Algorithms for Finite Fields

1 Introduction

This course will discuss efficient ways to do computations over finite fields. First, we will search for efficient ways to factor polynomials over finite fields. There is a probabilistic polynomial time algorithm for this, as we'll see. Second, we'll discuss computations involving discrete logarithms. That is, suppose $h \in < g >$, find n such that $h = g^n$. As of the first class day, there is no polynomial time algorithm for this problem. It is possible that there is no such algorithm. Third, we will look at primality testing for integers. There is a deterministic polynomial time algorithm for this problem, and it is essentially a finite field algorithm. It starts with a ring depending of the number to be tested, which is a field if and only if the number is prime.

For every prime number p and positive integer n, there exists a field with p^n elements, and any two such fields are isomorphic. $\mathbb{F}_{p^n}$ denotes "the" finite field with p^n elements. When $n = 1$, we have $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$. One way of describing $\mathbb{F}_{p^n}$ is as the splitting field of $x^{p^n} - x$ in the algebraic closure of $\mathbb{F}_p$. This is unsatisfactory, however, since this does not show how to compute the algebraic closure. A better way is to choose an irreducible $f(x) \in \mathbb{F}_p[x]$ of degree n and think of $\mathbb{F}_{p^n}$ as $\mathbb{F}_p[x]/(f(x))$. This is better since we may describe each element in this field as $a_0 + a_1 x + \ldots + a_{n-1}x^{n-1}$ for $a_i \in \mathbb{F}_p$. But, we have a new problem on our hands: Given p and n, find an irreducible polynomial of degree n in $\mathbb{F}_p[x]$. We would like the fastest possible procedure for this, i.e. one that is polynomial in $n\log\, p$. Indeed, we cannot do better than $n\log p$, since for $f(x) = f_0 + f_1 x + \ldots + f_n x^n$ we would need $\log\, p$ digits just to describe each f_i.

Example 1 (Easy) If $n = 2$ and $p \equiv 3 (\text{mod } 4)$, then $f(x) = x^2 + 1$ is good.

Example 2 (Hard) Same $n = 2$ but with $p \equiv 1 (\text{mod } 4)$.

How likely is a polynomial of degree n picked at random irreducible? Over $\mathbb{F}_p$, the probability is $1/n$. So, our algorithm is simple: pick a random polynomial of degree n, and test whether it is irreducible. If it is, great. If not, pick another.

Now, suppose you are lucky and find two irreducible polynomials $f(x)$ and $g(x)$ of degree n. Then $\mathbb{F}_p[x]/(f(x)) \cong \mathbb{F}_p[x]/(g(x))$. A natural question to ask is, what is the isomorphism? There is a deterministic polynomial time

algorithm for finding this; so now that we know the ending, we'll move along to the next example.

We know that $\mathbb{F}_q^\times = \mathbb{F}_q - \{0\}$ is a cyclic group of order $q-1$, but what is the generator? One algorithm to find it is to pick $g \in \mathbb{F}_q^\times$ at random, and check if $g^{(q-1)/l} \neq 1$ for all primes $l|(q-1)$. It is clear that this is a necessary and sufficient condition for g to generate $\mathbb{F}_q^\times$. This is not hard to do provided we have a factorization of $q-1$. (If we know the factorization, then this is a probabilistic polynomial time algorithm for finding the generator.) The number of generators is $\phi(q-1)$, so the probability that an element chosen at random is a generator is $\phi(q-1)/q-1$. This algorithm is efficient for small q, but for large q, we'd like to have a smaller set from which to search for our generator, as opposed to picking from all of $\mathbb{F}_q^\times$. Under the generalized Riemann Hypothesis, there is a g such that $2 < g < (\log p)^2$ which is a primitive root mod p. It may be $(\log p)^3$, or something close to that, but the point is that it is very specific. So, we have a deterministic algorithm if we pick sequentially, but we must assume the generalized Riemann Hypothesis.

Example 3 Look at $\mathbb{F}_{p^p} = \mathbb{F}_p[x]/(x^p - x - 1)$. $\mathbb{F}_{p^p}^\times$ has $p^p - 1$ elements, and $p^p - 1 = (p-1)\frac{p^p-1}{p-1}$. We have an exact sequence:

$$1 \longrightarrow \mathbb{F}_p^\times \longrightarrow \mathbb{F}_{p^p}^\times \longrightarrow G \longrightarrow 1.$$

Ker $N_{\mathbb{F}_{p^p}/\mathbb{F}_p}$ is a subgroup of $\mathbb{F}_{p^p}^\times$ of order $\frac{p^p-1}{p-1}$ splitting the exact sequence.

Conjecture: The image of x in $\mathbb{F}_p[x]/(x^p-x-1)$ has order $\frac{p^p-1}{p-1}$. (Only been checked for primes less than 20.)

Given a ground field $\mathbb{F}_q$, if we want to build an extension of degree n we need an irreducible polynomial $f(x)$ of degree n so that $\mathbb{F}_{q^n} = \mathbb{F}_q[x]/(f(x))$. We shall now study some algorithms in $\mathbb{F}_q[x]$ with a view towards an irreducibility test.

Division Algorithm

Given polynomials $a(x), f(x) \in \mathbb{F}_q[x]$ $\exists$ polynomials $b(x), r(x) \in \mathbb{F}_q[x]$ such that

1. $a(x) = b(x)f(x) + r(x)$
2. $\deg r(x) < \deg f(x)$

Algorithm

Let $a = a_0x^m + \dots$ and $f = f_0x^n + \dots$.
If $\deg a < \deg f \ \Rightarrow \ b = 0$, $r = a$
Else if $\deg a = m > \deg f = n$
Replace a by $a - \frac{a_0}{f_0}x^{m-n}f$ and b by $b + \frac{a_0}{f_0}x^{m-n}$
Do this until $\deg a < \deg f$.

This algorithm takes at most $m - n$ steps to get b and r. Each step takes $O(n)$ operations in $\mathbb{F}_q$ and the whole process takes $O(mn)$ operations.

Euclidean Algorithm

Given polynomials $a, b \in \mathbb{F}_q[x]$ with $\deg b \leqslant \deg a$ we want to compute $\gcd(a, b)$. We shall let $a\%f$ denote the remainder when a is divided by f.

Algorithm

$\gcd(a, b) = gcd(b, a\%b)$
Do this until $\gcd(a, 0) = a$

Iterating will compute $\gcd(a, b)$ in $O(\max\{\deg a, \deg b\}) = O(\deg a)$ division of polynomials.

Raising to an integral power

Given $a, f \in \mathbb{F}_q[x]$ and $m > 0$ an integer we would like to compute $a^m \mod f$. This can be done in $O(\log m)$ operations in $\mathbb{F}_q[x]/(f(x))$.

Algorithm

Let us look at the binary expansion of m.

$$m = m_0 + m_1 2 + \cdots + m_r 2^r, m_i \in \{0, 1\}$$

To improve efficiency of our algorithm we could use the base p representation of m.
We compute by squaring and then reducing $\mod p$ the previous term of the sequence.

$$\{a, a^2, \dots a^{2^r}\}$$

Then $a^m = a^{m_0}(a^2)^{m_1}\cdots(a^{2^r})^{m_r}$ can be computed by using at most $r+1$ multiplications from terms of the sequence.
Since we are reducing mod f each time the degree of a does not become large.

Irreducibility of Polynomials

Theorem 1. *Let $f(x) \in \mathbb{F}_q[x]$ be a polynomial of degree n. Then $f(x)$ is irreducible iff*

1. $f(x)|(x^{q^n} - x)$
2. $gcd(f(x), x^{q^d} - x) = 1, \forall\ d|n,\ d < n$

Proof. Assume 1 and 2.
$1 \Rightarrow f$ splits completely in $\mathbb{F}_{q^n}$ and has simple roots.
$2 \Rightarrow f$ has no roots in a smaller subextension of $\mathbb{F}_{q^n}/\mathbb{F}_q$
So f is irreducible, since a factor would have to have roots in a field smaller than $\mathbb{F}_{q^n}$. Converse is similar. □

Algorithm

We want to compute $(x^{q^d} - x)\%f$.
We compute x^{q^d} in $\mathbb{F}_q[x]/(f(x))$ which takes at most $d \log q$ steps.
$x^{q^d} \equiv b \bmod f\ (\deg b < \deg f)$.
$(x^{q^d} - x)\%f = (b - x)\%f$
When $d = n$ this is item 1 of the theorem. For $d < n$ this calculation is the first step of the Euclidean Algorithm. Subsequent steps involve only polynomials of degree at most n.

Example 1. *Consider $f(x) = x^5 + x + 1$ in $\mathbb{F}_2[x]$.*
Condition 2 of the Theorem is clearly satisfied i.e. $gcd(f(x), x^2 - x) = 1$.
We want to compute x^{2^5} mod f.
x, x^2, x^4

$$\begin{aligned} x^8 &\equiv x^4 + x^3 \ mod\ f \\ x^{16} &\equiv x^8 + x^6 \equiv x^4 + x^3 + x^2 + x \ mod\ f \\ x^{32} &\equiv x^8 + x^6 + x^4 + x^2 \\ &\equiv x^4 + x^3 + x^2 + x^4 + x^2 \equiv x^3 + x \ mod\ f \end{aligned}$$

Hence $(x^{32} - x)\%f = x^3 \neq 0$ and hence condition 1 of the Theorem is not satisfied. So f is reducible.

For large q and small n there might be better algorithms.

We have a fast deterministic test for irreducibility of polynomials over finite fields. How do we find an irreducible polynomial of given degree n over $\mathbb{F}_q$? There is no deterministic polynomial time algorithm to do this. There is a probabilistic algorithm.

Algorithm

Pick a monic polynomial $f(x) \in \mathbb{F}_q[x]$ of degree n at random. Test for irreducibility. Repeat until you find an irreducible polynomial.

The algorithm is based on the following theorem.

Theorem 2.

$$\lim_{q^n \to \infty} \frac{\#\{\textit{monic irreducible polynomials of degree } n \textit{ over} \mathbb{F}_q\}}{q^n} = \frac{1}{n}.$$

From this theorem we conclude that the probability of failure after k tries is $(1 - \frac{1}{n})^k \to 0$ as $k \to \infty$

Proof. Let $a_d = \#$monic irreducible polynomials of $\deg n$ over $\mathbb{F}_q$.

Claim.

$$\sum_{d|n} d a_d = q^n$$

To prove this note that an irreducible polynomial of degree $d|n$ divides $x^{q^n} - x$ because its roots generate $\mathbb{F}_{q^d} \subseteq \mathbb{F}_{q^n}$. Conversely an irreducible factor of $x^{q^n} - x$ has roots in $\mathbb{F}_{q^n}$. So $x^{q^n} - x =$ product of all irreducible polynomials of $\deg d|n$.

Möbius Inversion Formula

$$\begin{aligned} \mu(n) = &\quad 0, && \text{if n is not square free} \\ = &\quad (-1)^r, && \text{if } n = p_1 \dots p_r \text{ distinct primes} \\ = &\quad 1, && n = 1. \end{aligned}$$

If x_n, y_n where $n \geq 1$ are such that $\sum_{d|n} d x_d = y_n$ then

$$x_n = \sum_{d|n} \mu(\frac{n}{d}) \frac{y_d}{n}$$

Applying the inversion formula to Claim 2 we have

$$\begin{aligned} a_n &= \frac{1}{n}\sum_{d|n} \mu(\tfrac{n}{d}) q^d \\ &= \frac{q^n}{n} + \frac{1}{n}\sum_{d|n\,,\, d<n} \mu(\tfrac{n}{d}) q^d \\ \Rightarrow \frac{a_n}{q^n} &= \frac{1}{n} + \frac{1}{n}\sum_{d|n\,,\, d<n} \mu(\tfrac{n}{d}) q^{d-n} \end{aligned}$$

The last term $\to 0$ as $q^n \to \infty$. □

This theorem is the function field version of the Prime Number Theorem.

$$\frac{\#\{primes \leq x\}}{x} \sim \frac{1}{\log x}.$$

Theorem 3 (Shoup). *If we can factor cyclotomic polynomials deterministically in polynomial time over $\mathbb{F}_q$ then we can construct irreducible polynomials of given* $\deg n$ *in polynomial time over $\mathbb{F}_q$.*

Factoring Polynomials over $\mathbb{F}_q$

Theorem 4. *There exist a polynomial time probabilistic algorithm for factoring polynomials in $\mathbb{F}_q[x]$.*

First Step

We first remove repeated factors. Let $f(x) = a_0x^n + \cdots + a_n \in \mathbb{F}_q[x]$ and $f'(x) = na_0x^n + \cdots + a_{n-1}$ be its formal derivative. If $f' \equiv 0$ then $f(x) = g(x)^p$ for some $g(x) \in \mathbb{F}_q[x]$ where $p = \operatorname{char} \mathbb{F}_q$. In this case factoring f reduces to factoring g.

$f'(x) \equiv 0 \Rightarrow a_i = 0$ if p does not divide $n - i$. Hence,

$$\begin{aligned} f(x) &= \sum_{p|n-i} a_i x^{n-i} \\ &= \sum_{p|n-i} a_i (x^{\frac{n-i}{p}})^p \\ &= \sum_{p|n-i} b_i^p (x^{\frac{n-i}{p}})^p \quad \text{where } b_i = a_i^{p^{m-1}} \text{ and } q = p^m \\ &= (\sum_{p|n-i} b_i x^{\frac{n-i}{p}})^p \end{aligned}$$

If f has no multiple roots then $\gcd(f, f') = 1$. On the other hand if $f(x)$ has a multiple root then it may be factored as

$$f = \frac{f}{(f, f')}(f, f').$$

If $f(x)$ has a root α of multiplicity k then $(x-\alpha)^k | f(x)$. This implies $f'(x)$ has a root of multiplicity $\geq (k-1)$. Now (f, f') has a root of multiplicity $\geq k-1$ and $\leq k$ at α. If p does not divide k then (f, f') has a root of multiplicity $k-1$ at α and $f/(f, f')$ has a simple root at α. We thus have a fast deterministic algorithm which given f produces a polynomial h which is square free and has the same irreducible factors as f. From now on we will assume that $f(x) \in \mathbb{F}_q[x]$ is a squarefree polynomial. That is, $f(x) = g_1(x) \dots g_r(x)$ where the g_i are distinct irreducible polynomials.

2 Berlekamp's Algorithm

Berlekamp's Algorithm is used to factor a polynomial $f(x)$ over a finite field $\mathbb{F}_q$. Given a squarefree polynomial $f(x) \in \mathbb{F}_q[x]$ of degree n, the goal is to find distinct irreducible polynomials $g_i(x) \in \mathbb{F}_q[x]$ so that

$$f(x) = g_1(x) \dots g_r(x).$$

If a polynomial $u(x) \in \mathbb{F}_q[x]$ satisfies

$$u(x)^q \equiv u(x) mod\, f(x) \tag{1}$$

then it can be shown that

$$\prod_{c \in \mathbb{F}_q} \gcd(f(x), u(x) - c) = f(x)$$

which may provide us with a factorization of f.

Polynomials that satisfy (1) form a vector space of dimension r over $\mathbb{F}_q$ and are the kernel of a certain linear map on $\mathbb{F}_q[x]/(f)$. By computing the $n \times n$ matrix of this map and then reducing it to row-echelon form, we can easily determine a basis for this subspace and hence polynomials of type (1). The gcd's are then computed with the Euclidean algorithm (since $\mathbb{F}_q[x]$ is a Euclidean domain).

2.1 Details

Now we describe this process in detail. By the Chinese Remainder Theorem we have

$$\frac{\mathbb{F}_q[x]}{(f(x))} \cong \frac{\mathbb{F}_q[x]}{(g_1(x))} \oplus \cdots \oplus \frac{\mathbb{F}_q[x]}{(g_r(x))} \tag{2}$$

via the map that sends

$$u(x) \bmod f(x) \mapsto u(x) \bmod g_1(x), \ldots, u(x) \bmod g_r(x)$$

for any polynomial $u(x) \in \mathbb{F}_q[x]$.

Clearly $\mathbb{F}_q^r$ embeds into the right hand side of (2); in fact at *most* r copies embed, and $\mathbb{F}_q^r$ is isomorphic to the space spanned by polynomials satisfying (1).

Theorem. *The Frobenius map*

$$Fr : \frac{\mathbb{F}_q[x]}{(f(x))} \longrightarrow \frac{\mathbb{F}_q[x]}{(f(x))} \tag{3}$$

$$u \longmapsto u^q \tag{4}$$

is and $\mathbb{F}_q$-linear map, and $\ker(Fr - I) \cong \mathbb{F}_q^r$ *so in particular*

$$\dim_{\mathbb{F}_q} \ker(Fr - I) = r.$$

Proof. Linear:

$$Fr(g + h) = (g + h)^q = g^q + h^q = Fr(g) + Fr(h)$$

since q is a power of the characteristic of $\mathbb{F}_q$.

If $\lambda \in \mathbb{F}_q$, then $\lambda^q = \lambda$ so

$$Fr(\lambda g) = (\lambda g)^q = \lambda g^q = \lambda Fr(g).$$

Kernel:

$$\begin{aligned}
& (Fr - I)(h) = 0 \bmod (f) \\
\Leftrightarrow\ & (Fr - I)(h_1, \ldots, h_r) = (0, \ldots, 0) \quad (\text{where } h_i = h \bmod g_i) \\
\Leftrightarrow\ & ((Fr - I)h_1, \ldots, (Fr - I)h_r) = (0, \ldots, 0) \\
\Leftrightarrow\ & h_i \in \mathbb{F}_q
\end{aligned}$$

since elements satisfying $h^q = h$ are precisely the elements of $\mathbb{F}_q$. So

$$\ker(Fr - I) \cong \mathbb{F}_q \oplus \cdots \oplus \mathbb{F}_q \hookrightarrow \frac{\mathbb{F}_q[x]}{(g_1(x))} \oplus \cdots \oplus \frac{\mathbb{F}_q[x]}{(g_r(x))}$$

□

Next we find a matrix representation of $(Fr - I)$. $1, x, x^2, \ldots, x^{n-1}$ is an $\mathbb{F}_q$-basis for $\mathbb{F}_q[x]/(f(x))$, so we compute a_{ij}'s in $\mathbb{F}_q$ such that

$$Fr(x^i) = x^{qi} = \sum_{j=0}^{n-1} a_{ij} x^j \bmod f(x), \qquad i = 0, \ldots, n-1.$$

Then the matrix $\left(a_{ij} - \delta_{ij}\right)$ (where δ_{ij} is the Kronecker delta) is the matrix for $(Fr - I)$ with respect to this basis.

