Grade 11/12 Math Circles

February 9, 2022

Cryptography, Part 1

Introduction

In this lesson (and the next), we will explore the math behind *cryptography*, the science of sending secret messages. For this lesson, we'll focus on the state of cryptography leading up to World War II. In the lesson two weeks from now, we'll look at *public key cryptography*, the big post-war idea that made it possible to send credit card numbers and other confidential information over the internet.

The Basics of Cryptography

In essence, cryptography is about scrambling a message you don't want people to read, in such a way that only the intended recipient can unscramble and read it. The scrambling process is known as *encryption*, and the de-scrambling process is known as *decryption*. Before encryption, the message is called a *plaintext*, and after encryption, it is called a *ciphertext*. It sounds a lot like a spy movie!

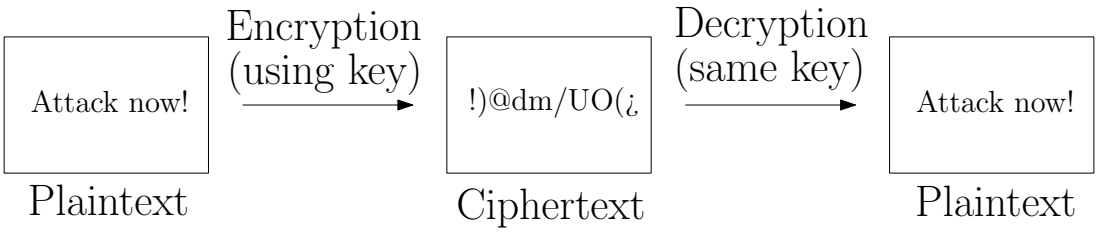
Stop and Think

Why would anyone wish to encrypt a message? Can you think of situations in your own life where cryptography might be useful?

There are many situations where sending a secure message is desired. Here is one historic application: imagine a general who wants to send her battle strategy to her soldiers, but knows it would be very bad if an enemy intercepted it. If the message is encrypted first, then it doesn't matter if it falls into enemy hands, because it will look like gibberish. Only the lieutenant on the battlefield with the decryption key has the ability to recover the original message from the gibberish (at least, that's the idea).

However, there are many other less poetic, more relevant applications these days. For instance, suppose you wish to send your credit card number to a store over the internet, and you don't want anyone but the store to know what the number is. Or, say you want to recover a password you've forgotten from a website, and you don't want someone intercepting the message to also learn what the password is.

Before the 1970s, encryption always worked in the same way. Two people wishing to communicate (let's call them Nick and Shefaza) would have to exchange a secret *key* beforehand. To send a message, Nick would use this key to convert the plaintext into ciphertext, and when Shefaza receives the message, she would use it to change the ciphertext back into the original plaintext. Anyone reading the ciphertext without the key is faced with a seemingly meaningless string of characters. A diagram indicating the general procedure is given here.



At the moment, the biggest question is: how can we actually encrypt and decrypt a message? You are encouraged to spend a few minutes thinking about this before reading the suggestions below.

Stop and Think

If your goal was to encrypt a message, how would you go about doing it? Keep in mind that you want the encrypted message to be hard for others to understand, but easy for the intended recipient to decrypt.

In essence, there are two approaches: *transposition* and *substitution*.

Encryption by Transposition

Encrypting a message by transposition means to re-arrange the characters of the plaintext in order to get the ciphertext. Let's look at an example illustrating this type of procedure.

Example 1

Suppose that Nick wants to send Shefaza the message

Retreat immediately.

Nick and Shefaza would first agree on a certain number of letters. In this case, we'll choose that number to be 5. Now, Nick removes all the spaces and punctuation and makes all the letters

uppercase in order to remove any distinguishing features that might give away something about the message. This leaves him with

RETREATIMMEDIATELY

Next, Nick arranges the letters in rows of five, the number he agreed upon with Shefaza:

R	E	T	R	E
A	T	I	M	M
E	D	I	A	T
E	L	Y		

Finally, Nick arranges the columns of this array in some order that has been pre-arranged with Shefaza, say last to first. He goes down each column and writes out the letters, resulting in the ciphertext

EMTRMATIIYETDLRAEE

This explains how encryption works. When Shefaza receives this message, she can reverse the procedure to decrypt. She knows the letters were arranged in rows of five. Since the message has 18 letters in total, the last two columns will be one letter shorter than the rest. Shefaza knows the columns were taken in reverse order, from last to first, so she puts the first three letters of the message in the last column, the next three letters in the fourth column, and then four letters per column working backwards after that. This will result in the same arrangement of letters that Nick started with. By reading the letters across each row, Shefaza can get the original message back.

