Discrete Math Lecture 4

1 Recurrences

Recurrences are a way of recursively defining a sequence of numbers or a function in terms of the values of the function or sequence for smaller values. Since, in computer science, we will often be dealing with recursively defined objects, you will often have to work with recurrences. In particular in your algorithms class, you will often be able to bound the run-time of your algorithm in terms of it's run time on smaller problems, especially if you are dealing with recursive algorithms.

You have already seen some examples of recurrences in this class. Recall the Fibonacci numbers. The n-th Fibonacci number, F_n, is defined by

$$F_n = F_{n-1} + F_{n-2}$$

where $F_1 = F_2 = 1$. Note how we need to specify the base cases for our recurrence by hand. We have seen how we can prove things about these numbers by using induction. Today we will look at a recurrence that come up a lot while studying common algorithms and prove some bounds on the solution of this recurrence. Once again induction will be our tool of choice.

Consider the classic Mergesort algorithm. We will not worry too much about what this algorithm does yet. Suffice to say that it takes a list of n numbers and sorts them. To do so it breaks the list into two halves, sorts each half and spends n time to combine the two sorted halves into a sorted whole. If $f(n)$ is the run time of Mergesort on a size n list, it turns out that, because of this recursive behaviour, we have the following recurrence for $f(n)$:

$$f(n) = 2f\left(\frac{n}{2}\right) + n$$

Let's say on a size 1 problem, the algorithm takes 1 unit of time, so $f(1) = 1$. You might rightly point out that this equation doesn't make sense if n is not divisible by 2, but let's ignore that concern and say n is a power of 2, i.e., $n = 2^k$ for some natural number k.

We claim that $f(n) \geq n \log_2 n$. Hence forth whenever I use logarithms without a base, I will mean logarithms to the base 2.

Proof. For $n = 1$, notice that $f(1) = 1 > 1 \cdot \log_2 1$, since $\log_2 1 = 0$.

Now for the inductive step. Assume $f(n) \geq n \log_2 n$ for $n = 2^k$. We now want to prove this for the next power of two, i.e., $2n = 2^{k+1}$.

$$\begin{aligned} f(2n) &= 2f\left(\frac{2n}{2}\right) + 2n \\ &= 2f(n) + 2n \\ &\geq 2(n \log_2 n) + 2n \\ &= 2n \log_2 n + 2n \log_2 2 \\ &= 2n \log_2(2n) \end{aligned}$$

This completes the proof.

□

Excercise 1. *Using a similar analysis, prove that if $f(1) = 1$, and for all other natural numbers n that are powers of 2:*

$$f(n) = 2f\left(\frac{n}{2}\right) + 1$$

Then $f(n) \geq \frac{1}{2}\log_2 n$.

You can also visualize these recurrences via a recursion tree. This is also a very useful way to reason about recurrences, but we will not discuss recursion trees in this course. Let's move on to the main topic of interest for the day, which is number theory. The natural numbers, while seemingly very simple objects, have a huge number of remarkable properties. Number theory is a centuries old field, with many many hard problems that are still unsolved. Today we will look at some basic properties about numbers, focusing on divisibility of numbers by other numbers and the powerful idea of modular arithmetic which has to do with the remainders left when you perform such divisions.

2 Divisibility and remainders

Let us begin with some definitions you are likely somewhat familiar with:

Definition 1. *Let a and b be integers. Then we say a divides b, $a|b$ or b is divisible by a if there exists an integer n such that $a \cdot n = b$. We also say that b is a multiple of a.*

Notice that by our definitions, every number divides O. This might seem fishy, but it's just because we said $a|b$ if $\exists c$ such that $b = ac$.

Notice that this also means that $0|0$ and it sounds scary to divide by 0! But this is nothing to worry about, we aren't actually dividing by 0, we are defining divisibility and that is being done only in turns of multiplication.

Recall that if we divide two integers b by a, then if $a|b$ we get a quotient such that $b = qa$, whereas if a does not divide b then we get both a quotient q and a remainder r. In general we have the division theorem

Theorem 1. *For all integers n and $d > 0$, there exist unique integers q and r such that:*

$$n = dq + r \text{ and } 0 \leq r < d$$

Proof. To prove this we show two things:

- q and r actually exist.
- They are unique.

