Cryptography and Finite Fields

Aaron Anderson

November 15, 2020

1 Fermat's Little Theorem

Let's start with a basic fact:

Theorem 1.1 (Bézout's Identity) *If $a, b \in \mathbb{Z}$, then there exist $c, d \in \mathbb{Z}$ such that $ac + bd = \gcd(a, b)$, where $\gcd(a, b)$ is the greatest common divisor of a and b.*

Problem 1 Say that a is invertible mod m when there exists some b such that $ab \equiv 1 \mod m$.

- Show that if p is prime, then every $1 \leq a < p$ is invertible mod p.
- Show that if a is relatively prime to m (that is, $\gcd(a, m) = 1$), then a is invertible mod m.

Problem 2 Let p be a prime, and $1 \leq a < p$.

- What is the set $\{a \mod p, 2a \mod p, 3a \mod p, \ldots, (p-1)a \mod p\}$?
- Comparing $(p-1)! \mod p$ with $a(2a)(3a)\ldots((p-1)a) \mod p$, can you calculate $a^{p-1} \mod p$? The result you should get is called *Fermat's Little Theorem.* Don't confuse it with Fermat's Last Theorem; Fermat actually managed to prove this one!

1.1 Euler's Totient Function

For $n \in \mathbb{N}$, let $\varphi(n)$ be the number of numbers $1 \leq a \leq n$ such that a is relatively prime to n. φ is called *Euler's Totient Function.*

Problem 3

- If p is prime, calculate $\varphi(p)$.
- If p, q are distinct primes, calculate $\varphi(pq)$.

Problem 4 Prove that

$$\sum_{d|n} \varphi(d) = n$$

In the sum, d should range over the factors of n. For instance,

$$\sum_{d|6} \varphi(d) = \varphi(1) + \varphi(2) + \varphi(3) + \varphi(6) = 6$$

(Hint: consider the list of fractions $\frac{1}{n}, \frac{2}{n}, \ldots, \frac{n}{n}$. When all fractions are expressed in reduced terms, how many have denominator d?)

Problem 5 Let $m \in \mathbb{N}$. Let U_m be the set of numbers between 1 and m that are relatively prime to m. ($|U_m| = \varphi(m)$ by definition.) Fix $a \in U_m$.

- What is the set $\{au : u \in U_m\}$?
- Comparing $\prod_{u \in U_m} u \mod m$ with $\prod_{u \in U_m} au \mod m$, can you calculate $a^{\varphi(m)} \mod m$? This result is known as the Fermat-Euler Theorem, because it is a better version of Fermat's Little Theorem, proven by Euler.

1.2 Primitive Roots

Let $1 \leq a < m$. Then we say that a is a *primitive root* mod m when for every b that is invertible mod m, there is some n such that $a^n \equiv b \mod m$.

Problem 6

- Find a primitive root mod 13.
- Show that there is no primitive root mod 8.

1.3 Discrete Logarithm

Let $1 \leq b < m$, and assume that b is invertible mod m. Then we define a function $\log_{b \mod m} : \{a : 1 \leq a < m \text{ and } a \text{ is invertible mod } m\} \to \mathbb{N}$ so that $\log_{b \mod m} a$ is always the smallest positive number such that $a^{\log_{b \mod m} a} \equiv b \mod m$.

Problem 7 Calculate $\log_{2 \mod 17} 9$.

2 Diffie-Hellman

In this section, we will start working with cryptography. Imagine two friends, Alice and Bob, who want to share messages, but can only communicate over a public channel. Their enemy Eve can sometimes eavesdrop on their communications, and they'd like to keep their messages secret.

Problem 8 Say that Alice and Bob can meet in person once, flip a coin n times, and write down the sequence of n coinflips, as 0s and 1s, before they have to leave.

- Explain how when they are apart again, they can share an n-bit secret message.
- If Eve eavesdrops on their communication channel, and attempts to guess their message, with what probability does she get it right?
- This cryptographic system is called a one-time pad system, because it stops being as secure after the first use. Explain how if Alice and Bob try to use the same sequence of coinflips as a keyword repeatedly, to send many n-bit messages, Eve might be able to break the code eventually.

Problem 9 Other forms of cryptography revolve around finding a computational process which can be performed quickly, but cannot be reversed quickly. Suppose that Alice, Bob, and Eve have the same type of computer, and that this type of computer can perform $2^{30} \approx 10^9$ basic computational steps per second, where for instance, multiplying two 1024-bit numbers counts as a step, or modding one 1024-bit number by another counts as a step.

