $f_d \neq 0$.

f_d is called the leading coefficient of f . $LC(f)$

$f_d x^d$ is called the leading monomial of f . $LM(f)$

(leading coefficient, leading monomial)

$$f(x) g(x) = h(x). \quad f(x) = f_0 + \dots + f_d x^d$$

$$g(x) = g_0 + \dots + g_t x^t$$

$$h(x) = h_0 + \dots + h_{d+t} x^{d+t}, \text{ where}$$

$$h_j = \sum_{i=0}^j f_i \cdot g_{j-i} \quad \text{for } 0 \leq j \leq d+t.$$

Moreover since R is an I.D., we have

$$LC(h) = LC(f) \cdot LC(g).$$

Lemma. (Inheritance)

$R \rightarrow$ ring, $R[x] \rightarrow$ polynomial ring.

Then,

(i) R is commutative $\Leftrightarrow R[x]$ is commutative.

(ii) R is an int. dom. $\Leftrightarrow R[x]$ is an int. dom.

(iii) R is a UFD $\Leftrightarrow R[x]$ is a UFD.

Cor. $\mathbb{F}[x]$ is a UFD if \mathbb{F} is a field.

(Any field is a UFD.)

Thus, $\mathbb{Q}[x]$, $\mathbb{R}[x]$, $\mathbb{C}[x]$, $\mathbb{F}_{p^r}[x]$ are all UFDs.

(Even $\mathbb{Z}[x]$ is, as per the lemma.)

Thus, it makes sense to talk about factorizations in these rings.

$R \rightarrow \text{UFD}$

Then, $R[x_1, \dots, x_n]$ is a UFD.

$R[x_1, \dots, x_{n-1}][x_n] \cong \dots$ induct.

$$f(x_1, \dots, x_n) = \sum_{\substack{\text{finite} \\ \text{sum}}} f_{i_1, \dots, i_n} x_1^{i_1} \cdots x_n^{i_n}$$

$$\text{total deg} (\prod_j x_j^{i_j}) = \sum_j i_j.$$

$$\deg(f) = \max \left\{ \text{total deg} (\prod_j x_j^{i_j}) : f_{i_1, \dots, i_n} \neq 0 \right\}.$$

Division of univariate polynomials.

F - field.

Lemma 1: $\forall g, h \in F[x]$ with $h \neq 0$, \exists unique $q, r \in F[x]$ s.t.
(i) $g = h \cdot q + r$
(ii) $r = 0$ or $\deg(r) < \deg(h)$.

Prof.

Uniqueness Assume q_1, q_2, r_1, r_2 s.t.

$$g = hq_1 + r_1 = hq_2 + r_2.$$

$$\text{Then, } h \cdot (q_1 - q_2) = r_2 - r_1.$$

If $q_1 \neq q_2$, then $r_2 = r_1$ and

$$\deg(r_2 - r_1) < \deg(h) \leq \deg(h \cdot (q_1 - q_2)).$$

Thus, $q_1 = q_2$ and so, $r_1 = r_2$ as well.

Existence. By induction on $\deg(g)$. If $\deg(g) < \deg(h)$,

we $q = 0$ and $r = g$.

Else define $g' = g - \frac{LM(g)}{LM(h)} h$

and proceed.

GCD of polynomials.

Defn. Let $h, g \in F[x]$.

f is said to be a **factor/divisor** of g , written $f|g$,
if $\exists q \in F[x]$ s.t. $g = fq$.

Given $h, g \in F[x]$, $p \in F[x]$ is a **GCD** of
 h, g if

(1) p is a common divisor of g and h , i.e.,
 $p | g$ and $p | h$.

(2) \forall common divisors of g and h , we have $u|p$.

Observations:

① GCD is not (necessarily) unique.

Indeed, if p is a GCD of g, h , then

kp is also a GCD $\forall k \in F^*$.

② Existence?

Follows from the fact that $F[x]$ is a UFD

Lemma. Let $g, h \in F[x] \setminus \{0\}$. Consider

$$\mathbb{I}(g, h) = \{ ug + vh : u, v \in F[x]\}.$$

Let p be a non-zero polynomial of lowest degree in $I(g, h)$.

Then,

- (i) p is a common divisor of g and h .
- (ii) p is a GCD of g and h .
- (iii) if q is a GCD(g, h), then $\exists \alpha \in F$ s.t.
$$q = \alpha p.$$

Obs. Lemma \Rightarrow GCD is defined up to a nonzero scalar multiple.

Thus, we can loosely talk about "the GCD".

(Possible convention: assume GCD to be monic, i.e.,
 $LC = 1$)

Proof (of lemma)

① Write $g = pq + r$ and $p = ug + vh$

Then, $r = (\)g + (\)h$.

But $\deg(r) < \deg(p)$. Thus, $r=0$. $\therefore p \mid g$.
if $p \nmid h$

② Suppose $g \mid g$ and $g \mid h$.

Since $p = ug + vh$ and $g \mid \text{RHS}$, we get
 $g \mid \text{LHS}$.

③ If g is a GCD, then $p \mid g$ as well.

$$\begin{aligned} \therefore p \mid g &\ \& g \mid p \\ \Rightarrow g &= \alpha p \text{ for} \\ \text{some } \alpha &\in F. \end{aligned}$$

Lecture 3 (06-08-2021)

06 August 2021

17:31

Agenda for the upcoming lectures:

- ① Fields of size p^k for p prime and $k \in \mathbb{N}$.
 - ② Algorithms for operations on polynomials.
 - Addition/multiplication of poly
 - Division with remainder
 - Multipoint evaluation of polynomials
 - GCD of two polynomials
- Takeaway:
Fast Fourier Transform
by Cooley-Tukey
("one of the most useful algorithms")
-

- \mathbb{F}_{p^k} .

$$k=1 : \mathbb{F}_p = \{0, 1, \dots, p-1\} \quad \text{Arithmetic is modulo } p$$

Theorem: $\forall p$ primes $\forall k \in \mathbb{N}, k \geq 1$, there is a (finite) field with p^k elements.

Moreover, (1) these are essentially unique
(2) these are the only finite fields.
(Not proving this.)

Today: A description of \mathbb{F}_{p^k} .

Irreducible polynomial: $f \in \mathbb{F}[x]$ is irreducible (over \mathbb{F}) if f cannot be written as a product of two nonconstant polynomials (in $\mathbb{F}[x]$).

Example: ① $x^2 + 3x + 2 \rightarrow$ reducible over \mathbb{R}

$$\textcircled{2} \quad x^2 + 1 \xrightarrow{\substack{\text{irred. over } \mathbb{R} \\ (x+i)(x-i) \xrightarrow{\text{red. over } \mathbb{C}}}}$$

Fact. \forall prime p and $d \in \mathbb{N}$, \exists irred. polynomial of deg. equal to d in $\mathbb{F}_p[x]$.

polynomial of deg. $\leq d$ in $\mathbb{F}_p[x] = p^{d+1}$

Roughly $\frac{1}{p^d}$ fraction of these polynomials are irreducible.

Now, let $g \in \mathbb{F}_p[x]$, $\deg(g) = k$, with g irred.

\mathbb{F}_{p^k}

Elements: $\{ f \in \mathbb{F}_p[x] : \deg(f) < k \}$
 $\hookrightarrow_{p^k \text{ many elements}}$

Arithmetic: happen in $\mathbb{F}_p[x]$ and then we go mod g .

Does this satisfy the field axioms?

① Commutativity, assoc., distributivity \rightarrow carry over from $\mathbb{F}_p[x]$.

② 0, 1, existence of additive inverse \rightarrow again easily follows from $\mathbb{F}_p[x]$

③ Non-trivial: existence of multiplicative inverse

\hookrightarrow Proof. Let $f \neq 0$ be with $\deg(f) < k$.

Let $d := \gcd(f, g)$.

Then, $\deg(d) \leq \deg(f) < k = \deg(g)$.

Since g is irred., $d = 1$.

Then, by last class' result, $\exists a, b \in \mathbb{F}_p[x]$ s.t.

$$af + bg = 1.$$

Going modulo g , we have

$$af = 1.$$

Thus, $\bar{a} := \underbrace{a \bmod g}_{\downarrow}$ has $\bar{a}f = 1$
in \mathbb{F}_{p^k} . \square

remainder obtained
upon dividing a by g

This is how we shall imagine fields of size p^k .

Properties.

① \mathbb{F}_{p^k} is a vector space over \mathbb{F}_p .

② $\dim_{\mathbb{F}_p} (\mathbb{F}_{p^k}) = k$.

③ One natural basis = $\{1, x, \dots, x^{k-1}\}$.

④ $\mathbb{F}_p \subseteq \mathbb{F}_{p^k}$

↳ prime subfield of \mathbb{F}_{p^k}

⑤ \mathbb{F}_{p^k} is called a degree k extension of \mathbb{F}_p .

⑥ $\forall \alpha \in \mathbb{F}_{p^k}, p \cdot \underset{\text{ii}}{\alpha} = 0$.
 $\underbrace{\alpha + \dots + \alpha}_{p \text{ times}}$ } same is true in $\mathbb{F}_p[x]$
as well

Thus, \mathbb{F}_{p^k} has characteristic p .

($\mathbb{Q}, \mathbb{R}, \mathbb{C}, \dots$ have char = 0)

_____ x _____ x _____

Arithmetic with polynomials.

$f(x) \in \mathbb{F}[x]$, $\deg = d$.

This defines a natural function

$$\text{Eval}_{f(x)} : \mathbb{F} \rightarrow \mathbb{F}, \\ a \mapsto f(a).$$

That is, if $f(x) = f_0 + f_1 x + \dots + f_d x^d$,

$$\text{then } f(a) = f_0 + f_1 a + \dots + f_d a^d.$$

Q. Does the evaluation of a polynomial tell us anything about the coefficients.

Ans. Given sufficiently many evaluations and hypothesis on degree,
YES.

Special case: $f(x) \in \mathbb{F}[x]$, $\deg \leq d$.

$$f(a) = 0 \text{ for some } a \in \mathbb{F}.$$

What can we say about f ?

Lemma $f(x) = 0 \Rightarrow (x-a) \mid f(x)$.

Proof. Use the division lemma to write

$$f(x) = q(x)(x-a) + r(x)$$

with $r(x) = 0$ or $\deg(r(x)) < 1$.

$\therefore r$ is a constant. Putting $x = a$ above gives $r = 0$.

$$\therefore f(x) = q(x)(x-a).$$

□

S. Let $0 \neq f \in \mathbb{F}[x]$, and $d := \deg(f)$.

Then, # zeroes of f in $\mathbb{F} \leq d$.

Proof. Induct on d . $d=1$ is clear.

$d \geq 2$: If f has no root in \mathbb{F} , then done.

Else, it has a root $a \in \mathbb{F}$ and then, we

can write $f(x) = (x - \alpha) g(x)$.

Now,

$\{ \text{roots of } f(x) \} \subseteq \{ \alpha \} \cup \{ \text{roots of } g(x) \}$. $\# \leq d-1$ by
ind.

A similar inductive argument also gives the following:

If $\alpha_1, \dots, \alpha_n \in \mathbb{F}$ are distinct roots of $f(x)$, then
 $(x - \alpha_1) \dots (x - \alpha_n) \mid f(x)$.

Theorem. (Interpolation)

$\forall \alpha_1, \dots, \alpha_t \in \mathbb{F}$ distinct and

$\forall \beta_1, \dots, \beta_t \in \mathbb{F}$ (not nec. distinct),

$\exists f(x) \in \mathbb{F}[x]$ s.t. $f(\alpha_i) = \beta_i$ for $i=1, \dots, t$, and
 $\deg(f(x)) \leq t-1$.

Moreover, such an f is unique.

Proof is part of HW-1.

Thus, the above gives a way of going from evaluations
to coefficients.



Question to think about:

Q. $f = f_0 + f_1 x + \dots + f_d x^d$ and $\alpha \in \mathbb{F}$.

Design algo to compute $f(\alpha) \in \mathbb{F}$.

(How costly is it?)

Lecture 4 (10-08-2021)

10 August 2021 17:32

Input: $f, g \in \mathbb{F}[x]$, $f = \sum_{i=0}^d f_i x^i$, $g = \sum_{i=0}^d g_i x^i$.

Goal: Output $f \cdot g$, as a list of co-efficients.

Naive algorithm:

$$(fg)_i = \sum_{j=0}^i f_j g_{i-j}. \quad \text{--- (1)}$$

Algo 1: For $i = 0$ to $2d$,

use (1) to compute deg i co-efficient in fg .

Running time: for i^{th} iteration, we have i mult. and $i+1$ additions. $\sim 2i$

$$\therefore \text{Total} = \sum_{i=1}^{2d} (2i) = O(d^2)$$

Input size = $O(d)$

Note: To compute running time, we will just count field operations for "algebraic algorithms".

quadratic complexity

TODAY: Algorithm for polynomial multiplication in time $O(d \log d)$

① Gödel-Tukey - $O(d \log d)$ over \mathbb{C} , some more fields.

② Schonage-Strassen $\rightarrow O(d \log d \log \log d)$ any field.

③ 2019: [Harvey, —] $\rightarrow O(d \log d 2^{\log^* d}) \rightarrow$ (all finite fields?)

④ Last month:
Ben-Sasson, Carmos, $\rightarrow O(d \log d) \rightarrow$ all finite fields

Let us look at ①.

Key idea. Two ways of representing polynomials:

① Coefficients representation \rightarrow use of coeffs
 $f = (f_0, f_1, \dots, f_d)$.

② Evaluation representation \rightarrow evaluation on $d+1$ distinct points.

$$\alpha_1, \dots, \alpha_{d+1} : f = (f(\alpha_1), \dots, f(\alpha_{d+1}))$$

$$g = (g(\alpha_1), \dots, g(\alpha_{d+1})).$$

Obs: The naive algo in coeff rep runs in $\Theta(d^2)$ time.

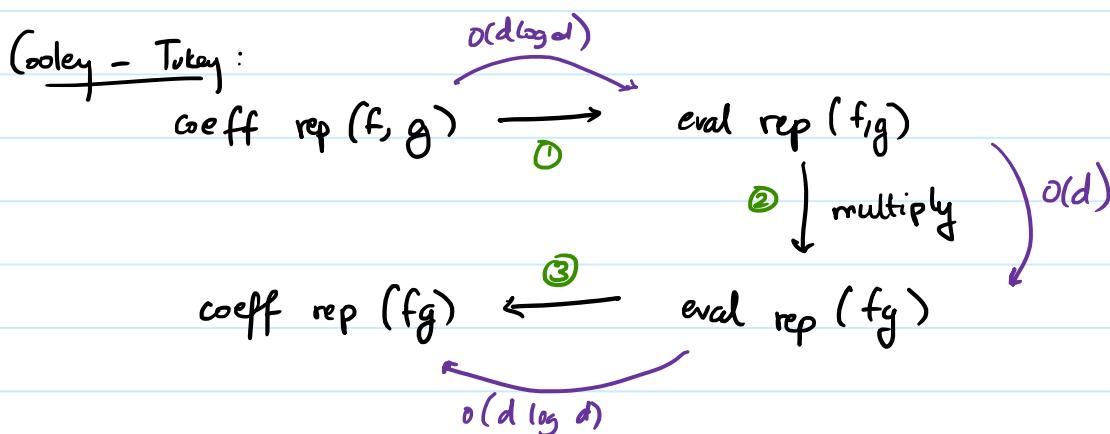
Eval representation?

Eval. repr. of product is simply

$$(fg)(\alpha_1), \dots, (fg)(\alpha_n) = (f(\alpha_1)g(\alpha_1), \dots, f(\alpha_n)g(\alpha_n)).$$

Can do in $\Theta(d)$.

Moral: Multiplication looks easy in eval. repr.



Note: $\deg(fg) = 2d$.

Thus, we need $(2d+1)$ evaluations.

Hence, we must start with $2d+1$ evaluations even though $\deg(f), \deg(g) \leq d$.

Fast Multipoint Evaluation

First, we see how to do \mathcal{O} fast.

$$f = f_0 + f_1 x + \dots + f_d x^d.$$

Want: evaluate f on $\sim 2d$ distinct points fast.

$$\alpha_1, \dots, \alpha_t \rightarrow \text{distinct.} \quad (t = 2d+1)$$

$\underbrace{\text{freedom} \rightarrow \text{choose}}$

$$\left[\begin{array}{c} 1 & \alpha_1 & \dots & \alpha_1^d \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_t & \dots & \alpha_t^d \end{array} \right] \left[\begin{array}{c} f_0 \\ \vdots \\ f_d \end{array} \right] = \left[\begin{array}{c} f(\alpha_1) \\ \vdots \\ f(\alpha_t) \end{array} \right]$$

$\underbrace{\text{even writing down the matrix will take too much time}}_{\text{since } t(d+1) \sim 2d^2 \gg d \log d}.$

CRUCIAL POINT: if $\alpha_1, \dots, \alpha_t$ are chosen carefully, can compute $f(\alpha_1), \dots, f(\alpha_t)$ in $\mathcal{O}(d \log d)$ time.

Choice of $\alpha_1, \dots, \alpha_t$.

[Field plays a role.]

Primitive m^{th} root of 1:

$\omega \in \mathbb{C}$ is said to be a primitive m^{th} root of 1 if

$$\textcircled{1} \quad \omega^m = 1,$$

$$\textcircled{2} \quad \omega^k \neq 1 \quad \forall k \in \{1, \dots, m-1\}.$$

Example: When working over \mathbb{C} , we can take

$\omega = \exp\left(\frac{2\pi i}{m}\right)$ as a primitive m^{th} root of 1.