Note: x^{qi} is an exponentiation in $\mathbb{F}_q[x]/(f(x))$ so it can be computed in polynomial time in $n \log q \log q^p \leq n \log q \log q^n \leq O((n \log q)^c)$.

Now we want to show how elements of the kernel actually provide us with factors of f. Suppose $u \in \ker(Fr - I) \setminus \mathbb{F}_q$ (where $\mathbb{F}_q$ is being viewed as the set of r-tuples, $(c, c, \ldots, c)$ in $\mathbb{F}_q^r$ with $c \in \mathbb{F}_q$) so $u = (c_1, \ldots, c_r)$, $c_i \in \mathbb{F}_q$ not all equal, say $c_i \neq c_j$.

Then $u - c_i$ can be viewed as a polynomial in $\mathbb{F}_q[x]/(f(x))$ via (2). By construction, $u \equiv c_i \bmod g_i$ so $u - c_i \equiv 0 \bmod g_i$, but modulo g_j, $u - c_i \equiv c_j - c_i \neq 0 \bmod g_j$. Hence $g_i \big| u - c_i$ but $g_j \not| u - c_i$ which implies $\gcd(u - c_i, f) \neq 1, f$. Thus u gives a proper factor of f and we get the following result:

Theorem. *Let $f(x) \in \mathbb{F}_q[x]$ be monic and squarefree. If $u \in \mathbb{F}_q[x]$ is such that $u^q \equiv u \bmod f$ but $u \not\equiv c \pmod f \; \forall\, c \in \mathbb{F}_q$ (ie $u \in \ker(Fr - I) \setminus \mathbb{F}_q$), then*

$$\prod_{c \in \mathbb{F}_q} gcd(u - c, f) = f.$$

(Note: None of the factors are equal to f since we required that $u \notin \mathbb{F}_q$, however some of the factors might be trivial since not every c is a c_i in the decomposition of u from above.)

Proof. As noted, if $c \neq c_i$ for any i in the decomposition of u, then $g_i \not| u - c$ for all i, hence $\gcd(u - c, f) = 1$ and so we are reduced to showing

$$\prod_{c_i \in \mathbb{F}_q} \gcd(u - c_i, f) = f.$$

where $u = (c_1, \ldots, c_r)$. From above we have that g_i divides $u - c_i$ and of course g_i divides f, so g_i divides $\gcd(u - c_i, f)$ hence

$$f = \prod_{i=1}^{r} g_i \Big| \prod_{c_i \in \mathbb{F}_q} \gcd(u - c_i, f).$$

On the other hand, for any $c, c' \in \mathbb{F}_q$, $c \neq c'$, we have $\gcd(u-c, u-c') = 1$ because $(u-c)-(u-c') = c-c' \in \mathbb{F}_q^*$ (so there exists $k \in \mathbb{F}_q$ such that $k(c-c') = 1 = k(u-c) - k(u-c')$). Thus $\gcd(u-c, f)$ and $\gcd(u-c', f)$ are coprime, and clearly divide f, hence

$$\prod_{c_i \in \mathbb{F}_q} \gcd(u - c_i, f) \Big| f.$$

□

Example 2. *Let $q = 2$ with $\mathbb{F}_2 = \{0, 1\}$ and let $u = (c_1, \ldots, c_r)$. Then by the theorem above*

$$f = gcd(u, f) \cdot gcd(u - 1, f).$$

We summarize:

Berlekamp's Algorithm.

1. *Construct the matrix $(Fr - I)$, (an $n \times n$ matrix over $\mathbb{F}_q$)*
2. *Compute the kernel (using Gaussian elimination). If $\dim(\ker(Fr - I)) = 1$, then $r = 1$ and f is irreducible, so stop.*
3. *If $\dim(\ker(Fr - I)) > 1$, find $u \in \ker(Fr - I) \setminus \mathbb{F}_q$ (linear algebra) and compute the gcd's in Theorem 2.1 (using the Euclidean algorithm) to split $f(x)$.*

Remark 2.1. This is a deterministic polynomial time factoring algorithm if q is small. For q large we have to do something else...

If $f(x)$ splits into linear factors over $\mathbb{F}_q$, then $\ker(Fr - I) \cong \mathbb{F}_q^n \cong \mathbb{F}_q[x]/(f(x))$ and

$$f = \prod_{c \in \mathbb{F}_q} \gcd(x - c, f).$$

Example 3. *Let $q = 2$ and $f(x) = x^5 + x + 1$. $\mathbb{F}_2[x]/(f)$ has as a basis $1, x, x^2, x^3, x^4$. The Frobenius applied to the basis elements gives*

$$\begin{aligned} Fr(1) &= 1 \\ Fr(x) &= x^2 \\ Fr(x^2) &= x^4 \\ Fr(x^3) &= x^6 = x^2 + x \\ Fr(x^4) &= x^8 = x^4 + x^3 \end{aligned}$$

We need to find a polynomial $u \in \mathbb{F}_2[x]/(f)$ *so that* $Fr(u) = u$. *Let*

$$u = \alpha x + \beta x^2 + \gamma x^3 + \delta x^4$$

Then

$$\begin{aligned} Fr(u) &= \alpha x^2 + \beta x^4 + \gamma(x^2 + x) + \delta(x^4 + x^3) \\ &\quad \gamma x + (\alpha + \gamma)x^2 + \delta x^3 + (\beta + \gamma)x^4 \end{aligned}$$

so take $\alpha = \gamma = \delta = 1$, $\beta = 0$ *and then*

$$u = x + x^3 + x^4$$

is in $\ker(Fr - I)$.

Computing the gcd's gives:

$$gcd(u, f) = x^3 + x^2 + 1, \qquad gcd(u + 1, f) = x^2 + x + 1$$

so

$$f = gcd(u, f) \cdot gcd(u + 1, f) = (x^3 + x^2 + 1)(x^2 + x + 1),$$

and in this case, both of the polynomials above happen to be irreducible, so we're done.

Remark 2.2. Note that

$$x+x^3+x^4 \equiv x+(x^2+1)+(x^2+1)x \equiv x+x^2+1+x^2+1+x \equiv 0 \bmod (x^3+x^2+1)$$

and

$$x+x^3+x^4 \equiv x+(x+1)x+(x+1)^2 = x+x^2+x+x^2+2x+1 \equiv 1 \bmod (x^2+x+1)$$

which demonstrates the isomorphism between $\ker(Fr - I)$ and $\mathbb{F}_2^2$.

We give a couple definitions, Any u satisfying the hypotheses of Theorem 2.1 is called a splitting polynomial for f. The values $c \in \mathbb{F}_q$ for which $\gcd(u - c, f) \neq 1$ are called splitting values for f and u.

We will now focus on different methods for computing splitting polynomials and their splitting values.

Theorem. *If* m *is an integer such that* $\deg g_i \big| m \, \forall \, i = 1, \ldots, r$ *then*

$$T_j := x^j + x^{jq} + x^{jq^2} + \cdots + x^{jq^{m-1}}$$

satisfies

$$T_j^q \equiv T_j \bmod f.$$

Furthermore $\exists \, j \leq n$ *such that* T_j *is a splitting polynomial (i.e. such that* $T_j \not\equiv c \bmod f \, \forall \, c \in \mathbb{F}_q$*).*

Proof. The second part of the theorem is due to McEliece and its proof will be omitted.

For the first part, observe that

$$\begin{aligned} \mathbb{F}_q[x]/(g_i(x)) &\cong \mathbb{F}_{q^k} \qquad \text{where } \deg g_i =: k & (5)\\ &\subseteq \mathbb{F}_{q^m} \qquad \Leftrightarrow k \big| m & (6) \end{aligned}$$

so if $\deg g_i \big| m$ then the roots of g_i are in $\mathbb{F}_{q^m}$ and $x^{q^m} \equiv x \bmod g_i$ which we use to compute:

$$\begin{aligned} T_j^q &\equiv x^{jq} + x^{jq^2} + \cdots + x^{jq^m} \\ &\equiv x^{jq^{m-1}} + \cdots + x^{jq} + x^j \\ &\equiv T_j \bmod g_i \end{aligned}$$

□

Remark 2.3. To compute the degrees of the g_i's, recall that $x^{q^d} - x$ is equal to the product of all irreducible polynomials in $\mathbb{F}_q[x]$ of degree k as k runs through the divisors of d. This gives a way to find all the factors of f of a certain degree; for example $\gcd(x^q - x, f)$ gives all linear factors of f and $\gcd(x^{q^2} - x, f)$ gives all linear *and* quadratic factors of f, so

$$\frac{\gcd(x^{q^2} - x, f)}{\gcd(x^q - x, f)}$$

gives only the quadratic factors of f. Using this method we can write

$$f = h_1 \cdots h_L$$

where each h_i is the product of the irreducible factors of f of degree i (and if $\deg h_i \big| m$ then so does i).

Here is a method for constructing a splitting polynomial in the special case of factoring a cyclotomic polynomial $f(x) = x^l - 1$ modulo $\mathbb{F}_q$.

Theorem. *If l is prime, $l \not| q$, H is a subgroup of $(\mathbb{Z}/l\mathbb{Z})^\times$ containing q, and C is a coset of H, then*

$$u := \sum_{c \in C} x^c$$

satisfies

$$u^q \equiv u \bmod (x^l - 1)$$

Proof.

$$u^q = \sum_{c \in C} x^{qc} = \sum_{c \in C} x^c = u \bmod (x^l - 1) \tag{7}$$

since $q \in H$. □

To apply this theorem we let H be the subgroup of squares in $(\mathbb{Z}/l\mathbb{Z})^\times$, C_1 the set of nonsquares, and then take

$$u := \sum_{j \in H} \left(\frac{j}{l}\right) x^j = \sum_{j \in H} x^j - \sum_{j \in C_1} x^j.$$

By the theorem, u satisfies $u^q \equiv u \bmod (x^l - 1)$ since it is a linear combination of elements of the form (7). Note that we must have $\left(\frac{q}{l}\right) = 1$ to use this, (so that q is an element of H as required by (2.1)).

Alternatively let m be the order of $q \bmod l$; then the factors of $x^l - 1$ have degree dividing m and we can write T_j from (2.1) as

$$T_j = \sum_{c \in j < q >} x^c$$

where $H :=< q >= \{1, q, \ldots, q^{m-1}\}$, and this will also satisfy $T_j^q \equiv T_j \bmod (x^l - 1)$.

We now turn to the task of computing the splitting values for f.

Theorem. *If $f(x)$ is a monic squarefree polynomial in $\mathbb{F}_q[x]$ of degree n and u is a splitting polynomial for f, then we can compute in deterministic polynomial time in $n \log q$ the polynomial whose roots are exactly the splitting values for u.*

Remark 2.4. Note that if f splits into linear factors, then we can take $u = x$ and then the polynomial whose roots are the splitting values of u is precisely f.

To show how to compute the splitting values we need to use the resultant.

2.2 Resultants

Let K be a field, $f(x) = a_0 + a_1 x + \cdots + a_n x^n$, and $g(x) = b_0 + b_1 x + \cdots + b_m x^m$ both in $K[x]$. The resultant of f and g, denoted $Res(f, g)$ is defined as the

determinant of an $(n+m)\times(n+m)$ matrix as follows,

$$Res(f,g)=\det\begin{vmatrix} a_n & a_{n-1} & a_{n-2} & \cdots & a_0 & 0 & \cdots & 0\\ 0 & a_n & a_{n-1} & \cdots & a_1 & a_0 & \cdots & 0\\ \vdots & & & \vdots & & & \vdots & \\ 0 & \cdots & \cdots & 0 & a_n & \cdots & \cdots & a_0\\ b_m & b_{m-1} & b_{m-2} & \cdots & b_0 & 0 & \cdots & 0\\ 0 & b_m & b_{m-1} & \cdots & b_1 & b_0 & \cdots & 0\\ \vdots & & & \vdots & & & \vdots & \\ 0 & \cdots & \cdots & 0 & b_m & \cdots & \cdots & b_0 \end{vmatrix}$$

One of the most important results regarding the resultant is that

$$Res(f,g)=a_n^m b_m^n\prod_{i=1}^{n}\prod_{j=1}^{m}(\alpha_i-\beta_j)$$

where $\alpha_1,\ldots,\alpha_n$, $\beta_1,\ldots,\beta_m$ are the roots of f, g respectively. From this we can conclude,

$$Res(f,g)=0 \Leftrightarrow f \text{ and } g \text{ have a common root}$$

Now we can prove Theorem 2.1,

Proof. Let

$$h(y):=Res_x\big(u(x)-y,f(x)\big)\in\mathbb{F}_q[y].$$

We claim that the roots of $h(y)$ are precisely the splitting values of u (with respect to f) : Let $c\in\mathbb{F}_q$, then

$$\begin{aligned} h(c)=0 &\Leftrightarrow Res_x\big(u(x)-c,f(x)\big)=0\\ &\Leftrightarrow u(x)-c \text{ and } f(x) \text{ have a common root}\\ &\Leftrightarrow u(x)-c \text{ and } f(x) \text{ have a common (nontrivial) factor}\\ &\Leftrightarrow c \text{ is a splitting value.} \end{aligned}$$

h can be computed by linear algebra in $\mathbb{F}_q[y]$ and hence in polynomial time. □

Here is a better way to compute $h(y)$ (in the case $n+1\le q$): Choose distinct $c_1,\ldots,c_{n+1}\in\mathbb{F}_q$, and compute $h(c_i)=Res_x\big(u(x)-c_i,f\big)$. This is just linear algebra in $\mathbb{F}_q$ (as opposed to $\mathbb{F}_q[y]$).

Since $\deg h \leq r \leq n$ (there are at most r splitting values), $n+1$ is guaranteed to be enough c_i to find an appropriate polynomial h, which we construct using Lagrange interpolation:

$$h(x) := \sum_{j=1}^{n+1} h(c_j) \prod_{i=0, i\neq j}^{n+1} \frac{x - c_i}{c_j - c_i}.$$

Here is the "best way" to compute h (due to Zassenhaus):

Theorem. *$h(y)$ is the monic polynomial of smallest degree such that*

$$h(u(x)) \equiv 0 \bmod f(x)$$

Proof. We'll prove this later in the course. □

Since $h(u(x))$ is a polynomial in $u(x)$, we want to find the smallest k such that

$$1, u(x), u(x)^2, \ldots, u(x)^k$$

is a linearly *dependent* set modulo f, and then apply the theorem to show this linear combination is equal to h.

Example 4. *Let $f(x) = x^4 + 3x^2 + 2$ in $\mathbb{F}_7[x]$. First we try $T_1 = x + x^7$ but it turns this is not a splitting polynomial since*

$$x^7 \equiv x^4 x^3 \equiv (4x^2 + 5)x^3 \equiv 4(4x^2 + 5)x + 5x^3 \equiv 21x^3 + 20x \bmod f$$

so

$$x^7 + x = 21x^3 + 21x \equiv 0 \bmod f$$

(and $0 \in \mathbb{F}_q$).

Next we try $T_2 = x^2 + x^{14}$. Since $x^4 \equiv 4x^2 + 5$, we have $x^8 \equiv 2x^4 + 5x^2 + 4 \equiv 6x^2$ so

$$x^{14} \equiv x^2 x^8 x^4 \equiv x^2(6x^2)x^4 \equiv 6x^8 \equiv 36x^2 \equiv x^2 \bmod f$$

therefore

$$T_2 \equiv 2x^2 \bmod f$$

is a splitting polynomial. We now compute h. The set $\{1, u\}$ are linearly independent but the set

$$\{1, u, u^2\} = \{1, 2x^2, 4x^4\}$$

satisfies

$$2u^2 + 5u + 2 \equiv 0 \bmod f.$$

Multiply through by 4, *i.e.* 2^{-1}, *to make this monic, then*

$$h(y) := y^2 - y + 1$$

is our h. *Its roots,* $c_1 = 3/2 \equiv 5 \bmod 7$ *and* $c_2 = -1/2 \equiv 3 \bmod 7$ *are the splitting values and when we compute the gcd's we get,*

$$gcd(2x^2 - 5, f) = x^2 + 1, \qquad gcd(2x^2 - 3, f) = x^2 + 2.$$

Both factors are clearly irreducible mod7 *(the only squares* mod7 *are* $1, 2$, *and* 4*), and multiply to* f *as wanted.*

Recall our setup from last week: $f(x) \in \mathbb{F}_q[x]$ is a square-free polynomial; $u(x) \in \mathbb{F}_q[x]$ is a splitting polynomial for f, so it satisfies

$$u^q \equiv u \pmod{f(x)} \qquad u \not\equiv c \pmod{f(x)} \text{ for any } c \in \mathbb{F}_q;$$

$C = \{c \in \mathbb{F}_q : (u - c, f) \neq 1\}$ is the set of splitting values for f and u; and $h(y) \in \mathbb{F}_q[y]$ is defined to be

$$h(y) = \prod_{c \in C} (y - c).$$

Let us clarify a misleading point from last week: In general, $h_1(y) = Res_x(u(x) - y, f(x))$ may not be equal to $h(y)$. We proved last time that the zeros of $h_1(y)$ are indeed the elements of C and $\deg(h_1(y)) \leq \deg(f(x))$ by construction. But generally h_1 factors as

$$h_1(y) = \prod_{c \in C} (y - c)^{\alpha_c}$$

where $\alpha_c \geq 1$. Thus h_1 may not have simple roots, though h does.

Example. Let $f(x) = x^4 + 3x^2 + 2 = (x^2 + 1)(x^2 + 2)$ be a polynomial in $\mathbb{F}_7[x]$. $u(x) = x^2$ is a splitting polynomial for f and $C = \{1, -2\}$. (All this was proven in a previous class.) Thus $h(y) = y^2 + 3y + 2$. Let's compute $h_1(y)$. An elementary property of the resultant is that it satisfies

$$Res(f, g) = a_n^m \prod_{i=1}^{n} g(x_i)$$

where x_i are the roots of f, a_n is the leading coefficient of f, and $m = deg(g)$ [See Dummit and Foote, p. 621]. In our case we have $h_1(y) = Res_x(x^2 - y, f(x)) = f(\sqrt{y})f(-\sqrt{y}) = h(y)^2$.