Let p be a prime number whose binary representation is 1024 bits long, and let $1 \leq g < p$ be a primitive root mod p. Let a be randomly selected such that $1 \leq a < p$.

- If Alice tries to calculate $g^a \mod p$ by computing $g^1 \mod p$, multiplying by g, then modding to calculate $g^2 \mod p$, and then repeatedly multiplying by g and then modding until she gets all the way to $g^a \mod p$, how long will this take on average?
- Prove that Alice can actually calculate $g^a \mod p$ in less than a second.

Problem 10 Suppose that Alice and Bob can't actually meet in person. There's still a way they can send a secret message, called the Diffie-Hellman system. They choose a prime number p, and a number $1 \leq g < p$, and announce both of these numbers publicly. Then Alice picks a random number $1 \leq a < p$, and Bob picks a random number $1 \leq b < p$. Then Alice sends Bob $g^a \mod p$, and Bob sends Alice $g^b \mod p$.

- Prove that each of them can calculate $g^{ab} \mod p$.
- Describe how they can use g^{ab} to send a secret message of about 1024 bits in length.
- Suppose Eve learns $g^a \mod p$ and $g^b \mod p$. Say she then tries to calculate $g^{ab} \mod p$ by calculating b, and tries to solve for b by calculating $g^1 \mod p, g^2 \mod p, \ldots$ until she finds $g^b \mod p$. On average, how many steps will this take? How long will it take Eve to decode the secret message? Is the message actually secret?

- Why do Alice and Bob want to pick g to be a primitive root?

Problem 11 The Diffie-Hellman system, and all similar systems, are not infallible. Say that Eve can not only eavesdrop their communications, but can intercept a message from Alice to Bob, or from Bob to Alice, read it, and replace it with her own message. If Alice and Bob try to use the Diffie-Hellman system to send a message, how can Eve decipher it?

2.1 The Future

There actually are efficient algorithms for calculating discrete logarithms, but unfortunately for Eve, they have drawbacks. Most of them can only calculate $\log_{b \mod m}$ given assumptions on m, and Alice and Bob can easily avoid picking those m. There are a few algorithms that can calculate any discrete logarithm efficiently, but all the ones we've discovered so far will only run on a quantum computer, which hasn't yet been invented. (There is no proof that you need a quantum computer to efficiently calculate discrete logarithms in general, but nearly every expert thinks this is true. If you prove it, you would have proven $P \neq NP$, and you could collect a $\$1,000,000$ prize.)

Problem 12 Briefly explain how Eve could break Diffie-Hellman with access to a quantum computer or an efficient algorithm for calculating discrete logarithms.

3 The Euclidean Algorithm

So far, we have taken Bézout's identity as a black box. However, it would be useful to be able to use it for computations. For instance, if x is invertible mod n, then x has an *inverse* mod n, a number x^{-1} such that $x \cdot x^{-1} \equiv 1 \mod n$. We used Bézout's identity to show that such a number exists, but how can we calculate it?

Problem 13 Verify that if x is invertible mod n, there is a *unique* x^{-1} such that $x \cdot x^{-1} \equiv 1 \mod n$.

Problem 14 Let's start by efficiently calculating $\gcd(a, b)$ for $0 < a < b$, according to the Euclidean algorithm:

1. Find q, r such that $b = qa + r$, where $0 \leq r < a$.
2. If $r = 0$, return a as our answer.
3. If $r > 0$, then repeat this process starting at at step 2, replacing a and b with r and a.

- Prove that at the end of step 3 in each repetition, $(b-a)$ and a are always positive integers.
- Prove that this process eventually terminates (that is, returns an answer in step 2).
- Prove that if $b = qa + r$, then $\gcd(r, a) = \gcd(a, b)$.
- Prove that the final answer is actually $\gcd(a, b)$.
- How many steps does this process take? If a, b are each 1024 bits long, will this take a practical amount of computer time for Alice, Bob, or Eve?
- What do we do if $a, b \in \mathbb{Z}$, but we don't assume $0 < a < b$?

Problem 15 If $0 < a < b$, then we can also use the Euclidean algorithm to calculate c, d such that $ac + bd = \gcd(a, b)$.

- If $b = qa + r$ where $0 \leq r < a$, and we have numbers s, t such that $qs + rt = \gcd(a, b)$, then find numbers c, d such that $ac + bd = \gcd(a, b)$.
- Use your previous answer and the Euclidean algorithm to describe an algorithm for finding c, d such that $ac + bd = \gcd(a, b)$. How many steps will it take?