Properties of primitive m^{th} root:

ω is a primitive m^{th} root of 1

① Then: $1, \omega, \dots, \omega^{m-1}$ are all the m^{th} roots of 1.

② $1 + \omega + \dots + \omega^{m-1} = 0$.

③ If m is even, then ω^2 is a primitive $(m/2)^{\text{th}}$ root of 1.

Pf. ① $(\omega^t)^{m/2} = 1$.

② If $1 \leq k \leq \frac{m}{2} - 1$, then $(\omega^2)^k = \omega^{2k} \neq 1$
since $2k \leq m-2$.

Back to algorithm:

ω = primitive t^{th} root of 1.

Points of evaluation = $\{1, \omega, \dots, \omega^{t-1}\}$.

Assume: t is a power of 2.

That is, we come up with an efficient
way of computing at t points when $t = 2^n$.

We can do this since we can always pick the next
power of 2. Moreover, this power is at most
twice the given number. (That is, $\exists n \in \mathbb{N}$ s.t.
 $n \leq 2^d \leq 2n$)

Recursive algorithm:

Evaluate a deg $t/2$ poly on all
 t^{th} roots of 1

↓ reduction

$2 \times [$ Evaluate a deg $t/4$ poly on all $(t/2)^{\text{th}}$ roots of 1 $]$

+ $O(t)$ processing time.

Running time: $R(t) \leq 2R(t/2) + O(t)$
 $\Rightarrow O(t \log t).$

$$\begin{aligned} f(x) &= f_0 + f_1 x + \dots + f_d x^d \\ &= (f_0 + f_2 x^2 + \dots) + (f_1 x + f_3 x^3 + \dots). \end{aligned}$$

$$\begin{aligned} f_{\text{even}}(y) &:= f_0 + f_2 y + f_4 y^2 + \dots + f_{2i} y^i + \dots \\ f_{\text{odd}}(y) &:= f_1 + f_3 y + f_5 y^2 + \dots + f_{2i+1} y^i + \dots \end{aligned}$$

Note. $\deg(f_{\text{even}}) \leq d/2$, $\deg(f_{\text{odd}}) \leq d/2$.

$$\text{Also, } f(x) = f_{\text{even}}(x^2) + x f_{\text{odd}}(x^2).$$

$$f(\omega^i) = f_{\text{even}}(\omega^{2i}) + \omega^i f_{\text{odd}}(\omega^{2i}).$$

Since ω is a primitive t^{th} root, we know that ω^2 is a primitive $(t/2)^{\text{th}}$ root.

Suffices to have evaluations of $f_{\text{even}}, f_{\text{odd}}$ on $\{(\omega^2)^i : i \in \{0, \dots, t/2\}\}$. $\deg \leq t/2$

Thus, we have done ① in $O(d \log d)$ time.

Now, we need to do ③:

Eval rep ($\deg 2d$ poly, $2d+1$ points) \rightarrow Coefficients.

Linear system:

Earlier

$$\left[\begin{array}{cccc|c} 1 & 1 & \dots & 1 \\ 1 & \omega & \dots & \omega^d \end{array} \right] \left[\begin{array}{c} f_0 \\ \vdots \end{array} \right] = \left[\begin{array}{c} f(1) \\ \vdots \end{array} \right]$$

Known \Rightarrow Unknown \Rightarrow

$$\begin{bmatrix} 1 & \omega & \dots & \omega^d \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{t-1} & \dots & (\omega^{t-1})^d \end{bmatrix} \begin{bmatrix} \vdots \\ f_d \end{bmatrix} = \begin{bmatrix} \vdots \\ f(\omega^{t-1}) \end{bmatrix}$$

Now: $h = fg$

$$h \rightarrow h(1), h(\omega), \dots, h(\omega^{t-1}).$$

$$\deg(h) \leq t-1.$$

want: coeff of h $h = h_0 + h_1 x + \dots + h_{t-1} x^{t-1}$.

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & \omega & \dots & \omega^{t-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{t-1} & \dots & \omega^{(t-1)(t-1)} \end{bmatrix}_{t \times t} \begin{bmatrix} h_0 \\ \vdots \\ h_t \end{bmatrix}^? = \begin{bmatrix} h(1) \\ \vdots \\ h(\omega^{t-1}) \end{bmatrix}$$

!! known

V_ω is full rank. Then,

$$\text{coeff}(h) = (V_\omega)^{-1} \cdot \text{eval}(h).$$

Define $N_\omega := \frac{1}{t} \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & (\omega^{-1}) & (\omega^{-1})^2 & \dots & (\omega^{-1})^{t-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & (\omega^{-1})^{t-1} & (\omega^{-1})^2 & \dots & (\omega^{-1})^{(t-1)^2} \end{bmatrix}$

Claim. $N_\omega = V_\omega^{-1}$.

Note that by Claim, we are done since we are in the same position as in ①.

($\because \omega^{-1}$ is also a primitive t^{th} root of 1.)

Proof. Simply compute. If $\omega^i \neq 1$, then $\sum_i (\omega^i)^j = 0$.

$$\text{Imply compute } \omega^t \text{ such that } \sum_{i=0}^{t-1} (\omega^t)^i = 0.$$

This algorithm is called the FFT (Fast Fourier Transform).

- Where did we use the field?

- (1) Existence of primitive roots.
- (2) Existence of $\frac{1}{t}$. (t should not be divisible by $\text{char}(\mathbb{F})$)

Next : (1) Division with remainder.

(2) Fast multipoint evaluation for any $\alpha_1, \dots, \alpha_t$.

(3) (May be) multipoint evaluation for multivariate polynomials.

Lecture 5 (13-08-2021)

13 August 2021

17:35

Input: $f, g \in \mathbb{F}[x]$, $d \geq \deg(f) \geq \deg(g)$ with $g \neq 0$.

Output: $q, r \in \mathbb{F}[x]$ s.t. $f = g \cdot q + r$, $\deg(r) < \deg(g)$.

Naive: Long division

— Running time: $O(d^2)$

Today: An algorithm which runs in $O(d \log^2 d)$.

we had seen
 $M(d) = O(d \log(d))$
for "nice" fields

More precisely: $O(M(d) \log(d))$,
where $M(d)$ is time complexity of
multiplying two polynomials in $\mathbb{F}[x]$.

New ideas:

- 1) Reversal of a polynomial
- 2) Newton iteration / Hensel lifting

← very useful

Reversal

$$f = f_0 + f_1 x + \dots + f_d x^d, \quad f_d \neq 0.$$

$$\begin{aligned} \text{Rev}(f) &= f_d + f_{d-1} x + \dots + f_1 x^{d-1} + f_0 x^d \\ &= x^d \cdot f(x^{-1}) \\ &= x^{\frac{d}{\text{deg}(f)}} \cdot f(x^{-1}). \end{aligned}$$

↪ note that not linear, i.e.,
 $\text{Rev}(f+g) = \text{Rev}(f) + \text{Rev}(g)$

MAU NOT hold!

Reversal and division

$$f = g \cdot q + r$$

(Assume $r \neq 0$ for the time being.)

$$\begin{aligned}\Rightarrow \text{Rev}(f) &= x^d f(y_x) \\ &= x^d (g(y_n) q(y_x) + r(y_n)) \\ &= x^d g(y_x) q(y_x) + x^d r(y_n)\end{aligned}$$

$$\textcircled{1} \quad \text{rev}(g) = x^{\deg(s)} g(y_n), \quad \text{rev}(q) = x^{\deg(q)} q(y_n), \\ \text{rev}(r) = x^{\deg(r)} r(y_n).$$

$$\textcircled{2} \quad \deg(g) + \deg(g_1) = \deg(f) = d.$$

$$\therefore x^d g(y_2) g_1(y_2) = \text{res}(g) \text{res}(g_1)$$

$$\textcircled{3} \quad \deg(r) < \deg(g) \leq d.$$

$$\therefore \text{rev}(f) = \text{rev}(g) \text{req}(g) + x^{d - \deg(r)} \text{rev}(r)$$

Note: $d - \deg(r) > d - \deg(q) = \deg(q)$.
 !!
 f

$$\therefore \boxed{\text{rev}(f) = \text{rev}(g) \text{ rev}(g) + x^l \text{ rev}(g).} \quad \text{--- (2)}$$

$$e > \deg(q) \geq \deg(\text{rev}(g)).$$

Ques. Degree of every monomial in $x^l \cdot \text{rev}(r)$ is $\geq l > \deg(q)$.

$$(2) + \underline{\text{Obs.}} \Rightarrow \left[\begin{array}{l} \text{rev}(f) \equiv \text{reg}(g) \text{ rev}(q) \pmod{x^e} \\ \downarrow \text{Known} \end{array} \right] \quad \rightarrow (3)$$

Plan : 1. Use (3) to solve for $\text{rev}(q) \pmod{x^k}$.

However, $\ell > \deg(q) \geq \deg(\text{rearr})).$

True, solving it modulo 2^k is sufficient.

(That is: $\text{rev}(g) = \text{rev}(g) \bmod x^e$.)

the canonical remainder of $\deg < L$

$$\textcircled{2} \quad \text{rev}(f) \rightarrow g \quad (\text{easy})$$

$$\textcircled{3} \quad r := f - gq \quad (\text{easy})$$

Thus, it reduces to solving $\textcircled{3}$.

Note: we don't know ℓ but taking $\ell = 1 + \deg(f) - \deg(g)$ works. (Even if $r=0$)

Want: A fast algorithm for solving $\textcircled{3}$.

Idea: Suppose we have $h(x) \in \mathbb{F}[x]$ such that

$$h(x) \cdot \text{rev}(g) \equiv 1 \pmod{x^\ell}$$

That is, $h(x)$ = inverse of $\text{rev}(g)$ in $\mathbb{F}[x]/\langle x^\ell \rangle$.

Multiply $\textcircled{3}$ with $h(x)$ to get:

$$\text{Then, } \text{rev}(q) \equiv h \cdot \text{rev}(f) \pmod{x^\ell}$$

Reduces to: Inverse computation for $\text{rev}(g)$ in $\mathbb{F}[x]/\langle x^\ell \rangle$.

Existence of such an h :

$$g = g_0 + \dots + g_m x^m \quad \text{with } g_m \neq 0. \\ (m = \deg(g))$$

$$\text{rev}(g) = g_0 x^m + \dots + g_{m-1} x + g_m.$$

Note: $g_m \neq 0$ and thus $x \nmid \text{rev}(g)$.

In turn, $\gcd(\text{rev}(g), x^\ell) = 1$ and so,
such an h exists.

Proof: let $h(x) = h_0 + h_1 x + \dots + h_n x^n$ be an arbitrary poly.

Want: h_0, \dots, h_n s.t.

$$(h_0 + h_1 x + \dots + h_n x^n)(g_m + g_{m-1} x + \dots + g_0 x^m) \\ = 1 \pmod{x^\ell}$$

$\Leftrightarrow h_i \in \mathbb{F}, \text{ coeff of } x^i \text{ in } h \cdot \text{rev}(g)$

$$= \text{coeff of } x^i \text{ in } 1 \\ = \begin{cases} 0 & ; i \neq 0, \\ 1 & ; i = 0. \end{cases}$$

$\Leftrightarrow \begin{cases} \text{① coeff of } x^0 \text{ in } h\text{rev}(g) = 1, \\ \text{② coeff of } x^i \text{ in } h\text{rev}(g) = 0 \quad \forall 1 \leq i \leq l. \end{cases}$

$$\left. \begin{array}{l} h_0 g_m = 1 \\ h_0 g_{m-1} + h_1 g_m = 0 \\ \vdots \\ h_0 g_1 + h_1 g_2 + \dots + h_{l-1} g_m = 0 \end{array} \right\} \text{linear system in } h_i$$

wlog $n = l - 1$.

$$\text{rewrite} \quad \left[\begin{array}{c|c} g_m & 0 \\ g_m & \vdots \\ * & g_m \end{array} \right] \left[\begin{array}{c} h_0 \\ h_1 \\ \vdots \\ h_{l-1} \end{array} \right] = \left[\begin{array}{c} 1 \\ 0 \\ \vdots \\ 0 \end{array} \right]$$

Since $g_m \neq 0$, the left matrix is full-rank.
Thus, unique solⁿ exists.

Solve the linear system to get h .

↪ Still quadratic : /

Want : fast!

Enter: Newton Iteration / Hensel lifting.

Want: $h \cdot \text{rev}(g) \equiv 1 \pmod{x^l}$.

Notation: $b := \text{rev}(g)$

$$= b_0 + \dots + b_m x^m \quad \text{with } b_0 \neq 0. \\ (b_i = g_{m-i} \forall i \in [0, m])$$

Want: $h \in \mathbb{F}[x]$ s.t. $h \cdot b \equiv 1 \pmod{x^l}$.

Induction: At the end of i^{th} iteration, we have:

Induction: At the end of i^{th} iteration, we have:
 $H_i \in \mathbb{F}[x]$ s.t. $H_i \cdot b \equiv 1 \pmod{x^2}$
and $\deg(H_i) < 2^i$.

Base case : 0

Stop when: $2^i > l$, $i = \lceil \log l \rceil$

(If we do this fast, we are done.)

Runtime: $\log l \leq O(\log d)$ iterations.

(As we shall see: constant # of polynomial multiplications
 $M(d)$ ↪ in each iteration.)

Base: $H_0 b = 1 \pmod{x}$

Take $H_0 = b_0^{-1}$.

Induction: Suppose we have H_k with $\deg < 2^k$ s.t.
 $H_k b \equiv 1 \pmod{x^{2^k}}$.

Will find an $A(x)$, $\deg(A) < 2^k$ and set
 $H_{k+1} := H_k + x^{2^k} \cdot A$.

Want: $(H_k + x^{2^k} \cdot A) b \equiv 1 \pmod{x^{2^{k+1}}}$

Sketch: $b = b_0 + b_1 x + \dots + b_m x^m$
 $= (b_0 + \dots + b_{2^k-1} x^{2^k-1})$
 $+ x^{2^k} (b_{2^k} + \dots + b_m x^{m-2^k}).$
 $= B_0 + x^{2^k} B_1$

Want: A s.t.
 $(H_k + x^{2^k} A) (B_0 + x^{2^k} B_1) \equiv 1 \pmod{x^{2^{k+1}}}.$

$$\begin{aligned}
 & (H_k + x^{2^k} A) (B_0 + x^{2^k} B_1) \equiv 1 \pmod{x^{2^{k+1}}} \\
 \Leftrightarrow & H_k B_0 + x^{2^k} (A B_0 + H_k B_1) + x^{2^{k+1}} A B_1 \equiv 1 \pmod{x^{2^{k+1}}} \\
 \Leftrightarrow & H_k B_0 + x^{2^k} (A B_0 + H_k B_1) \equiv 1 \pmod{x^{2^{k+1}}}.
 \end{aligned}$$

Induction Hyp: $H_k b \equiv 1 \pmod{x^{2^k}}$

$$\Leftrightarrow \exists c \in \mathbb{F}[x] \text{ s.t. } H_k B_0 = 1 + c \cdot x^{2^k}$$

$$\begin{aligned}
 & 1 + x^{2^k} (c + A B_0 + H_k B_1) \equiv 1 \pmod{x^{2^{k+1}}} \\
 \Leftrightarrow & (c + A B_0 + H_k B_1) \equiv 0 \pmod{x^{2^k}} \\
 \Leftrightarrow & A B_0 \equiv - (c + H_k B_1) \pmod{x^{2^k}}
 \end{aligned}$$

note $H_k B_0 \equiv 1 \pmod{x^{2^k}}$

$$\Leftrightarrow A \equiv - H_k (c + H_k B_1) \pmod{x^{2^k}}$$

Remains: Getting $c \cdot x^{2^k} c = (H_k B_0 - 1)$.

$$\begin{aligned}
 H_{k+1} := H_k + x^{2^k} A &= H_k - H_k c x^{2^k} - H_k B_1 x^{2^k} \\
 &= H_k - H_k (H_k B_0 - 1) \\
 &\quad - H_k^2 B_1 x^{2^k}. \quad \square
 \end{aligned}$$

Lecture 6 (17-08-2021)

17 August 2021

17:32

Univariate multipoint evaluation.

Input: $f(x) \in \mathbb{F}[x]$, $\deg f = d$. $\alpha_1, \dots, \alpha_n \in \mathbb{F}$.

Output: $f(\alpha_1), \dots, f(\alpha_n)$.

Naive. Iteratively (from $i=1$ to n), evaluate $f(\alpha_i)$.

$$f(x) = f_0 + f_1 x + \dots + f_d x^d.$$

Time complexity : $\mathcal{O}(nd)$.

↳ Calculating $f(x)$.

- ↳ Digression: A#1
- ① Compute $\alpha^0, \alpha^1, \dots, \alpha^d \rightarrow d$ products
 - ② Compute $f_0, \dots, f_d \alpha^d \rightarrow d$ products
 - ③ Sum them up $\rightarrow d+1$ sums

A#2. Horner's Method.

$$f(x) = f_0 + x(f_1 + x(f_2 + x(f_3 + \dots + f_d))).$$

→ (1)

Work with (1) (instead of monomial by monomial).

Here, we use $-d$ products and $-d+1$ summations.

Today:

Thm There is a deterministic algo. for univariate multipoint evaluations, which runs in time

$$\mathcal{O}((n+d) \log^3(nd)).$$

(if \mathbb{F} is "FFT friendly")

(if \mathbb{F} is "FFT friendly")
 $\hookrightarrow O((n+d) \log^3 n)$ and $O(\log \log n)$ otherwise

Open question: No simpler algorithm for multivariate.

Partial results: ① Nearly linear time algo over finite fields,
 if # vars is small.