Let $S = \{n - dq \mid q \in \mathbb{Z}\}$, here n and $d > 0$ are arbitrary integers. ($\mathbb{Z}$ normally represents the integers). Since $d > 0$ and q can be any integer, there must be some element in S, i.e., S is not empty. Let r be the smallest non-negative element in S. Let q be such that $r = n - dq$. Now we want to show that $r < d$.

Suppose it is not. Then, $r \geq d$ so $0 \leq r - d$. But then $r - d$ is also an element of S (Why?) and it is non-negative. However this is a contradiction. Thus we conclude that for any $n, d > 0$:

There exist q and r such that $n = dq + r$ and $0 \leqslant r < d$.

Now let's show that these q and r are unique. Suppose they are not:

$$n = dq_1 + r_1 \qquad (1)$$
$$n = dq_2 + r_2 \qquad (2)$$

We want to conclude that $q_1 = q_2$ & $r_1 = r_2$. This would show that q & r are unique because saying $\exists$ unique $x, P(x)$ is the same as saying:

$$\exists x, (P(x) \wedge (\forall y, P(y) \Rightarrow x = y))$$

Without loss of generality, let's say $r_1 \geq r_2$. Subtracting equations (1) and (2) yields $0 = (q_1 - q_2)\, d + (r_1 - r_2)$. Or,

$$(q_2 - q_1)\, d = r_1 - r_2$$

But then $d \mid (r_1 - r_2)$. As $r_1 - r_2 < d$, $r_1 - r_2 = 0$. Since $(q_2 - q_1)\, d = 0$ and $d > 0$ so $q_2 - q_1 = 0$.

We have shown $q_1 = q_2$ & $r_1 = r_2$, completing the proof of uniqueness. □

We just described the following:

- n is the dividend
- d is the divisor
- q is the quotient
- r is the remainder

We are particularly interested in divisors. The set of divisors of n is defined as

$$\text{Div}(n) = \{a : a \mid n\}.$$

For example what are the divisors of 24? They are: $\{1, 2, 3, 4, 6, 8, 12\}$. Integers often have common divisors. If $m, n \in Z$ then the greatest common divisor of m and n, denoted by $\gcd(m, n)$ is the largest element in $\text{Div}(m) \cap \text{Div}(n)$. For example, for $m = 18$ and $n = 21$

$$\gcd(m, n) = 3$$

Finding the set of all divisors of a number turns out to he a hard problem. However we do know an easy way to find the greatest common divisor. This is where modular arithmetic comes in handy.

3 Introduction to modular arithmetic and Euclid's algorithm

We actually use modular arithmetic all the time in day-to-day life. Suppose it is $10am$ now. What will be the time in 15hrs ? Yow think for a bit and answer 1am. How did you do this?

Well $10 + 15 = 25$, so why isn't it 25 am? This is because every 24 hours is a whole day and the time 'resets' after every 24 hours. Multiples of 24 can thus be ignored and counted as 0. In our mathematical notation, we say

$$25 = 1 \bmod 24$$

This idea is very useful, as we will see today.

Let n, d, q and r be integers satisfying $n = dq + r$. Then we say

$$n = r \mod d$$

The idea here is that n and r are equivalent unto multiples of d. Notice that, by the above definition, if $n = r \mod d$ then $r = n \mod d$ as well. Really, this is a notion of equivalence, and

in other sources you might find this represented as $n \equiv r \mod d$, but we will just use the $=$ sign in these notes. So for example

$$31 = 19 = 7 = 1 \mod 3$$

because all these numbers leave the same remainder (1) when divided by 3. In most programming languages this is represented by the modulus operator %. Indeed we can think of 31 mod 3 as a set of numbers: all integers that have the remainder 1 when divided by 3, and 31%3 yields the smallest non-negative representative of 31 modulo 3. So $31\%3 = 1$. You could get negative representatives as well, by subtracting multiples of 3 from 1, for example

$$-29 = 1 (\mathrm{mod} 3)$$

However we know, by the division theorem, that if we ask what some number n is modulo d, there is always some non-negative integer r, such that $n \mod d = r$ and $0 \leq r < d$. This r is precisely the remainder you get when you divide n by d, and this is what is generally meant when you are asked to calculate $n \mod d$.

But how does any of this help us get the greatest common divisor? It is through the following observation:

Observation 1. *If* $d = \gcd(m, n)$, $m > n$ *and* $n \neq 0$ *Then* $d = \gcd(m - n, n)$.