Theorem. *$h(y)$ is the monic polynomial of smallest degree such that $f(x)$ divides $h(u(x))$.*

Proof. Observe that $I = \{g(y) \in \mathbb{F}_q[x] : f(x)|g(u(x))\}$ is an ideal in $\mathbb{F}_q[x]$, which is a prinicipal ideal domain and hence I is generated by a single element (of minimal degree). Our claim is equivalent to the claim that $I = (h)$. First observe that $h \in I$: recall that $f = g_1 \cdots g_r$ where each g_i is irreducible and that $u \equiv c_i \pmod{g_i}$ for some $c_i \in \mathbb{F}_q$. (For this last claim, recall $u^q \equiv u \pmod{f}$, so $u^q \equiv u \pmod{g_i}$ for all i. Since the g_i's are irreducible, $\mathbb{F}_q[x]/(g_i)$ is a finite field containing a copy of $\mathbb{F}_q$ as the solutions of $x^q \equiv x \pmod{g_i}$. Thus $u \equiv c_i \pmod{g_i}$ for some $c_i \in \mathbb{F}_q \subseteq \mathbb{F}_q[x]/(g_i)$.) Thus g_i divides $u - c_i$ and hence also $\prod_{c \in \mathbb{F}_q}(u - c) = h(u)$. Thus every g_i divides $h(u)$, so f does as well.

Now assume $I = (k)$ for some $k \in \mathbb{F}_q[y]$, $k \neq h$. Since $h \in I$, we know k divides h, so

$$k(y) = \prod_{c \in C'} (y - c)$$

for $C' \subsetneq C$. So there exists a $c_i \in C$ such that $c_i \notin C'$. Recall that we know $u \equiv c_i \pmod{g_i}$ for some i. We claim for this fixed i, g_i does not divide $k(u)$. Assume this is not true, so g_i divides $\prod_{c \in C'}(u - c)$. But g_i is irreducible and hence a prime in $\mathbb{F}_q[x]$, so g_i divides $u - c$ for some $c \in C'$, so $u \equiv c \pmod{g_i}$, but $u \equiv c_i \pmod{g_i}$. Thus $c = c_i$, but this contradicts the fact that $c_i \notin C'$. Thus for some i, g_i does not divide $k(u)$, hence neither does f, so $k \notin I$, a contradiction. Hence k must be h, so $I = (h)$. □

Problem: Given a square-free $f(x) \in \mathbb{F}_q[x]$ that splits completely in $\mathbb{F}_q$ (i.e. $f(x)$ divides $x^q - x$), find the (distinct) roots of $f(x)$.

We have a probabilistic algorithm that is quite fast in practice due to Legendre. The idea is to split up $\mathbb{F}_q$ into two disjoint halves and hope that this will induce a splitting of f. After at most $\deg f$ successful splits, we will have found its roots. Here is the procedure:

Algorithm: Suppose first that q is odd. Observe that

$$x^q - x = x(x^{(q-1)/2} - 1)(x^{(q-1)/2} + 1).$$

The last two factors on the right distinguish the squares from the non-squares in $\mathbb{F}_q$, respectively, and the first factor distinguishes 0. With this in mind, pick a $b \in \mathbb{F}_q$ at random and consider

$$f(x) = (f(x), x - b) \cdot (f(x), (x - b)^{(q-1)/2} - 1) \cdot (f(x), (x - b)^{(q-1)/2} + 1).$$

If this is a non-trivial splitting of f, recurse over these other factors by choosing different b's. If not, choose a different b and try again.

Now suppose q is even, so say $q = 2^m$. Let $S(x) = x + x^2 + \cdots + x^{2^{m-1}}$. Observe $S(x)(S(x)+1) = x^q - x$. So for any $a \in \mathbb{F}_q$, $S(a) = Tr_{\mathbb{F}_q/\mathbb{F}_2}(a) \in \mathbb{F}_2$. Again, select a $b \in \mathbb{F}_q$ at random and consider

$$f(x) = (f(x), S(bx)) \cdot (f(x), S(bx) + 1).$$

If this is a non-trivial splitting of f, again recurse over these factors with new b's. If not, choose a different b and try again.

Remark 2.5. (q odd.) Computing $(x-b)^{(q-1)/2}$ using modular exponentiation is fast, as is computing the gcd's. Hence the only difficulty is finding a satisfactory b. So how often do we have a "bad" b, i.e. a b for which we get a non-trivial splitting of f? Suppose $f(x) = \prod_{i=1}^{n}(x - c_i)$. Then $f(x)$ divides $(x-b)^{(q-1)/2} - 1$ if and only if $x - c_i$ divides $(x-b)^{(q-1)/2} - 1$ for all i if and only if $(c_i - b)^{(q-1)/2} = 1$ for all i. Equivalently, $c_i - b$ is a square for every i. We will prove later that the probability of this event is approximately 2^{-n} where n is the degree of f.

Remark 2.6. (q even.) We will show below that picking a "bad" b in this case also has small probability.

We have the following theorems:

Theorem. *For $c_1, \ldots, c_n \in \mathbb{F}_q$ distinct, q odd,*

$$\#\{b \in \mathbb{F}_q : b - c_i \text{ is a square for all } i\} = \frac{q}{2^n} + O(n\sqrt{q})$$

Proof. Consider the system of equations

$$x - c_i = y_i^2 \tag{8}$$

for $i = 1, \ldots, n$ in the variables $x, y_1, \ldots, y_n$. Any b for which $b - c_i$ is a non-zero square in $\mathbb{F}_q$ for all i gives rise to solutions $x = b, y_i = \pm\sqrt{b - c_i}$. Thus we see that every such $b \neq c_i, \forall i$ yields 2^n solutions to the system (8). Our claim will follow if and only if (8) has $q + O(2^n n\sqrt{q})$ solutions in $\mathbb{F}_q$. Observe that (8) defines an affine algebraic curve X over $\mathbb{F}_q$. We have the map $\phi : X \to \mathbb{A}^1$ given by $(x, y_1, \ldots, y_n) \mapsto x$. This extends to a rational map $\bar{X} \to \mathbb{P}^1$, where $\bar{X}$ is the projective closure of X. (In fact this is a $(2, \ldots, 2)$ Galois cover of $\mathbb{P}^1$.) The genus of (the normalization of) $\bar{X}$ is $g = 2^{n-2}(n-3) + 1$, which can be calculated using Hurwitz's formula. Recall the Weil bound:

Lemma. *If X is a smooth irreducbile projective curve over $\mathbb{F}_q$ of genus g, then*

$$|\#X(\mathbb{F}_q) - (q+1)| \leq 2g\sqrt{q}.$$

Thus the estimate $q + O(2^n n\sqrt{q})$ for the number points on X is just the Weil bound. This completes the proof. □

Theorem. *For $c_1, \ldots, c_n \in \mathbb{F}_q$ distinct, q even,*

$$\#\{b \in \mathbb{F}_q : S(bc_i) = 0 \text{ for all } i\} = \frac{q}{2^k} = 2^{m-k}$$

where k is the dimension of the $\mathbb{F}_2$-vector space spanned by $c_1, \ldots, c_n$ inside $\mathbb{F}_q$.

Proof. Observe that S is $\mathbb{F}_2$-linear: $S(x+y) = S(x) + S(y)$ since we are in characteristic 2. Let $V = \{b \in \mathbb{F}_q : S(bc_i) = 0 \text{ for all } i\}$ be the $\mathbb{F}_2$-subspace of $\mathbb{F}_q$. We want to show that $\dim_{\mathbb{F}_2}(V) = m - k$. We recall the following lemma from field theory:

Lemma. *If L/K is a finite separable extension of fields then the map $L \times L \to K$ given by $(x, y) \mapsto Tr_{L/K}(xy)$ is a non-degenerate bilinear pairing of K-vector spaces, where $Tr_{L/K}$ is the trace of L/K.*

Now $Tr_{\mathbb{F}_q/\mathbb{F}_2}(x) = S(x)$ for all $x \in \mathbb{F}_q$. Observe that V is the orthogonal complement to $c_1, \ldots, c_n$ with respect to the trace pairing since $S(bc_i) = 0$ implies b is orthogonal to c_i with respect to this pairing. Thus $\dim_{\mathbb{F}_2}(V) = \dim_{\mathbb{F}_2}(\mathbb{F}_q) - \dim_{\mathbb{F}_2}(V^\perp) = m - k$ where $k = \dim_{\mathbb{F}_2}(span(c_1, \ldots, c_n))$. This completes the proof. □

Remark 2.7. (q even.) Recall that we have

$$f = \prod_{i=1}^{n}(x - c_i)$$

and we also have

$$f(x) = (f(x), S(bx))(f(x), S(bx) + 1)).$$

We have already discussed the "bad" b's arising from the first factor. For the second factor, consider the set $V_1 : \{b \in \mathbb{F}_q : S(bc_i) = 1 \text{ for all } i\}$. Next observe that if there exists a $b_0 \in \mathbb{F}_q$ such that $S(bc_i) = 1$ for all i, then $V_1 = b_0 + V$ since S is $\mathbb{F}_2$-linear, so V_1 is just a translation of V by b_0. In such a case, we see that $\#V_1 = \#V$. A priori though, there may exist no

such b_0. For observe that $V \cap V_1 = \emptyset$, thus if $V = \mathbb{F}_q$, $V_1 = \emptyset$ and we will have no such b_0.

Now writing f as

$$\begin{aligned} f(x) &= (f(x), S(bx))(f(x), S(bx)+1) \\ &= \prod_{c_i \in \mathbb{F}_q, S(bc_i)=0} (x - c_i) \prod_{c_i \in \mathbb{F}_q, S(bc_i)=1} (x - c_i), \end{aligned}$$

we see this is a (non-trivial) splitting of f if and only if $b \notin V \cup V_1$ (otherwise one of the products contains all of the c_i's, and hence is f). Now $\#(V \cup V_1) = \#V + \#V_1 \leq 2\#V = q/2^{k-1} \leq q/2$ if $k \geq 2$. So if k is at least 2, then at least half of the b's are "good." For $k = 0$, we see that $\{c_1, \ldots, c_n\} = \{0\}$ since they span just 0, which corresponds to the polynomial $f(x) = x$, which is already factored. For $k = 1$, we have that all of the c_i's are $\mathbb{F}_2$-linear combinations of one $c \in \{c_1, \ldots, c_n\}, c \neq 0$. This means f divides $x(x - c)$, hence is a quadratic which we can factor easily. These becomes trivial cases, which we check for, and otherwise we use the above algorithm.

Another significant improvement can be made though. We can make this probabilistic algorithm into a deterministic one via the following proposition:

Proposition 5. *Let $b_1, \ldots, b_m$ be a basis for $\mathbb{F}_q$ over $\mathbb{F}_2$. Then there exists a j such that $b_j \notin V \cup V_1$ so long as $m \geq 2$.*

In fact, thinking of $\mathbb{F}_q$ as $\mathbb{F}_2/(g(x))$ where $g(x) \in \mathbb{F}_2[x]$ is an irreducible polynomial of degree m, we have the natural basis $\{1, \bar{x}, \bar{x}^2, \ldots, \bar{x}^{m-1}\}$ where $\bar{\cdot}$ is reduction modulo g.

Proof of Proposition. We argue by contradiction: suppose every b_j is an element of $V \cup V_1$. Then for all j, $S(b_j c_i)$ are all equal. In particular $S(b_j c_1) = S(b_j c_2)$ for all j. So $S(b_j(c_1 - c_2)) = 0$ for all j by linearity of S. So $c_1 - c_2$ is orthogonal to every basis element. But S is a non-degenerate bilinear pairing, so $c_1 - c_2 = 0$, contradicting the assumption that f has distinct roots. □

Thus for q even, we have a deterministic polynomial time factoring algorithm by picking a b from a basis of $\mathbb{F}_q/\mathbb{F}_2$ instead of a random b.

Remark 2.8. (q odd.) A similar strategy works for $q = p^m$, p a small odd prime. Define

$$S(x) = x + x^p + \cdots x^{p^{m-1}},$$

which is just $Tr_{\mathbb{F}_q/\mathbb{F}_p}(x)$. Then

$$f(x) = \prod_{j=0}^{p-1} (f(x), S(bx) - j).$$

We want to find a b such that this is a (non-trivial) splitting. As before, there always exists such a b within a basis of $\mathbb{F}_q/\mathbb{F}_p$. This then gives us a deterministic polynomial time algorithm for factoring f in $\mathbb{F}_q$ where $char(\mathbb{F}_q)$ is small. This algorithm has running time $O((np \log q)^c)$.

Thus we have deterministic polynomial time algorithms for even q and odd q's with small characteristic (with respect to $n \log q$). This leaves only $\mathbb{F}_q$ with large characteristic, such as $\mathbb{F}_p$ with p large. We will tackle this next week.

Problem. Given $a \in \mathbb{F}_q$ (with q odd) such that a is a square; find x such that $x^2 = a$. This is equivalent to factoring the polynomial $x^2 - a$. Note that $a \in \mathbb{F}_q^*$ is a square if and only if $a^{\frac{q-1}{2}} = 1$.

The general techniques for polynomial solving give probabilistic polynomial time algorithms. For example, we can take the greatest common divisor

$$(x^2 - a,\ (x - b)^{\frac{q-1}{2}} - 1)$$

for random b. (See last lecture).

Today we will see some algorithms that are specific for square roots and outperform the general polynomial solving algorithms in some situations.

3 An algorithm

Let $q \equiv 3\ (mod\ 4)$. If $a^{\frac{q-1}{2}} = 1$ then $a^{\frac{q-1}{2}} a = a$ but $a^{\frac{q-1}{2}} a = a^{\frac{q+1}{2}} = (a^{\frac{q+1}{4}})^2$. Hence $x = \pm a^{\frac{q+1}{4}}$ are the square roots of a in $\mathbb{F}_q$.

Note that this is a consequence of a fact that holds for every group of odd cardinality: $|G| = m \equiv 1\ (mod\ 2)\ \Rightarrow\ (g^{\frac{m+1}{2}})^2 = g\ \forall\ g \in G$. Now $q \equiv 3 \pmod 4 \Rightarrow$ $\#[(\mathbb{F}_q^*)^2] = \frac{q-1}{2}$ is odd.

This is a nice way to take square roots in groups of odd order. We can try to generalize this idea. Write $q - 1 = 2^r m$ with m odd.

Since $\mathbb{F}_q^*$ is a cyclic group, it has a unique subgroup of order m, let's call it G :

$$G\ =\ \{x \in \mathbb{F}_q^*\ |\ x^m = 1\}.$$

Then the following sequence is exact, and $|\mathbb{F}_q^*/G| = 2^r$:

$$0 \to G \hookrightarrow \mathbb{F}_q^* \to \mathbb{F}_q^*/G \to 0$$

$|\mathbb{F}_q^*/G| = 2^r$ and $\mathbb{F}_q^* \simeq G \times \mathbb{F}_q^*/G$.

Suppose $c \in \mathbb{F}_q^*$ is not a square. Then using c we will have a deterministic algorithm for taking square roots in $\mathbb{F}_q^*$ which is good if r is small. We discuss how to find c below.

Given $a \in \mathbb{F}_q^*$ set $e_0 = 0$, and for $i = 1, ..., r$ do: if $(ac^{-e_{i-1}})^{\frac{q-1}{2^{i-1}}} \neq 1$ put $e_i = e_{i-1} + 2^{i-1}$, otherwise, $e_i = e_{i-1}$. We will show below that $ac^{-e_r} \in G$,i.e. $(ac^{-e_r})^m = 1$.

Claim. e_r even $\Rightarrow$ $(ac^{-e_r})^{\frac{m+1}{2}} c^{\frac{e_r}{2}}$ is a square root of a.

Proof. Since m is odd and e_r is even, all the exponents in the expression are integers. Now let's take the square:

$$\begin{array}{ll} [(ac^{-e_r})^{\frac{m+1}{2}} c^{\frac{e_r}{2}}]^2 & = \\ (ac^{-e_r})^{m+1} c^{e_r} & = \\ (ac^{-e_r})^m (ac^{-e_r}) c^{e_r} & = \\ a(c^{-e_r} c^{e_r}) & = a \end{array}$$

□

Claim. For all $i = 0, \ldots, r$ $(ac^{-e_i})^{\frac{q-1}{2^i}} = 1$.

Proof. By induction.

$\underline{i = 0:}$ $(ac^{-e_0})^{q-1} = 1$ automatically. (Nothing to be checked).

$\underline{1 \leq i \leq r-1:}$ $(ac^{-e_{i-1}})^{\frac{q-1}{2^{i-1}}} = 1$ by our inductive hypothesis. Taking square roots on both sides we obtain:

$$(ac^{-e_{i-1}})^{\frac{q-1}{2^i}} = \pm 1,$$

so there are two cases to check.

Case 1: $= +1$. Here $e_i = e_{i-1}$, so $(ac^{-e_i})^{\frac{q-1}{2^i}} = 1$.

Case 2: $= -1$. Here $e_i = e_{i-1} + 2^{i-1}$, so

$$\begin{aligned}
(ac^{-e_i})^{\frac{q-1}{2^i}} &= (ac^{-e_{i-1}-2^{i-1}})^{\frac{q-1}{2^i}} \\
&= (ac^{-e_{i-1}})^{\frac{q-1}{2^i}} (c^{-2^{i-1}})^{\frac{q-1}{2^i}} \\
&= (-1)c^{-\frac{q-1}{2}} \\
&= (-1)^{\frac{1}{-1}} \quad \text{(since } c \text{ is not a square).} \\
&= 1
\end{aligned}$$

$\underline{i = r:}$ $1 = (ac^{-e_r})^{\frac{q-1}{2^r}} = (ac^{-e_r})^m$, hence $ac^{-e_r} \in G$.

□

Note that we get an isomorphism

$$\begin{array}{ccc} \mathbb{F}_q^*/G & \xrightarrow{\sim} & \mathbb{Z}/2^r\mathbb{Z} \\ a & \mapsto & e_r \end{array}$$

The problem with this algorithm is finding c. If we assume the generalized Riemann hypothesis, for q an odd prime there exists c, $1 \leq c \leq 4(\log q)^2$, with $c^{\frac{q-1}{2}} \neq 1 \pmod{q}$. Note that if for a given q such a c doesn't exist, then this would be a counterexample to GRH. Alternatively, we can do a random search for c and each try will have a 0.5 probability of success.