- does weird bit arithmetic
- not algebraic, in particular
- Kedlaya-Umans (2008)

② # vars $\leq d^{O(1)}$, char(\mathbb{F}) small $d^{O(1)}$
 Nearly linear time algebraic algorithm.

Remainder - Tree based algorithm (Borodin et al. '70s)

IP. $f; \alpha_1, \dots, \alpha_n$.

$$\textcircled{1} \quad g_1(x) := \prod_{i=1}^{\lfloor n/2 \rfloor} (x - \alpha_i),$$

$$g_2(x) := \prod_{i=\lceil n/2 \rceil + 1}^n (x - \alpha_i).$$

$$\textcircled{2} \quad \begin{aligned} f_1 &:= f(x) \mod g_1(x) \\ f_2 &:= f(x) \mod g_2(x). \end{aligned}$$

③ Recursion $\left[\begin{array}{l} \textcircled{1} \text{ Eval } f_1 \text{ on } \alpha_1, \dots, \alpha_{\lfloor n/2 \rfloor}, \\ \textcircled{2} \text{ Eval } f_2 \text{ on } \alpha_{\lceil n/2 \rceil + 1}, \dots, \alpha_n. \end{array} \right]$

Return the values.

END.

Proof (of correctness)

Proof (of correctness)

$$\text{To show: } \textcircled{1} \quad f_1(\alpha_i) = f(\alpha_i) \quad \forall i \in \{1, \dots, \lfloor n/2 \rfloor\}$$

$$\textcircled{2} \quad f_2(\alpha_i) = f(\alpha_i) \quad \forall i \in \{\lfloor n/2 \rfloor + 1, \dots, n\}.$$

We prove \textcircled{1}. By $\lfloor n/2 \rfloor$, we shall mean $\lfloor \frac{n}{2} \rfloor$.

$$g_1 = \prod_{i=1}^{\lfloor n/2 \rfloor} (x - \alpha_i).$$

$$f_1 = f \bmod g_1.$$

$$\Rightarrow \exists q_1 \in \mathbb{F}[x] \text{ s.t.}$$

$$f = q_1 g_1 + f_1.$$

$$\therefore f(x) = q_1 \cdot \prod_{i=1}^{\lfloor n/2 \rfloor} (x - \alpha_i) + f_1.$$

Plug $x = \alpha_i$ for $i \in \{1, \dots, \lfloor n/2 \rfloor\}$ and conclude. \square

Time complexity: $n, d ; m := \max \{n, d\}$

$$T(m, m) \leq 2T\left(\frac{m}{2}, \frac{m}{2}\right) + \Theta\left(\begin{smallmatrix} \text{steps} & 1 \times 2 \\ \text{recursion} & \end{smallmatrix}\right)$$

Step 2: Division with remainder (Assume nice field.)
 $\Theta(n \log^2 m)$

$$\underline{\text{Step 1:}} \quad g_1 = (x - \alpha_1) \cdots (x - \alpha_{\lfloor n/2 \rfloor}).$$

Naively: $\lfloor n/2 \rfloor$ multiplications of deg $\leq \lfloor n/2 \rfloor$ each.

$$= \Theta(n^2 \log n) \rightarrow \text{Too long!}$$

(Even worse than original $\Theta(nd)$.)

But we can do multiplication itself recursively.

Comes down to $R(n) \leq 2R(n/2) + c \cdot n/2 \log n$.

Thus, $R(n) = \Theta(n \log^2 n)$.

Plug back in to get

$$T(m, m) \leq 2T(m, m) + \Theta(m \log^2 m).$$

Thus, $T(m, m) \leq \Theta(m \log^3 m)$.

(If it is not FFT friendly, then
we just have an extra $\log \log m$ factor.)

Exercise:

I/P : $b, a_1, \dots, a_n \in \mathbb{N}$.

O/P : $b \bmod a_1, \dots, b \bmod a_n$

Count bit operations

Design a fast algorithm for this.

Some other similar algorithms:

- ① Fast interpolation. \rightarrow maybe in homework
- ② Fast GCD computation. $\stackrel{?}{\rightarrow}$ Project topic
- ③ Fast CRT.

Next up:

- ① Algorithm for GCD, Extended Euclid's Algorithm,
- ② Chinese Remainder Theorem,
- ③ Resultants and Connections to GCD,
- ④ Gauss' Lemma.

Note: many of these 'fast' poly arithmetic algos also work over rings (under some mild conditions).

H/W: Check this + figure out missing details.

GCD computation.

I/P: $f, g \in \mathbb{F}[x], \deg \leq d,$ (Not both zero.)
O/P: $\text{GCD}(f, g).$

Euclid's algorithm:

- 1) If $g = 0$, output $f.$
- 2) Else, $q, r \rightarrow$ quotient, remainder of $f \div g.$
Output $\text{GCD}(g, r).$

Runtime (naively): Analyse two steps at a time.

$$\underset{\leq d}{\text{GCD}}(f, g) \rightarrow \underset{\leq d}{\text{GCD}}(g, r) \rightarrow \underset{\leq d-1}{\text{GCD}}(r, r-1)$$

$$\begin{aligned} T(d) &\leq T(d-1) + O(d \log^2 d) \\ \Rightarrow T(d) &\leq \Theta(d^2 \log^2 d) \end{aligned} \quad (\text{FFT friendly})$$

→ Faster algorithms are known.

Extended GCD

Recall: $\text{GCD}(f, g) = a \cdot f + b \cdot g \quad \text{for some } a, b \in \mathbb{F}[x].$

Q. Want an algo to compute a, b given f, g

Exercise: Extend Euclid's algorithm to solve this question.

(That is called Extended Euclid's Algorithm.)

(That is called Extended Euclid's Algorithm.)
(EEA)

Lemma. Let $f, g \in \mathbb{F}[x]$ be nonconstant.

$h = \text{GCD}(f, g)$. Then, $\exists a, b \in \mathbb{F}[x]$ s.t.

$$\textcircled{1} \quad h = af + bg,$$

\textcircled{2} $\deg(a) < \deg(g)$ and $\deg(b) < \deg(f)$,

\textcircled{3} there is an efficient algorithm to compute
 a, b given f, g .

Proof. Take any a, b s.t. $h = af + bg$.

(Can we EEA to get, for example.)

Apply division lemma to get

$$a = q \cdot g + a^*$$

$\deg(a^*) < \deg(g)$.

$$\Rightarrow h = (qg + a^*)f + bg$$

$$= \underline{\underline{a^*}} f + g(b + qf)$$

satisfies
degree condition

Claim. $\deg(b + qf) < \deg(f)$.

Note that proving this claim finishes the proof of \textcircled{2} \textcircled{3}.

Proof. We have

$$g \cdot (b + qf) = \underbrace{h - a^*f}_{\deg \leq \deg(f) + \deg(g) - 1}$$

$$\text{Thus, } \deg(b + qf) \leq \deg(f) - 1.$$

◻ ◻

Lecture 7 (20-08-2021)

20 August 2021 17:31

Chinese Remainder Theorem

Theorem: $f_1, \dots, f_k \in \mathbb{F}[x]$ are pairwise coprime, i.e., $\forall i \neq j : \gcd(f_i, f_j) = 1$.
Then, $\forall a_1, \dots, a_k \in \mathbb{F}[x] \exists! g \in \mathbb{F}[x]$ satisfying:
(i) $g \equiv a_i \pmod{f_i} \quad \forall i \in \{1, \dots, k\}$, and
(ii) $\deg(g) < \sum_{i=1}^k \deg(f_i)$.

Proof. ① Existence of a solution of 'low' degree.
② Uniqueness

We prove it for $k = 2$. The general case is left as an exercise.

Want to find g s.t.

$$\begin{aligned} g &\equiv a_1 \pmod{f_2}, \\ g &\equiv a_2 \pmod{f_1}. \end{aligned}$$

Since $\gcd(f_1, f_2) = 1$, the EEA gives us u and v s.t.
 $uf_1 + vf_2 = 1$.

Thus, $a_1(uf_1 + vf_2) = a_1$, and
 $a_2(uf_1 + vf_2) = a_2$.

Then, $g = a_1vf_2 + a_2uf_1$ satisfies the modular congruences.

Replace g with $g \pmod{f_1f_2}$.

Still satisfies the congruences and

$$\deg(g) < \deg(f_1) + \deg(f_2)$$

Uniqueness. If g' is another such, then

f_1 and f_2 divide $g - g'$.

~~.....~~ f_1 and f_2 divide $g - g'$.
 $\therefore f_1, f_2 \mid (g - g')$. ($\because \gcd(f_1, f_2) = 1$)

By degree hypothesis, $g = g'$. □

Applications we see in this course:

① Multivariate multipoint evaluation (Kedlaya-Umans)

② Primality testing
 (Agrawal-Biswas)
 (Agrawal-Kayal-Saxena)

Applications seen so far (in disguise):

(In HW) ① Interpolation : $(a_1, b_1), \dots, (a_n, b_n)$, $a_i \neq a_j$.

Wanted: $f(x)$ of deg $< n$ s.t.
 $f(a_i) = b_i$.

Note : $f(a_i) = f(x) \pmod{(x-a_i)}$.

Thus, CRT gives existence and uniqueness of such an f .

(In HW) ② The last question of HW as well.

Resultant

$f, g \in \mathbb{F}[x]$; $\deg(f) = m \geq 1$, $\deg(g) = n \geq 1$.
 $I(f, g) = \{uf + vg : u, v \in \mathbb{F}[x]\}$.
 $\Gamma_{f,g} : \mathbb{F}_{\leq n}[x] \times \mathbb{F}_{\leq m}[x] \rightarrow \mathbb{F}_{\leq m+n}[x]$.

$$\Gamma_{f,g}(u, v) = uf + vg.$$

Observations :

Observations :

① $\Gamma_{f,g}$ is a linear map over \mathbb{F} .

② $\dim_{\mathbb{F}}(\text{domain}) = m+n = \dim_{\mathbb{F}}(\text{codomain}).$

Q. What is the kernel?

When is the kernel trivial, i.e., when is $\Gamma_{f,g}$ invertible?

Fix f, g and write $\Gamma_{f,g}$ as Γ for simplicity.

$$\ker(\Gamma) = \{(u, v) : uf + vg = 0\}$$

Example: $\gcd(f, g) = h, \deg(h) \geq 1.$

→ Kernel is non-trivial.

$\left(\frac{g}{h}, -\frac{f}{h}\right)$ is in the kernel.

↙ this is a valid input
since $\deg(h) \geq 1$.

Thm. $\ker(\Gamma_{f,g})$ is non-trivial $\Leftrightarrow \gcd(f, g) \neq 1.$

Proof. (\Leftarrow) done above

(\Rightarrow) by contrapositive. That is, $\gcd(f, g) = 1 \Rightarrow$ trivial kernel.

Pick $(u, v) \in \ker(\Gamma).$

Then, $uf = -vg.$

$$\therefore f \mid (-v) \xrightarrow{\deg v < \deg f} v = 0.$$

Similarly $u = 0.$

◻

Matrix corresponding to Γ :

Pick the "obvious" bases. $u = \sum_{i=0}^{n-1} u_i x^i,$

pick the obvious bases.

$$u = \sum_{i=0}^m u_i x^i$$

$$v = \sum_{j=0}^{n-1} v_j x^j$$

$$uf + vg = h = \sum_{l=0}^{m+n-1} h_l x^l$$

$$\begin{bmatrix} f_0 & \dots & 0 & g_0 & 0 & \dots & 0 \\ f_1 & f_0 & \dots & 0 & \dots & g_1 & g_0 & \dots & 0 \\ f_2 & f_1 & f_0 & \dots & g_2 & g_1 & g_0 & \dots & 0 \\ f_3 & f_2 & f_1 & f_0 & \dots & g_3 & g_2 & g_1 & \dots & 0 \\ \vdots & \vdots \\ & & & & & & & & & V_{n-1} \end{bmatrix}_{(mn) \times (mn)} = \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ \vdots \\ V_0 \\ V_1 \\ \vdots \\ V_{n-1} \end{bmatrix} = \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \\ \vdots \\ h_{mn} \end{bmatrix}$$

$$h_0 = f_0 u_0 + g_0 v_0$$

$$h_1 = u_0 f_1 + u_1 f_0 + v_0 g_1 + v_1 g_0$$

Exercise: Write Γ when $\deg(f) = 2$, $\deg(g) = 3$.

This matrix is called the Sylvester matrix $S(f, g)$.

$$\text{Resultant}(f, g) := \det(S(f, g)).$$

Thus, we have:

Thm. $\text{GCD}(f, g) = 1 \iff \text{ker}(\Gamma) = 0 \iff \text{resultant}(f, g) \neq 0$.

Note that the resultant is simply a polynomial in coefficients of f and g !

A useful property:

Lemma. $R(f, g) \in I(f, g)$

Proof. $R = \det \begin{bmatrix} f_0 & g_0 & & \\ f_1 & f_0 & g_1 & g_0 \\ \vdots & \vdots & \vdots & \vdots \\ a_n & a_n & a_n & a_n \end{bmatrix}$

$$\text{Punkt.} \quad R: \det \begin{pmatrix} r_0 & & & \\ f_1 & f_0 & & g_1 & g_0 \\ f_2 & f_1 & \ddots & g_2 & g_1 \\ \vdots & f_2 & \ddots & \vdots & \vdots \\ \ddots & \ddots & & \ddots & \ddots \end{pmatrix}$$

View the above in $F(x) \geq F$.

Apply the following transformation: $m+n-1$

$$R_1 \mapsto R_1 + xR_2 + \dots + x^{m+n} R_{m+n}.$$

$$R = \det \begin{bmatrix} f & xf & \cdots & x^{n-1}f & g & xg & \cdots & x^{m-1}g \end{bmatrix}.$$

Same

Now, expand along first row to get

$$f(x) + x f'(x) + \dots + x^{n-1} f^{(n-1)}(x) + g(x) + x^n g'(x)$$

Lecture 8 (24-08-2021)

24 August 2021

17:32

Recall: $R(f, g) = \det \underbrace{S(f, g)}_{\text{Sylvester matrix}}$

$$R(f, g) \neq 0 \Leftrightarrow \text{GCD}(f, g) = 1.$$

$$R(f, g) = uf + vg \quad \text{for some } u, v \in \mathbb{F}[x].$$

Discriminant

Square free polynomial

$f \in \mathbb{F}[x]$ is square free if there does not exist $g \in \mathbb{F}[x]$ s.t. $\deg(g) \geq 1$ and $g^2 \mid f$.

Q. How do we detect square freeness?

For a quadratic polynomial $p(x) = ax^2 + bx + c$, ($a \neq 0$)
we see that $p(x)$ is square free iff $b^2 - 4ac \neq 0$.

In general, the answer is: discriminant.

Defⁿ. (Formal derivatives)

$$\frac{d}{dx} : \mathbb{F}[x] \rightarrow \mathbb{F}[x],$$

If $i > 0$, we have

$$\frac{d}{dx}(x^i) = i \cdot x^{i-1}.$$

For $i = 0$: $\frac{d}{dx}(1) = 0$.

Extend this linearly. This is called the formal derivative.

Check. $\frac{d}{dx} f$ satisfies the product rule.

Obs. f is squarefree iff $\text{GCD}(f, \frac{d}{dx} f) = 1$



$$R(f, \frac{d}{dx} f) \neq 0.$$

Def. The discriminant of f is defined as

$$D(f) = R(f, \frac{d}{dx} f).$$

Example. $f(x) = ax^2 + bx + c,$

$$\frac{d}{dx} f(x) = 2ax + b.$$

$$R(f, \frac{d}{dx} f) = \det \begin{bmatrix} c & b & 0 \\ b & 2a & b \\ a & 0 & 2a \end{bmatrix}$$

$$= 4a^2c - b(2ab - ab)$$

$$= 4a^2c - ab^2$$

$$= -a(b^2 - 4ac).$$

Proof of Obs:

• f is not square $\Rightarrow \text{GCD}(f, \frac{d}{dx} f) \neq 1.$

Proof. Let $g^2 \mid f$ where $\deg(g) \geq 1$.

Then, $f = g^2 h$ for some $h \in A[\lambda]$.

$$\Rightarrow f' = g^2 h' + 2gg'h.$$

Thus, $g \mid f'$ as well. $\therefore g \mid \text{GCD}(f, \frac{d}{dx} f).$

Conversely, suppose $\text{GCD}(f, \frac{d}{dx} f) = 1$.

Pick an irreducible ^{common} factor p .

$$\begin{aligned} f &= ph \\ \Rightarrow f' &= ph' + p'h \end{aligned}$$

$$\begin{aligned} &\Rightarrow p \mid p' h \quad \text{deg}(p') < \text{deg}(p) \\ &\Rightarrow p \mid h \quad \text{and } p \text{ is irreducible} \\ &\Rightarrow p^2 \mid f. \quad (\text{assuming } p' \neq 0) \end{aligned}$$

∴

Vanishing of $\frac{d}{dx} f$.

$$\text{Let } F = F_{p^k}. \text{ Then, } \frac{d}{dx} f(x) = 0$$

$$f(x) = f_0 + f_p x^p + \cdots + f_{kp} x^{kp}.$$

In such a case, define

$$g(x) = (f_0)^{1/p} + (f_p)^{1/p} x + \cdots + (f_{kp})^{1/p} x^k.$$

To show : (i) $g \in F_{p^k}[x]$, i.e., all those p^k roots exist,
(ii) $g^p = f$.
this follows since $\text{char} = p$.

Lemma. Let $F = F_{p^k}$. Then,

$\forall \alpha \in F \exists \beta \in F \text{ s.t. } \beta^p = \alpha.$
(Every element has a p^{th} root.)

Proof.

Define

$$\psi: F \rightarrow F \quad \text{by} \\ \psi(\alpha) = \alpha^p$$

Note that $\psi(a+b) = \psi(a) + \psi(b)$.