Exercise 1

(a) Try encrypting the message RETREATIMMEDIATELY using a different number of columns (and possibly a different ordering of columns).

(b) Suppose you have received the ciphertext CTAIMFROPSUUYGHSCNPRYOH, knowing the encryption was done by transposition with four columns, taken in the order first-to-last. What is the decryption of this ciphertext?

Shift Ciphers

As you will see in the problem set, there are many reasons why using transposition alone to encrypt a message is not a good idea. Historically, substitution was the preferred technique. Here, instead of scrambling the letters of the original message, we take each letter of the plaintext and replace it with a different symbol (usually also a letter), and keeping the order of the replaced letters the same.

The simplest example of encryption by substitution is known as a *shift cipher*, and it is thousands of years old. The general idea is to encrypt by shifting every letter of the alphabet by a certain number of positions. For example, suppose we wanted to shift by two letters. We would replace every "A" in the message by a "C", which is two places further in the alphabet. Similarly, "B" would be replaced by "D", "C" would be replaced by "E", and so on, finishing by replacing every "Z" with a "B" (looping back around to the beginning of the alphabet).

We can write out these substitutions nicely in the form of a table:

Plaintext	A	B	C	D	E	F	G	H	I	J	K	L	M
Ciphertext	C	D	E	F	G	H	I	J	K	L	M	N	O
Plaintext	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Ciphertext	P	Q	R	S	T	U	V	W	X	Y	Z	A	B

Let's illustrate this with an example.

Example 2

Suppose Nick wants to use the shift cipher to send RETREATIMMEDIATELY to Shefaza, with a shift by two letters. Using the table above, Nick takes the first plaintext letter, R, and looks it up in the Plaintext row of the table. He then replaces the R with T, the letter below it in the table. Next, he takes the second letter of the message, E, and replaces it with G, the letter below E in the Ciphertext row. Doing the same thing for the entire message, Nick sends the following ciphertext to Shefaza:

TGVTGCVKOOGFKCVGNA

To decrypt the ciphertext, Shefaza looks up each letter of the ciphertext in the Ciphertext row of the table and replaces it with the corresponding letter in the Plaintext row of the table (in this case, shifting each letter two places backward in the alphabet).

For the shift cipher, the number of alphabet positions to shift by is what forms the key. For instance,

Nick and Shefaza could decide to shift forward by 10 positions, which would result in the following encryption/decryption table:

Plaintext	A	B	C	D	E	F	G	H	I	J	K	L	M
Ciphertext	K	L	M	N	O	P	Q	R	S	T	U	V	W
Plaintext	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Ciphertext	X	Y	Z	A	B	C	D	E	F	G	H	I	J

Exercise 2

(a) Try encrypting the message RETREATIMMEDIATELY using a shift cipher with a shift number different from 2.

(b) Suppose you receive the ciphertext HIGVCTXMSRMWWLMJXCFYWMRIWW, encrypted using a shift cipher with a forward shift of four letters. What is the decryption of this message?

Monoalphabetic Substitution Cipher

While the shift cipher may look like a promising form of encryption, there are in fact several major problems with the security of the shift cipher. As you may already have noted, there are only 25 possible keys, one for every possible size of shift (not counting the shift by 0 places, which does nothing to the message). If someone intercepts a ciphertext and suspects a shift cipher, they only have to try all 25 possible keys. Usually, only one does not lead to a string of gibberish, and that will be the key that gives the plaintext.

Happily, this particular problem is easily fixed. Rather than shifting each letter of the alphabet forward by the same fixed number of places, we can instead replace each letter in a less predictable fashion. More specifically, we will allow any arrangement of the alphabet in the Ciphertext row of the encryption table. This more general encryption technique is called a *monoalphabetic substitution cipher*.

For example, we could use this encryption table:

Plaintext	A	B	C	D	E	F	G	H	I	J	K	L	M
Ciphertext	G	Z	I	C	N	P	F	H	X	D	L	Q	T
Plaintext	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Ciphertext	M	Y	W	A	J	B	S	R	V	U	K	E	O

By allowing any arrangement of letters in the Ciphertext row of the table, the number of possible keys increases dramatically, enough to make trying all the keys an impossible task. In the problem set, you'll work out exactly how many keys are available, and also how long it would take to search through all the keys, under some modest assumptions.

That's good news, isn't it? The monoalphabetic substitution cipher looks unbreakable! Of course, you already know that can't be true, or else cryptography would not be the blossoming subject area that it is today. It is possible to perform a *statistical analysis* on the ciphertext, using the fact that not every letter appears in text with the same frequency. For example, in English, the most common letter is "e", and it is reasonable to guess that the most common ciphertext letter corresponds to "e" in the plaintext.