Now suppose $a \in \mathbb{F}_q$ is a square, and let $t \in \mathbb{F}_q$ such that $f(x) = x^2 - tx + a$ is irreducible in $\mathbb{F}_q[x]$. Then for α a root of f, $\mathbb{F}_q(\alpha) = \mathbb{F}_{q^2}$ and the norm of α is given by the product of α itself and its image under the Frobenius morphism, hence:

$$N_{\mathbb{F}_{q^2}/\mathbb{F}_q}(\alpha) = \alpha^{q+1} = a.$$

So $\alpha^{\frac{q+1}{2}}$ is a square root of a in $\mathbb{F}_{q^2}$. Since we are assuming that a has a square root in $\mathbb{F}_q$, we conclude that $\alpha^{\frac{q+1}{2}} \in \mathbb{F}_q$.

$\underline{\text{When is } f \text{ irreducible?}}$ $x^2 - tx + a$ is irreducible iff $t^2 - 4a$ is not a square. So again we need to find a non square in $\mathbb{F}_q$. We can proceed randomly to find it or sequentially if we believe GRH.

This algorithm is much faster than the previous one when r, (the 2-adic valuation of q), is large.

4 Schoof's algorithm

Given $a \in \mathbb{Z}$ and a prime p, Schoof's algorithm is a deterministic algorithm to compute a square root of d mod p in polynomial time in $|d| \log p$. It is based on a method to count rational points on elliptic curves over finite fields.

Let $a, b, \in \mathbb{F}_q$, with $(b, q) = 1$ and let's consider the elliptic curve over the finite field $\mathbb{F}_q$ given by the equation $y^2 = x^3 + ax + b$. Thus

$$E(\mathbb{F}_{q^n}) = \{(x, y) \in \mathbb{F}_{q^n} \times \mathbb{F}_{q^n} \mid y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\}$$

Elliptic curves have a group structure where $P + Q + R = \mathcal{O} \Leftrightarrow P, Q, R$ are colinear.

We know $\#E(\mathbb{F}_q) = q + 1 - t$ where $|t| \leq 2\sqrt{q}$.

Schoof's algorithm computes $\#E(\mathbb{F}_q)$ in polynomial time in $\log q$.

Let's say we compute the square root of $d \pmod p$, $d \in \mathbb{Z}$ and p prime. For simplicity, let's assume $d < 0$. (Think of $0 < d < p$ and take $d := d - p$).

We need to find an elliptic curve with complex multiplication [1]by $\mathbb{Q}(\sqrt{d})$ and reduce it modulo p. We won't see here how to find such a curve, but it exists. Unfortunately, such a curve is hard to find and that's why the running time depends badly on $|d|$. Assume that E is such a curve.

Let $f(x) = x^2 - tx + q$ where $\#E(\mathbb{F}_q) = q + 1 - t$. Let α be a root of f. Then $\begin{cases} \alpha.\overline{\alpha} &= q \\ \alpha + \overline{\alpha} &= t \end{cases}$

The complex multiplication hypothesis is essentially equivalent to $\alpha, \overline{\alpha} \in \mathbb{Q}(\sqrt{d})$ and this is all we will use of it. Let's write $\alpha = u + \sqrt{d}v$ where $2u,\ 2v \in \mathbb{Z}$. Then:

$\begin{cases} t &= 2u (= \alpha + \overline{\alpha}) \\ p &= u^2 - dv^2 \end{cases}$

We will compute u using Schoof, and from that we compute $v = \sqrt{(u^2 - p)/d}$. Now $p = u^2 - dv^2 \ \Rightarrow (\frac{u}{v})^2 \equiv d \bmod p$. Hence we would have found a square root of d mod p, namely $\frac{u}{v}$.

Example.

$d = -1$

$E:\ y^2 = x^3 - x \pmod p$.

E has complex multiplication by $\mathbb{Q}(i)$. If $p \equiv 1 \pmod 4$, then the number of points on the curve is $p + 1 - 2u$, and $u^2 + v^2 = p$. This gives a square root of -1 modulo p. As an aside, $(\frac{p-1}{2})!$ is also a square root of -1 modulo p but is a horrible way to compute it.

[1]See Silverman's *Advanced topics in the arithmetic of elliptic curves.*

EXAMPLE.

$E: \ y^2 = x^3 - 1$

E has complex multiplication by $\mathbb{Q}(\sqrt{-3})$.

How do we count the points? Let's consider the Frobenius morphism and its square.

$$\begin{array}{ccccccc} E & \xrightarrow{Fr} & E & \qquad & E & \xrightarrow{Fr^2} & E \\ (x,y) & \mapsto & (x^q,\ y^q) & & (x,y) & \mapsto & (x^{q^2},\ y^{q^2}) \end{array}$$

It can be proved that Fr satisfies $x^2 - tx + q = 0$, hence

$$Fr^2 - tFr + qI = 0.$$

$$(x^{q^2},\ y^{q^2}) + [q](x,\ y) = [t](x^q,\ y^q)$$

We don't know t. We will compute it modulo l for every $l \leq L$, where L is chosen as the smallest such that $M := \prod_{\text{primes } l \leq L} l > 4\sqrt{q}$. Once we know $t \bmod l$ for every $l \leq L$, then we know $t \bmod M$ by the Chinese Reminder Theorem. But we also know that $|t| \leq 2\sqrt{q} < M/2$, hence $|t| < \frac{M}{2}$. This uniquely determines t.

Note that the Prime Number Theorem implies that $M \sim e^L$.

Since we want $M > 4\sqrt{q}$ it is enough to take $L = O(\log q)$. There are $O(\log q)$ primes $l \leq O(\log q)$. Remember that the **l-torsion** of the elliptic curve is given by the elements whose order is divisible by l :

$$E[l] = \{p \in E(\overline{\mathbb{F}_q}) \mid lp = 0\}$$

$$Fr^2 - tFr + qI = 0 \text{ on } E[l].$$

For each l, the algorithm computes $Fr^2 + qI$ on $E[l]$ and then, for each $\tau = 0, ..., l-1$, it computes τFr on $E[l]$ until $Fr^2 + qI = \tau Fr$ in $E[l]$. Once a match is found, we get $\tau \equiv t (\bmod\ l)$.

Using the algebraic description of the group law we can write

$$[n](x,y) = (\frac{u_n(x)}{f_n(x)^2}, \frac{v_n(x)y}{f_n(x)^3}),$$

where u_n, v_n, f_n are certain polynomials. This implies that

$$(x,y) \in E[l] \Leftrightarrow f_l(x) = 0.$$

To test the equation $(x^{q^2}, y^{q^2}) + [q](x, y) = [\tau](x^q, y^q)$ in $E[l]$ can be tested by computing in $\mathbb{F}_q[x]/(f_l(x))$.

For instance, compute $\frac{u_n(x)}{f_n(x)^2}$ mod f_l where $n \equiv q \bmod l$ will give the x coordinate of $[q](x, y)$ in $E[l]$. Likewise, $(\frac{u_\tau(x)}{f_\tau(x)^2})^q$ is the x coordinate of $[\tau](x^q, y^q)$.

We know that deg $f_l = \frac{l^2-1}{2}$ if l is odd. $O(\log q^{\frac{l^2-1}{2}}) = O(l^2 \log q) = O((\log q)^3)$ is required for doing one computation on the ring $\mathbb{F}_q[x]/(f_l(x))$.

The case $l = 2$ is a little special but it can be done directly.

$\#E(\mathbb{F}_q) = q+1-t$ and since q is odd, $\#E(\mathbb{F}_q) \equiv t \bmod 2$. Hence $\#E(\mathbb{F}_q)$ is even $\Leftrightarrow$ $(E[2]\backslash\{\mathcal{O}\}) \cap E(\mathbb{F}_q) \neq \emptyset$.

Note that the points of odd order come in pairs, whereas $E[2] - \{\mathcal{O}\}$ has odd cardinality, since its elements correspond to the roots of $x^3 + ax + b$ in $\mathbb{F}_q$.

We will apply Schoof's algorithm, previously discussed, to compute the number of points of $y^2 = x^3 + 1$ over $\mathbb{F}_5$ and $\mathbb{F}_7$.

Recall the following: $\#E(\mathbb{F}_q) = q+1-t$, where $|t| \leq 2q^{1/2}$. Furthermore, if

$$(x^{q^2}, y^{q^2}) + [q](x, y) = \tau(x^q, y^q)$$

in $E[\ell]$ for some ℓ, then $t \equiv \tau \bmod \ell$.

We have defined the Frobenius morphism, for $P = (x, y)$, to be $Fr(P) = (x^q, y^q)$, and so the previous equation could also be written, for $P \in E[\ell],\ P \neq 0$, that $Fr^2(P) + qP = \tau Fr(P)$.

If $P \in E[\ell] \cap E(\mathbb{F}_q),\ P \neq 0$, then $Fr(P) = P$ and $Fr^2(P) = P$, so in calculations we will check to see if $P + qP = \tau P$, i.e. $\tau \equiv (q + 1) \bmod \ell$, or $t \equiv (q + 1) \bmod \ell$. This is the same as saying that if $E(\mathbb{F}_q)$ has a point of order ℓ, then $\ell | \#E(\mathbb{F}_q)$.

With our given curve, $y^2 = x^3 + 1$, it is trivial to see that $(-1, 0) \in E[2]$ and $(0, 1) \in E[3]$. Also, since $t \equiv (q + 1) \bmod 6$, if $q = 5$, then $t \equiv 6 \equiv 0 \bmod 6$. Since $|t| \leq 2\sqrt{5}$, this implies that $t = 0$, so $y^2 = x^3 + 1$ has six points over $\mathbb{F}_5$, including the point at infinity. Note, of course, that there are other ways to calculate the number of points of this curve over $\mathbb{F}_5$.

However, things become less trivial over $\mathbb{F}_7$: When $q = 7$, then $t \equiv 8 \equiv 2 \bmod 6$. Since $|t| \leq \lfloor 2\sqrt{7} \rfloor$, it follows that $t = 2$ or $t = -4 \equiv 3 \bmod 7$. This means that we can't decide between these options just considering points mod2 and mod3, so we must attempt to calculate $t \bmod 5$, which requires that we work in $E[5]$. (Note that $\#E[5] = 25$, including one zero point). It is too cumbersome to carry out these calculations by hand, so we resort to a Pari script, which can be found at the following URL.

`http://www.ma.utexas.edu/users/voloch/FFnotes/schoof.gp`.

Notes about the Pari Script

Note that

$$5(x,y) = (\frac{u_5(x)}{f_5(x)^2}, \frac{v_5(x)}{f_5(x)^3}y),$$

and by reading the denominators in the output of the script, we see that $\deg f_5(x) = 12$; its roots are the x-coordinates of the non-zero points of $E[5]$. We can check that f_5 is irreducible mod7.

As a matter of notation, the script has a variable a, which is given by $a = \bar{t} \in \mathbb{F}_7[t]/(f_5(t)) = \mathbb{F}_{7^{12}}$, which gives some indication of why it would be unreasonable to carry out these calculations by hand.

Furthermore, $P = (a, v) \in E[5]$, where $v = \sqrt{a^3+1}$. So $v = \overline{x}$, the image of x in the field $\mathbb{F}_{7^{12}}[x]/(x^2 - (a^3+1)) = \mathbb{F}_{7^{24}}$.

After running the script, we find that $t \equiv 1 \bmod 5$, which allows us to decide that $t = -4$.

Now we know t mod 30. Moreover, this will work not only for 7: For any prime q such that $4\sqrt{q} < 30$, or, solving for q, for any prime $q < 50$. we can find t once we know $t \bmod 2$, $t \bmod 3$, and $t \bmod 5$.

Incidental Comments

There is one point to consider that is incidental to the calculation: one step includes factoring a polynomial with integer coefficients. How do we do this?

Note that

$$\begin{aligned} f(x) &= f_1 \cdots f_r \qquad \in \mathbb{Z}[x] \\ f(x) &\equiv g_1 \cdots g_k \pmod{p} \in \mathbb{F}_p[x], \end{aligned}$$

We can factor in $\mathbb{F}_p[x]$, but we don't know how to lift factorizations to $\mathbb{Z}[x]$, besides trying all combinations. Applying Hensel's lemma allows us to find the factorization modp^n, given the factorization mod$p$, for all n. This provides congruences modulo p^n for the coefficients of the unknown f_i. However, this isn't quite enough. Additionally, we must infer a bound for the size of the coefficients of the f_i, based on the size of the coefficients of f. To find such a bound, we must use the Mahler measure, specifically the property that $\mu(fg) = \mu(f)\mu(g)$, and the fact that there are bounds for the coefficients based on the Mahler measure.

5 Primitive Roots

An element $g \in \mathbb{F}_q^*$ is called a *primitive root* if $< g >= \mathbb{F}_q^*$.
How many primitive roots are there? $\varphi(q-1)$.
If you pick $g \in \mathbb{F}_q^*$ at random, then g is a primitive root with probability

$$\frac{\varphi(q-1)}{q-1} = \prod_{l|q-1,\ l\ prime} (1 - 1/l) \gg 1/log\ q.$$

So there are many primitive roots.

How can we test if g is a primitive root? g is a primitive root $\Leftrightarrow g^{\frac{q-1}{l}} \neq 1\ \forall\ l|q-1,\ l\ prime$.
This is easy to do, *provided* we know the prime factors of $q-1$.

To construct finite fields, we construct $\mathbb{F}_q = \mathbb{F}_p[x]/(f(x))$.
A polynomial $f(x)$ is primitive if $f(x) \in \mathbb{F}_p[x]$ is irreducible and $< x >= (\mathbb{F}_p[x]/(f(x)))^*$.

Example: $\mathbb{F}_{p^p} = \mathbb{F}_p[x]/(x^p - x + g)$, where g is a primitive root *mod* p.

Conjecture: $x^p - x + g$ is primitive.
This is equivalent to the root of $x^p - x + 1$ having order $\frac{p^p-1}{p-1}$.
We know this element has order $\gg (5.9)^p$.

To see this: $\overline{x}^p = \overline{x} - 1$, so induction implies $\overline{x}^{p^k} = \overline{x} - k$.
So if $n = \sum n_k p^k$, then $\overline{x}^n = \prod(\overline{x} - k)^{n_k}$.
If $\sum n_k < p$, then we get distinct elements of $\mathbb{F}_p[x]/(x^p - x + 1)$, at least 2^p of them. So the order is at least 2^p, and getting to $(5.9)^p$ is another technique.

Example: $p = 2$, $\mathbb{F}_4 = \mathbb{F}_2[x]/(x^2 + x + 1)$. This case can be extended to all fields with 2^{2^n} elements by a construction of Wiedemann.

Theorem. *For $n \geq 0$, define $\alpha_0 = 1$, and α_n to be a root of $x^2 + \alpha_{n-1}x + 1 = 0$. Then $\mathbb{F}_{2^{2^n}} = \mathbb{F}_2(\alpha_n)$.*

Lemma. *If $a \in \mathbb{F}_{2^m}$, then $x^2 + x + a$ is irreducible in $\mathbb{F}_{2^n}[x] \Leftrightarrow Tr_{\mathbb{F}_{2^m}/\mathbb{F}_2}(a) = 1$.*

Proof. Consider $\wp : \mathbb{F}_{2^m} \longrightarrow \mathbb{F}_{2^m}$ where $x \mapsto x^2 + x$. $\wp$ is $\mathbb{F}_2$-linear and $ker\wp = \mathbb{F}_2$, so $Im\wp$ is of codimension 1 in $\mathbb{F}_{2^m}$.

On the other hand, $Im\wp \subseteq ker(Tr_{\mathbb{F}_{2^m}/\mathbb{F}_2})$ because $Tr(x^2+x) = Tr(x^2) + Tr(x) = 2Tr(x) = 0$ (since x^2 and x are conjugates).
Since Tr is non-trivial (an algebra fact), we get $Im(\phi) = ker(Tr)$.
Since quadratics are irreducible when they don't have roots, we get the lemma. □

So now back to the Theorem:

Proof of Theorem. $\frac{1}{\alpha_{n-1}^2}(x^2 + \alpha_{n-1}x + 1) = (\frac{x}{\alpha_{n-1}})^2 + (\frac{x}{\alpha_{n-1}}) + \frac{1}{\alpha_{n-1}^2}$.
We need to prove that $Tr(\frac{1}{\alpha_{n-1}^2}) = 1$.
We prove this by induction.

$$\begin{aligned} Tr(\frac{1}{\alpha_{n-1}^2}) &= Tr(\frac{1}{\alpha_{n-1}}) \text{ since they are conjugates} \\ &= Tr(\alpha_{n-1} + \alpha_{n-2}) \text{ since } \alpha_{n-1}^2 + \alpha_{n-2}\alpha_{n-1} + 1 = 0 \Rightarrow \alpha_{n-1} + \alpha_{n-2} + 1/\alpha_{n-1} = 0 \\ &= Tr_{\mathbb{F}_{2^{2^{n-1}}}/\mathbb{F}_2}(\alpha_{n-1}) + Tr_{\mathbb{F}_{2^{2^{n-1}}}/\mathbb{F}_2}(\alpha_{n-2}) \end{aligned}$$

Now

$$Tr_{\mathbb{F}_{2^{2^{n-1}}}/\mathbb{F}_2}(\alpha_{n-2}) = Tr_{\mathbb{F}_{2^{2^{n-2}}}/\mathbb{F}_2}(Tr_{\mathbb{F}_{2^{2^{n-1}}}/\mathbb{F}_{2^{2^{n-2}}}}(\alpha_{n-2})) = 0,$$

since

$$Tr_{\mathbb{F}_{2^{2^{n-1}}}/\mathbb{F}_{2^{2^{n-2}}}}(\alpha_{n-2}) = \alpha_{n-2} + \alpha_{n-2} = 0$$

since $\alpha_{n-2} \in \mathbb{F}_{2^{2^{n-2}}}$.
And

$$Tr_{\mathbb{F}_{2^{2^{n-1}}}/\mathbb{F}_2}(\alpha_{n-1}) = Tr_{\mathbb{F}_{2^{2^{n-2}}}/\mathbb{F}_2}(Tr_{\mathbb{F}_{2^{2^{n-1}}}/\mathbb{F}_{2^{2^{n-2}}}}(\alpha_{n-1})).$$

But α_{n-1} is a root of $x^2 + \alpha_{n-2}x + 1$ over $\mathbb{F}_{2^{2^{n-2}}}$, we get

$$Tr_{\mathbb{F}_{2^{2^{n-1}}}/\mathbb{F}_{2^{2^{n-2}}}}(\alpha_{n-1}) = \alpha_{n-2},$$

so $Tr_{\mathbb{F}_{2^{2^{n-1}}}/\mathbb{F}_2}(\alpha_{n-1}) = 1$ by induction. □

We will now use the notation $q = 2^{2^{n-1}}$. So $\mathbb{F}_{q^2} = \mathbb{F}_q(\alpha)$, where $\alpha = \alpha_n$. Every element of $\mathbb{F}_{q^2}$ is of the form $u + v\alpha,\ u, v \in \mathbb{F}_q$. Arithmetic is

$$(u_1 + v_1\alpha)(u_2 + v_2\alpha) = (u_1u_2 + v_1v_2) + (u_1v_2 + u_2v_1 + \alpha_{n-1}v_1v_2)\alpha.$$

So to do multiplication in $\mathbb{F}_{q^2}$, one must do multiplication in $\mathbb{F}_q$, so it's recursive and can be very efficient in larger fields.

