PAWS 2025: MATHEMATICAL CRYPTOGRAPHY
PROBLEM SET 0

GIACOMO BORIN, JOLIJN COTTAAR, ELI ORVIS, GABRIELLE SCULLARD

Welcome to PAWS! Below are the exercises for Problem Set 0, which are intended to provide some background on common vocabulary and notation that will be used in this course. We have also prepared a notebook that will guide you through an introduction to the use of Sagemath for computer algebra and cryptography. More information in the last section of the file.

You are absolutely not obligated (or expected) to finish all of these problems before our first meeting, but we recommend that you at least attempt the problems in the review section if you have not seen the material before. Most importantly, work on the problems that interest you!

Review Problems

(1) The Euclidean algorithm computes the greatest common divisor of two positive integers $a > b$ by the following procedure: Use division with remainder to write

$$a = q_1 b + r_1,$$

where the remainder r_1 satisfies $0 \leq r_1 < b$ and q_1 is the quotient. If $r_1 = 0$, then $b \mid a$ and $\gcd(a, b) = b$. Otherwise, we can continue, using b as the new dividend and r_1 as the new divisor, and obtain a new remainder $0 \leq r_2 < r_1$. We continue until $r_{k+1} = 0$, in which case $r_k = \gcd(a, b)$.

$$\begin{aligned} a &= q_1 \cdot b + r_1 \\ b &= q_2 \cdot r_1 + r_2 \\ r_1 &= q_3 \cdot r_2 + r_3 \\ &\vdots \\ r_{k-1} &= q_k \cdot r_k \end{aligned}$$

(a) Use the Euclidean algorithm to compute $\gcd(30030, 257)$. Use your result and the fact that $30030 = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13$ to prove that 257 is prime.

(b) Use the Euclidean algorithm to compute $\gcd(4883, 4369)$. Use your work to factor 4833 and 4369 into a product of primes.

(2) The *extended* Euclidean algorithm uses the sequence of quotients $q_1, q_2, \ldots, q_{k-1}$ obtained from the Euclidean algorithm to compute two integers x, y such that $ax + by = \gcd(a, b)$, by forming the two sequences:

$$\begin{aligned} x_0 &= 1, & x_1 &= 0, & x_j &= -q_{j-1}x_{j-1} + x_{j-2} \\ y_0 &= 0, & y_1 &= 1, & y_j &= -q_{j-1}y_{j-1} + y_{j-2} \end{aligned}$$

Then $ax_k + by_k = \gcd(a, b)$.
For a more detailed explanation and example of how this works, see Example 1.10 in An Introduction to Mathematical Cryptography (an online copy is here).

(a) Show that if there exist integers x and y such that $ax + by = 1$, then $\gcd(a, b) = 1$.

(b) Show that a is invertible mod b if and only if $\gcd(a, b) = 1$. ("Invertible mod b" means there exists an integer $z \pmod b$ such that $az \equiv 1 \pmod b$. We denote $z \pmod b$ by $a^{-1} \pmod b$.)

(c) Use the extended Euclidean algorithm to compute x and y such that $17x + 101y = 1$. What is $17^{-1} \pmod{101}$?

(3) The Chinese Remainder Theorem states that if $\gcd(n, m) = 1$ and a, b are integers, then there is a unique solution $x \pmod{mn}$ to the simultaneous congruence,

$$x \equiv a \pmod{n}, \qquad x \equiv b \pmod{m}$$

(a) Show with a counterexample that the theorem is no longer true if the condition $\gcd(n, m) = 1$ is dropped.

(b) Solve the simultaneous congruence

$$\begin{cases} x \equiv 2 \pmod{17} \\ x \equiv 9 \pmod{101}. \end{cases}$$

(Hint: You could start listing numbers which are congruent to 9 (mod 101), but here's another approach using the work you've already done: Write $x = 17k + 2$, and solve for k in the congruence $17k + 2 \equiv 9 \pmod{101}$.)