In fact, ψ is \mathbb{F}_p -linear since $a^p = a \forall a \in \mathbb{F}_p$.

Also, $\psi(a) = 0 \Rightarrow a^p = 0 \Rightarrow a = 0$.

Thus, $\ker \psi = 0$. Since \mathbb{F} is a f.d.v.s. over \mathbb{F}_p , it follows that ψ is onto. \square

Gauss' Lemma

Let $f(x) \in \mathbb{Z}[x] \subseteq \mathbb{Q}[x]$.

$f(x)$ has a nontrivial factor over \mathbb{Z}
iff

$f(x)$ has a nontrivial factor over \mathbb{Q} .

(Moreover, given a factorisation over \mathbb{Q} efficiently gives one over \mathbb{Z} .)

More generally: We can replace \mathbb{Z} with a UFD R and \mathbb{Q} with $\text{Frac}(R)$.

Examples: ① $\mathbb{Z} \rightarrow \mathbb{Q}$

② $\mathbb{F}[x,y] \rightarrow \mathbb{F}(x)[y]$

Proof. (↓) is trivial.

(↑) Given: $f = gh$ with $\deg g, h \geq 1$, $g, h \in \mathbb{Q}(x)$.

Want: $f = \tilde{g}\tilde{h}$ with $\deg \tilde{g}, \tilde{h} \geq 1$, $\tilde{g}, \tilde{h} \in \mathbb{Z}(x)$.

Lecture 9 (27-08-2021)

27 August 2021 17:31

Multivariate midpoint evaluation (MME)

I/P: $f(x_1, \dots, x_m) \in F[x_1, \dots, x_m]$, $\deg_{x_i}(f) < d$
 $a_1, \dots, a_n \in F^m$

O/P: $f(a_1), \dots, f(a_n)$

Seen so far: $m=1 \rightarrow \Theta(d \cdot \text{poly}(\log(d)))$ Trivial: $\Theta(d^2)$
↳ quadratic

Input vars: d var, d points
~ $2d$ field inputs

Solved in '71.

$m=2$. "Open" until recently.
3
2008 Trivial: $\Theta(d^2 \cdot N)$

Thm. [Kedlaya-Umans '08]

There is an $1 + o(1)$ time algorithm for multivariate multipoint evaluation if
Not algebraic
(input-size)
① $m \leq d^{O(1)}$,
② F - finite field.

Algebraic Another algorithm which works over field of small characteristic.

Open: - Algorithms for infinite fields (#arithmetic operations)
- Algebraic algorithm over all finite fields (\rightarrow)
- Large m

- Large m

Simplifying assumption:

$F = F_p$ for a prime p (Not too tough to extend to if p d.)

Input size: $d^m + Nm$

Desirable runtime: $\Theta((d^m + Nm) \text{ poly}(d, m))$.

Toy (but useful) example:

Set of evaluation points is nice.

$$- N = p^m$$

$$- \text{Set of eval points} = F_p^m$$

$$\underline{m=2}: f(x, y) = \sum_{i=0}^{d-1} \sum_{j=0}^{d-1} f_{ij} x^i y^j \in F[x, y]$$

$$= \sum_{j=0}^{d-1} y^j \left(\sum_{i=0}^{d-1} f_{ij} x^i \right) \in F[x][y]$$

$$= \sum_{j=0}^{d-1} y^j \cdot f_j(x),$$

$$f_j(x) = \sum_{i=0}^{d-1} f_{ij} x^i.$$

- Algorithm:

① Evaluate $f_j(x)$, $j = 0, \dots, d-1$ on F .

② for each $x \in F$, evaluate $f(x, y)$ on F .

$$\sum_{j=1}^{d-1} " f_j(x) y^j "$$

d univariate multi-point eval - deg $d-1$


 d univariate multipoint eval - deg $d-1$
 | points
 p -iter \rightarrow univariate multipoint eval
 of deg $d-1$, on p points.

~~$\tilde{\mathcal{O}}(f(n)p^{\log_2(f(n))})$~~ Runtime: $\tilde{\mathcal{O}}(d(d+p) + p\tilde{\mathcal{O}}(d+p))$
 $= \tilde{\mathcal{O}}((d+p)^2)$
 $\leq \tilde{\mathcal{O}}(d^2 + p^2)$ | $(d+p)^2 \in 2(d^2 + p^2)$

Generalisation to $m > 2$ is clear.

$$f(x_1, \dots, x_m) = \sum_{i=0}^{d-1} x_m^i f_i(x_1, \dots, x_{m-1})$$

Step 1. Eval f_0, \dots, f_{d-1} on \mathbb{F}^{m-1} .

Step 2. $\forall a \in \mathbb{F}^{m-1}$, evaluate $f(a, x_m)$ on \mathbb{F} .

Runtime: $T(m) \leq d \cdot T(m-1) + p^{m-1} \cdot \tilde{\mathcal{O}}(d+p)$

solve \downarrow

$$T(m) \leq \tilde{\mathcal{O}}(d^m + p^m)$$

High level idea: Want to go from evaluating f on arbitrary sets of points in \mathbb{F}_p^m to something like \mathbb{F}_p^m .

Step 1. Go from $\mathbb{F}_p + \mathbb{Z}$. (Think of $\mathbb{F}_p = \{0, 1, \dots, p-1\}$.)

$$\begin{aligned}
 f \in \mathbb{F}_p[x] &\rightsquigarrow \tilde{f} \in \mathbb{Z}[x] \\
 a_1, \dots, a_n \in \mathbb{F}_p^N &\rightsquigarrow \tilde{a}_1, \dots, \tilde{a}_m \in \mathbb{Z}^m \\
 &\text{get fast algo to do} \\
 &\quad \tilde{f}(\tilde{a}_1), \dots, \tilde{f}(\tilde{a}_m)
 \end{aligned}$$

and then reduce mod p .
 (Assume \downarrow this is efficient.)

Step 2. Pick 'many' small primes p_1, \dots, p_t .
 $\in \mathbb{F}_{p_i}[\bar{x}]$

$$\forall i \in [t] : Q_i(\bar{x}) := \tilde{f}(\bar{x}) \pmod{p_i}$$

$$b_{ij} \in \mathbb{F}_{p_i}^m := \tilde{a}_j \pmod{p_i}, \quad j=1, \dots, N.$$

Step 3. $\forall i \in [t]$: solve $MME(Q_i, \mathbb{F}_{p_i}^m)$.

Step 4. $\forall i \in [t]$: read off $Q_i(b_{ij})$.
 $\forall j \in [N]$

Step 5. (Recovery)

$\forall i \in [N]$, recover $f(a_i)$ from
 $\{Q_i(b_{ij}) : i \in [t]\}$.

via CRT

CRT: p_1, \dots, p_t primes

Then, the map

$$\Psi: \mathbb{Z}/p_1 \dots p_t \rightarrow \mathbb{Z}/p_1 \times \dots \times \mathbb{Z}/p_t,$$

$a \pmod{p_1 \dots p_t} \mapsto (a \pmod{p_1}, \dots, a \pmod{p_t})$
is a bijection.

Correctness

Claim: The algorithm is correct if

$$|\tilde{f}(\tilde{a}_j)| \leq \prod_{i \in [t]} p_i.$$

\mathbb{Z}

How many primes do we need?

Well, max out all coeffs and entries to see that
 $|\tilde{f}(\tilde{a}_j)| \leq d^m (p-1)^m \leq d^m p^m$.

Suffices to take t large enough so that the product of the smallest t primes is $> d^m p^{nd}$.

Clearly, we have $p_1 \cdots p_t > 2^t$ for $t > 1$.

Taking

$$t \sim \log_2(d^m p^{nd})$$

$$\sim m \log d + nd \log p$$

does the job.

Note : $p_t \leq O(t \cdot \log t)$

FACT : #primes $\ll n$ is $\sim \frac{n}{\log n}$. Conclude

Running time :

Step 1. $O(N + d^m)$

Step 2. $\text{poly}(t) \xrightarrow{\text{getting the primes}} t \tilde{O}(N + d^m) \xrightarrow{\text{going modulo}}$

Step 3. $t \cdot \tilde{O}(d^m + p_t^m) \quad (p_1 < \dots < p_t)$

Step 4. $t \cdot O(N + d^m)$

Step 5. N. Time (CRT)

Fact! Can do CRT in $\text{poly}(\text{input-size})$

Here, input-size: $t \cdot \log(p_t)$

element of $\prod p_i$ has
 $\leq \log(p_i)$ bits
 $\approx \log(p_t)$

Thus, the time is $\mathcal{O}(n \cdot \text{poly}(t \log p_t))$.

Step 3 is the slowest. Thus,
Running Time $\leq t^5 \log^5(p_t) \tilde{\mathcal{O}}(d^m + p_t^m)$

Lecture 10 (31-08-2021)

31 August 2021 17:21

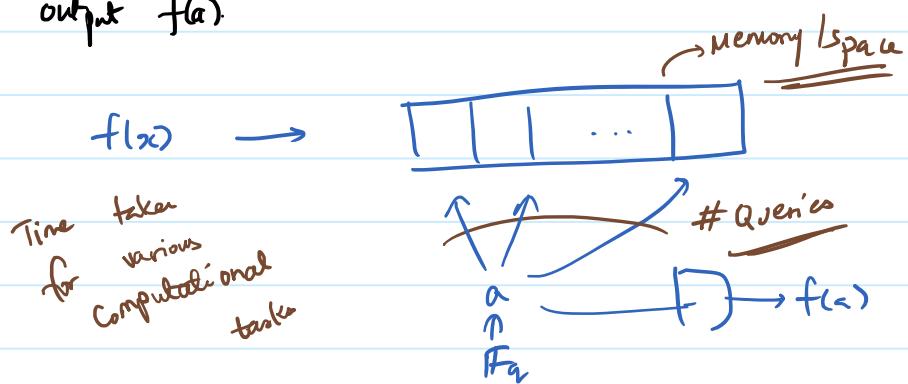
Data Structure for polynomial evaluation

\mathbb{F}_q, n

(think of q as n^3 or so.)

Data. $f(x) \in \mathbb{F}_q[x], \deg \leq n$.

Queries. $a \in \mathbb{F}_q; \text{ output } f(a)$.



Think of memory as a collection of field elements.

(think of each cell as $\log(q)$ bits.)

Want: small space + few queries for every evaluation

Example.

① Storing coefficients.

Memory :

$$f(x) = f_0 + \dots + f_{n-1}x^{n-1}$$

f_0	f_1	\dots	f_{n-1}
-------	-------	---------	-----------

Queries: To find $f(a) \rightarrow$ Read all cells and compute.

cells : n ; Space = $n \log(q)$

queries: n

$n \log(q)$

optimal (why?!)

is this optimal?

(we want output of one field element.)

↳ feels too much. '

② Store evaluations of $f(x)$ on all of \mathbb{F}_q .

Memory : $\boxed{f(x_1) \mid f(x_2) \mid \dots \mid f(x_s)}$

Query : Read the appropriate cell.

Space : q -cells $q \log q$ bits \rightarrow ^{$\gg n \log q$} not so good
Query : 1-cell $\log q$ bits \rightarrow optimal

Q. Can we get the "best of both worlds"?

(*) $O(n)$ cells and $O(1)$ queries?

Thm. (Larsen '12)

If #cells is $O(n)$, then #queries $\geq \Omega(\log n)$.

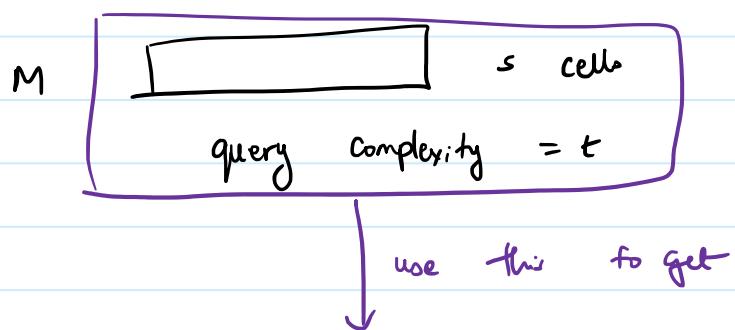
Thus, (*) cannot be achieved.

Thm. (Kedlaya-Umans '08)

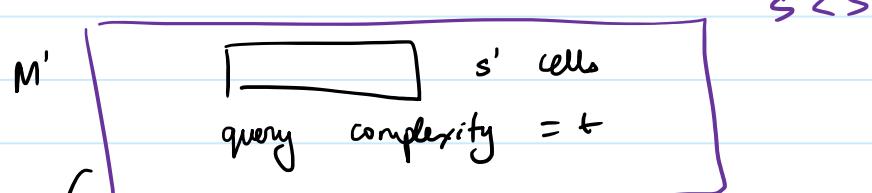
There exists a data structure which needs $O(n^{1+o(1)})$ space and $O(\text{poly}(\log n, \log q))$ queries.

Prof (Larsen)

Suppose we have a structure with



use this to get



query complexity = t

Catch: Not all answers are correct.

BUT evaluates $f(a)$ correctly for at least n distinct $a \in \mathbb{F}_q$

Claim: Can recover f correctly from M' .

Thus, in particular, $s' > n$. (We'll also show $s \gg s'$ and conclude.)

More concretely:

① Going from M to M' :

$p \in [0, 1]$ — parameter

For every cell in M , delete the content with probability $1-p$. (Independently across cells.)

- Expected # non-empty cells in M' = $p \cdot s$.

- Expected # queries that M' answers correctly?

$$\geq p^t q$$

(Suppose $a \in \mathbb{F}_q$ has i_1, \dots, i_t in M which are required.

Then, it is answered correctly if all t are saved.)

Suppose we have an M' which has the above properties.

If $p^t q \geq n$, then the description of M' should be at least $n \log q$ bits long.

Description of M' : cell numbers + content for those

$\log s$ bits $\log q$ bits

length of desc : $ps \log s + ps \log q$

If $p^t q \geq n$, then $ps \log(qs) \geq n \log(q)$.

$$q = n^3 \left\{ \begin{array}{l} \text{pick } p \geq Y_2/t \\ \text{the } p^t q \geq n \text{ and hence} \\ s \geq \frac{n \cdot \log q}{p \cdot \log(q)} = n \frac{\log(n^3)}{p \log(n^3)} \\ \geq \Omega(n \cdot n^{2/t}) \end{array} \right.$$

Can replace n^3 with n^c and get similarly.

But we hadn't actually shown that such an M' exists.
Want to show: Existence of M' st. #cells = ps
correctly answered queries $\sim p^t q$.

Will show: If we pick a random subset of cells in M of size ps , then with prob > 0 , it answers $p^t q$ queries correctly.

Random process: Pick $U \subseteq [s]$ of size ps uniformly at random from all subsets of $[s]$ of size ps .

For $a \in \mathbb{F}_q$, define the w

$$Y_a = \begin{cases} 1 & \text{if } f(a) \text{ can be answered correctly on } U, \\ 0 & \text{else.} \end{cases}$$

$$\text{Define } Y = \sum_{a \in \mathbb{F}_q} Y_a.$$

Will show : $E[Y] \geq p^t q (1 - \text{tiny}).$
 $\Rightarrow P[Y \geq p^t q (1 - \text{tiny})] > 0.$

$$E[Y] = \sum_{a \in \mathbb{F}_q} E[Y_a]$$

$$= \sum_{a \in \mathbb{F}_q} P[Y_a = 1]$$

to answer $f(a)$, we queried in ... it in ..

$$= \sum_{a \in \mathbb{F}_q} \Pr[Y_a = 1]$$

queried in \dots ^{at}
 M.

$$= \sum_{a \in \mathbb{F}_q} \frac{\binom{s-t}{ps-t}}{\binom{s}{ps}}$$

$$= \sum_{a \in \mathbb{F}_q} \frac{(s-t)!}{(ps-t)!(s-ps)!} \cdot \frac{(ps)! (s-ps)!}{s!}$$

$$= \sum_{a \in \mathbb{F}_q} \frac{(s-t)!}{s!} \frac{(ps)!}{(ps-t)!}$$

$$= \sum_{a \in \mathbb{F}_q} \frac{(ps)^t}{s^t} \cdot \frac{(1 - \gamma_{ps})(1 - 2\gamma_{ps}) \dots (1 - \frac{t-1}{ps})}{(1 - \gamma_s)(1 - 2\gamma_s) \dots (1 - \frac{t-1}{s})}$$

$$\geq q \cdot p^t (1 - \text{tiny}) \quad \checkmark \text{Homework}$$

Lecture 11 (03-09-2021)

03 September 2021 17:30

Thm. (Kedlaya Umans)
^(KU)

There exists a data structure for poly eval for deg n polynomials in $\mathbb{F}_q[x]$ with

$$\begin{aligned} \text{Memory : } & n^{1+o(1)} \\ \text{Query complexity : } & \text{poly log}(n, q). \end{aligned}$$

(Think of q as $\text{poly}(n)$.)

Recall: Multipoint Multivariate Evaluation. ^(MME) ($q = \text{prime}$)

I/P : $g(x_1, \dots, x_m) \in \mathbb{F}_q[x], \deg_{x_i}(g) \leq d,$
 $a_1, \dots, a_n \in \mathbb{F}_q^m$

O/P : $g(a_1), \dots, g(a_n)$.

KU: An algo for MME in time $\tilde{\Theta}((n+d^m) \log q)$.