Why is this? Well if d divides both m and n, it will divide $m - n$ as well. This is because if $m = k_1 d$ and $n = k_2 d$ then

$$m - n = (k_1 - k_2)\, d$$

Furthermore, if d divides $m - n$ and n then

$$\begin{aligned} m - n &= k_1 d \\ n &= k_2 d \end{aligned}$$

so

$$m = (k_1 + k_2)\, d$$

Thus m is also divisible by d.

Extending this idea further

Proposition 1. *If* $d = \gcd(a, b)$, $r = a \mod b$, *and* $b \neq 0$, *then*

$$d = \gcd(r, b)$$

Proof. Let a and $b > 0$ be arbitrary. Let q be such that $a = qb + r$. If $d|a$ and $d|b$ then d must divide r since we can write $a = k_1 d$, $b = k_2 d$ so

$$r = (k_1 - k_2 q)d$$

Similarly let c be a common divisor of b and r. Then we can write $b = ck_1$ and $r = ck_2$ so

$$a = c(qk_1 + k_2)$$

and thus $c|a$ as well. What we have shown is that a and b share a divisor if and only if $a \mod b$ and b do, which completes the proof. □

Here's an example of a calculation done using our observation:

$$\begin{aligned}
&\gcd(232, 72) \\
&= \gcd(232 \bmod 72, 72) \\
&= \gcd(16, 72) \\
&= \gcd(16, 72 \bmod 16) \\
&= \gcd(16, 8) \\
&= \gcd(16 \bmod 8, 8) \qquad = \gcd(8, 0) \\
&= 8.
\end{aligned}$$

Meanwhile, our observation suggests an algorithm to find the gcd of the two numbers.

```
GCD(a,b)
{
   if(a=0)
     return b;
   else if(b=0)
     return a;
   else if (a>=b)
     return gcd(b,a mod b);
   else if (a<b)
     return gcd(a,b mod a);
}
```

This is Euclid's algorithm, which has been known since 300BC ! It was probably discovered even earlier. People have been doing mathematics for a very long time.

We will next prove a related lemma that tells us something very useful about the gcd.

Lemma 1 (Bezout)**.** *Let a and b be integers that have $\gcd(a, b) = d$. Then there exist integers x and y such that $ax + by = d$.*

Let's see an example. $\gcd(15, 35) = 5$. What does $15x + 35y$ look like? If we start plugging in integers x and y, we get only multiples of 5 and $x = -2, y = 1$ does in fact give us exactly 5. This suggests how we can prove Bezout's lemma: the smallest positive value of $ax + by$ should be the gcd.

Proof. (Sketch) Consider the set

$$S = \{ax + by | x, y \in \mathbb{Z} \wedge ax + by > 0\}$$

S contains either a or $-a$ so it is non empty. By well-ordering, there is some least element $d \in S$.

Divide a by d. Let $a = dq + r$, where $0 \leq r < d$. This is by the Division theorem. But we know that $d = as + bt$ for some s, t. Then

$$\begin{aligned}
r &= a - dq \\
&= a - (as + bt)q \\
&= a(1 - sq) + b(tq)
\end{aligned}$$

But this means that unless r is 0, r itself is in S, which is a contradiction because r is smaller than d. The only possibility is that r is 0, so d divides a. The exact same method can be used to

show that d divides b.

Ok, so d is a common divisor of a and b. Now to show that it is the gcd. Suppose c is any common divisor of a and b. Then $a = cp, b = cq$ for some p, q. But then

$$\begin{aligned} d &= as + bt \\ &= cps + cqt \\ &= c(ps + qt) \end{aligned}$$

So $c|d$ but $d > 0$ so $c \leq d$. Thus d is the gcd of a and b.

□

The coefficients x and y which are guaranteed to exist by Bezout's lemma can in fact be calculated by an extension of Euclid's algorithm, but we will skip over these details today. This concludes our discussion of Euclid's algorithm and gcd's for the moment. Let's move on to modular arithmetic.

One reason modular arithmetic is very useful is that if we only care about calculations up to some multiplicative factor, we can just do computations modulo that factor. The following rules are very useful for this.