Problem 16 Now that you have an algorithm that proves Bézout's identity, describe how you would efficiently compute $x^{-1} \mod n$.

4 RSA

Now let's examine the RSA system, which will also allow Alice and Bob to communicate without meeting in person, but has a slightly different structure. Instead of using cryptography to agree on a shared secret ($g^{ab} \mod p$ in Diffie-Hellman), they can directly encrypt and decrypt messages.

Let's say that Bob wants to be able to receive messages from Alice. First, Bob finds two large prime numbers, p and q, which he keeps secret. Bob publicly announces their product $n = pq$. Then he picks a random number $1 < e < n - p - q - 1$, checks that it is coprime to $n - p - q - 1$, and also publicly announces e. The pair (n, e) is known as Bob's *public key.*

Then, whenever Alice wants to send a message to Bob, she just encodes her message as a binary number $0 \leq m < n$, calculates $m^e \mod n$, and sends it to Bob.

Problem 17

- How can Bob efficiently calculate m from $m^e \mod n$? (Hint: use your algorithm for computing inverses)
- How long would it take Eve to calculate m by brute force?

Problem 18 Now say that Alice gets a message from Bob, and wants to check that the message is actually from Bob. Say that Bob has posted (n, e) on his website, and that she knows for sure that he actually posted it. If Bob sends her the message m, encoded as a binary number less than n, then he also sends her a signature, $f(m)$, for a carefully chosen function f.

- How can he choose f such that Alice can calculate $f(m)^e \equiv m \mod n$?
- How confident should Alice be that Bob actually sent the message?

4.1 The Future

There are also efficient algorithms for factoring numbers, but these also have only been invented so far for quantum computers.

Problem 19 Briefly explain how Eve could break RSA with access to a quantum computer or an efficient algorithm for factoring numbers. (Note that even without efficient factoring, RSA still has potential security issues, for instance, if your computer's "random" number generator isn't actually random enough, there will be detectable patterns.)

5 Existence of primitive roots

Earlier, we defined primitive roots, and found an example, but why do we know that Alice and Bob can find a suitable primitive root g to go with their prime p in the Diffie-Hellman system?

First, let's see this theorem.

Theorem 5.1 *Let q be a polynomial of degree d, with integer coefficients, and let p be a prime. Assume that $q(x) \neq 0$ for some $x \in \mathbb{Z}$. Then $q(x)$ has at most d zeroes in $\{0, 1, \ldots, p-1\}$.*

This hopefully should make sense by analogy to the real or complex numbers, where a nonzero polynomial of degree d always has at most d zeroes.

Problem 20 If x is an integer and n a positive natural number, then we say the *order* of x mod n is the smallest positive natural number k such that $x^k \equiv 1 \mod n$, or ∞ if there are no positive numbers such that $x^k \equiv 1 \mod n$.

- Show that the order of x mod n is ∞ if and only if x is not invertible mod n.
- Show that $x^k \equiv 1 \mod n$ if and only if the order of x mod n divides k.
- If x is a primitive root mod n, what is the order of x?
- Is the converse to the last part true? That is, if x has the right order, is it guaranteed to be a primitive root mod n?

Problem 21 We will show that there exists a primitive root mod p for every prime p. In fact, we will show that there are $\varphi(p-1)$ of them!

Let $f(n)$ be the number of elements of $\{0, 1, \dots, p-1\}$ with order n mod p.

- Show that there are exactly $p-1$ zeroes (mod p) of $x^{p-1}-1$ in $\{0, 1, \dots, p-1\}$.
- Let $n \mid p-1$. Then show that $x^{p-1}-1 = (x^n-1)q(x)$ for some polynomial $q(x)$.
- Again, let $n \mid p-1$. Show that there are exactly n elements $x \in \{0, 1, \dots, p-1\}$ such that the order of x divides n.
- Show that $\sum_{d|n} f(n) = n$.
- Prove by induction that if $n \mid p-1$, then $f(n) = \varphi(n)$.
- Conclude that primitive roots mod p exist.