Outline :

$$① \quad g \in \mathbb{F}_q[\bar{x}] \rightarrow \tilde{g} \in \mathbb{Z} \in [\bar{x}]$$

$$a_i \in \mathbb{F}_q^m \rightarrow \tilde{a}_i \in \mathbb{Z}^m$$

② Pick primes $p_1 < p_2 < \dots < p_t$ (smallest)
s.t.

$$\prod p_i > d^m q^{dm}.$$

$$t \sim dm \cdot \log(q) + m \log(d).$$

③ $\forall i \in [t]$

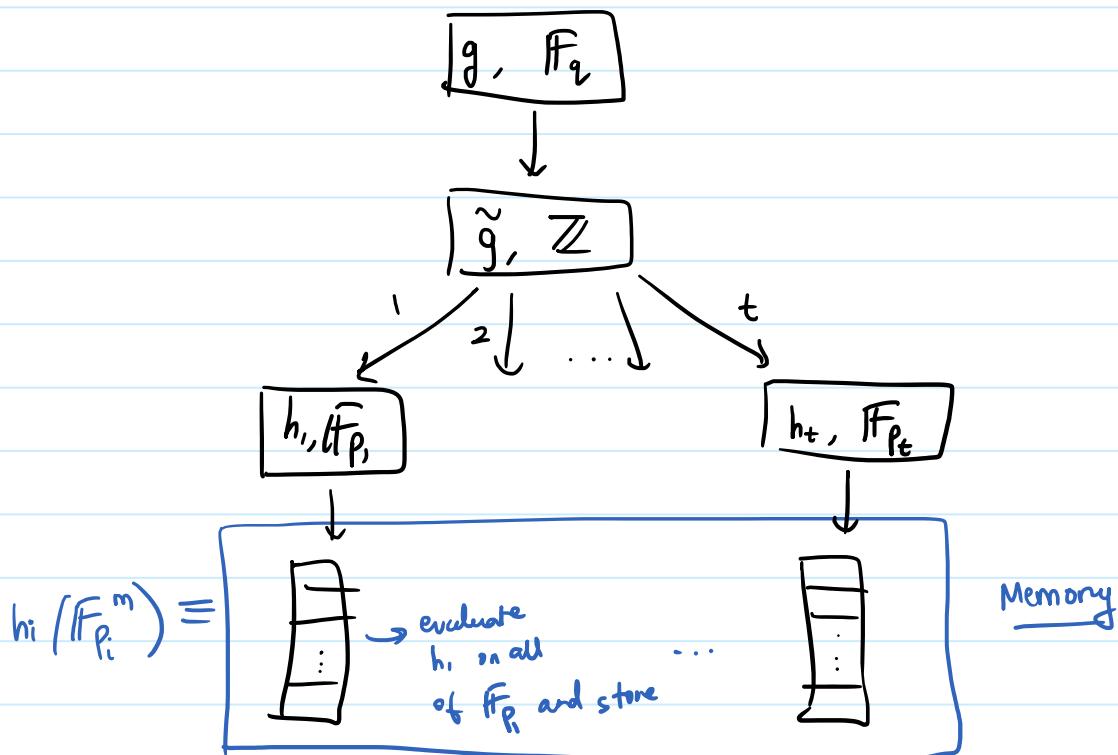
$$\mathbb{F}_{p_i}[x] \ni h_i = \tilde{g}^{\alpha_i} \pmod{p_i}$$

$$\mathbb{F}_{p_i}^m \ni \beta_{ij} = \tilde{x}_j^{\alpha_i} \pmod{p_i}$$

$\left[\text{④ } \forall i \in [t] : \text{ eval } h_i \text{ on } \mathbb{F}_{p_i}^m \right] \rightarrow \text{Can reuse here!}$
 Read off $h_i(\beta_{ij})$.

⑤ Use CRT to recover $g(a_i)$ by solving
 $\tilde{g}(\tilde{a}_i) = h_i(\beta_{ij}) \pmod{p_i}$

Data structure from above:



Query: ① $a_i \in \mathbb{F}_q^m$, $a \rightarrow \tilde{a} \in \mathbb{Z}^m$,

$$\textcircled{2} \quad b_i = \tilde{a} \pmod{p_i}$$

③ Read off $h_i(b_i) \rightarrow$ only step accessing memory!

④ CRT

Resources: For each prime, we look at one cell from $h_i(\mathbb{F}_{p_i}^m)$.

$$\# \text{ cells used} = t \sim \log q$$

$$\begin{aligned} \# \text{ Memory} &\leq t \cdot p_t^m \\ &= (\log q) \cdot (\log q \cdot \log(\log q)) \end{aligned}$$

In principle: Optimal memory: d^m

Query: $\Theta(1)$

+ Data structure for $m=1$:

Aim: $n^{1+\epsilon}$, $\text{poly}(\log n)$

$$\# \text{ cells} \leq n \log q$$

$$\text{Query} \leq \text{poly}(n, \log q)$$

not so good

Univariate to multivariate reduction:

Claim 1. $\forall g(n), \deg(g), \exists R(y_1, \dots, y_{\deg(g)})$ s.t.

$$\textcircled{1} \quad \deg_i(R) \leq 1$$

$$\textcircled{2} \quad R(x, x^{2^1}, x^{2^2}, \dots, x^{2^{\log n}}) = g(x).$$

Proof. Consider x^i for $i \leq n$.

Then write i in binary and conclude. \square

Claim 2 $\forall g(n), \deg(g) + d, m \in \mathbb{N}$ s.t. $d^m > n$

$\exists R_{d,m}(y_1, \dots, y_m)$ s.t.

$$\textcircled{1} \quad \deg_i(R) \leq d-1,$$

$$\textcircled{2} \quad R(x, x^d, \dots, x^{d^{m-1}}) = g(x).$$

Choose deg param = d.
 $m = \frac{\log n}{\log d}$ suffices.

Back to data structure.

$g(x) \rightarrow R_{d,m}(y_1, \dots, y_m) \xrightarrow{K^0} \text{Data structure.}$

Will need to set d appropriately.
 Then set $m = \frac{\log n}{\log d}$.

Set $d = \log^{100} n$.

Analysis:

$$\begin{aligned}
 \# \text{primes} &\leq md \log q = \frac{\log n}{\log d} \log^{100} n \log q \\
 &= \text{poly}(\log n).
 \end{aligned}$$

$$\begin{aligned}
 \text{Memory} &\sim (md \log q)^{m+1} \\
 &\sim \left(\frac{\log n}{100 \log \log n} \cdot \log^{100} n \log n \right)^{\frac{\log n}{100 \log \log n}} \\
 &\sim O\left((\log^{102} n)^{\frac{\log n}{100 \log \log n}}\right) \\
 &\sim O\left(2^{\frac{102}{100} \log \log n \frac{\log n}{\log \log n}}\right) \\
 &= O(n^{1+\epsilon})
 \end{aligned}$$



Lemma. (Schwartz - Zippel - DiMello - Lipton - Ore)

Let \mathbb{F} be any field.

Let $g(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$ be a polynomial of total-degree $\leq d$.

Let $S \subseteq \mathbb{F}$.

If $g \neq 0$, then

$$\# \text{zeros of } g \text{ on } \underbrace{S \times \dots \times S}_{n \text{ times}} \leq d \cdot |S|^{n-1}.$$

Remarks. ① Holds for any \mathbb{F} but interesting when S large.

② Interesting when $|S| > d$.

Take $|S| = 100d$, then

$$\# \text{zeros} \leq \frac{|S|^n}{100}.$$

③ Note that polynomials can have inf. many roots.

④ $n=1$ gives the usual result.

Proof. Induction on n .

$n=1$ is clear. ✓

Assume for $n-1$.

$$g(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_{n-1}][x_n].$$

$$\sum_{i=0}^l g_i(x_1, \dots, x_{n-1}) x_n^i,$$

where $l = \deg_n(g) \leq d$,
and $g_l \neq 0$.

$$\# \text{zeros}_{S^n}(g) = \sum_{(a_1, \dots, a_{n-1}) \in S^{n-1}} \# \text{zeros}_S(g(a_1, \dots, a_{n-1}, x_n))$$

① If $g_l(a_1, \dots, a_{n-1}) \neq 0$, then

$$\#\text{zeroes}_S(g(a_1, \dots, a_{n-1}, x_n)) \leq l.$$

② If $g_l(a_1, \dots, a_{n-1}) = 0$, then

$$\#\text{zeroes}_S(g(a_1, \dots, a_{n-1}, x_n)) \leq |S|.$$

Let $A := \{(a_1, \dots, a_{n-1})^{S^{n-1}} : g_l(a_1, \dots, a_{n-1}) = 0\}$.

$$\begin{aligned}\#\text{zeroes}_{S^n}(g) &\leq l \cdot (|S|^{n-1} - |A|) + |S| \cdot |A| \\ &= l \cdot |S|^{n-1} + (|S| - l) |A|.\end{aligned}$$

$$\text{By induction, } |A| \leq (d-l) |S|^{n-2}.$$

Note that it suffices to show it when $d < |S|$.

$$\begin{aligned}\therefore \#\text{zeroes}_{S^n}(g) &\leq l \cdot |S|^{n-1} + (|S| - l)(d-l) |S|^{n-2} \\ &= d \cdot |S|^{n-1} + (d-l) |S|^{n-1} - l(d-l) |S|^{n-2} \\ &\leq d \cdot |S|^{n-1}.\end{aligned} \quad \square$$

Lecture 12 (07-09-2021)

07 September 2021 17:30

Univariate polynomial factorisation

- Algorithms over finite fields

- Cantor Zassenhaus $\text{poly}(\deg, \log q)$
Randomised

- Berlekamp $\text{poly}(\deg, \log q, P)$
Deterministic

① Open: Deterministic $\text{poly}(\deg, \log |\mathbb{F}|)$ is not known.

② Design an algo (possibly randomised) which runs in time $\tilde{\mathcal{O}}(d \cdot \log(q))$. \approx fastest known $(d \log q)^{1+o(1)}$
(Kedlaya-Umans)

- Algorithms over rationals

- Lenstra-Lenstra-Lovasz (LLL)

Talk about lattices, shortest vectors on lattices

Baby question: Finding square roots in a finite field.

(char $\neq 2$)

Input: $a \in \mathbb{F}_q$.

Output: Does there exist $b \in \mathbb{F}_q$ s.t. $b^2 = a$?
If yes, output such a b .

Consider: $x^2 - a \in \mathbb{F}_q[x]$

Claim: Suffices to factor $x^2 - a$. (To solve \uparrow)

$x^2 - a$ factors iff a is a square.
(The monic linear factors give us the square roots.)

Factoring $x^2 - a$

High level: Both ZZ and Berlekamp
proceed via reduction from

Baby Case : $q \equiv 3 \pmod{4}$ factorisation to GCD.

$$\text{Let } G(x) := \prod_{i=0}^{q-1} (x - i) = x^q - x.$$

Obs 1. If $a \in \mathbb{F}_q^2$, then
 $(x^2 - a) \mid G(a)$.

In this case, $\text{GCD}(x^2 - a, G(x)) = x^2 - a$.

Strategy : Come up with $H(x)$ s.t. $\text{GCD}(x^2 - a, H)$ has $\deg = 1$.

$$G(x) = x(x^{q-1} - 1) \\ = x(x^{\frac{q-1}{2}} - 1)(x^{\frac{q-1}{2}} + 1).$$

$$\text{Take } H(x) = x^{\frac{q-1}{2}} - 1.$$

Now, we will claim that if a is a nonzero square, then $\text{GCD}(x^2 - a, H)$ has $\deg 1$.
Algorithmically, we will be done since the GCD will give a non-trivial factor (and hence, a square root of a).

Moreover this algorithm is efficient :

Step 1. If $a = 0$, output 0.

Step 2. $R(x) = \text{GCD}(x^2 - a, H(x))$. \rightarrow monic GCD

Step 3. If $\deg(R(x)) \neq 1$, then a has no square root.
Else, output the constant term.

\rightarrow Need to do this quick. Our method of $\text{GCD}(x^2 - a, H(x))$ gives $\Theta(q)$ but not good enough.

Suffices to compute $(x^{\frac{q-1}{2}} - 1) \pmod{(x^2 - a)}$ in time

$\text{poly}(\log q)$.
 \hookrightarrow Multipoint?

Solution : Repeated squaring. Compute $x^{\frac{q-1}{2}} \bmod x^2 - a$
via repeated squaring.

$$x \bmod x^2 - a = x, \quad x^2 \bmod x^2 - a = a, \quad x^4 \bmod x^2 - a = a^2$$

$$\dots \quad x^{2^i} \bmod x^2 - a = a^{2^{i-1}} \quad \forall i \geq 1$$

$\hookrightarrow \text{poly}(\log q) \text{ operations now}$

Proof of correctness relies on following claim:

Claim. If $a \in (\mathbb{F}_q^\times)^2$, then $\deg(\text{GCD}(x^2 - a, H(x))) = 1$.

Proof. We have

$$\prod_{b \in \mathbb{F}_q^\times} (x - b) = (x^{\frac{q-1}{2}} - 1)(x^{\frac{q-1}{2}} + 1).$$

$H(x)$

Suffices to show that $H(b) = 0 \Rightarrow H(-b) \neq 0$.

But this is true because $\frac{q-1}{2}$ is odd and $-1 \neq 1$. \square

The above finishes computation of square root
in \mathbb{F}_q with $q \equiv 3 \pmod{4}$.



Square roots over arbitrary \mathbb{F}_q .

High level: Give with a polynomial $H(x)$ s.t

- These ensure that previous algo works
- ① With High Probability (w.h.p.) ($\sim \frac{1}{2}$)
 $\deg(\text{GCD}(H(x), x^2 - a)) = 1$ if $a \in (\mathbb{F}_q^\times)^2$.
 - ② $\text{GCD}(H(x), x^2 - a)$ is computable in $\text{poly}(\log q)$ time.

$$f(x) = x^2 - a, \quad H(x) = x^{\frac{q-1}{2}} - 1,$$

,

roots of $H(x) = (\mathbb{F}_q^\times)^c$.

Define $\tilde{f}_{c,d}(x) := (x - c)^2 - a \cdot d^2$ for $c \in \mathbb{F}_q$ and $d \in \mathbb{F}_q^\times$.

Note: Sufficient to factor $\tilde{f}_{c,d}$ in order to factor f .

Lemma. With probability ≥ 0.48 over $c \sim \mathbb{F}_q$,
 $d \sim \mathbb{F}_q^\times$.

(Choose c and d uniformly & independently from \mathbb{F}_q).
 $\text{GCD}(\tilde{f}_{c,d}, x^{q-1} - 1)$ has $\deg = 1$.

Proof. Sufficient to show

$$\Pr_{c,d \in \mathbb{F}_q} \left(\begin{array}{l} \text{Exactly one of } \{c+db, c-db\} \text{ is} \\ \text{a root of } x^{\frac{q-1}{2}} - 1 \end{array} \right) \geq 0.48.$$

Fix $\alpha, \beta \in \mathbb{F}_q$. The system $\begin{cases} c+db = \alpha \\ c-db = \beta \end{cases}$

has exactly one soln.

Thus,

$$\Pr_{c,d \sim \mathbb{F}_q} [c+db = \alpha \text{ AND } c-db = \beta] = \frac{1}{q^2}.$$

Thus, $c+db$ and $c-db$ are also independent.

Thus,

$$\begin{aligned} \Pr_{c,d \in \mathbb{F}_q} & \left(\begin{array}{l} \text{Exactly one of } \{c+db, c-db\} \text{ is} \\ \text{a root of } x^{\frac{q-1}{2}} - 1 \end{array} \right) \\ &= 2 \cdot \frac{\left(\frac{q-1}{2}\right)\left(\frac{q+1}{2}\right)}{q^2} \end{aligned}$$

$$= \frac{q^2 - 1}{2q^2} = \frac{1}{2} \left(1 - \frac{1}{q^2}\right)$$
$$\geq \frac{1}{2} \left(1 - \frac{1}{25}\right) = \frac{1}{2} \cdot \frac{24}{25}$$
$$= \frac{12}{25} \quad \text{#}$$

Lecture 13 (21-09-2021)

21 September 2021 17:25

Input: $f \in \mathbb{F}_q[x]$, deg d . Given as list of coefficients.

As before, assume q odd.

Output: Irreducible polynomials $g_1, \dots, g_k \in \mathbb{F}_q[x]$
s.t.

$$f = g_1 \cdot \dots \cdot g_k.$$

Suffice to have an algorithm ② for a nontrivial factorisation,
i.e., same input but output is $g \in \mathbb{F}_q[x]$ s.t.
 $g \mid f$ and $1 \leq \deg(g) < \deg(f)$.

Algorithm ② + Division + Recurse finishes original task.

Preprocessing phase:

① Ensure squarefreeness. (That is, $\nexists g \in \mathbb{F}_q[x]$ s.t. $\deg(g) \geq 1$ & $g^2 \mid f$.)

Compute f' .

If $f' \neq 0$, compute $\text{GCD}(f, f')$:

- if $\deg \geq 1$, already done!
- if $\deg = 0$, then we know f is squarefree.

If $f' = 0$, then $f = (\tilde{f})^p$ for some $\tilde{f} \in \mathbb{F}_q[x]$,
where $p := \text{char}(\mathbb{F}_q)$.

Repeat with \tilde{f} .

Can do quickly

From now on, we assume f is squarefree.

② Distinct degree factorisation.

At the end of this step, we want the input to

be a product of distinct irreducible polynomials of the same degree.