(4) Recall that a *group* is a set S together with a binary operation $m : S \times S \to S$, such that

(a) for all $s_0, s_1, s_2 \in S$, $m(s_0, m(s_1, s_2)) = m(m(s_0, s_1), s_2)$,

(b) there exists $s^* \in S$ such that $m(s, s^*) = m(s^*, s) = s$ for all $s \in S$, and,

(c) for all $s \in S$ there exists $s^{-1} \in S$ such that $m(s, s^{-1}) = m(s^{-1}, s) = s^*$.

We think of m as being a (not necessarily commutative!) multiplication on S, s^* as being a multiplicative identity, and (as the notation indicates) s^{-1} as being the multiplicative inverse of s, and will usually denote m as a product. When the operation m is commutative, we will sometimes denote it with $+$. In this exercise we will recall the basic properties of groups with an emphasis on examples that will be useful in this course.

(a) Prove that the identity element in any group is unique. Prove that each element of a group has a *unique* multiplicative inverse (this justifies the notation s^{-1} used above).

(b) Let G be a group and $g \in G$. Show that the function $m_g : G \to G$ defined by $m_g(h) = hg$ is a bijection.

(c) Let $\mathrm{GL}_2(\mathbb{R})$ denote the set of 2×2 matrices with real entries and determinant 1. Show that $\mathrm{GL}_2(\mathbb{R})$ is a group under multiplication. Is $\mathrm{GL}_2(\mathbb{R})$ commutative?

(d) Let $\mathbb{Z}/p\mathbb{Z}$ be the set of integers modulo p, i.e., the equivalence classes of the integers under the equivalence relation $a \sim b$ if and only if $p \mid a - b$. We define addition and multiplication on $\mathbb{Z}/p\mathbb{Z}$ by $[a] + [b] \coloneqq a + b \pmod{p}$ and $[a][b] \coloneqq ab \pmod{p}$ (we will usually drop the brackets, but it will be understood that we are multiplying equivalence classes). Prove that $\mathbb{Z}/p\mathbb{Z}$ together with the operation of addition is a group. Prove that $(\mathbb{Z}/p\mathbb{Z})^\times = \mathbb{Z}/p\mathbb{Z} - \{[0]\}$ is a group under multiplication.

(e) Is the set $\mathbb{Z}/15\mathbb{Z} - \{[0]\}$ a group under multiplication? Can you identify the maximal subset of $\mathbb{Z}/15\mathbb{Z}$ that is a group under multiplication?

(f) The number of integers less than or equal to N that are coprime to N is denoted by $\varphi(N)$. The function φ is called "Euler's phi function," or sometimes "Euler's totient function." Prove that

$$\varphi(N) = N \prod_{p \mid N, p \text{ prime}} \left(1 - \frac{1}{p}\right).$$

(**Hint:** One way is to proceed as follows, first show the result when N is a power of a prime. Next show that $\varphi(mn) = \varphi(m)\varphi(n)$ when m and n are relatively prime. Finally, put the two together to get the general result. For those who have seen the Möbius inversion formula, there is an elegant proof, which begins by arguing directly that $N = \sum_{d \mid N} \varphi(d)$...)

(5) Euler's Theorem states that for any integer a coprime to N, $a^{\varphi(N)} \equiv 1 \pmod{N}$.

(a) Prove Euler's Theorem. Use the group theory fact that the order of an element divides the order of the group, applied to the group of integers mod N which are coprime to N under multiplication, $(\mathbb{Z}/N\mathbb{Z})^\times$.

(b) Compute $7^{-1} \pmod{30}$ using the extended Euclidean algorithm.

(c) Suppose for some unknown integer $m \pmod{31}$, you are given the value of $m^7 \pmod{31}$. How can you find m? (Raise m^7 to a certain power mod 30 and use Euler's Theorem.)

(6) A *group homomorphism* is a function $\phi : G \to H$, where G and H are groups, such that

$$\phi(g_1 g_2) = \phi(g_1)\phi(g_2), \tag{1}$$

for all $g_1, g_2 \in G$.
(Equation (1) does not look so surprising, but notice that the "multiplication" on each side of the equals sign can be different!) In this question, we will prove some of the basic facts about group homomorphisms.