Conjecture. *$\alpha_n\alpha_{n-1}\cdots\alpha_1$ is a primitive root for $\mathbb{F}_{2^{2^n}}$ for all n.*
($\Leftrightarrow$ α_n has order $q+1$, i.e. $2^{2^{n-1}}+1\ \forall\ n$)

Note that $q^2-1=(q-1)(q+1)$ and that $q^2-1=\#\mathbb{F}_{q^2}^*, q-1=\#\mathbb{F}_q^*$, and $q+1=\#\{x\in\mathbb{F}_{q^2}^*|N_{\mathbb{F}_{q^2}/\mathbb{F}_q}(x)=1\}$. On the other hand, $\alpha^2+\alpha_{n-1}\alpha+1=0$, so $N_{\mathbb{F}_{q^2}/\mathbb{F}_q}(\alpha)=1$. So the conjecture is that α is a generator of the subgroup of norm 1 elements of $\mathbb{F}_{q^2}$.

The conjecture is known for $n\leq 11$. The hard part is factoring $F_n=2^{2^n}+1$.

Question: Give a lower bound to the order of α_n.
A good lower bound combined with the partial factorizations of $\mathbb{F}_n$'s might allow us to check the conjecture for a few more values of n.

6 The Discrete Logarithm Problem

6.1 The Problem

Let G be a cyclic group of order n and $g\in G$ a generator, i.e. $\langle g\rangle=G$. The Discrete Logarithm Problem is the following. Given $h\in G$, we want to find some $m\in\mathbb{Z}/n\mathbb{Z}$ such that $h=g^m$.

We currently have no good algorithm for solving the Discrete Log Problem (henceforth called the DLP), but we can reduce the number of steps it takes from the order n taken by the stupid algorithm of just computing every power of g until we find the one desired. For example, for a group of non-prime order n, if we can write $n=n_1n_2$ with $(n_1,n_2)=1$, then we know that $G\cong G_1\times G_2$, where $|G_i|=n_i$. The group isomorphism is given by

$$\begin{aligned} G &\cong G_1\times G_2 \\ x &\to (x^{n_2},x^{n_1}) \\ y^az^b &\leftarrow (y,z), \quad an_2+bn_1=1. \end{aligned}$$

So the DLP in a group of order n reduces to the DLP on the groups of order n_1 and n_2, respectively.

For a general group, the best algorithm for the DLP takes $O(\sqrt{n})$ steps. Notice that $\sqrt{n}=e^{\frac{\ln n}{2}}$, so the algorithm is exponential in $\ln n$. In $\mathbb{F}_q^*$, there is an algorithm taking $O(e^{(\ln n)^{1/3+\epsilon}})$ steps, known as the "index calculus

algorithm", which is probabilistic in nature. This algorithm performs better than exponential time, but still not as good as an ideal polynomial-time algorithm:

$$(\log n)^c < e^{(\log n)^\alpha} < n^c$$

There is currently no polynomial time algorithm for solving the DLP.

6.2 An Algorithm in $O(\sqrt{n})$ (Shank's "Baby-step/Giant-step" Algorithm)

As above, let G be a cyclic group of order n with generator g. The following is a deterministic algorithm that, given $h \in G$, computes m such that $h = g^m$ in $O(\sqrt{n})$ time.

- (Baby Steps) Compute and store the values $1, g, g^2, g^3, \ldots, g^{\lfloor\sqrt{n}\rfloor}$
- (Giant Steps) Given $h \in G$, compute $hg^{-i\lfloor\sqrt{n}\rfloor}$ for $i = 1, 2, \ldots$, and compare it with the list $1, g, \ldots, g^{\lfloor\sqrt{n}\rfloor}$, until you find a match. If a match is found, stop, else proceed to next value of i.

  Once you find a match, you've just found i, j such that

$$hg^{-i\lfloor\sqrt{n}\rfloor} = g^j \Longrightarrow h = g^{i\lfloor\sqrt{n}\rfloor + j}$$

Theorem. *There is a match with $0 < i \leq \lfloor\sqrt{n}\rfloor + 1$. Therefore, the algorithm above stops in $O(\sqrt{n})$ steps.*

Proof. Choose $h \in G$. Then since G is cyclic with generator g, $h = g^m$ for some m. Use the division algorithm to write

$$m = i\lfloor\sqrt{n}\rfloor + j, \quad 0 \leq j \leq \lfloor\sqrt{n}\rfloor - 1$$

Then we have

$$i = \frac{m-j}{\lfloor\sqrt{n}\rfloor} \leq \frac{m}{\lfloor\sqrt{n}\rfloor} \leq \frac{n-1}{\lfloor\sqrt{n}\rfloor} \leq \frac{n-1}{\sqrt{n}-1} = \sqrt{n}+1$$

and it follows that $i \leq \lfloor\sqrt{n}\rfloor + 1$. □

6.3 The Index Calculus Algorithm "Framework"

Suppose that you have the following.

- G a cyclic group of order n.
- g a generator of G
- $B = \{g_1, g_2, \ldots, g_r\} \subseteq G$ a "factor base" for G, and
- An algorithm $(**)$ that, given $h \in G$, outputs with a certain probability $\epsilon > 0$ integers $a_1, \ldots, a_r$ with $h = g_1^{a_1} \cdots g_r^{a_r}$.

Then the Index Calculus Algorithm performs the following computations. First, we must find $\{x_1, \ldots, x_r\}$ such that $g^{x_i} = g_i$. In order to do this, pick $k \in \mathbb{Z}/n\mathbb{Z}$ at random. Compute g^k and feed it to our algoritm $(**)$ above. Repeat enough times to get a system of equalities

$$g^{k_j} = g_1^{a_{j_1}} \cdots g_r^{a_{j_r}}, \quad j = 1, \ldots, R, \quad R > r.$$

This gives us the system of R linear equations in the x_i's:

$$k_j \equiv a_{j_1} x_1 + \cdots + a_{j_r} x_r \bmod n$$

If we have found r linearly independent equations, then we can solve for the x_i. We hope to succeed if R is a little bit bigger than r.

Now we are ready to compute the discrete log. Given $h \in G$, pick $k \in \mathbb{Z}/n\mathbb{Z}$ at random, compute hg^{-k}, and feed it to the algorithm $(**)$ above, until you get $hg^{-k} = g_1^{a_1} \cdots g_r^{a_r}$. Then

$$h = g^m, \quad m = k + a_1 x_1 + \cdots + a_r x_r.$$

Next time, we will work on finding the specific algorithm $(**)$.

7 Diffie-Hellman Key Exchange

The Diffie-Hellman Key Exchange is an encoding scheme that relies on the difficulty of the DLP. If we could find a faster algorithm for computing the DLP, then this scheme would become drasically less effective.

7.1 The Algorithm

Let $G = \langle g \rangle$ be a cyclic group of order n. Suppose Alice and Bob want to communicate over an insecure channel. The Diffie-Hellman algorithm proceeds as follows:

- Alice chooses at random some $a \in \mathbb{Z}/n\mathbb{Z}$, keeps a a secret and sends Bob g^a.
- Bob chooses at random some $b \in \mathbb{Z}/n\mathbb{Z}$, keeps b a secret and sends Alice g^b.
- Alice can compute g^{ab} by computing $(g^b)^a$
- Bob can compute g^{ab} by computing $(g^a)^b$. Alice and Bob now have a shared secret key.

Notice that if the discrete logarithm problem is hard, then knowledge of g^a, g^b doesn't reveal the numbers a and b. Thus Alice and Bob's secret is safe.

7.2 Open Problem

One other way to crack the Diffie-Hellman code would be to find a way to compute g^{ab} knowing only g^a and g^b. i.e. without first computing a and b. An algorithm to compute this would be interesting even if it ran in $O(e^{(\log n)^\alpha})$ time, $\alpha < 1$.

8 More on the Index Calculus Algorithm

8.1 Running Time of the Index Calculus Algorithm

The average number of tries in the algorithm $(**)$ before success is $\frac{1}{\epsilon}$. We need a total of R successes before we attempt to calculate the discrete logarithms of the g_i, so it follows that we must run $(**)$ a total of $\frac{R}{\epsilon}$ times on average before success. This step is parallelizable, so we can improve the time it takes to run this algorithm by running on several machines simultaneously.

After finding our R linear equations, we must then attempt to solve them with linear algebra, however, which is not in general a parallelizable algorithm. The time it takes to perform this step is $O(R^3)$. For our purposes, we may assume that $R \sim r$ for studying the running time.

8.2 An Example

Let $G = \mathbb{F}_q^*$, $q = p^n$, p a prime (think p small and n large). [NOTE: We have changed notation - the order of our group is no longer n, but $p^n - 1$.] Then

$$\mathbb{F}_q = \mathbb{F}_p[x]/(f(x)), \quad \deg(f) = n.$$

Given an $h \in \mathbb{F}_q^*$, we can view h as the reduction of $h(x) \in \mathbb{F}_p[x]$, with $\deg(h(x)) \leq n$.

Let $B = \{$ monic irreducible polynomials of degree $\leq m\} \cup \{g_0\}$ where $\langle g_0 \rangle = \mathbb{F}_p^*$. Then

$$r = \#B \sim \frac{p^m}{m} + \frac{p^{m-1}}{m-1} + \cdots \sim cp^m$$

and we have

$$\epsilon = \frac{\#\{h \mid \deg(h) < n \text{ and } h \text{ factors as a product of polynomials of degree } \leq m\}}{p^n = \#\{\text{ polynomials of degree } < n\}}$$

We want to find a lower bound on ϵ to give us an upper bound on $\frac{1}{\epsilon}$.

Let $v = \lfloor \frac{n}{m} \rfloor$. Assume that m does not divide n so that $vm < n$. Take v irreducible polynomials of degree $\leq m$ and multiply them out to get a polynomial of degree $< n$ which factors the way we want. There are at least $\binom{r}{v}$ choices for how to do this. For our purposes, $\binom{r}{v} \sim \frac{r^v}{v!}$. Thus we get an (approximate) lower bound on ϵ:

$$\epsilon \geq \frac{r^v/v!}{p^n}.$$

We can then attempt to figure out the running time of the index calculus method in this case. We have

$$\begin{aligned}
\frac{R}{\epsilon} &\sim \frac{r}{\epsilon} \ll p^m \cdot \frac{p^n}{p^{mv}} \cdot v! \quad (r \ll p^m) \\
&= \frac{p^m p^{mv+a} v!}{p^{mv}} \ll p^{2m} v! \quad (n = mv + a, 0 \leq a < m) \\
&\ll p^{2m} e^{v \log v} \quad (v! \ll e^{v \log v}) \\
&\ll p^{2m} e^{\frac{n}{m} \log n}
\end{aligned}$$

Here we must balance the p^{2m} term (becomes large if m is large) with the $e^{\frac{n}{m}}$ term (which becomes small if m is large). It turns out that the best way

to do this is to set $m = \sqrt{n}$. In this case, we have

$$\begin{aligned}
\frac{R}{\epsilon} &\ll p^{2\sqrt{n}} e^{\sqrt{n}\log n} && (9)\\
&\ll e^{\sqrt{n}\log n + 2\sqrt{n}\log p} && (10)\\
&\ll e^{(n\log p)^{\left(\frac{1/2+\delta}{1+\delta}\right)}} && (11)\\
&\sim e^{(\log|\mathbb{F}_q^*|)^{\alpha}} && (12)
\end{aligned}$$

Where we get from (2) to (3) by assuming that p is small in the sense that $\log p < n^{\delta}$. It follows that if p is small, then we can take $\log p = n^{\delta}$, and hence $\sqrt{n}\log p = n^{1/2+\delta} = (n\log p)^{\alpha} = n^{(1+\delta)\alpha} \Longrightarrow \alpha = \frac{1/2+\delta}{1+\delta}$. We get from step (3) to step (4) by assuming that $\log p \leq n^{\delta}$.

In practice, the performance of the linear algebra computation will always dominate the running time.

8.3 Another Example

Let $G = \mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$. Then for $h \in \mathbb{F}_p^*$, we can lift h to $\mathbb{Z}$, with $1 \leq h \leq p-1$. Our factor base B is the set of primes $l \leq x$. In other words, we're hoping that h factors as a product of small primes. Then, by the Prime Number Theorem, we have

$$\#B = r \sim \frac{x}{\log x}$$

Let $v = \lfloor \frac{\log p}{\log x} \rfloor$. Then as before we have at least $\binom{r}{v}$ choices that factor as a product of small primes less than x (i.e. factor as we want them to). Thus we have

$$\#\{h < p \mid h \text{ is a product of primes } \leq x\} \geq \binom{r}{v}$$

So we must have as before that (approximately)

$$\epsilon \geq \frac{r^v/v!}{p}.$$

For comparison to the previous example, $\log p = n$ and $\log x = m$. So now we must choose $\log x = \sqrt{\log p}$, or $x = e^{\sqrt{\log p}}$.

The running time of this algorithm is $e^{(\log p)^{\alpha}}$, $\alpha < 1$ (in fact, α is close to $\frac{1}{2}$).

For p^n where p is big and n is small, take an extension $K/\mathbb{Q}$ of degree n where p is inert and unramified so that

$$\mathcal{O}_k/(p) \cong \mathbb{F}_{p^n}$$

and proceed along similar lines. (This is in fact never done in practice for $n > 3$.)

8.4 Other examples

The function field sieve is another way of approaching the index calculus algorithm for $\mathbb{F}_q$. Instead of working in $\mathbb{F}_p[x]$, work in a ring of integers on an extension of $\mathbb{F}_p(x)$, and use the fact that there may be more prime divisors of low degree and use them as the factor base.

Suppose $C/\mathbb{F}_q$ is an algebraic curve of genus $g \geq 1$. Then the Jacobian $J_C(\mathbb{F}_q)$ is an abelian group. Thus we can do index calculus with the factor base

$$B = \{\text{positive divisors of small degree}\}$$

which ends up working very well when g is very big.

If $G = \mathbb{F}_q^*$, $q = r^m$, then $\{x \in \mathbb{F}_q^* \mid N_{\mathbb{F}_q/\mathbb{F}_r} = 1\} = G$, and

$$|G| = \frac{q-1}{r-1}, \quad \text{(i.e. we've made } G \text{ smaller)}$$

It turns out however that it is currently not known whether the DLP is any easier on this smaller group than it is on the larger one!

9 Primality Testing

A primality testing algorithm takes an integer n as input, and decides whether n is prime or not. (Note: It does not necessarily give factors - this is a different (and harder) problem.)

We will eventually present the AKS algorithm, which is a deterministic, polynomial time primality test.

Many primality tests (including AKS) are based on the following principle: If $\mathbf{R}$ is a ring of prime characteristic n, then $(x+y)^n = x^n + y^n \; \forall x, y \in \mathbf{R}$.

An example of this is in Fermat's Little Theorem: If n is prime, then $a^n \equiv a \pmod{n} \; \forall a \in \mathbb{Z}$. This follows from the principle because $(1 + 1 + \ldots + 1)^n = 1^n + 1^n + \ldots + 1^n = a$ in $\mathbb{Z}/n$.

9.1 A pseudoprimality test

For random a: test if $a^n \equiv a \pmod{n}$.

If not, then n is composite.

If yes, then we don't know.

By repeating this test, we can usually know if n is probably a prime, but there are cases where it will never work: A Carmichael number is a composite integer n for which $a^n \equiv a \pmod{n}$ $\forall a, (a,n) = 1$. For example, $561 = 3 * 11 * 17$ is a Carmichael number, and there are infinitely many such numbers.

9.2 Legendre and Jacobi symbols

The Legendre symbol is defined for p an odd prime as follows:

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{if } a \text{ is a square mod } p, a \neq 0 \\ -1 & \text{if } a \text{ is not a square mod } p, a \neq 0 \\ 0 & \text{if } a = 0 \end{cases}$$

The Jacobi symbol is defined for $a, n \in \mathbb{Z}$, n odd, $n = p_1^{\alpha_1} \cdots p_r^{\alpha_r}$ by

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right)^{\alpha_1} \cdots \left(\frac{a}{p_r}\right)^{\alpha_r}$$

where $\left(\frac{a}{p_i}\right)$ is the Legendre symbol.

Note: $\left(\frac{ab}{n}\right) = \left(\frac{a}{n}\right)\left(\frac{b}{n}\right)$, and if $a \equiv b \bmod n$ then $\left(\frac{a}{n}\right) = \left(\frac{b}{n}\right)$.

The definition is not useful for computing, because it requires factoring. Instead we use Quadratic Reciprocity: If a, n are odd and $(a,n) = 1$, then

$$\left(\frac{a}{n}\right)\left(\frac{n}{a}\right) = (-1)^{\frac{a-1}{2}\frac{n-1}{2}}$$

.

We can use this to compute Jacobi symbols by repeatedly factoring out powers of 2 and reducing:

$$\left(\frac{a}{n}\right) = \left(\frac{n}{a}\right)(-1)^{\frac{a-1}{2}\frac{n-1}{2}} = \pm\left(\frac{n\%a}{a}\right)$$

If we iterate, the convergence is polynomial: It will take O(log n) steps because after two steps, the numbers will get halved.