At this point, we know

$$f = g_1 \cdots g_k \quad \text{for } \underline{\text{distinct}} \text{ irreducibles } g_i.$$

Toy Example : $x^q - x = \prod_{\alpha \in F_q} (x - \alpha).$

$f \rightarrow \text{square-free}.$

Let $h = \text{GCD}(f, x^q - x).$

Then, $h = \text{product of all irreducible factors of deg 1.}$

$\deg \text{ factors} \leftarrow h, f/h \rightarrow \deg \geq 2 \text{ irreducible factors}$

FACT: Let $k \in \mathbb{N}.$

Then, $x^{q^k} - x = \text{product of all monic irreducible } g \in F_q[x] \text{ s.t. } \deg(g) \mid k.$

for $i = 1 \text{ to } d = \deg(f):$

$$h_i = \text{GCD}(f, x^{q^i} - x)$$

$f \rightarrow f/h_i$

} can do this quickly as in earlier lec

Output $h_1, h_2, \dots, h_d.$

OBSERVE: For all $i \in \{1, \dots, d\}:$

$$h_i = \prod_{\substack{g \in F_q[x] \\ \text{irred} \\ g \mid f}} g.$$

If any of h_i is a proper factor, the done.

Else exactly one is f and thus, f is in desired form. Moreover, we know the degree.

Going forward:

f is a product of distinct irreducibles of known degree.

(In particular, if $\deg(f)$ = this degree, then f is irred.)

Goal: Find a nontrivial factor.

Will use randomness.

For simplicity, we assume

$$f = g_1 \cdot g_2, \quad \deg(f) = 2r.$$

Algorithm 2:

- Pick $A \in \mathbb{F}_q[x]$, $\deg(A) < 2r$, uniformly at random.
- O output $\text{GCD}(f, A^{q^{\frac{r-1}{2}}} - 1)$.

Running time: Fast as before, go by repeated squaring.

Correctness (Probabilistic):

Lemma: If $f = g_1 \cdot g_2$, $\deg(g_1) = r$, $g_1 \neq g_2$ irreducibles.
Then,

$$\Pr_{\substack{A \in \mathbb{F}_q[x] \\ \deg(A) < 2r}} \left(\text{GCD}(f, A^{\frac{q^{r-1}-1}{2}} - 1) \text{ is } \deg r \right) \approx \frac{1}{2}.$$

Equivalent but cleaner phrasing:

Lemma: $\forall g_1, g_2 \in \mathbb{F}[x]$, irred., distinct, $\deg r$.

Then,

$$\Pr_{\substack{A \in \mathbb{F}_q[x] \\ \deg(A) < 2r}} \left[\left(A(x)^{\frac{q^{r-1}-1}{2}} - 1 \right) \begin{array}{l} \text{is divisible} \\ \text{by exactly one of } g_1 \text{ or } g_2 \end{array} \right] \approx \frac{1}{2}.$$

Note: Lemma \Rightarrow Correctness

Proof. Let us compute

$$\underset{A}{\mathbb{P}} \left[g_1 \mid A^{\frac{q^r-1}{2}} - 1 \right].$$

Claim 1. $\underset{A}{\mathbb{P}} \left[g_1 \mid A^{\frac{q^r-1}{2}} - 1 \right] = \gamma_2 - \frac{1}{2q^r}.$

Proof. Note that $g_1 \mid (A^{\frac{q^r-1}{2}} - 1) \Leftrightarrow g_1 \mid (A \bmod g_1)^{\frac{q^r-1}{2}} - 1.$

Consider the field

$$L = \frac{\mathbb{F}_q[x]}{\langle g_1(x) \rangle} \cong \mathbb{F}_{q^r}.$$

Consider $B(y) = y^{\frac{q^r-1}{2}} - 1.$

Obj: # zeroes of $B(y)$ in $L = \frac{q^r-1}{2}$
 ↓ precisely the nonzero squares

Thus, the claim is reduced to:

$$\mathbb{P} \left[(A \bmod g_1) \text{ is a zero of } B(y) \right]$$

If $A \bmod g_1$ is uniformly distributed in L ,
 then the probability above is $\frac{q^r-1}{2q^r} = \frac{1}{2} - \frac{1}{2q^r}.$

And it IS uniformly distributed in L .

(Every such A is of the form $c + hg$
 for unique c, h with $\deg c \leq r-1$.)

Similarly, $\mathbb{P} \left[g_2 \mid A^{\frac{q^r-1}{2}} - 1 \right] = \frac{1}{2} - \frac{1}{2q^r}.$

If divisibility by q_1 and q_2 are indep events, then we are done.

And the independence is due to Chinese Remainder Theorem.

Thus, the desired probability is:

$$\begin{aligned} & 2 \left[\left(\frac{1}{2} - \frac{1}{2q^r} \right) - \left(\frac{1}{2} - \frac{1}{2q^r} \right)^2 \right] \\ &= 2 \left\{ \frac{1}{2} - \frac{1}{2q^r} - \frac{1}{4} + \frac{1}{2q^r} - \frac{1}{4q^{2r}} \right\} \\ &= 2 \left\{ \frac{1}{4} - \frac{1}{4q^{2r}} \right\} \\ &= \frac{1}{2} - \frac{1}{2q^{2r}}. \end{aligned}$$

3

Lecture 14 (24-09-2021)

24 September 2021 17:30

Today: Deterministic algorithm for factoring univariate poly over \mathbb{F}_q .
(Berlekamp)

- Running time : $\text{poly}(\deg, \log q, \text{char})$
(So if we have prime field, not so good.)

Input: $f \in \mathbb{F}_q[x]$, given as a list of coefficients f_0, \dots, f_d .
 $p = \text{char}(\mathbb{F}_q)$

Output: A non-trivial factor of f .

Assume that preprocessing has been done (as in Lec 13).

→ f is square free

→ product of distinct irreducible of some deg.

Two steps :

Algo 1: $\text{poly}(d, q)$

Algo 2: Change above slightly to get $\text{poly}(d, \log q, p)$

Algo 1:

1. Find a poly $g \in \mathbb{F}_q[x]$ s.t.

① $1 \leq \deg(g) < \deg(f)$,

② $f \mid g^q - g$.

This step won't work if f is irr.

2. For every $\alpha \in \mathbb{F}_q$, compute $\text{GCD}(f, g(x) - \alpha)$.

Theorem 1: If $f \in \mathbb{F}_q[x]$ is NOT irreducible, then Algo 1

finds a non-trivial factor.

Moreover, runtime of algo $\leq \text{poly}(d, q)$.

Proof. 1) Assume that step 1 works, i.e., we have found

$g \in \mathbb{F}_q[x]$ satisfying ① and ②.

Want to show that step 2 finds a nontrivial factor of f in time $\text{poly}(d, q)$.

Runtime is easy: q iterations, GCD computation is $\text{poly}(d)$.

Correctness:

$$\text{Note } z^q - z = \prod_{\alpha \in \mathbb{F}_q} (z - \alpha).$$

$$\text{Thus, } g(x)^q - g(x) = \prod_{\alpha \in \mathbb{F}_q} (g(x) - \alpha).$$

$$f \mid g(x)^q - g(x) \Rightarrow f \mid \prod_{\alpha \in \mathbb{F}_q} (g(x) - \alpha).$$

NOTE: Did not assume f is irreduc. for this part. Thus, Step 1 will not work if f is irreduc. $\Rightarrow \exists \alpha \in \mathbb{F}_q$ s.t. $\text{GCD}(f, g(x) - \alpha)$ is nonconstant. Since $\deg(g(x)) < d$, we get that GCD is a non-trivial factor.

2) Step 1 works: That is, if f is reducible, then

- ② a desired g exists, and
- ③ can find one in time $\text{poly}(d, q)$.

Note that condition ② is an \mathbb{F}_q -linear constraint: If

$f \mid g_1^q - g_1$ and $f \mid g_2^q - g_2$, then

$$f \mid (g_1 + \alpha g_2)^q - (g_1 + \alpha g_2).$$

Will try to set up linear system in coefficients of g .

let $g \in \mathbb{F}_q[x]$ be arbitrary with $\deg g \leq d-1$.

$$g(x) = g_0 + g_1 x + \dots + g_{d-1} x^{d-1}.$$

$$\text{Then, } g(x)^q = g_0 + g_1 x^q + \dots + g_{d-1} x^{q(d-1)}.$$

Want: $g_0, \dots, g_d \in \mathbb{F}_q$ s.t.

$$(g^q - g)(x) = \sum_{i=0}^{d-1} g_i (x^{iq} - x^i)$$

is divisible by $f(x)$.

- Compute

$$R_i(x) := (x^{iq} - x^i) \pmod{f(x)} \quad \text{for } i = 0, \dots, d-1.$$

$\hookrightarrow \deg \leq d-1$.

Want $g_0, \dots, g_{d-1} \in \mathbb{F}_q$ s.t.

$$g_0 R_0(x) + \dots + g_{d-1} R_{d-1}(x) = 0.$$



Collect all coefficients of x^i

and equate to 0

- We get a linear system in g_i over \mathbb{F}_q .

- Solve the system to get a solution.



BUT we need a solution which gives a sol" with $g_i \neq 0$ for some $i > 0$.

Runtime of this step = $\text{poly}(d, q)$.

Note: each $\alpha \in \mathbb{F}_q$ satisfies $\alpha^q - \alpha = 0$.

Thus, we have q "trivial" solutions.

Suffices to show that the linear system has $\geq q+1$ distinct solutions.

Now we use that f is reducible.

Write $f = A \cdot B$ s.t. $\gcd(A, B) = 1$ with $A \& B$ nonconst.

Observe: $\forall \alpha, \beta \in \mathbb{F}_q$, if $g(x) \equiv \alpha \pmod{A}$, and

$$g(x) \equiv \beta \pmod{B},$$

$$\text{then } f \mid g(x)^q - g(x).$$

(That is, if g is constant mod $A \& B$, then $f \mid g^q - g$.)

Proof:
$$g^q(x) - g(x) = \alpha^q - \alpha = 0 \pmod{A}.$$

$$\text{By CRT, } g^q - g = 0 \pmod{B}.$$

$$g^q - g = 0 \pmod{AB}.$$

□

Now, for each $(\alpha, \beta) \in \mathbb{F}_q^2$, consider the polynomial

$$g_{\alpha, \beta}(x) \text{ obtained by CRT s.t. } \begin{cases} g_{\alpha, \beta} \equiv \alpha \pmod{A}, \\ g_{\alpha, \beta} \equiv \beta \pmod{B}. \end{cases}$$

Since $q^2 > q$, we have a non-deg=1 sol.

NOTE: The algo to find g DOES NOT use A and B .

The algo just solves the linear system.

The analysis above shows (using A and B) that the linear system WILL have a solution of desired form. □

Algo 2: (Improving runtime to $\text{poly}(d, \log q, p)$).

Step 1: Find $g \in \mathbb{F}_q[x]$ s.t.

$$\deg(g) \leq d-1 \quad \text{and} \quad g^q - g = 0.$$

Step 1: Find $g \in \mathbb{F}_q[x]$ s.t.

$$\textcircled{1} \quad 1 \leq \deg(g) \leq d-1,$$

$$\textcircled{2} \quad f \mid g^p - g.$$

Step 2. For all $\alpha \in \mathbb{F}_p$, compute $\text{GCD}(f, g(x) - \alpha)$.

↳ prime subfield of \mathbb{F}_q

Using the earlier proof, it suffices to show Step 1 works in desired time.

$$g(x) = g_0 + g_1 x + g_2 x^2 + \dots + g_{d-1} x^{d-1}$$

Condition $\textcircled{2}$ is not \mathbb{F}_q linear anymore, only \mathbb{F}_p .

$g_i \in \mathbb{F}_q$. Consider $\mathbb{F}_q = \overline{\mathbb{F}_p[y]}_{(h(y))}$ for h irreducible of $\deg k$.

$$g_i = g_{i,0} + g_{i,1}[y] + \dots + g_{i,k-1}[y]^{k-1}$$

$$g_{i,j} \in \mathbb{F}_p$$

$$g_i^p = g_{i,0} + g_{i,1}^p [y]^p + \dots + g_{i,k-1}^p [y]^{p(k-1)}$$

$$[y] = Y \bmod h.$$

$f \mid g^p - g$ leads to a linear system (over \mathbb{F}_p) in the g_{ij} . Solve this system to get a solution.

Now, need to argue that \exists non-trivial solution.

The same as before! Note that the deg 0 sol's are only when $g(x) = \alpha \in \mathbb{F}_p$.

Thus, $p^{\deg g}$ sol's.

But p^t many solutions exist via CRT as before. \square

But p^2 may have many solutions exist via CRT as before. A

Q. Do the following:

Input: $f(x) \in \mathbb{F}_q[x]$, $\deg(f) = d$.

Output: $\alpha \in \bar{\mathbb{F}}_{q^d}$, for some extension $\bar{\mathbb{F}}_{q^d} \supseteq \mathbb{F}_q$ s.t $f(\alpha) = 0$.

Lecture 15 (28-09-2021)

28 September 2021 17:27

Today and next class: Bivariate factorisation

Input: $f(x, y) \in \mathbb{F}[x, y]$, $\deg f \leq d$, given as a list of monomials.

Goal: Output a factorisation of f into a product of irreducibles.
(over \mathbb{F} or maybe an extension.)

As in the case of univariate, the following suffices:

Output: A nontrivial factor of f .

Will use the following univariate problem as a subroutine:

Input: $h(x) \in \mathbb{F}[x]$

Output: $\alpha \in \mathbb{K}$ s.t. $h(\alpha) = 0$, where \mathbb{K} is an ext' of \mathbb{F} .

Note: If \mathbb{F} is finite, we have seen how to get an irreducible factor of a polynomial. Then we can get an extension where there is a root.

For \mathbb{F} infinite, we haven't seen any such routine but will assume that we have one such.

High level view:

- Preprocess f to put it in a "necessary" form:

① f is not a p^{th} power, where $p = \text{char}(\mathbb{F})$.
(If $p=0$, nothing to do.)

② Make sure that $f(x, y)$ is monic when viewed
as a univariate in $(\mathbb{F}[x])[y]$ (or maybe in

$(F[y])[x]).$

③ $\frac{\partial f}{\partial y} \neq 0.$

[Non-singularity w.r.t. $y.$]

④ $f(x,y)$ is squarefree.

⑤ $f(0, y) \in F[y]$ is squarefree

(Idea is to change f to f^* at each stage s.t. f^* satisfies the conditions and a factorisation of f^* gives one of $f.$)

At the end, we will have a polynomial \tilde{f} with factors of $\tilde{f} \leftrightarrow$ factors of $f.$

— Iterative derivative based algorithm to factor $f,$ assuming that it is preprocessed.
(Will resemble Newton iteration.)

Will use the following form of Gauss' Lemma:

Lemma. $f(x, y) \in F[x][y]$ is monic in y and $g(x, y) \in F(x)[y]$ is monic in y and divides f over $F(x).$

Then, $g(x, y) \in F[x][y]$

Will also use Resultants, GCD algorithms, etc. from earlier classes.

Today: preprocessing

PREPROCESSING.

① Is not a p^e power.

- 1.1. Compute $e \in \mathbb{N}$ s.t. f is a p^e -th power but

NOT a p^{e+1} -th power.

For every monomial $m_{i,j} = x^i y^j$ with a nonzero coefficient in f , define

$$e_{i,j} = \max_e (p^e \text{ divides both } i \text{ and } j).$$

Then, put $e := \min_{m_{i,j}} e_{i,j}$.

f is a p^e -th power but not p^{e+1} -th power
 \Updownarrow Assume F is perfect, i.e.,
 $F = F^p$.

$\forall m_{i,j}$ in f , we have $p^e | i$ & $p^e | j$
AND

$\exists m_{i,j}$ in f s.t. $p^{e+1} \nmid i$ or $p^{e+1} \nmid j$.
For concreteness, we shall assume $\cancel{p^{e+1} \mid j}$

- 1.2. $f \rightarrow f^*$ s.t. ① f^* is not a p^e -th power
② factors of f and f^* are "closely related"

Write $f(x, y) = \sum_{j=0}^d f_j(x) y^j$.

$p^e | j$ s.t. $f_j(x)$ is nonzero.

$$f^*(x, y) := \sum_{j=0}^d f_j(x) y^{j/p^e} \rightarrow \text{NOT a } p^e\text{-th power.}$$

Let us now go from a nontrivial factor of f^* to one of f in an efficient manner:

Algo. Note that $f^*(x, y^{p^e}) = f(x, y)$.

Let $h(x, y)$ be a nontrivial factor of $f(x, y)$.
 $\Rightarrow h^*(x, y^{p^e}) \mid f^*(x, y^{p^e}) = f(x, y)$.

Moreover, $h(x, y) := h^*(x, y^{p^e})$ is a nontrivial factor of $f(x, y)$. \Rightarrow

Now, if f is reducible, we need to ensure f^* is reducible. Let h be an irred factor (non trivial). Then, $h^{p^e} \mid f$.

Write $h(x, y)^{p^e} = h^*(x, y^{p^e})$.

Then, $h^* \mid f^*$ and h^* is nontrivial.

D&E Assume f is not a p^{\pm} power. $\exists j$ s.t. coeff. of y^j in $f^*(x, y)$ is nonzero and $p \nmid j$.
 Now, we process f to make it monic in y and $\frac{\partial f}{\partial y} \neq 0$.

- Want to find a matrix

$$Z = \begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} \in F^{2 \times 2}$$

s.t.

① Z is invertible,

② $f\left(Z \cdot \begin{bmatrix} x \\ y \end{bmatrix}\right)$ is monic in y and $d = \deg(f)$,

③ derivative condition holds for.

$$F = f(z_{11}x + z_{12}y, z_{21}x + z_{22}y) \in F[x, y, \bar{z}]$$

$$= \sum_{i=0}^d y^i \cdot f_i(\bar{z}, x) \quad \text{→ deg in } y \text{ cannot increase}$$

Claim : ① $f_d(\bar{z}, x) \neq 0$,

② $f_1(\bar{z}, x) \in F[x]$,

② $f_j(\bar{z}, x) \neq 0$, where j is s.t. $p+j$ from earlier.