(a) Let 1 be the multiplicative identity in G. Prove that $\phi(1)$ is the multiplicative identity in H.
(b) For any $g \in G$, show that $\phi(g^{-1}) = \phi(g)^{-1}$.
(c) A *subgroup* of a group is a subset that is also a group under the same operation. Prove that $\phi(G)$ is a subgroup of H.
(d) The *kernel* of a group homomorphism is the set of elements $g \in G$ such that $\phi(g) = 1$. We denote this by $\ker(\phi)$. Prove that $\ker(\phi)$ is a subgroup of G with the additional property that $g \ker(\phi) g^{-1} = \ker(\phi)$ for all $g \in G$. Such a subgroup is called a *normal subgroup* of G.
(e) Let $G/\ker(\phi)$ denote the set of equivalence classes of G under the equivalence $g \sim h$ if and only if $gh^{-1} \in \ker(\phi)$. Define a multiplication on $G/\ker(\phi)$, and prove that your multiplication is well-defined and makes $G/\ker(\phi)$ into a group.

Algorithm 1 Square and Multiply for Modular Exponentiation

```
Input: Integers x, e, n
Output: x^e mod n
Convert e to binary: e = (e_k e_{k-1} ... e_0)_2
result ← 1
for i from k down to 0 do
    result ← result^2 mod n
    if e_i = 1 then
        result ← (result · x) mod n
    end if
end for
return result
```

(7) In the rest of this course it is going to be paramount that we can do exponentiation in modular arithmetic as fast as possible. One of the most used algorithms is the square and multiply method which can be found in Algorithm 1.

(a) Calculate $23^{71} \pmod{31}$ using the square-and-multiply method (using at most 7 squarings and 4 multiplications).
(b) Given that a multiplication costs $\mathcal{O}(n^{\log 3})$ (Karatsuba), what is the expected runtime of the square and multiply method?

(8) Some cryptographic computations need to calculate something of the form $x^e y^f \pmod{n}$. Devise an efficient algorithm (as an adaptation on Algorithm 1) that outputs $x^e y^f \pmod{n}$, for some integers x, y, e, f, n. (It should be able to compute $x^{22} y^{13} \pmod{n}$ in at most 4 multiplications and 4 squarings, plus one precomputation).

Introduction to Sagemath

Sagemath is a free open-source computer algebra system that is widely used in research and education. It is particularly useful for number theory and cryptography, as it includes many built-in functions for working with modular arithmetic, groups, and other mathematical structures. You can download and install Sagemath on your computer, or you can use it online through the online platform CoCalc.

Once you have installed it on your computer you can open the Sagemath console by opening a terminal and typing

```
$ sage
```

If you have installed Jupyter Notebook you can also run Sagemath in a Jupyter Notebook by typing in a terminal:

```
$ sage -n jupyter
```

Note: there are plenty of guide on how to install Sagemath, depending on your operating system and installation preferences. If you have any trouble, please don't hesitate to reach out to us!

We have also prepared a notebook that will guide you through an introduction to the use of Sagemath for computer algebra and cryptography. Try to open it on Cocalc or download it, run the cells, do (some of) the exercise and play with it to get confidence. Don't be scared if this is your first experience with writing code, try to play around and if you feel stuck feel free to reach to us!

Here some useful links:

- Notebook with introduction to Sagemath: `https://cocalc.com/share/public_paths/3722a39b9bc5617b718a368c6e6cb6007597aa0b`
- Installation instructions: `https://doc-gitlab.sagemath.org/html/en/installation/index.html`
- Sagemath documentation: `https://doc.sagemath.org/html/en/index.html`
- Sagemath tutorial: `https://doc.sagemath.org/html/en/tutorial/index.html`
- Jupyter Notebook: `https://jupyter.org/install`
- CoCalc: `https://cocalc.com/`