- if $a = b$ then $a \mod r = b \mod r$.
- $a + b \mod r = ((a \mod r) + (b \mod r)) \mod r$.
- $ab \mod r = ((a \mod r) \cdot (b \mod r)) \mod r$.

Excercise 2. *Prove that the above rules work.*

The first implication certainly doesn't work in the other direction though! Also, notice that we said nothing about division. Why? Well, first of all dividing an integer by another doesn't necessarily yield an integer. But what about when we consider the integers modulo some natural number n? It turns out that if you consider the integers modulo a *prime* number, then for any number m, you can find a natural number m^{-1}, such that $m \cdot m^{-1} = 1$ modulo the prime. This is one of the things which makes prime numbers special! We will prove this fact later today.

Let's see how our rules for modular arithmetic can be useful: one feature that will come up again later, is that modular arithmetic can be used to check things quickly. For example if I told you $2^{2023} + 3^{2024} = 7^{777}$ you could immediately tell me this is nonsense, without having to do some crazy computation. Let's see how.

We will consider the whole equation modulo 7. The right hand side is simply 0 because it is clearly divisible by 7. Let us look at the remainders when 2^i is divided by 7. These are: $2, 4, 1, 2, 4, 1, \ldots$. We notice a pattern, the remainders cycle through $2, 4, 1$. This is a cycle of length 3. That means 2^{2023} will leave a remainder of 2, when divided by 7, since $2023 = 1 \mod 3$.

What about the remainders when 3^i is divided by 7? Those cycle through $3, 2, 6, 4, 5, 1$. That means 3^{2024} will also leave a remainder of 2, when divided by 7, since $2024 = 2 \mod 6$. But then the sum on the LHS leaves a remainder of 4 when divided by 7, while the RHS is divisible by 7, which means the equality cannot hold.

Our calculations above while finding $2^{2023} \mod 7$ and $3^{2024} \mod 7$ suggest the following

Proposition 2. *Consider the sequence $a^i \mod b$ for any natural numbers a and b and all natural numbers i. These remainders must form a cycle.*

Proof. (Sketch) The possible remainders after dividing any number by b are $0, 1, \ldots b-1$. Consider the sequence of remainders on dividing a^i by b. Eventually a remainder must repeat (by the pigeonhole principle). So there will be some i and k such that $a^i = a^{i+k} \mod b$. Now argue that this guarantees that the cycle of remainders from $a^i \mod b$ to $a^{i+k} \mod b$ will keep repeating. □

This result tells us that we can actually do exponentiation quite fast when we just want the answer modulo some number.

4 Prime numbers

Definition 2. *An integer p greater than 1 is called prime if the only positive divisors of p are 1 and p. Otherwise, p is called composite.*

A large portion of number theory is devoted to the study of prime numbers and their properties. A big reason for this is that they are considered the building blocks for numbers. This idea is so important that the primary result concerning this is called the Fundamental Theorem of Arithmetic.

Theorem 2. *Every natural number $n > 1$ can be written uniquely as a product of primes.*

Note here the condition that $n > 1$. Sometimes there is some controversy among non-mathematicians over the fact that 1 is not considered a prime number. There are a number of reasons one can give to justify this idea. The main justification is due to this theorem: if 1 were a prime number, then we could no longer say that every number has a unique prime factorization.

Let's now prove this fundamental theorem. We first need a lemma, due to Euclid:

Lemma 2. *(Euclid) If p is prime and a and b are integers such that p divides $a \cdot b$ then p divides at least one of a or b.*

Proof. Without loss of generality, suppose p does not divide a. Then $\gcd(p, a) = 1$ and there exist integers x and y such that $xa + yp = 1$. Then multiplying both sides by b, we have

$$b = xab + ypb.$$

Since $p \mid ab$, let n be an integer such that $ab = pn$. Then we have

$$b = xab + ypb = xpn + ypb = p \cdot (xn + yb).$$

Therefore, by definition of divisibility, $p \mid b$. □

Now, we will state, but not prove the following lemma, which we make use of later on. It makes a nice exercise (try proving it with induction).

Excercise 3. *If $p, p_1, \ldots, p_k$ are prime and $p \mid p_1 \cdots p_k$, then $p = p_i$ for some i.*

Now, we're ready to prove the Fundamental Theorem of Arithmetic. Since the theorem claims existence and uniqueness of prime factorization, we'll prove this in two parts: first we prove the fact that we can factor numbers into primes and then show that this factorization is unique.