9.3 Solovay-Strassen Primality Test

If n is an odd prime and $(d,n) = 1$, then let $\mathbf{R} = (\mathbb{Z}/n[x])/(x^2 - d)$.

Then by the earlier principle,

$$(a + b\sqrt{d})^n = a^n + b^n\sqrt{d}^n = a + bd^{\frac{n-1}{2}}\sqrt{d} = a \pm b\sqrt{d}$$

.

(Note that $d^{\frac{n-1}{2}} = \left(\frac{d}{n}\right) \bmod n$ since n is prime.)

So the test is as follows: Test for a random d whether $d^{\frac{n-1}{2}} \equiv \left(\frac{d}{n}\right)$. We can simply compute each side. If they are not equal, then n is composite. If they are equal, we don't know, but we don't have any Carmichael-like problems.

Theorem 6. *If n is odd and composite, then $G = \{d \in (\mathbb{Z}/n)^* : \left(\frac{d}{n}\right) \equiv d^{\frac{n-1}{2}} \bmod n\} \neq (\mathbb{Z}/n)^*$*

Remark: G is a subgroup of $(\mathbb{Z}/n)^*$, because both sides are multiplicative. So if $G \neq (\mathbb{Z}/n)^*$, then $|G| \leq \frac{1}{2}|(\mathbb{Z}/n)^*|$. Thus at least half of the d's will show that n is composite in the Solovay-Strassen test. So it follows from the theorem that this test is a probabilistic polynomial time primality test.

In fact, if GRH is true and n is composite, then $\exists\, d \notin G, 1 \leq d \leq 4(\log n)^2$, so this would be a polynomial time determinstic test by testing those d's.

Proof. Suppose by contradiction that n is composite and $G = (\mathbb{Z}/n)^*$. $\forall a \in (\mathbb{Z}/n)^*, a^{n-1} = (a^{\frac{n-1}{2}})^2 \equiv \left(\frac{a}{n}\right)^2 = 1 \bmod n$. Thus n is Carmichael, so n is squarefree.

Then we can write $n = pr$, p prime, $p \nmid r, r > 1$. Let c be a quadratic nonresidue mod p. Find a s.t.

$$a \equiv c \bmod p$$

$$a \equiv 1 \bmod r$$

We know this exists by CRT. Now

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p}\right)\left(\frac{a}{r}\right) = \left(\frac{c}{p}\right)\left(\frac{1}{r}\right) = (-1)(1) = -1$$

Thus if $a \in G$, then $a^{\frac{n-1}{2}} \equiv -1 \bmod n$, so $a^{\frac{n-1}{2}} \equiv -1 \bmod r$, so $1 \equiv -1 \bmod r$, contradiction. Thus $a \notin G$, but this contradicts our original assumption. □

Note: The Miller-Rabin test is similar - it uses $d^{\frac{n-1}{2^r}}$, and is stronger although not as mathematically pretty.

Suppose $\left(\frac{d}{n}\right) = -1$. If n is prime, then

$$(a + b\sqrt{d})^n = a - b\sqrt{d}$$

$$(a + b\sqrt{d})^{n+1} = (a - b\sqrt{d})(a + b\sqrt{d}) = a^2 - db^2 \in \mathbb{Z}/n$$

Thus $(a + b\sqrt{d})^{n+1} - (a - b\sqrt{d})^{n+1} = 0$. This is the Lucas-Lehmer Test (which is usually phrased in terms of linear recurrences). A version of this test can be set up by taking say $a = b = 1$ and d minimal with $\left(\frac{d}{n}\right) = -1$. No composite number has been observed to pass this test, although such a number should exist.

Consider the case of Mersenne primes: If $n = 2^l - 1, l$ prime, then $\left(\frac{3}{n}\right) = -1$. So we can use the Lucas-Lehmer test with $a = 2, b = 1$:

Define $S_k = (2 + \sqrt{3})^{2^k} + (2 - \sqrt{3})^{2^k}, S_0 = 4$. (Note that $S_{k+1} = S_k^2 - 2$, so it's easy to compute these mod n.) Then $n = 2^l - 1$ is prime $\iff S_{l-2} \equiv 0 \mod n$.

To prove this: Proving $\Rightarrow$ comes from the identity above. The idea behind proving $\Leftarrow$ is that $(\frac{2+\sqrt{3}}{2-\sqrt{3}})^{2^{l-2}} \equiv -1 \mod n$ so the order of $\frac{2+\sqrt{3}}{2-\sqrt{3}}$ in $\mathbf{R}^*$ is 2^{l-1}, so $\mathbf{R}$ must be a field, so n is prime.

9.4 AKS Preliminaries

If $\mathbf{R}$ is a ring of prime characteristic n, then $(x+y)^n = x^n + y^n$, for $x, y \in \mathbf{R}$.

Proposition 7. *If, for some $a \in (\mathbb{Z}/n)^*, (x + a)^n = x^n + a^n$ in $\mathbb{Z}/n[x]$ then n is prime.*

Remark: Computing $(x + a)^n$ is hard (order n), so this is not a useful algorithm.

To prove this proposition, we will use the following lemma:

Lemma. (Lucas Lemma) *Let $n = a_0 + a_1 p + \ldots + a_s p^s$, and $m = b_0 + b_1 p + \ldots + b_s p^s$, where $0 \le a_i, b_i \le p$. Then*

$$\binom{n}{m} \equiv \binom{a_0}{b_0} \cdots \binom{a_s}{b_s} \ mod\ p$$

In particular, $\binom{n}{m} \not\equiv 0 \ mod\ p \iff b_i \le a_i \ \forall i$.

Proof. (Lucas Lemma) $\binom{n}{m}$ is the coefficient of x^m in $(x+1)^n$. OTOH,

$$\begin{aligned}
(x+1)^n &= (x+1)^{a_0+a_1p+\ldots+a_sp^s} \\
&= \prod_{i=0}^{s}(x+1)^{a_ip^i} \\
&= \prod_{i=0}^{s}(x^{p^i}+1)^{a_i} \text{ in } \mathbb{F}_p[x] \\
&= \prod_{i=0}^{s}\left(\sum_{j=0}^{a_i}\binom{a_i}{j}x^{p_ij}\right) \\
&= \sum_{j_0,\ldots,j_s}\binom{a_0}{j_0}\cdots\binom{a_s}{j_s}x^{j_0+pj_1+\ldots+p^sj_s}
\end{aligned}$$

Since $0 \le j_i \le a_i < p, j_0 + j_1p + \ldots + j_sp^s = m \iff j_i = b_i\ \forall i$. Thus $\binom{n}{m} = \binom{a_o}{b_0}\cdots\binom{a_s}{b_s}$ in $\mathbf{F}_p$. □

Proof. (Proposition) Suppose n is composite. Assume first that n is not a prime power. Let p be a prime, $p^r|n, p^{r+1} \nmid n$. Then $n/p^r \neq 1$. We'll show that $\binom{n}{p^r} \not\equiv 0 \bmod p$, so $\binom{n}{p^r} \not\equiv 0 \bmod n$, so $(x+a)^n$ has a nonzero coefficient in x^{n-p^r} and thus cannot be $x^n + a^n$.

Now $n = a_rp^r + a_{r+1}p^{r+1} + \ldots, a_r \neq 0$ and $p^r = 1 * p^r + 0$. Thus by Lucas Lemma, $\binom{n}{p^r} \equiv \binom{a_r}{1} \equiv a_r \neq 0 \bmod p$.

Suppose instead that $n = p^r$ for some prime $p, r \ge 2$. Then

$$(x+a)^{p^r} = \sum\binom{p^r}{j}a^jx^{p^r-j}$$

$$\binom{p^r}{p} = \frac{p^r(p^r-1)\cdots(p^r-p+1)}{p\cdots 1}$$

The only terms not prime to p in this are p^r in the numerator and p in the denominator, so $\binom{p^r}{p} = p^{r-1} * u$, where $p \nmid u$, so it cannot be 0 mod p^r. □

Theorem 8. *Suppose n is not a prime power and $r < n$ is a prime, $r \nmid n$, such that the order of n in $(\mathbb{Z}/r)^*$ is at least $4(\log n)^2$. Then $\exists a, 1 \le a \le 2\sqrt{r}\log n$ such that $(x+a)^n \neq x^n + a$ in $(\mathbb{Z}/n[x])/(x^r-1)$.*

9.5 AKS Primality Test

Input n odd.

1. If $n = a^b, b \geq 2$, then output that n is composite.

   To do this, simply take the bth root of n numerically as a real number and check if it's an integer. We only need to check for values of b up to $\log n$.

2. Find the smallest r prime, $r < n, r \nmid n$, such that the order of n in $(\mathbb{Z}/r)^*$ is at least $4(\log n)^2$.

3. Test for $a = 1, \ldots, \lfloor 2\sqrt{r} \log n \rfloor$ whether $(x + a)^n = x^n + a$ in $(\mathbb{Z}/n[x])/(x^r - 1) = \mathbf{R}$.

   (a) If yes for all a, then n is prime.

   (b) If no for some a, then n is composite.

   (c) If no such r exists, then n is prime.

For step 3, we have a ring with n^r elements. Computing $(x + a)^n$ in $\mathbf{R}$ takes $O((r \log n)^c)$ steps. So for the algorithm to be polynomial time in $\log n$, we must have $r = O((\log n)^c)$ for some c. Note that $r > 4(\log n)^2$ from step 2.

Lemma. *If $n \in \mathbb{Z}, \exists$ prime $r, r \nmid n$, such that the order of n in $(\mathbb{Z}/r)^*$ is at least $4(\log n)^2$ and $r = O((\log n)^6)$.*

Proof. Let $M = n \prod_{j=1}^{\lfloor 4(\log n)^2 \rfloor} (n^j - 1)$

If r is prime, $r|n$, then $r|M$. If $r \nmid n$ and the order of n in $(\mathbb{Z}/r)^*$ is less than $4(\log n)^2$ then $r|M$.

What we want is a prime $r, r \nmid M$. Claim: at most $\log_2 M$ primes divide M. This is clear. (Note that $M = p_1^{\alpha_1} \cdots p_k^{\alpha_k} \geq 2^k$.)

Then

$$\begin{aligned} \log M &\leq \log n + \sum_j j \log n \\ &\leq \log n \left(1 + \sum_{j=1}^{4(\log n)^2} 4(\log n)^2 \right) \\ &\leq 5(\log n)^2 * 4(\log n)^2 * \log n = O((\log n)^5) \end{aligned}$$

□

Thus step 2 is ok, and the algorithm is polynomial in $\log n$.

Theorem 9.1. *Let $n \geq 1$ be an odd integer and r a prime, $r \nmid n$, such that the order of n in $\left(\mathbb{Z}/r\right)^\star$ is at least $4(\log n)^2$ and such that $(x+a)^n = x^n + a$ in $R_n := \mathbb{Z}/n[x]/(x^r - 1)$ for all a with $1 \leq a \leq 2\sqrt{r}(\log n)$.*
Then n is a prime power.

Proof. We will proceed via a series of Lemmas.

Let $p \mid n$ be prime and set $l = \lfloor 2\sqrt{r}\log n \rfloor$, and $R_p := \mathbb{Z}/p[x]/(x^r - 1)$ then we have $(x+a)^n = x^n + a$ in R_n for $a = 1, \ldots, l$. Let

$$I := \{m \in \mathbb{N} \mid (x+a)^m = x^m + a \text{ in } R_p \ \forall a = 1, \ldots, l\}$$

then $1, p, n \in I$. We aim to show that I consists of the powers of p.

Lemma 9.2. *If $m, m' \in I$ then $mm' \in I$ (I is a multiplicative semigroup).*

In R_p we have $(x+a)^{mm'} = (x^m + a)^{m'}$. On the other hand

$$(x+a)^{m'} - x^{m'} - a = (x^r - 1)u(x) \quad \text{in} \quad \mathbb{Z}/p[x]$$

Replacing x with x^m yields

$$(x^m+a)^{m'} - x^{mm'} - a = (x^{mr}-1)u(x) = (x^r-1)(1+x^r+\cdots+x^{r(m-1)})u(x^m) = 0 \quad \text{in} \quad R_p$$

So $(x^m + a)^{m'} = x^{mm'} + a$ in R_p and $mm' \in I$ ■

Define I_0 to be the group generated by p and n in $\left(\mathbb{Z}/r\right)^\star$ and let $t = |I_0|$. Then $t \geq 4(\log n)^2$ since t was assumed to be the order of n in $\left(\mathbb{Z}/r\right)^\star$.

Lemma 9.3. *If $1 \leq a_1, \ldots, a_k \leq l$ and $f(x) = \prod_{i=1}^{k}(x + a_i) \in \mathbb{Z}/p[x]$ then $f(x)^m = f(x^m)$ in R_p for all $m \in I$.*

We proceed via induction on k. The base case, $k = 1$, is clear from the definition of I. Now assume that $f_k(x) = f_{k-1}(x)(x + a_k)$ where the result holds for f_{k-1}. Then in R_p

$$f_k(x)^m = f_{k-1}(x)^m(x + a_k)^m = f_{k-1}(x^m)(x^m + a_k) = f_k(x^m)$$

■

For the next two Lemmas we introduce the following notation:
Let $h(x)$ be an irreducible factor of $\frac{x^r-1}{x-1}$ in $\mathbb{Z}\!\!\diagup\!\!_p[x]$, and ζ a root of h in $\overline{\mathbb{Z}\!\!\diagup\!\!_p}$. Also let $F := \mathbb{Z}\!\!\diagup\!\!_p(\zeta) = \mathbb{Z}\!\!\diagup\!\!_p[x]\!\!\diagup\!\!_{(h(x))}$ be a finite field with $G := \langle\, \zeta + a \;\mid\; 1 \le a \le l \,\rangle \subseteq F^\star$.

Lemma 9.4. $|G| \ge \binom{t+l+1}{t-1}$

Let $f(x) = \prod_{i=1}^{k}(x + a_i)$, $g(x) = \prod_{i=1}^{k'}(x + b_i)$, $1 \le a_i, b_i \le l$; $k, k' \le t - 1$. We will show that $f(\zeta) \ne g(\zeta)$ in F so they yield distinct elements of G and so the size of G is bigger than the number of polynomials of the same form as f.

If $f(\zeta) = g(\zeta)$ then for all $m \in I_0$ we have

$$f(\zeta^m) \overset{9.3}{=} f(\zeta)^m = g(\zeta)^m \overset{9.3}{=} g(\zeta^m)$$

so the polynomial $f(x)-g(x)$ has roots ζ^m, $m \in I_0$ (since $\zeta^r = 1$, we are working mod r) and these are all distinct so $f - g$ has at least t roots. But $\deg(f-g) \le \max(k, k') \le t-1$ so $f(x) = g(x)$, thus distinct polynomials give distinct elements of G. ■

Lemma 9.5. *If n is not a power of p then $|G| \le \frac{1}{2}n^{2\sqrt{t}}$*

Let $J := \{n^i p^j \mid 0 \le i, j \le \lfloor\sqrt{t}\rfloor\} \subseteq I$ (recall 9.3). If n is not a power of p then all these terms are distinct and $|J| \ge (\lfloor\sqrt{t}\rfloor + 1)^2 > t$.

Now $\exists m, m' \in J$ with

$$m \equiv m' \bmod r$$

because the image of J in $\left(\mathbb{Z}\!\!\diagup\!\!_r\right)^\star$ is contained in I_0 but $|I_0| = t < |J|$ so reduction mod$r$ cannot be injective.

Again, let $f(x) = \prod_{i=1}^{k}(x + a_i)$, $1 \le a_i \le l$ and set $g = f(\zeta) \in G$. Then

$$g^m = f(\zeta)^m \underset{m \in J \subseteq I}{=} f(\zeta^m) \underset{\substack{m \equiv m' \bmod r, \\ \zeta^r = 1}}{=} f(\zeta^{m'}) = f(\zeta)^{m'} = g^{m'}$$

so

$$G \subseteq \{ \text{ roots of } x^m - x^{m'} \} \subseteq \bar{F}$$

and $|G| \le \max(m, m') \le (np)^{\sqrt{t}} \le \frac{1}{2}n^{2\sqrt{t}}$. ■

Finally we return to the proof of the main theorem:

If n is not a power of p then by Lemma 9.5

$$|G| \leq \frac{1}{2} n^{2\sqrt{t}}$$

on the other hand Lemma 9.4 gives

$$|G| \geq \binom{t+l+1}{t-1}.$$

Recall $\sqrt{t} \geq 2\log n \implies t \geq 2\sqrt{t}\log n$ and $l = \lfloor 2\sqrt{r}\log n \rfloor \geq 2\sqrt{t}\log n$ since $I_0 \leq \left(\mathbb{Z}/r\right)^\star$, so

$$|G| \geq \binom{t+l+1}{t-1} \geq \binom{l+1+\lfloor 2\sqrt{t}\log n \rfloor}{\lfloor 2\sqrt{t}\log n \rfloor} \geq \binom{\lfloor 4\sqrt{t}\log n \rfloor}{\lfloor 2\sqrt{t}\log n \rfloor} > \frac{1}{2} n^{2\sqrt{t}}$$

by Stirling's formula. Thus n is a prime power. ■

Conjecture 9.6. *If r is an odd prime which does not divide the odd n and*

$$(x+1)^n = x^n + 1 \quad \text{in} \quad R_n$$

then either n is prime of $n^2 \equiv 1 \pmod{r}$

This seems to be too strong according to the counter conjecture

Conjecture 9.7. *For any prime $r \geq 5$ $\exists n$ such that $r \nmid n(n^2-1)$ and*

$$(x+1)^n = x^n + 1 \quad \text{in} \quad R_n$$

What is the smallest n for a given r? Denote it by $n_0(r)$. Now

$$(x+1)^n = x^n + 1 \,\text{in}\, R_n \quad \iff (x+1)^n \equiv x^n + 1 \bmod x^r - 1 \,\text{in}\, \mathbb{Z}/r[x]$$

If $r > n$ this implies

$$(x+1)^n = x^n + 1 \,\text{in}\, \mathbb{Z}/n[x] \quad \implies \quad n \text{ is prime}$$

So $n_0(r) > r$. (numerical evidence shows $n_0(5)$ is large)

10 Polynomial Reconstruction

Let $x_1, x_2, \ldots, x_n$ be distinct elements of the field F, and $t, d \geq 1$ integers.