Note: $m_{ij}(x, y) = x^i y^j$

↓

$$(z_{11}x + z_{12}y)^i (z_{21}x + z_{22}y)^j \in \mathbb{F}[\bar{z}, x][y]$$

$$\begin{aligned} \text{deg in } y &= i+j. \\ \text{coefficient of } y^{i+j} &= z_{12}^i z_{22}^j \end{aligned}$$

$$= m_{ij}(z_{12}, z_{22}).$$

Write $f(x, y) = h_0(x, y) + \dots + h_d(x, y)$.

↙ homogeneous components ↘
of f of corresp. degree

Then, coefficient of y^d in F is equal to $h_d(z_{12}, z_{22})$.

x_0

This proves ① and ②.

To prove ③, put $\bar{z} = \bar{z}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ to get

$$f_j(\bar{z}_0, x) = \text{coefficient of } y^j \text{ in } f \neq 0. \quad \text{③}$$

Claim: Can efficiently find $A \in \mathbb{F}^{2 \times 2}$ s.t.

① $\det(A) \neq 0$,

② $f_d(A) \neq 0$,

③ $f_j(A, x) \neq 0$.

Proof: Take $Q(\bar{z}) := \det(\bar{z}) f_d(\bar{z}) f_j(\bar{z}, x)$.

We have shown Q is not the zero poly.

Thus, $\exists A$ s.t. $Q(A) \neq 0$. (Use a polynomial)

identity theorem.)

Assume $|F|$ large enough. P
(Or take extension.)

This fulfills our requirement.

Lecture 16 (01-10-2021)

01 October 2021 17:23

Preprocessing phase:

DID THIS

- ① Ensure $f(x, y)$ is not a p^{th} power.
- ② $f(x, y)$ is monic in y
- ③ $\frac{\partial f}{\partial y}(x, y) \neq 0$
- ④ $f(x, y)$ is squarefree
- ⑤ $f(0, y) \in \mathbb{F}[y]$ is squarefree

x ————— x —————

④

Squarefreeness

$$g^2(x, y) \mid f(x, y) \Rightarrow \text{GCD}\left(f(x, y), \frac{\partial f}{\partial y}(x, y)\right) \text{ is nontrivial}$$

we have ensured that this
is nonzero!

Thus, if f is not squarefree, then we already have a nontrivial factor.

Just need to see if we can compute GCD. We can do it in the usual way since f and $\frac{\partial f}{\partial y}$ are monic in y .

(Possibly get a polynomial in $\mathbb{F}(x)[y]$ but we Gauss to conclude that it is in $\mathbb{F}[x][y]$.)

- ⑤ To ensure: $f(0, y) \in \mathbb{F}[y]$ is square free.

Know so far: $f(x, y)$ — monic in y , $\frac{\partial f}{\partial y} \neq 0$, square free

Recall: $f(x, y) \in (\mathbb{F}[x])[y]$ is squarefree iff
 $\text{Disc}_y(f) = \text{Res}_y\left(f, \frac{\partial f}{\partial y}\right) \neq 0$.

$$f = f_0(x) + y \cdot f_1(x) + \dots + y^d \cdot f_d(x)$$

nonzero
 field constant,
 i.e., $\in \mathbb{F}^\times$

$$\frac{\partial f}{\partial y} = \tilde{f}_0(x) + y \cdot \tilde{f}_1(x) + \dots + y^e \cdot \tilde{f}_e(x)$$

$(\tilde{f}_e \neq 0, e \leq d-1)$

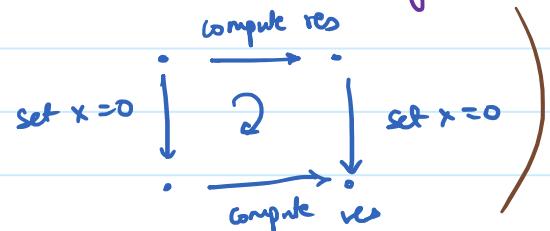
$$\det \begin{bmatrix} f_d & \tilde{f}_e & \tilde{f}_{e-1} & \dots \\ f_{d-1} & f_d & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ f_0 & \vdots & \tilde{f}_0 & \tilde{f}_{e-1} \\ & f_0 & & \dots \end{bmatrix} \in \mathbb{F}[x]$$

$$r(x) := \text{Res}(f, \frac{\partial f}{\partial y}) \in \mathbb{F}[x], \deg r \leq 2d^2$$

If $r(0) \in \mathbb{F}$ is nonzero, then we are done already.

Why?!

The following diagram commutes.



Otherwise, $\exists \alpha \in \mathbb{F}$ s.t. $r(\alpha) \neq 0$. ($\because r(x) \neq 0$)

Now, put $f^*(x, y) = f(x + \alpha, y)$. (Assume \mathbb{F} large enough.)

Everything now works nicely.

X END OF PRE PROCESSING X

Going forward: Assume f has ① - ⑤.

Special case: f has a factor of the form $y - g(x)$ for some $g(x) \in \mathbb{F}[x]$.

Things we know:

① $f(x, y)$ is square free

$$\Rightarrow (y - g(x)) \mid f(x, y)$$

AND

$$(y - g(x))^2 \nmid f(x, y)$$

$$\textcircled{2} \quad y - g(x) \mid f(x, y) \iff f(x, g(x)) = 0.$$

Assume: We have an efficient algorithm for computing roots of univariate over \mathbb{F} ?

Can we use this to get anything about g ?

Note: $f(x, g(x)) \in \mathbb{F}[x]$ of deg $\leq O(d^2)$

Want a $g(x)$ of deg $< d$ s.t.
 $f(x, g(x)) = 0$. \leftarrow o polynomial



$\forall k \in \mathbb{N}, k \leq d^2 : \text{coeff of } x^k \text{ in } f(x, g(x)) \text{ is } 0.$



\textcircled{2} [$\forall k \in \mathbb{N}, k \leq d^2 :$
 $f(x, g(x)) = 0 \pmod{x^k}$

Put $k=1$ in \textcircled{2}:

$$f(x, g(x)) = 0 \pmod{x}$$

$$g = \underline{g_0} + \underline{g_1}x + \dots + \underline{g_r}x^r$$

\downarrow want to find these

$$f(x, g_0 + g_1x + \dots + g_r x^r) = 0 \pmod{x}$$



$\underbrace{\text{going mod } x \text{ really}}_{\text{means setting } x=0}$

$$f(0, g_0) = 0 \quad \text{in } \mathbb{F}$$

↓

g_0 is a root of $f(0, y) \in \mathbb{F}[y]$

'Finding' g_0 : $f(x, y) \xrightarrow{x=0} f(0, y) \in \mathbb{F}[y]$

↓ use efficient root finder

get a list: r_0, \dots, r_d of roots
of $f(0, y)$

One of them is g_0 . (Dunno which one.)

Pseudo Algo:

- ① For each g_0 in $\{r_0, \dots, r_d\}$,
find a g_1 .
- ② Keep building inductively.

Will show: Going from g_0 gives a unique g_1 , so no more branching.

What we have:

- $\alpha \in \mathbb{F}$ s.t. $f(0, \alpha) = 0$.
- $f(0, Y)$ is square free.

Want: $\beta \in \mathbb{F}$ s.t. $f(x, \alpha + \beta x) = 0 \pmod{\langle x^2 \rangle}$.

Lemma: $\forall \alpha \in \mathbb{F} : \left(\begin{array}{l} f(x, \alpha) = 0 \pmod{\langle x \rangle} \text{ AND } \frac{\partial f}{\partial y}(0, \alpha) \neq 0 \\ \Downarrow \\ \exists! \beta \in \mathbb{F} \text{ s.t. } f(x, \alpha + \beta x) = 0 \pmod{\langle x^2 \rangle} \end{array} \right)$

Moreover, β can be efficiently computed.

Lemma*: $\forall k \in \mathbb{N}, \forall \alpha_0, \dots, \alpha_k \in \mathbb{F} :$

$$f(x, \alpha_0 + \dots + \alpha_k x^k) = 0 \pmod{\langle x \rangle^{k+1}}$$

$$\text{AND } \frac{\partial f}{\partial y}(0, \alpha_0) \neq 0$$

And $\frac{\partial f}{\partial y}(0, \alpha_0) \neq 0$

↓

$\exists! \beta \in F$ s.t. $f(x, \alpha_0 + \dots + \alpha_k x^k + \beta x^{k+1}) = 0 \pmod{\langle x \rangle^{k+2}}$.

can be efficiently computed.

Proof of Lemma: Want $\beta \in F$ s.t. $f(x, \alpha + \beta x) = 0 \pmod{x^2}$.

Know: $f(x, \alpha) = 0 \pmod{x}$, $\frac{\partial f}{\partial y}(0, \alpha) \neq 0$.

Taylor:

$$f(x, \alpha + \beta x) = f(x, \alpha) + (\beta x) \frac{\partial f}{\partial y}(x, \alpha) + \frac{(\beta x)^2}{2!} \frac{\partial^2 f}{\partial y^2}(x, \alpha) + \dots$$

Going mod x^2 , our desired condition becomes

$$f(x, \alpha) + \beta x \cdot \frac{\partial f}{\partial y}(x, \alpha) = 0 \pmod{\langle x^2 \rangle}$$

{our hypothesis ensures that $\exists! \beta \in F$ satisfying above}

Exercise: Prove Lemma* and verify that we have found $y_-(x)$.

Next: Arbitrary factors

Do not assume $y - g(x)$ is a factor.

Algo:

① Find α_0 s.t. $f(x, \alpha_0) = 0 \pmod{y}$, i.e., $f(0, \alpha_0) = 0$.

Since $f(0, y)$ is squarefree, this ensures $\frac{\partial}{\partial y} f(0, \alpha_0) \neq 0$.

② For $k = 2d^2 + 1$, use lemma* to get

$$A(x) = a_0 + \dots + a_k x^k \text{ s.t.}$$

$$f(x, A(x)) \equiv 0 \pmod{y^{k+1}}$$

③ Find a polynomial $H(x, y) \neq 0$ s.t.

$$\text{① } \deg_y(H) \leq d-1,$$

$$\text{② } \deg_x(H) \leq d,$$

$$\text{③ } H(x, A(x)) \equiv 0 \pmod{y^{k+1}}$$

If no such H , output "f irred".

(we have
prepared to
make some
 $\deg_y(f) = d$)

④ Output $\text{GCD}_y(f, H)$.

If $\text{GCD} = 1$, output "f irred".

How do we find H ? Note that ③ is a linear condition.

Thus, we get a linear system in coefficients of H .

$$H(x, y) = \sum_{i=0}^d \sum_{j=0}^{d-1} h_{ij} x^i y^j ; \quad h_{ij} \in \mathbb{F}.$$

$$\text{Want } h_{ij} \text{ s.t. } H(x, A(x)) = \sum_i \sum_j h_{ij} x^i (A(x))^j \equiv 0 \pmod{y^{k+1}}$$

System of homog. linear solution in h_{ij} .

Size $O(d^2)$. Solve for a nonzero solution.



If no nonzero sol: output f is irred.

Running time : $\text{poly}(d) + \text{Root finding call.}$

Correctness:

Lemma: Assume f is reducible.

① There is a nonzero sol^{*} to the linear system.

② Any nonzero sol^{*} will give a nontrivial GCD.

Proof: ① Write $f(x, y) = g_1(x, y) \cdots g_t(x, y)$

for $t \geq 2$, and

① g_i — irred.

$$(ii) g_i \neq g_j \quad \forall i \neq j.$$

Obs: $1 \leq \deg_y(g_i) \leq d-1$, since f is monic in y with $\deg_y(f) = d$.

Claim: $\exists i \in \{1, \dots, t\}$ s.t. $g_i(x, A(x)) = 0 \pmod{x^k}$.

Then, can take $H = g_i$.

Proof:

$$(f) \quad g_1(x, A(x)) \cdot \dots \cdot g_t(x, A(x)) = f(x, A(x)) \\ = 0 \pmod{x^{k+1}}.$$

In particular, the product is 0 mod x , i.e.,

$$H(0, A(0)) = g_1(0, A(0)) \cdot g_2(0, A(0)) \cdots g_t(0, A(0)) = 0$$

Since $f(0, y)$ is square free, $\exists! i \in \{1, \dots, t\}$ s.t.

$$g_i(0, A(0)) = 0.$$

Thus, for $j \neq i$, $g_j(x, A(x))$ is NOT divisible by x . Thus, $(*)$ tells us that $g_j(x, A(x)) \equiv 0 \pmod{x^{k+1}}$

② Pick a non-trivial solⁿ H .

Claim 1: $\deg_y(H) \geq 1$.

Proof: Else it is just $H(y, Y) = h(x) \in F[x]$.

Then, $H(x, A(x)) = h(x) \equiv 0 \pmod{x^{k+1}}$.

But $\deg_h h = \deg_x H \leq d \leq k+1$.

$$\Rightarrow h = 0 \text{ itself.} \quad \square$$

View: $f(x, y), H(x, y) \in F[x][y]$.

$$f(x, y) = y^d + f_{d-1}(x)y^{d-1} + \dots + f_1 y + f_0.$$

$$H(x, y) = H_d(x)y^d + \dots + H_1(x)y + H_0(x).$$

$$S_y(f, H) = \begin{bmatrix} & & H_d \\ 1 & & & H_{d-1} \\ f_{d-1} & 1 & & H_{d-2} & H_d \\ f_{d-2} & f_{d-1} & \ddots & \vdots & H_{d-1} \\ \vdots & \vdots & & \vdots & \ddots \end{bmatrix}_{(d+1) \times (d+1)}$$

$$\text{If: } \det S_y(f, H) = \text{Res}_y(f, H) = 0.$$

Note $\text{Res}_y(f, H) \in F[x]$ of $\deg \leq d(d-1)$

$$< 2d^2.$$

Thus, suffices to show it is 0 mod x^{k+1} .

'Recall': $\text{Res}_y(f, h) = f(x, y)B(x, y) + h(x, y)C(x, y)$
 for $B, C \in \mathbb{F}[x, y]$.
 This is in $\mathbb{F}[x]$!

Thus, putting $y = A(x)$ gives

$$\begin{aligned} \text{Res}_y(f, h) &= f(x, A(x))B(x, A(x)) \\ &\quad + h(x, A(x))C(x, A(x)) \\ &\equiv 0 \pmod{x^{2d+1}}. \end{aligned}$$

This finishes the proof. \square

Exercise. Can you adapt this to multivariate. (Assume it is preprocessed.)

X ————— X ————— X

Algebraic / Arithmetic circuits.

Typical computational question:

I/P: $f(x_1, \dots, x_n) \in \mathbb{F}[x_1, \dots, x_n]$

Goal: Do something with f .

Examples: ① a $\in \mathbb{F}^n$, evaluate $f(a)$

② Check if $f(x) = 0$.

(Polynomial identity testing (PIT))

③ Is f irreducible?

④ Output irreducible factors of f

⑤ Given g , output $f+g$, fg .

⑥ Does $f \mid g$?

More interested in: How is f given?

① List of coefficients. $\binom{n+d}{d} \sim 2^{2n}$ Very large.

Ex: $f(x_1, \dots, x_n) = \sum_{S \subseteq [n]} \prod_{i \in S} x_i$.

I/P: $\alpha = (\alpha_1, \dots, \alpha_n) \in \mathbb{F}^n$.

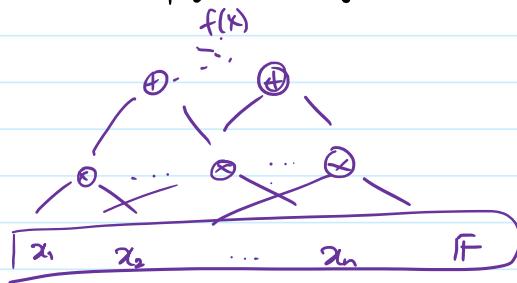
Want: $f(\alpha)$.

Naive way: 2^n operations.

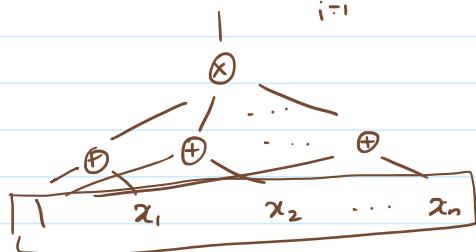
But note: $f(x_1, \dots, x_n) = \prod_{i=1}^n (1 + \alpha_i)$.

Now, $f(\alpha)$ is computable in $O(n)$ time.

In most applications, polynomials given in arithmetic circuits.



For example, $f(\dots) = \prod_{i=1}^n (1+x_i)$



Formal def : Directed acyclic graph. (Directed upwards)

Leaves \rightarrow variables & field constants.

Edges $\rightarrow +$ and \times

Size = # edges / # vertices

Depth = length of the longest path from input to output.



'Recall': P = Decision problem with det. poly time algo.

VP = poly. families with $d = \text{poly}(n)$,
circuits of size $\text{poly}(n)$

Valiant,

after Leslie Valiant

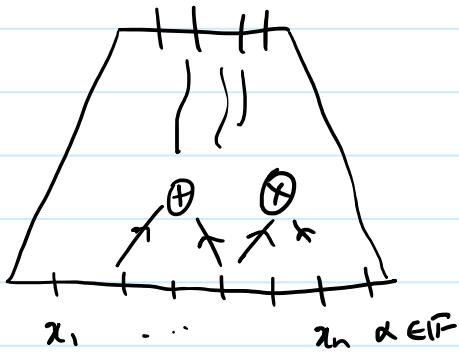
NP = Easy to verify
Analog
VNP

(non uniform) P v/s NP closely related VP v/s VNP
assuming stuff like GRH.