Theorem 3. *Every natural number $n > 1$ can be written as a product of primes.*

Proof. We will prove that for $n > 1, n$ can be written as a product of primes via strong induction on n.

Base case. $n = 2$. Clearly, 2 is a product of primes because 2 itself is a prime number.

Inductive case. Let $k \in \mathbb{N}$. Assume that for all natural numbers m with $2 \leq m \leq k$, that m can be written as a product of primes. We want to show that $n = k + 1$ can be written as a product of primes. There are two cases to consider.

- If $k+1$ is prime, then $k+1$ can clearly be written as a product of primes since $k+1$ is itself a prime number.
- If $k+1$ is composite, then we can write $k+1 = a \cdot b$ for some $a, b \in \mathbb{N}$. Observe that $2 \leq a, b < k+1$. Therefore, by the inductive hypothesis, a and b can be written as a product of primes. Let $a = p_1 p_2 \cdots p_s$ and $b = q_1 q_2 \cdots q_t$, where $p_1, p_2, \ldots, p_s, q_1, q_2, \ldots, q_t$ are primes. Then
$$k+1 = a \cdot b = p_1 p_2 \cdots p_s q_1 q_2 \cdots q_t$$
and therefore $k+1$ can be written as a product of primes.

□

And now for the uniqueness:

Proof. (Fundamental Theorem of Arithmetic) We just showed that every natural number $n > 1$ has a prime factorization. Suppose that prime factorizations are not unique. Let n be the smallest number that doesn't have a unique prime factorization, so

$$n = p_1 \cdots p_k = q_1 \cdots q_\ell$$

where p_i is prime for $1 \leq i \leq k$ and q_j is prime for $1 \leq j \leq \ell$. Clearly, $p_1 \mid n$. Then by Lemma 3.8, $p_1 = q_j$ for some j. Now, we divide each factorization by p_1. This gives us

$$\frac{n}{p_1} = p_2 p_3 \cdots p_k = q_1 q_1 \cdots q_{j-1} q_{j+1} \cdots q_\ell$$

But this is a smaller number with more than one prime factorization, contradicting the minimality of n. □

A question that greatly interests number theorists is how are the prime numbers distributed among the natural numbers. Today we will see that there are infinitely many prime numbers.

Theorem 4. *There are infinitely many prime numbers.*

Proof. Suppose not. Then there must be only finitely many prime numbers. Let's say there are k of them, arrange them in increasing order $p_1, p_2, \ldots, p_k$ so that p_k is the largest one. But $N = p_1 \cdot 2 \cdots p_k + 1$ is not divisible by any of the prime factors. But by the Fundamental theorem Every natural number is a product of primes. So N itself must be prime. But $N > p_k$. Contradiction. □

Modular arithmetic works especially nicely when we do everything modulo a prime number. In particular, every number has a multiplicative inverse:

Proposition 3. *For any prime p, every natural number n, that is not divisible by p, has a natural number n^{-1} that is its multiplicative inverse modulo p, i.e., $n \cdot n^{-1} = 1 \mod p$.*

Proof. (Sketch.) Consider $nx \mod p$ for all the numbers $\{0, 1, p-1\}$. Suppose $nx = ny \mod p$. Then $n(x-y) = 0 \mod p$. But n is not divisible by p and $|x-y| < p$ so $(x-y)$ can't be divisible by p. So $f(x) = nx \mod p$ maps $\{0, 1, p-1\}$ to $\{0, 1, p-1\}$ and different inputs are sent to different outputs. By the pigeonhole principle, $f(x) = 1$ for some x and this x is the multiplicative inverse of n. □

Excercise 4. *Prove the above result using Bezout's lemma.*

5 Fermat's little theorem

We will now see a result of Pierre de Fermat's about prime numbers and $\mathbb{Z}_p$ called Fermat's Little Theorem. This is to distinguish it from the more famous Fermat theorem, Fermat's Last Theorem (there are no integers a, b, c such that $a^n + b^n = c^n$ for $n > 2$). Much like Fermat's Last Theorem, Fermat stated the result in the mid 1600s but declined to give a proof of it because the proof was "too long".

While it would be about 400 years before Fermat's Last Theorem gets proven (by Andrew Wiles in 1995), Fermat's Little Theorem was proved only about 100 years later, by Leonhard Euler.