Problem P Given $y_1, \ldots, y_n \in F$, find all polynomials $f(x) \in F[x]$, $\deg f \leq d$ such that $f(x_i) = y_i$ for at least t values of i.

[*cf* Reed-Solomon codes]

Now if $f(x) \neq g(x)$ are polynomials of degree $\leq d$ then $\deg(f(x)-g(x)) \leq d$ so $f(x_i) = g(x_i)$ for at most d values of i.

When is there at most one solution?

Proposition 10.1. *The Problem P has at most one solution $f(x) \in F[x]$ if $2t - n > d$.*

Consider $I, J \subseteq \{1, 2, \ldots, n\}$ of size $\geq t$, then if

$$f(x_i) = y_i \text{ for } i \in I \qquad g(x_i) = y_i \text{ for } i \in J$$

then $f(x_i) = g(x_i)$ for $i \in I \cap J$ so $f = g$ if $|I \cap J| > d$ but

$$|I \cap J| = |I| + |J| - |I \cup J| \geq 2t - |I \cup J| \geq 2t - n.$$

So if $2t - n > d$ the problem has at most one solution. ■

If $t = n$, that is we need all $f(x_i) = y_i$, and $n > d$ then we have at most one solution, and Lagrange interpolation using the first $d + 1$ elements

$$f(x) = \sum_{j=1}^{d+1} y_j \prod_{i=1, i \neq j}^{d+1} \frac{x - x_i}{x_j - x_i}$$

gives a candidate which can then be checked to see if $f(x_i) = y_i$ for the elements $i > d + 1$.

An approach to problem P when $t \geq d + 1$ would be to take every $I \subseteq \{1, 2, \ldots, n\}$ with $|I| = d + 1$ and compute the unique f_I of degree $\leq d$ with $f(x_i) = y_i$ for $i \in I$ and then compute the $f_I(x_i)$ for $i \notin I$; keeping those f_I for which $f_I(x_i) = y_i$ for at least $t - (d + 1)$ values of $i \notin I$.
This process requires one to investigate $\binom{n}{d+1}$ polynomials.

There is an algorithm due to Berlekamp and Massey that when $2t-n > d$ either decides there is no solution or produces the unique solution efficiently (in finite fields).

The general reconstruction problem is as follows. Let F be a field (not necessarily finite). Given two sequences $\{x_1, \ldots, x_n\} \in F$ and $\{y_1, \ldots, y_n\} \in$

F, we want to find a polynomial $f(x) \in F[x]$ such that $f(x_i) = y_i$ for "enough" values of i.

The notable difference here between interpolation and reconstruction is that we do not specify the particular values of i for which $f(x_i) = y_i$ and by doing so we avoid the $\binom{n}{\deg f}$ running time of the naïve algorithm which interpolates every possible subset.

10.1 Specialized Polynomial Reconstruction

By restricting the degree of $f(x)$ and setting a lower bound on the number of values of i for which $f(x_i) = y_i$, we can narrow the set of possible solutions down to either one or zero solutions and obtain a simple algorithm to find a solution if it exists. Specifically, given a field F, $x_1, \ldots, x_n \in F$ distinct and $y_1, \ldots, y_n \in F$ choose a $k \in \mathbb{N}, k < n$ and seek f such that

1. $\deg f(x) < k$
2. $\#\{i : f(x_i) \neq y_i\} \leq \frac{n-k}{2}$

then the algorithm will either give a unique $f(x)$ which satisfies the conditions or will signal that no such $f(x)$ exists.

10.2 Algorithm for Polynomial Reconstruction

To find an $f \in F[x]$ which satisfies the above restrictions, this algorithm finds polynomials $E, N \in F[x]$ such that

1. E monic with $\deg E \leq \frac{n-k}{2}$
2. $\deg N \leq \frac{n+k}{2} - 1$
3. $N(x_i) = y_i E(x_i) \ \forall \ i = 1, \ldots, n$

If $E \mid N$ and $\deg N/E < k$, then N/E is a polynomial of degree $< k$ for which $N(x_i)/E(x_i) \neq y_i$ for at most $\deg E \leq \frac{n-k}{2}$ values. Hence $f = N/E$ is a solution to the original problem.

If $E \not| \ N$ or $\deg N/E \geq k$, then (as we shall see later), there is no solution to the original problem.

10.2.1 Description of Algorithm

To find $N, E \in F[x]$, note that condition (3) implies that

$$N(x_i) = \sum_{j=0}^{\frac{n+k}{2}-1} n_j x_i^j = y_i \sum_{j=0}^{\frac{n-k}{2}} e_j x_i^j = y_i E(x_i)$$

for all (x_i, y_i). But this is a linear system of n equations in $\frac{n+k}{2} + \frac{n-k}{2} + 1 = n+1$ unknowns, hence there is at least one solution and, by dividing by the leading nonzero coefficient of E, we can assume that E is monic.

Note also that there is a solution with $E \neq 0$ since, if $E = 0$, then $N = 0$ because deg $N < n$ and $N(x_i) = 0$ for all $1 \leq i \leq n$.

10.2.2 Proof of Uniqueness

Although the solution to the linear system above is not unique, we show that N/E is unique.

Lemma. *If N_1, E_1 and N_2, E_2 are two sets of solutions to conditions (1),(2), and (3) above with $E_i \mid N_i$ then $N_1/E_1 = N_2/E_2$.*

Proof. Note that, by condition (3), $(N_1(x_i)E_2(x_i) - N_2(x_i)E_1(x_i))y_i = 0$ for all i. If $y_i \neq 0$, we get $N_1(x_i)E_2(x_i) - N_2(x_i)E_1(x_i) = 0$. If $y_i = 0$, we get $N_1(x_i) = N_2(x_i) = 0$ and it follows that, again, $N_1(x_i)E_2(x_i) - N_2(x_i)E_1(x_i) = 0$. Since deg $(N_1E_2 - N_2E_1) \leq \frac{n-k}{2} + \frac{n+k}{2} - 1 = n - 1$ we conclude that $N_1E_2 = N_2E_1$ and so $N_1/E_1 = N_2/E_2$ as required. □

10.2.3 Proof of Sufficiency

We need to know whether the algorithm will always produce a solution if a solution to the original problem exists. Suppose f is a solution to the original problem. Define E as

$$E(x) := \prod_{\substack{i=1 \\ f(x_i) \neq y_i}}^{n} (x - x_i)$$

and define $N = fE$ so that $N/E = f$.

Now we check that N, E satisfy the three conditions. Since $\#\{i : f(x_i) \neq y_i\} \leq \frac{n-k}{2}$ we have deg $E \leq \frac{n-k}{2}$ and E is monic by construction so condition (1) holds. Furthermore, since deg $f < k$ and deg $E \leq \frac{n-k}{2}$ we have that deg $N =$ deg $f +$ deg $E \leq k - 1 + \frac{n-k}{2} = \frac{n+k}{2} - 1$ and so condition (2) holds.

Finally, $N(x_i) = f(x_i)E(x_i)$ so $N(x_i)$ is equal to $y_iE(x_i)$ when $f(x_i) = y_i$. If $f(x_i) \neq y_i$ then $E(x_i) = N(x_i) = 0$ so again $N(x_i)$ is equal to $y_iE(x_i)$.

By the previous section, we know that for any N, E which satisfy the three conditions and for which $E \mid N$, then N/E is unique. Hence if the algorithm does not find a suitable N, E, then no such pair exists and so there is no solution to the original problem.

10.3 Reconstruction Example

To see how the algorithm works in detail, let us consider the relatively simple case where $n = 3$ and $k = 1$. Then we are looking for a constant polynomial $f \in F$ such that $f = y_i$ for at least two values of i. Clearly, this can only occur when two of the y_i are the same.

The above algorithm finds polynomials $N(x) = n_0 + n_1x$ and $E(x) = e_0 + e_1x$ such that $N(x_i) = y_iE(x_i)$. For simplicity, let $x_i = i$ and then, to find the coefficients n_0, n_1, e_0, e_1 we must solve the following equation

$$\begin{pmatrix} 1 & 1 & -y_1 & -y_1 \\ 1 & 2 & -y_2 & -2y_2 \\ 1 & 3 & -y_3 & -3y_3 \end{pmatrix} \begin{pmatrix} n_0 \\ n_1 \\ e_0 \\ e_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

which has the general solution

$$\begin{pmatrix} n_0 \\ n_1 \\ e_0 \\ e_1 \end{pmatrix} = c \begin{pmatrix} 3y_1y_2 - 4y_1y_3 + y_2y_3 \\ -y_1y_2 + 2y_1y_3 - y_2y_3 \\ -y_1 + 4y_2 + 3y_3 \\ y_1 - 2y_2 + y_3 \end{pmatrix}$$

for any $c \in F$ except when $y_1 = y_2 = y_3$ in which case a solution is $N(x) = y_1x$ and $E(x) = x$ or $N(x) = y_1$ and $E(x) = 1$.

To see what happens when two of the y_i are the same, suppose that $y_1 = y_2 \neq y_3$, then

$$\begin{aligned} N(x) &= 3y_1(y_1 - y_3) + y_1(y_3 - y_1)x \\ E(x) &= 3(y_1 - y_3) + (y_3 - y_1)x \\ N(x)/E(x) &= y_1 \end{aligned}$$

and so $N/E = y_1$ is a solution to the original problem.

To see what happens when all the y_i are distinct, note that the general case is rather complicated so take $y_1 = 1, y_2 = 2, y_3 = 4$ to be an illustrative

example. Then $N(x) = -2 - 2x$ and $E(x) = -5 + x$ so that $N(1) = E(1) = -4$, $N(2) = 2E(2) = -6$ and $N(3) = 4E(3) = -8$, but $E \not| N$. In this case, the algorithm signals that there is no solution.

10.4 Sudan's Algorithm

Theorem 10.2. *Sudan's algorithm solves problem P in polynomial time if $t > 2\sqrt{nd}$ and shows (by generating them) that there are at most $\sqrt{\frac{n}{d}}$ solutions in this case.*

We can summarize the different regimes as follows:

		e.g. $n = 16d$
$t > \frac{n+d}{2}$	at most one solution	$t > 17d/2$
$t > 2\sqrt{nd}$	few solutions	$t > 8d$, at most 3 solutions
	$\vdots$	
$t = d + 1$	$\binom{n}{d+1}$	$\binom{16d}{d+1} \gg c^d$ solutions

The idea of Sudan's approach is

> Construct a curve through all the points (x_i, y_i), then a polynomial passing through many of the points will intersect this special curve numerous times and so by Bezout's theorem must share a common factor with this curve. The list of candidate polynomials will be constructed from the components of the curve.

Set $D = \lfloor \sqrt{nd} \rfloor$ and $l = \lfloor \sqrt{\frac{n}{d}} \rfloor$.

We want to a construct $0 \neq Q(x, y) \in F[x, y]$ such that $Q(x_i, y_i) = 0$ for $i = 1, \ldots, n$ and $\deg_x Q \leq D$ and $\deg_y Q \leq l$. Write

$$Q(x, y) = \sum_{i=0}^{n} \sum_{j=0}^{l} a_{ij} x^i y^j \quad \text{for some } a_{ij} \in F$$

then the conditions $Q(x_i, y_i) = 0$ for $i = 1, \ldots, n$ give linear equations in the a_{ij}.

We have n equations in

$$(D + 1)(l + 1) \quad > \sqrt{nd}\sqrt{\frac{n}{d}} = n$$

unknowns so a non-zero solution exists.

Suppose $f(x) \in F[x]$ has degree at most d and $f(x_i) = y_i$ for $i \in I \subseteq \{1, \ldots, n\}$ with $|I| = t$. Then for $i \in I$

$$Q(x_i, f(x_i)) = Q(x_i, y_i) = 0$$

so $Q(x, f(x))$ has t zeros, and as a polynomial in x,

$$\deg Q(x, f(x)) \leq D + ld \leq \sqrt{nd} + \sqrt{\frac{n}{d}}d = 2\sqrt{nd} \underbrace{\leq t}_{\text{by hypothesis}}$$

so $Q(x, f(x))$ is identically zero.

This means that any solution $f(x)$ to problem P under these hypotheses satisfies $Q(x, f(x)) \equiv 0$ so $y - f(x) \mid Q(x, y)$ or

$$Q(x, y) = \prod_{k=1}^{K} \underbrace{(y - f_k(x))}_{\text{irred. in } F[x,y]} Q_1(x, y)$$

where the solutions to problem P must be one of the f_k.

Since $\deg_y Q \leq l$ we get $K \leq l \leq \sqrt{\frac{n}{d}}$.

To exhibit the solutions mentioned in the theorem we need to learn how to factor $Q(x, y)$ as a product of irreducibles.

Example 10.3. *Let $\mathbb{F}_4 = \{0, 1, \omega, 1 + \omega\}$, and set $d = 1$ so we want linear polynomials $f(x) = ax + b$, and $n = 5$ (the fifth value will be the slope [value at ∞]).*

Suppose you are given $y_1, \ldots, y_5$; construct Q as a quadratic with the fifth condition that the line $y = y_5 x$ is asymptotic to $\{Q = 0\}$.

A conic has 6 coefficients to match the 5 conditions allowing one to determine 1 solution.

If

- *Q is irreducible, there are no solutions.*
- *$Q = \text{constant} \cdot (y - f_1)^2$, there is one $f = f_1$*
- *$Q = \text{constant} \cdot (y - f_1)(y - f_2)$, there are two solutions $f = f_1, f_2$*

As long as $t \geq 3$ then the solutions to P are factors of Q [A conic meets a line in at most 2 points unless the conic contains the line]

Let F be a field and consider $Q(x, y) \in F[x, y] \setminus F$. Our task is to factor $Q(x, y)$ into a product of irreducible polynomials.

Remark.

1. $F[x, y]$ is a UFD.

2. $Q(x, y)$ may be irreducible over F but may factor over some extension field L/F.

Factoring a Polynomial in Two Variables

Step 0. Write $Q(x, y) = \sum q_j(x)y^j$, compute the gcd of the $q_j(x)$, and factor it out. WLOG $\gcd(q_j(x))$=1. So, content$(Q) = 1$ as a polynomial in y with coefficients in $F[x]$.

Step 1. Regard Q as a polynomial in y with coefficients in $F(x)$. Take $\gcd(Q, \frac{\partial Q}{\partial y})$ to remove multiple factors.

Combining the previous two steps we can assume that Q has no factors in $F[x]$, and is square-free.

Step 2. Find $(x_0, y_0) \in \overline{F}^2$ with

$$(*) \qquad Q(x_0, y_0) = 0 \text{ and } \frac{\partial Q}{\partial y}(x_0, y_0) \neq 0.$$

The purpose of this is to find a point on the plane curve $Q = 0$ which is non-singular and such that the tangent line at this point is not vertical. This will allow us to write y as a power series in x. Now we justify the existence of such a point before moving on to finding the power series.

Only finitely many points will satisfy the first equation in $(*)$ and not the second. First, compute $\text{Res}_y(Q, \frac{\partial Q}{\partial y}) = r(x) \neq 0$ (by step 1). We want x_0 such that $r(x_0) \neq 0$. (This x_0 may not be in F, but this is why we have dealt with $\overline{F}$.) Once we have found x_0, we can solve for $y_0 \in \overline{F}$. One of them will satisfy $(*)$.

WLOG assume $x_0, y_0 \in F$, by extending F if necessary.

Step 3. Expand y as a power series: $y = y_0 + y_1(x - x_0) + y_2(x - x_0)^2 + \cdots$. Now we use a recursive algorithm to compute y_i, $i > 0$. (You may have seen this as Newton's algorithm or Hensel's lemma). Suppose $y_0, \ldots, y_n$ are known to satisfy

$$Q(x, y_0 + y_1(x - x_0) + \cdots + y_n(x - x_0)^n) \equiv 0 \bmod (x - x_0)^{n+1}.$$

When $n = 0$,

$$Q(x, y_0) \equiv Q(x_0, y_0) \equiv 0 \bmod (x - x_0).$$

Now, for ease of notation, let $z = y_0 + y_1(x - x_0) + \cdots + y_n(x - x_0)^n$. Then, for some $b \in F$, the induction hypothesis gives

$$Q(x, z) \equiv b(x - x_0)^{n+1} \bmod (x - x_0)^{n+2}.$$

To find y_{n+1}, let $h = y_{n+1}(x-x_0)^{n+1}$ and compute

$$\begin{aligned} Q(x, z+h) &= \Big(Q(x,z) + \tfrac{\partial Q}{\partial y}(x,z)\cdot h\Big) \bmod (x-x_0)^{n+2} \\ &= \Big(b(x-x_0)^{n+1} + \tfrac{\partial Q}{\partial y}(x,z)\cdot h\Big) \bmod (x-x_0)^{n+2} \\ &= \Big(b(x-x_0)^{n+1} + \tfrac{\partial Q}{\partial y}(x,z)\cdot y_{n+1}(x-x_0)^{n+1}\Big) \bmod (x-x_0)^{n+2} \\ &= \Big((b + \tfrac{\partial Q}{\partial y}(x_0,y_0)\cdot y_{n+1})(x-x_0)^{n+1}\Big) \bmod (x-x_0)^{n+2} \end{aligned}$$

So, $y_{n+1} = \dfrac{-b}{\frac{\partial Q}{\partial y}(x_0,y_0)}$ does the trick. We now have our power series expansion of y in terms of x. Use this method to compute y_n for $n \le (\deg\, Q)^2$.

Step 4. For $m = 1, 2, \ldots,$deg Q; try to find $P(x,y) \in F[x,y]$ of degree m, starting with $m = 1$, with

$$P(x, y_0 + y_1(x-x_0) + \cdots + y_n(x-x_0)^n) \equiv 0 \bmod (x-x_0)^{n+1}.$$

Stop if you find P, else go to next value of m. This is a system of linear equations in the coefficients of P. So find the nullspace. If it is zero, go to the next step.

Claim. The P of minimal degree m found in Step 4 is an irreducible factor of Q.

Assuming the claim, then $P \mid Q$, so if $Q \neq P$, we replace Q with Q/P and repeat steps 2-4.