Lecture 18 (08-10-2021)

08 October 2021 17:24

Arithmetic/Algebraic circuits



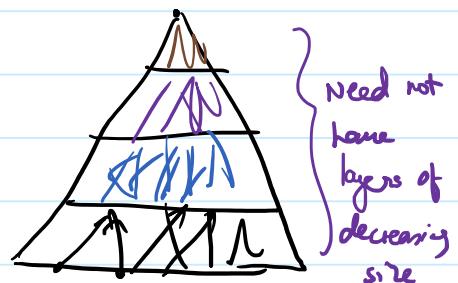
Will assume
fan-in 2, i.e.,
 \oplus and \otimes take
only two inputs

- Size \rightarrow either # wires or # gates
 - Depth vertices
- $n = \# \text{ variables}$
 $d = \text{degree}$
 $s = \text{circuit size}$
 $d = \text{poly}(n)$

Note, we can compute
high degree poly
using low s

$$\begin{array}{c} z^8 \\ z^4 \\ z^2 \\ z \end{array} \quad \left(\begin{array}{c} \oplus \\ \otimes \\ \oplus \end{array} \right) \quad z^{2^8}$$

Can assume circuit is layered:
 Can partition gates into layers
 s.t. only edges from lower
 to upper (nothing with same layer)



Layered circuit \equiv Parallel computation

gates in a layer \equiv # processors

layers \equiv # Rounds of computation

$\rightarrow \text{poly}(n)$
 $\rightarrow \text{poly}(\log(n))$

Considered efficient in circuit theory

NC
 ↳ Nick's class

VP/VNP

VP/VNP

$$\text{Det}_n(x) := \det \begin{bmatrix} & & \\ & & \\ x_{ij} & & \end{bmatrix} \quad n^2 \text{ variables}$$

$$= \sum_{\sigma \in S_n} \text{sign}(\sigma) \prod_{i=1}^n x_{i,\sigma(i)}$$

↓
n! monomials

However, this has a circuit of size $O(n^4)$
(Exponent might be diff.)

Thm. Det_n has a circuit of size $O(n^4)$ and depth $O(\log^2 n)$.

Berkowitz / Ciaky → Chan > 0 or $= 0$.

'94 Schöning → all fields

'98 Mahajan - Vinay → Combinatorial circuit

Fact. Det_n is 'almost' complete for VP.

$P \in \mathbb{F}[x_1, \dots, x_n]$, $\deg(d)$, size $s \leq \text{ckt}$
 Then, \exists matrix M_P of size $(sd)^{\text{poly}(\log(sd))}$ with
 entries being x_1, \dots, x_n or $\alpha \in \mathbb{F}$ s.t.
 $P = \det(M_P)$.

Open problem: Can we remove 'almost'?

Theorem. (Valiant - Skyum - Berkowitz - Rackoff '82)

P - n variate, $\deg d$, size $s \leq \text{ckt}$, $s = d = \text{poly}(n)$.

Then, P has a ckt of size $\text{poly}(sd)$ and depth $O(\log s \cdot \log d)$.

Permanent:

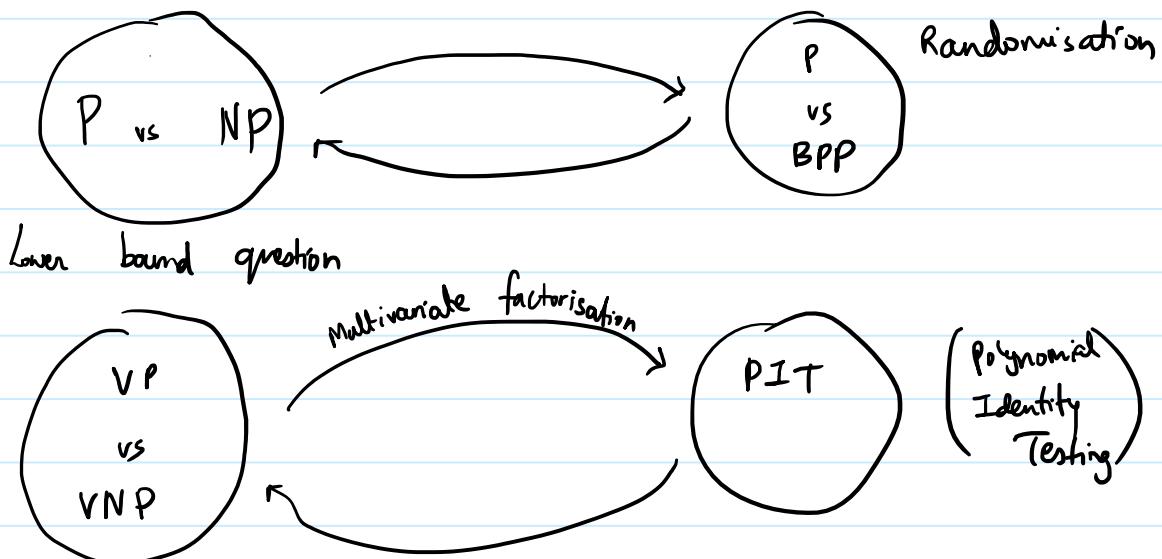
$$\text{Perm}_n = \sum_{\sigma \in S_n} \prod_{i \in [n]} x_{i, \sigma(i)}.$$

$\xrightarrow{\text{perm}} \downarrow$

Valiant (conjecture): Perm_n does not have a ckt of size $\text{poly}(n)$.

Fact: Perm_n is complete for VNP.

$\xrightarrow{\quad}$ $\text{VP} \neq \text{VNP}$ (conjecture)



PIT: I/p: Ckt C, size s, deg d, n-variables
Goal: Is $C \equiv b$?

White box setup : Look inside ckt

Black box : recs s, d, n and queries to C.

- PIT has a simple randomised algorithm.

$S \subseteq \mathbb{F}$, $|S| = \text{load}$, S2-lemma: $\Pr_{a \in S^n} [P(a) = 0] \leq \frac{1}{10^9}$

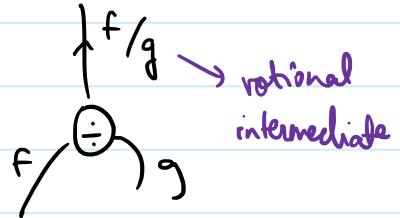
Fact: $\exists c \in \mathbb{N}$ s.t. $\forall s, d, n \in \mathbb{N} \exists H_{s,d,n} \subseteq \mathbb{F}^n$, $|H| \leq (sdn)^c$ s.t.

$\nexists P \in \mathbb{F}[x_1, \dots, x_n]$, deg d, ckt size s,

$\forall P \in \mathbb{F}[x_1, \dots, x_n]$, deg d, ckt size s,
 if $P \neq 0$, then $\exists a \in \mathbb{H}_{s, d, n}$ s.t. $P(a) \neq 0$.

Getting back to circuit:

What if we got a division gate?



Theorem [Strassen '70]

F - large enough.

If $P \in \mathbb{F}[x_1, \dots, x_n]$, deg d, size s ckt with
 $x, \div, +$ gates,
 then P has a ckt of size $(sd)^{10}$ with $x, +$ gates.

Q. P , deg d, size s ckt.

$$P = P_0 + P_1 + \dots + P_d,$$

P_i = homogeneous component of P of deg i.

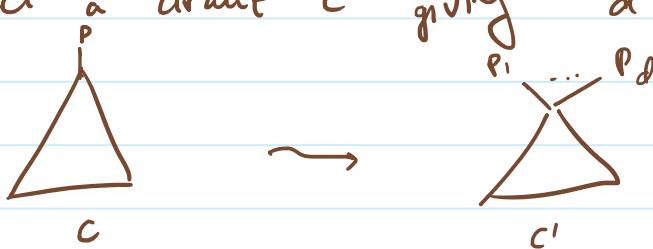
What is the complexity of the homogeneous components?

Thm. (Strassen)

H1, P_i has a ckt of size $\leq O(sd^2)$

Moreover, the ckt can be assumed homogeneous, i.e.,
 the polynomial computed at every gate is homog.

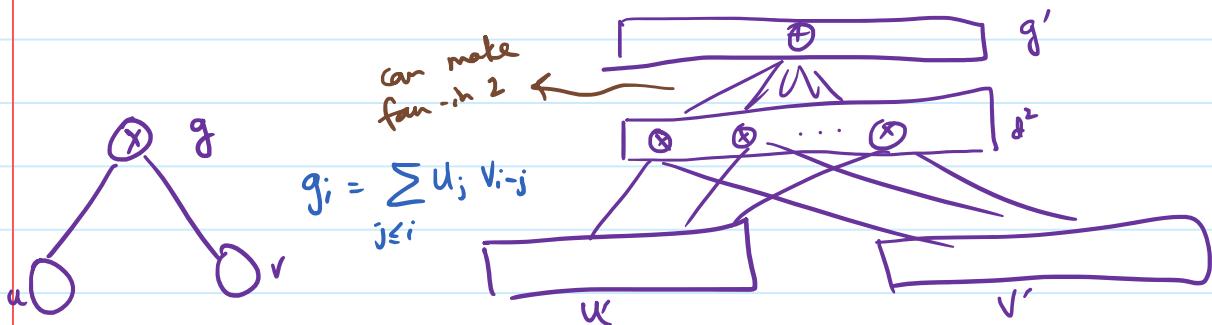
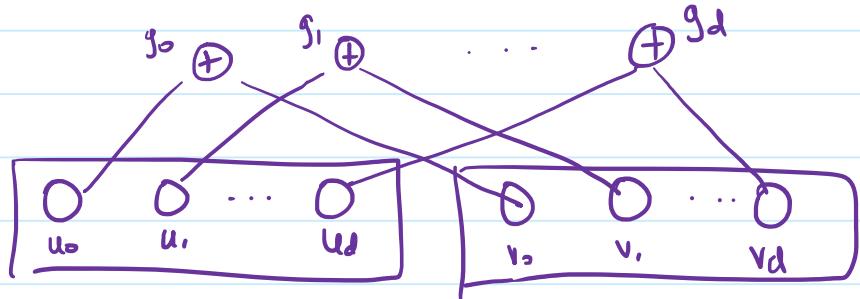
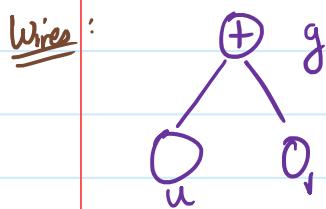
Proof. We construct a circuit C' giving d outputs each P_i .



Gates: For each gate in C , we create $d+1$ gates.



$g_i \rightarrow$ computes degree i component
of g



The above construction should make everything clear. \square

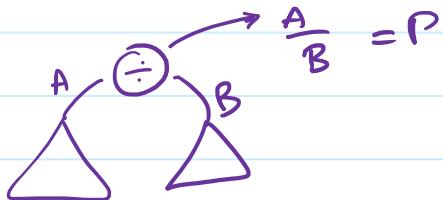
The division reduction is similar. Keep track of numerator and denominator (not even necessarily in lowest form).

But in the final stage, we need to do one division.

Lecture 19 (12-10-2021)

12 October 2021 17:30

from the last time, we can eliminate division by introducing one division gate at last point. At that point, we know that one input divides the other.



① Ensure that the denominator has a non-zero constant term.

Since $B \neq 0$, $\exists a \in \mathbb{F}$ s.t. $B(a) \neq 0$.

Then, can translate and do it.

$$\hookrightarrow P(x+a) \rightarrow P(x)$$

② Use ① for division elimination.

$$P = \frac{A}{1+B^1}, \quad B(0)=0.$$

(Can scale to make $B(0)=1$)

$$\text{Note : } (1-B')(1+B') = 1 - B'^2 \equiv 1 \pmod{\langle x_1, \dots, x_n \rangle^2}$$

$$(1-B' + B'^2)(1+B') = 1 + B'^3,$$

$$(1-B' + \dots + (-1)^k B^k)(1+B') = 1 + (-1)^k B'^{k+1}$$

$$\equiv 1 \pmod{\langle x_1, \dots, x_n \rangle^{k+1}}$$

Note $P(x) \rightarrow$ polynomial of deg d.

$$P(x) \pmod{\langle x_1, \dots, x_n \rangle^{d+1}} = P(x).$$

Thus, computing P modulo $\langle x_1, \dots, x_n \rangle^{d+1}$ is sufficient.

Corollary. Define

$$r_0, r_1, \dots, r_{d-1}, \dots, r_{d+1} = P(x)$$

$$Q = (1 - B' + B'^2 + \dots + (-1)^d B^{d+1}) \cdot A.$$

Homogeneous components of deg $\leq d$ of Q is P .



From this we are done.

Theorem: Detn can be computed by a circuit of size $\text{poly}(n)$ and depth $O(\log^2 n)$.
 (Our proofs will break down over some fields. Does work over fields like $\mathbb{Q}, \mathbb{R}, \mathbb{C}$, etc.)

Def. $X = \begin{pmatrix} & & j \\ & & | \\ i & \cdots & x_{ij} \\ & & \end{pmatrix}_{n \times n}$ \leftarrow formal variables x_{ij}

Eigenvalues of X are root of $\text{Det}(X - \lambda I)$.

$$\text{↑ } \mathbb{F}(\bar{x})[\lambda] \hookrightarrow n^2 \text{ variables}$$

let $\mathbb{K} = \text{Algebraic closure of } \mathbb{F}(\bar{x})$.

$$\text{Det}(X - \lambda I) \in \mathbb{K}[\lambda].$$

$\lambda_1, \dots, \lambda_n \in \mathbb{K}$ eigenvalues of X

$$\begin{aligned} ① \quad \lambda_1 + \dots + \lambda_n &= \text{Tr}(X) \\ &= \sum_i x_{ii}. \end{aligned}$$

$$② \quad \lambda_1 \cdots \lambda_n = \pm \text{Det}(X).$$

$$③ \quad \lambda_1^2 + \dots + \lambda_n^2 = \text{Tr}(X^2).$$

$$④ \quad \dots$$

λ_i^j are e-val of x_i^j with same mult.

① In fact, $\lambda_1^i + \dots + \lambda_n^i = \tau_k(x^i)$. (of x^i , with same mult.)

Digraph on Symmetric Polynomials

② $P_i(y_1, \dots, y_n) = y_1^i + \dots + y_n^i$.

Power symmetric polynomials.

③ $E_i(y_1, \dots, y_n) = \sum_{\substack{s \subseteq [n] \\ |s|=i}} \prod_{j \in s} x_j$.

Elementary symmetric polynomials.

In particular, $E_n(y_1, \dots, y_n) = y_1 \cdots y_n$.

Fact: Both $\{P_i\}_{i=1}^n$ and $\{E_i\}_{i=1}^n$ generate the ring of symmetric polynomials.

Newton - Girard Identities:

$\forall k : E_k = P_1 E_{k-1} - P_2 E_{k-2} + \dots + (-1)^{k-1} P_k$.

(e.g. $n=2, k=2 : 2ab = (a+b)(a+b) - (a^2 + b^2)$)

in matrix form

Char $> n$ $\begin{bmatrix} 1 & & & & \\ -P_1/2 & 1 & & & \\ -P_1/3 & \frac{P_2}{3} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ 0 & 0 & \dots & \dots & 1 \end{bmatrix} \begin{bmatrix} E_1 \\ E_2 \\ E_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} P_1 \\ -P_2/2 \\ P_3/3 \\ \vdots \\ \dots \end{bmatrix}$

$$\text{Char } \gamma_n \leftarrow \begin{array}{c|c} \vdots & \vdots \\ -P_1/k & P_2/k \\ \parallel & \cdots \\ M & \parallel \\ \parallel & I + M' \end{array} \quad \left[\begin{array}{c|c} \vdots & \vdots \\ E_n & \parallel \\ \parallel & P \end{array} \right]$$

$$E = (I + M')^{-1} \cdot P$$

$$= (I - M' + \cdots + (-1)^{n-1} M'^{n-1}) P \quad \therefore I + M' = I - M' + M'^2 + \cdots$$

↑ finite sum

Circuit for $E_n(\lambda_1, \dots, \lambda_n) = \text{Det}_n$:

① Compute x^2, x^3, \dots, x^n .

↳ Compute x^2, x^4, x^8, \dots and then compute each.

$\Theta(\log n)$ depth and $\text{poly}(n)$ size.

② Compute P_1, \dots, P_n of $\lambda_1, \dots, \lambda_n$.

$$P_i = \text{Tr}(x^i)$$

③ Have access to entries of M' .

Compute $M'^2, M'^3, \dots, M'^{n-1}$ as before.

Depth increase by $\Theta(\log n)$.

Fan-in = $D(n)$.

④ Compute $(I - M' + M'^2 + \dots) P$.

⑤ Convert to fan-in 2 \rightarrow depth increases by factor of $\Theta(\log n)$



Parallel Algo for Bipartite Matching:

Given a bipartite graph, does it have a perfect matching?

$G \rightarrow$ graph on $\{1, \dots, n\} \sqcup \{1, \dots, n\}$.

Define

$$G(i,j) = \begin{cases} x_{ij} & \text{if } (i,j) \text{ is an edge} \\ 0 & \text{else} \end{cases}$$

Lemma: $\det(\tilde{G}) \neq 0 \Leftrightarrow G$ has a perfect matching.
} as a polynomial

Pf. (\Leftarrow) Suppose $(1, i_1), \dots, (n, i_n)$ is a perfect matching.

Then, the monomial $x_{1,i_1} \dots x_{n,i_n}$ has a nonzero coeff in $\det(\tilde{G})$.

(\Rightarrow) A nonzero monomial gives a perf. matching. \square

Combine Schwartz-Zippel with the circuit to finish the problem.