Theorem 5. *(Fermat's Little Theorem). For all prime numbers p and integers a, $a^p = a \mod p$.*

Proof. (Sketch.) We will do the proof for all positive integers a. You can easily extend the result to negative integers as well.

We will begin with an identity colloquially referred to as the Freshman's dream:

$$(x + y)^p = x^p + y^p \mod p$$

This is actually true if p is a prime. To show this, use the binomial theorem and observe that $\binom{p}{r}$ is divisible by p except when r is 0 or p. Thus all the middle terms in the binomial expansion disappear modulo p.

Now we will do a proof by induction. For $a = 1$ check that the claim is true for all primes p. Suppose the claim is true for any prime p and all natural numbers a from 1 to m. Now we will prove it for $m+1$. For any prime p, we can use the Freshman's dream and our inductive hypothesis to obtain:

$$(m + 1)^p = m^p + 1^p = m^p + 1 = m + 1 \mod p$$

This completes the proof.

□

The following corollary follows immediately from the fact that if we are operating modulo a prime, every number, except 0, has a multiplicative inverse.

Corollary 1. *For all prime numbers p and integers a that are not divisible by p, $a^{p-1} = 1 \mod p$*

What we learnt so far today tell us that we don't always need to perform tedious computations to discover solutions to interesting problems involving large numbers. If all we need to know about are the remainders when integers are divided by n, then we can work directly with those remainders in modulo n. We already saw an example of this when we used modular arithmetic to check that $2^{2023} + 3^{2024}$ cannot equal 7^{777}. Let's see another example, this time using Fermat's little theorem.

Example 1. *What is $111^{31} \mod 31$? Applying the theorem, we immediately get the answer to be $111 \mod 31$, which is 18.*

In particular, Fermat's little theorem allows us to check if numbers are prime. To check if x is prime, calculate $a^{x-1} \mod x$ for some values of x. Remember, we can do exponentiation very fast with modular arithmetic. If you ever get an answer that is not 1, then the number is not prime. Sadly, you can't guarantee the number is prime if you do get 1 each and every time, but if you do this for enough random values of a it becomes unlikely that x is composite since a^{x-1} keeps yielding a remainder of 1. Still, it is not a guarantee. However you can guarantee that a number is composite if you find an a such that the remainder $a^{x-1} \mod x$ is not 1.

Doing a bunch of calculations using huge numbers is all well and good, but let's see another example where modular arithmetic is invaluable: cryptography.

6 Cryptography

Number theory is essential in the construction of numerous security protocols that are used today. Some of the more famous ones that you can read about include the Diffie-Hellmann Key Exchange protocol and the RSA system. A key idea in a lot of these applications in cryptography involves a so-called "one-way function". This is a function that is invertible and easy to compute, but it's inverse is hard to compute. We use the function for encrypting our messages and anyone eavesdropping on us would have to invert the function in order to decrypt the message. Today we will use the things we learnt about the gcd as well as Fermat's little theorem to see an example of a cryptosystem which is also based on this principle. However lets first spend some time on a very low tech version of this cryptosystem. Try to come up with a scheme to securely transmit an object based on the guidelines in your handouts.

One scheme that works is as follows:

- You put the diamond in the box, lock the box and send it to your friend via post.
- Your friend receives the locked box and locks it again with his own lock. He then posts it to you.
- You receive the doubly locked box and remove your lock. Now the box is locked using only your friends lock. You post this to your friend.
- Your friend removes his own lock and takes out the diamond.

However, notice that the postman, if he has a lock of his own, can still just impersonate your friend. When you give him the locked box, he will put his own lock on it. Then he will give you back the box with both locks on it and you will hand him the box after removing your lock. Now the box is locked using only the postman's lock, and he will be able to unlock it and steal the diamond. Actual cryptosystems must set up ways to deal with this kind of attack as well.

We are now going to formalize a cryptosystem based on the scheme from the handout. The idea of the double locks will be the same. Recall how we mailed the box three times in total. This protocol will do the same, it is known as Shamir's three pass protocol.