6 Challenge option 1: Fields

A *field* is a set $\mathbb{F}$ with addition, subtraction, multiplication, and inverse operations and special elements 0 and 1 such that the following properties hold:

Addition and subtraction properties:

- Associativity: For all $a, b, c \in \mathbb{F}$, $(a+b)+c = a+(b+c)$
- Commutativity: For all $a, b \in \mathbb{F}$, $a+b = b+a$
- Identity: For all $a \in \mathbb{F}$, $a+0 = a$
- Inverse: For all $a, b \in \mathbb{F}$, $a+(b-a) = b$

Multiplication and inverse properties:

- Associativity: For all $a, b, c \in \mathbb{F}$, $(a*b)*c = a*(b*c)$
- Commutativity: For all $a, b \in \mathbb{F}$, $a*b = b*a$
- Identity: For all $a \in \mathbb{F}$, $a*1 = a$
- Inverse: For all $a \in \mathbb{F}$, if $a \neq 0$, then $a*a^{-1} = 1$.

Distributivity: For all $a, b, c \in \mathbb{F}$, $a*(b+c) = a*b+a*c$.

Problem 22 Which of $\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$ are fields?

Problem 23 We will now define a field $\mathbb{Z}_p$ for each prime p. Let $\mathbb{Z}_p = \{0, 1, \dots, p-1\}$. It already comes with 0 and 1, and we can define $+, -, *, ^{-1}$ by addition, subtraction, multiplication and inverse mod p.

Verify that the axioms are satisfied.

Problem 24 For all positive $n \in \mathbb{N}$, we can define $+, -, *$ on $\{0, 1, \ldots, n-1\}$ by addition, subtraction, and multiplication mod n, like in the previous problem. Can we define an inverse operation that makes this set a field?

Problem 25 Let $\mathbb{F}$ be a field, and let $p(x)$ be a nonzero polynomial of degree d with coefficients in $\mathbb{F}$.

- Let $r \in \mathbb{F}$ be such that $p(r) = 0$ (evaluated using the addition and multiplication of $\mathbb{F}$). Show that there exists a polynomial $q(x)$ with coefficients in $\mathbb{F}$ such that $p(x) = (x - r)q(x)$, and that the degree of $q(x)$ is less than the degree of $p(x)$.
- Show by induction that $p(x)$ has at most d zeroes.
- Prove Theorem 5.1.

Problem 26 Let $\mathbb{F}$ be a field. If there exists a positive integer n such that $\underbrace{1 + 1 + \cdots + 1}_{n \text{ times}} = 0$, then we call the smallest such number the *characteristic* of $\mathbb{F}$. Otherwise, we say that the characteristic of $\mathbb{F}$ is 0.

- Find the characteristics of $\mathbb{Q}, \mathbb{R}, \mathbb{C}$, and $\mathbb{Z}_p$ for all primes p.
- Show that the characteristic of $\mathbb{F}$ is either 0 or a prime.

Problem 27 Taking inspiration from Fermat's Little Theorem and the Fermat-Euler Theorem, show that if $\mathbb{F}$ is a finite field, then for all $a \in \mathbb{F}$, $a^{|\mathbb{F}|} = a$.

Problem 28 If $\mathbb{F}$ is a finite field, show that it has a primitive root, defined as some $a \in \mathbb{F}$ such that for every $b \in \mathbb{F}$, either $b = 0$ or there is some $n \in \mathbb{F}$ such that $a^n = b$.

7 Challenge option 2: Computer implementation

Grab your favorite programming language (I recommend python), and try to implement some of the algorithms discussed in this worksheet. The following is a recommended order, with some sample numbers that should work fine (if you're not using python, remember to use 64-bit integers).

Problem 29

- Implement the Euclidean algorithm using recursion.
- Implement the algorithm for Bézout's identity.

Problem 30 Implement fast exponentiation of $a^b \mod n$. (In python, this is already implemented as `pow(a,b,n)`, so you can check your work, or use it to skip this problem.)

Problem 31 Implement a miniature Diffie-Hellman. Use prime $p = 4294967291$, and $g = 2$. First test that 2 is in fact a primitive root mod p. (You'll want to know that $p - 1$ factors as $2 * 5 * 19 * 22605091$, and then you'll want to check that for any three of those prime factors p_1, p_2, p_3, $2^{p_1 p_2 p_3} \not\equiv 1 \mod p$.)

You won't really be able to encode any substantial amount of text with a number less than 4294967291, so just encode numbers.

Another number option is $p = 2147483647, g = 7$.

Problem 32 Implement a miniature RSA with $p = 4294967291, q = 2147483647$, and $e = 65537$ (why do we know that e will be relatively prime to $\varphi(pq)$?).