Why is the claim true? The point (x_0, y_0) about which the power series expansion of y was given is on the plane curve $Q = 0$ and also on $P = 0$. The conditions on P and Q ensure that the intersection multiplicity of $P = 0$ and $Q = 0$ at (x_0, y_0) is at least $(\deg Q)^2 + 1 > \deg P \deg Q$. Bezout's theorem then implies that $P \mid Q$.

Example. Let us illustrate with an easy example. Let $Q(x,y) = x^2 - y^2$. Then $(1,1) = (x_0, y_0)$ is a point on the curve $Q = 0$. And $\frac{\partial Q}{\partial y}(1,1) = -2$.

Expanding y in a power series: $y = 1 + y_1(x-1) + \cdots$. We have

$$\begin{aligned} x^2 - (1 + y_1(x-1) + \cdots)^2 &\equiv x^2 - 1 - 2y(x-1) \bmod (x-1)^2 \\ &\equiv (x-1)(x+1+2y) \bmod (x-1)^2. \end{aligned}$$

So, $1 - 2y_1 = -1$, giving $y_1 = 1$.

Then, $y = 1 + (x-1) + \cdots$. Now, find P of degree 1 satisfying

$$P(x, 1 + (x-1) + \cdots) \equiv 0 \bmod (x-1)^2.$$

Say, $P(x,y) = y - x$, so $P \mid Q$.

A few weeks ago, we talked about how to factor in $\mathbb{Z}[x]$. So, $Q(Y) \in \mathbb{Z}[Y]$. The analogy here is between $\mathbb{Z}$ and $\mathbb{F}[x]$. Find a prime p and a y_0 such that

$$\begin{aligned} Q(y_0) &= 0 \bmod p \\ Q'(y_0) &\neq 0 \bmod p. \end{aligned}$$

This is the same as finding (x_0, y_0). Now, use Hensel's lemma: find $y \in \mathbb{Z}/p^n$ with $Q(y) \equiv 0 \bmod p^n$, for n large in relation to the coefficients of Q. Step 3 is to find the power series expansion of y:

$$y = y_0 + y_1 p + y_2 p^2 + \cdots + y_n p^n$$

Now find $P(Y) \in \mathbb{Z}[Y]$ of small degree and with small coefficients such that $P(y) \equiv 0 \bmod p^n$. (This can no longer be done with linear algebra.) This congruence defines a lattice in $\mathbb{Z}^{\deg P+1}$. To find a short vector in a lattice, there is an algorithm called the LLL-algorithm which we won't explain. Then a height calculation will replace Bezout to prove that $P \mid Q$.

11 List decoding and discrete log problem:

The decoding problem of Reed Solomon codes can be reformulated into the problem of curve fitting or polynomial reconstruction. If we are given n points, $(x_1, y_1), (x_2, y_2)..., (x_n, y_n)$ in $\mathbb{F}_q^2$, we want to find all polynomials $f(x)$ of degree d that pass through at least t points. Recall that Sudan gave an algorithm for $t \geq 2\sqrt{nd}$. Cheng and Wan prove that if we decrease t much further(made precise later), then decoding Reed Solomon codes becomes as hard as solving the discrete log problem.

11.1 Index Calculus versus Sudan's algorithm

We use the index calculus technique for $\mathbb{F}_q^*$ where , $\mathbb{F}_q = \mathbb{F}_{p^n} = \mathbb{F}_p[x]/(f(x))$. We consider p midsize, which means that $\log p$ and n^α are of comparable size.

We want to use the factor base

$$B = \{g, x, x+1, ...x+p-1\}$$

$$< g >= \mathbb{F}_p^*$$

The first question is how often $h(x)$ of $\deg h \leq n$ factors as

$$h(x) = g^b \prod_{i=0}^{p-1} (x+i)^{a_i}$$

Total number of h's is $(p^n - 1)$.
The number of h's which factor is $(p-1)\binom{n+p-1}{p}$
If p is big, n is small, this is approximately equal to $\frac{p^n}{n!}$ or c^p when p and n are comparable.
Probability of h factoring $= \frac{1}{n!}$.

But we want the polynomial to factor in the factor base in $\mathbb{F}_q^*$ which is weaker than factoring in $\mathbb{F}_p[x]$.

Cheng's Idea: Lets ask what it means to write h as a product of elements in B in the quotient ring?

$$h(x) = g^b \prod_{i=0}^{p-1} (x+i)^{a_i} + f(x)k(x) \qquad 1$$

But this is not useful to us because we don't know k.

However, if $a_i \neq 0$, for some $i = 0, 1, ..., p-1$,

$$h(-i) = f(-i)k(-i)$$

So,

$$k(-i) = \frac{h(-i)}{f(-i)}$$

So, though we don't know k, we know many of the values of k at elements of $\mathbb{F}_p$. This works only when many a_i's are non-zero.

Question: Reconstruct $k(x)$ given that we know $k(-i)$ for all $i = 0, ..., p-1$ with $a_i \neq 0$.

$$d = \deg k(x) = \sum a_i - n$$

$$t = \#\{i = 0....p-1 | a_i \neq 0\}$$

Sudan's algorithm produces a list of solutions k to polynomial reconstruction problem $k(-i) = h(-i)/f(-i)$ for t values of i provided $t \geq 2\sqrt{pd}$ and the list has at most $\sqrt{\frac{p}{d}}$ elements.

If,

$$k(-i) = \frac{h(-i)}{f(-i)}, i \in I \subset \{0, 1..., p-1\}, |I| = t$$

We can deduce $k(x)f(x) - h(x)$ vanishes at $x = -i, i \in I$. This only gives us:

$$k(x)f(x) - h(x) = g^b \prod_{i \in I}(x+i)u(x)$$

This is not quite enough. We want to get rid of $u(x)$.

11.2 How to improve the index calculus algorithm

Fix some d. Given $h(x) \neq 0$ and $\deg h < n$.
Try to find all $k(x)$ of degree $\leq d$, such that

$$k(-i) = \frac{h(-i)}{f(-i)}$$

for at least $\lceil 2\sqrt{pd} \rceil$ values of i and check whether $k(x)f(x) - h(x)$ factors as a product of elements in the factor base.
Note that if d is small, it is better to use Berlekamp's algorithm than Sudan's.

Problem: Count how many more factorizations we are going to achieve by this improvement.

$$2\sqrt{pd} < t < p$$

Thus

$$d < \frac{p}{4}$$

List decoding will not succeed if $d > \frac{p}{4}$.

Theorem. *If* $p > \max(g^2, (n-1)^{2+\epsilon})$ *and* $g \geq (\frac{4}{\epsilon} + 2)(n+1)$ *for some* $\epsilon > 0$, *then* $\forall h, \exists k, a_i, b$, *satisfying equation 1 with* $\sum a_i \leq g$.

Question: Can this be improved?
Gain: Count the possible a_i with $\sum a_i \leq (n+d)$ and with $\#\{a_i \neq 0\} \geq 2\sqrt{pd}$. The good h's are $(g^b)\prod (x+i)^{a_i} \% f$

11.3 Open questions

Consider: $\mathbb{F}_{p^n}^*$, $\mathbb{F}_{p^n} = \mathbb{F}_p[x]/f(x)$. Let B be the set of monic irreducible polynomials with degree at most b. $h(x) = g^r \prod_{r(x) \in B} r(x)^{\alpha_r} + f(x)k(x)$ $k(z) = \frac{h(z)}{f(z)}$ if $r(z) = 0$ and $\alpha_r > 0$

Sudan's algorithm gives solution for the field in which we are working. Is there a version of Sudan's algorithm taking into account the field where the values are? Can we use it for the discrete log problem?

11.4 Numerical example of Cheng's idea:

Consider the field $\mathbb{F}_{125} = \mathbb{F}_{5^3} = \mathbb{F}_5[x]/(x^3 - x + 2)$. Instead of working with $\mathbb{F}_{125}^*$, we will work with $G = \mathbb{F}_{125}^*/\mathbb{F}_5^*$. Thus, $|G| = 31 = 124/4$

$G =< x >$. Factor Base $B = x, x+1, x+2, x+3, x+4$. Also, $\deg h(x) \leq 2$

We ask the following questions: Does $h(x)$ factor over B? How many $h(x)$ factor over B as polynomials? The number of $h(x)$ that factor over B as polynomials are 1(constant) + 5(elements of B,linear) +5(squares)+ 10(2 distinct factors)=21.

So there are 10 irreducible polynomials of degree 2.

How many of these h can be expressed as

$$h(x) = c\prod_{i=0}^{4}(x+i)^{a_i} + f(x)k(x)$$

with $c, k \in \mathbb{F}_5^*$? $\deg f = 3, \deg h = 2, \sum a_i = 3$.

Example 1: Consider $h(x) = x^2 + 3$. Remember $f(x) = x^3 - x + 2$.

The values of $h(-i)/f(-i)$ for $i = 1, 2, 3, 4, 0$ are $2, 2, 4, 2, 4$

The degree of k is 0, and we try value $k = 2$ which works.

Example 2: $h(x) = x^2 + 2$.

The values of $h(-i)/f(-i)$ for $i = 1, 2, 3, 4, 0$ are $4, 1, 2, 4, 1$

Try $k = 4$. We get $h(x) - 4f(x) = (-4)(x+1)^2(x+4)$.

Example 3: $h(x) = x^2 + 3x + 4$.

The values of $h(-i)/f(-i)$ for $i = 1, 2, 3, 4, 0$ are $1, 2, 3, 4, 2$

Try $k = 2$ $h(x) - 2f(x) = -2(x^2)(x+2)$. Here, another interesting observation was that the values $1, 2, 3, 4$ correspond to the values of the polynomial $-x$. So we also get: $h(x) + xf(x) = (x+1)(x+2)(x+3)(x+4)$.

12 Complexity of Multiplication

Suppose $V = \{v_1, ..., v_n\}$ is a basis of $\mathbf{F}_{q^n}/\mathbf{F}_q$. So every element of $\mathbf{F}_{q^n}$ is $\sum a_i v_i$ $a_i \in \mathbf{F}_q$.

$$(\sum a_i v_i)(\sum b_i v_i) = \sum a_i b_j v_i v_j \tag{13}$$

Now we need to know how to multiply the basis elements. Suppose:

$$v_i v_j = \sum m_{ijk} v_k, \tag{14}$$

then

$$\sum a_i b_j v_i v_j = \sum_k (\sum_{i,j} a_i b_j m_{ijk}) v_k \tag{15}$$

This requires about n^3 multiplications in $\mathbf{F}_q$ plus several additions, unless several of the m_{ijk} vanish. This motivates the following definition:
Definition:The multiplicative complexity of the basis V is $\mu(V)$ defined as $\#\{i,j,k|m_{ijk} \neq 0\} \leq n^3$
Question (open): What is the minimum $\mu(V)$ over all V for a fixed extension $\mathbf{F}_{q^n}/\mathbf{F}_q$?
If $\mathbf{F}_{q^n} = \mathbf{F}_q[x]/f(x)$ and α is a root of $f(x)$ in $\mathbf{F}_{q^n}$, then $V = \{1, \alpha, \alpha^2, ..., \alpha^{n-1}\}$ is a power basis of $\mathbf{F}_{q^n}$. We need to compute $\alpha^i \alpha^j = \alpha^{i+j}$ which is in V for $i + j < n$.
So $m_{ijk} = \left\{ \begin{array}{ll} 1 & k = i+j, \\ 0 & if\ not \end{array} \right\}$ in this case. So $\mu(V) \leq n^2/2 + n^3/2 + O(n)$.
If $f(x) = x^n + g(x)$ where $\deg(g) = t$ is small, $\alpha^i \alpha^j = \alpha^{i+j} = \alpha^{i+j-n}\alpha^n = -\alpha^{i+j-n} g(x)$, $i + j \geq n$.
$\alpha^{i+j-n}, \alpha^{i+j-n+1}, ..., \alpha^{i+j-n+t}$ are basis elements if $i + j - n + t \leq n$, i.e. $i + j < 2n - t$. We have $\mu(V) \leq n^2/2 + (n^2 - t^2)(t+1)/2 + t^2 n/2 + O(n)$. The main term, when $t \neq 0$ is $n^2 t/2$. So if $t \sim \log n$, we get a basis with multiplicative complexity $n^2 \log n$. We conjecture that we can always find a polynomial that gives us this.
Question: Is the minimum of $\mu(V)$ of the order $O(n^2 \log n)$?

Proposition 9. *$\mu(V) \geq n^2$, for any basis.*

Proof: Multiplication by v_i is an invertible linear map $\mathbf{F}_{q^n} \to \mathbf{F}_{q^n}$ with matrix $(m_{ijk})_{j,k}$ so that $\det(m_{ijk})_{j,k} \neq 0$. Each row must be nonzero. i.e. for all i, j there exists k such that $m_{ijk} \neq 0$. So $\mu(V) \geq n^2$.

Returning to the example $f(x) = x^n + g(x)$, if $t = 0$ (i.e. $g(x) = c \in \mathbf{F}_q$ you get $\mu(V) = n^2/2 + n^2/2 = n^2$. If $f(x) = x^n + c$, we can write a multiplication table:
$\alpha_i \alpha_j = \left\{ \begin{array}{ll} \alpha^{i+j} & i+j < n, \\ -c\alpha^{i+j-n} & i+j \geq n \end{array} \right\}$. This covers all the cases, so $\mu(V) = n^2$ in this case.
Exercise: You can find irreducible $x^n + c \in \mathbf{F}_q[x]$ if and only if $n|(q-1)$.

12.1 Heuristic for irreducibles of the type $x^n + g(x),\ \deg(g) = t$

The probability that a polynomial of degree n in $\mathbf{F}_q[x]$ is irreducible is about $1/n$. So the probability that all $x^n + g(x)$ are reducible should be $(1-1/n)^{q^t}$. If

$$q^t = n,\ (1 - 1/n)^n \sim 1/e \tag{16}$$

If

$$q^t = n^2,\ (1 - 1/n)^{n^2} \sim 1/e^n. \tag{17}$$

So we hope $q^t = n^2$ is enough, i.e. $t = 2\log n/\log q$ should do it for large n and fixed q. This motivates the conjecture that $\min \mu(V) \sim (n^2 \log n)$.

Example:Suppose $n = l - 1$, l prime, l does not divide q, and assume that $\frac{x^l-1}{x-1}$ is irreducible in $\mathbf{F}_q[x]$. So$\mathbf{F}_{q^n} = \mathbf{F}_q(\zeta)$, ζ is a primitive l^{th} root of unity, and $\{1, \zeta, \zeta^2, ..., \zeta^{l-2}\}$ is a basis for $\mathbf{F}_{q^n}/\mathbf{F}_q$; since $\zeta^l = 1$, we have: $\zeta^i\zeta^j =$

$$\left\{ \begin{array}{cc} \zeta^{i+j} & i+j \leq l-2 \\ -1-\zeta-\zeta^2-...-\zeta^{l-2} & i+j = l-1 \\ \zeta^{i+j-l} & i+j \geq l \end{array} \right\}$$

Then $\mu(V) \sim 2l^2 \sim 2n^2$. So this is a good basis.

Exercise: Under the same hypothesis, $\mathbf{F}_{q^{n/2}} = \mathbf{F}_q(\zeta + \zeta^{-1})$. So $V = \{\zeta + \zeta^{-1}, \zeta^2 + \zeta^{-2}, ..., \zeta^{n/2} + \zeta^{-n/2}\}$ is a basis for $\mathbf{F}_{q^{n/2}}/\mathbf{F}_q$. Playing around with this gives: $\mu(V) \sim 2(n/2)^2 \sim n^2/2$.

If L/K is Galois with group G, a normal basis of L/K is a basis of the form $\{\sigma\alpha\}_{\sigma\in G}$ for some $\alpha \in L$.

Theorem. *Every Galois extension has a normal basis.*

In the case of finite fields, $\mathbf{F}_{q^n}/\mathbf{F}_q$, G is generated by $x \mapsto x^q$. So a normal basis is $\{\alpha, \alpha^q, \alpha^{q^2}, ..., \alpha^{q^{n-1}}\}$.
The two examples with l^{th} roots of unity are examples of normal bases.

Proposition 10. *$\mu(V) \geq n(2n-1)$ for a normal basis V.*

When this lower bound is obtained, V is called an optimal normal basis. The two previous examples are the only two examples of optimal normal bases. **Proof**: Let $\sigma(x) = x^q$. If

$$\alpha\sigma^i\alpha = \sum m_{ij}\sigma^j\alpha, \tag{18}$$

then

$$\sigma^i \alpha \sigma^j \alpha = \sigma^i(\alpha \sigma^{j-i} \alpha) = \sum (m_{(j-i)k} \sigma^{j+k} \alpha). \tag{19}$$

So $m_{ijk} = m_{(j-i)(k-j)}$. Each of the m_{ij} repeats n times. So

$$\mu(V) = n(\#\{i, j | m_{ij} \neq 0\}) \tag{20}$$

On the other hand,

$$\alpha Tr(\alpha) = \sum_{i=0}^{n-1} \alpha \sigma^i \alpha = \sum_j (\sum_i m_{ij}) \sigma^j \alpha)) \tag{21}$$

which implies that $\sum m_{i0} = Tr(\alpha), \sum m_{ij} = 0, j \neq 0$. By the same determinant argument from the previous proposition, then for all j there exists i with $m_{ij} \neq 0$. But if $j \neq 0$ since $\sum m_{ij} = 0$ there must be at least two nonzero m_{ij}. So $\#\{i, j \mid m_{ij} \neq 0\} \geq 2n - 1$.
Exercise:Compute $\mu(1, \alpha, \alpha^2, ..., \alpha^{n-1})$ if $\alpha^n + \alpha^m + 1 = 0 \;\; 1 \leq m < n/2$.

Going back to

$$(\sum a_i v_i)(\sum b_j v_j) = \sum_k (\sum m_{ijk} a_i b_j) v_k = \sum B_k v_k \tag{22}$$

where

$$B_k(a_1, ..., a_n, b_1, ..., b_n) = \sum m_{ijk} a_i b_j \tag{23}$$

is a bilinear form. One can try to write B_k as $B_k = \sum_{m=1}^{M} L_{mk}(a) M_{mk}(b)$ where the L_{mk}, M_{mk} are linear forms. Then the number of multiplications in the ground field decreases if M is smaller than n^2 there is a gain. There is some clever trickery with algebraic curves to do this.