First of all let's model all our messages as natural numbers. Since computers represent everything as binary numbers, this actually isn't that much of a stretch. Suppose Alice possesses a number m, and $m \mod p$ is the secret message she wants to send to Bob. We will assume that m is not $0 \mod p$. Alice also has another number r which she will use as a lock/key and Bob has a similar number s. We want to keep these keys secret. Alice and Bob both publicly agree that they will use computation modulo some large prime p in this protocol. This is known to everybody, including the evil third party Eve, but she won't be able to use this to learn anything about the message or the keys.

r and s should have the property that $\gcd(r, p-1) = 1$ and $\gcd(s, p-1) = 1$. Such an r and s exist, for example $r = p$ or $p-2$ works, by the **Euclidean algorithm**. $\gcd(r, p-1) = 1$ and $\gcd(s, p-1) = 1$ guarantee that for any a there exist r_1 and s_1 such that $a^{rr_1} = 1 \mod p$ and $a^{ss_1} = 1 \mod p$.

Why is this? We will choose r_1 such that $rr_1 = 1 \mod (p-1)$. We know that there is an r_1 such that $rr_1 = 1 \mod (p-1)$. This is because $\gcd(r, p-1) = 1$, so, by **Bezout's lemma**, there are x and y such that $rx + (p-1)y = 1$. An extension of the Euclidean algorithm actually lets us find x and y. Now set $r_1 = x$. Taking the equation $rr_1 + (p-1)y = 1$ modulo $(p-1)$ yields that

$rr_1 = 1 \mod (p-1)$.

Since $rr_1 = 1 \mod (p-1)$, thus $rr_1 = k(p-1)+1$ for some k and now it follows from **Fermat's little theorem** that

$$a^{rr_1} = a^{k(p-1)+1} = (a^{p-1})^k \cdot a = 1^k \cdot a = a \mod p$$

.

What all this tells us is that we can find r_1 and s_1 such that $a^{rr_1} = a \mod p$ and $a^{ss_1} = a \mod p$.

The three pass protocol is:

- Alice first computes $m^r \mod p$ and sends this to Bob. This can be done quite quickly, but neither Bob nor anyone else can easily compute the value of r by knowing just $m^r \mod p$ and p. This problem is called the discrete logarithm problem and there is no fast algorithm known for this problem.
- Using $m^r \mod p$ Bob exponentiates this to the power s and sends back $m^{rs} \mod p$ to Alice.
- Alice computes r_1 using her knowledge of what r is. This lets her to raise m^{rs} to r_1 working of course all the time modulo p, to compute $m^{rsr_1} = m^{rr_1 s} = m^s \mod p$. Then she sends back $m^s \mod p$ to Bob.
- Bob can raise this to the power s_1. This yields $m^{ss_1} \mod p = m \mod p$.

Notice that throughout the process Alice couldn't learn what s is , nor does Bob learn what r is. This is because given $m^s \mod p$ and p, it is hard to calculate s. r and s can be thought of as locks. Also notice how we used a bunch of thing we studied today: the **Euclidean Algorithm**, **Bezout's lemma** as well as **Fermat's little theorem**, to guarantee that there exist suitable r_1 and s_1 to perform the role of doing the 'unlocking' of r and s respectively. r_1 and s_1 are the multiplicative inverses of r and s modulo $(p-1)$.

Notice that the three pass system is still vulnerable to the so-called man in the middle attack. That is, if Alice and Bob use Eve to communicate with each other, then Eve could actually just pretend to be Bob. She receives m^s from Alice and returns m^{st}, where t is just another number such that $gcd(t, p-1) = 1$. Alice then unlocks her lock, returning m^t to Eve, who now calculates m. At least Eve is not able to get to know Alice's secret number r.

The security of the three pass protocol relies on the hardness of the discrete log problem. This is in fact a common feature of cryptosystems. We rely on "one -way functions" for the security. Here exponentiation modulo a prime is the one way function. Doing the exponentiation is easy, but undoing it (i.e. taking the log) is hard. That is, given a, r and b, you can calculate $a^b \mod r$ quite easily, but given $a^b \mod r$, even knowing a and r, it is still hard to find b. Most cryptographic schemes are built in this way, where the hardness of a computational problem provides the security guarantee since to break the scheme the attacker would have to solve an instance of the hard problem.

Another concrete application of number theory upon which we regularly rely in day to day life is the notion of a checksum. This provides a quick way to authenticate transactions or credit card numbers. We won't go into details today, but you can learn more about it by watching this excellent video.