Onetime Pad Cipher

Despite the flaws of the monoalphabetic substitution cipher, it would take over 1000 years before anyone could make a significant improvement on the method. The idea behind the breakthrough is this: the reason statistical analysis of the ciphertext works for a monoalphabetic substitution cipher is because each plaintext letter is encrypted using the same rule.

What if we instead encrypt each plaintext letter according to a different rule? For example, what if each letter is encrypted by a shift cipher, but the shift is different every time? Ciphers like this are called *polyalphabetic substitution ciphers*.

If the shifts for each letter are selected entirely randomly, the resulting encryption method is called the *onetime pad cipher*. A version of this cipher was first described in 1882 by an American named Frank Miller, but ironically, no one paid attention at the time! The technique was not used until its rediscovery in a different form in 1917, by a research engineer named Gilbert Vernam.

Example 3

To use the onetime pad cipher, Nick and Shefaza must share a large string of randomly generated letters, one that is at least as long as the message they wish to send. For example, suppose Nick's

message to Shefaza is "The Prime Minister is ill". As usual, he strips away all punctuation and spacing to get the plaintext

THEPRIMEMINISTERISILL.

There are 21 letters in this message, so Nick and Shefaza need a string of 21 random letters to use as the key. For the sake of example, suppose that key is

MSWZTFAOHAOHAJIERZYAL.

The first letter of the key is M, so Nick draws up an encryption table for a shift cipher, with M as the first letter in the ciphertext row:

Plaintext	A	B	C	D	E	F	G	H	I	J	K	L	M
Ciphertext	M	N	O	P	Q	R	S	T	U	V	W	X	Y
Plaintext	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Ciphertext	Z	A	B	C	D	E	F	G	H	I	J	K	L

He uses this table to encrypt the first letter of the plaintext, which is T, to find that the first ciphertext letter should be F. The second letter of the key is S, which means Nick must now use the encryption table starting with S in the second row.

Plaintext	A	B	C	D	E	F	G	H	I	J	K	L	M
Ciphertext	S	T	U	V	W	X	Y	Z	A	B	C	D	E
Plaintext	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Ciphertext	F	G	H	I	J	K	L	M	N	O	P	Q	R

The second plaintext letter, H, is therefore encrypted as Z.
Proceeding in this way through the entire message, Nick sends the following message to Shefaza:

FZAOKNMSTIBPSCMVZRGLW

For Shefaza to decrypt the message, she only needs to reverse the shift cipher on each letter.

Exercise 3

(a) Encrypt the word FALSE using a onetime pad cipher with random key FRQEX.

(b) Suppose you have received the ciphertext EJIKBKYXOS, encrypted with the onetime pad key QWERTYUIOP (the top row of a keyboard). What is the corresponding plaintext?

The main advantage of the onetime pad cipher is that it completely defeats statistical analysis. For instance, in the THEPRIMEMINISTERISILL example above, the three Es in the plaintext are all encrypted differently: one becomes an A, one becomes an S, and one becomes an M in the ciphertext.

In fact, as long as the onetime pad key is chosen truly randomly, performing decryption without the key is completely hopeless. If every possible random key is equally likely, then every plaintext could result in the ciphertext you receive, provided a certain key was used. Mathematicians have actually proved that the onetime pad offers *perfect secrecy*. This has a specific technical meaning, but essentially tells us that breaking the encryption without the key is impossible.

Stop and Think

Why isn't this the end of the story of cryptography? Doesn't this mean the onetime pad is the perfect cipher, and that we have no reason to seek better techniques? Why might the onetime pad not be a "perfect solution"?

Theoretically speaking, the onetime pad is as good a cipher as we could ask for. Practically speaking however, there are a couple hurdles that make use of the onetime pad difficult. Firstly, perfect secrecy is only guaranteed if the key is selected truly at random, and each key can be used *only once*. Breaking either of these conditions compromises the security of the cipher (and we'll explore why in the problem set).

Therefore, in order to use the onetime pad correctly, a large amount of random information is required to generate the keys. Such randomness is rare in nature and hard to harness, making the procedure costly in terms of time and equipment. As a result, use of onetime pad ciphers are generally reserved for top-secret diplomatic communications (such as the hotline between Moscow and Washington D.C. during the Cold War).

Other Mechanical Ciphers

Because the onetime pad is so impractical, the search continued for ciphers that were more secure than a monoalphabetic substitution cipher, but not so cumbersome to use as the onetime pad. By the early- to mid-20th century, people were relying on machines (and eventually, computers) to make their calculations, so it made sense that the next generation of ciphers would be mechanized.

One particularly famous example is the Enigma Machine, used widely by the German army during World War II, and pictured below (obtained from an image by Alessandro Nassiri - Museo della

Scienza e della Tecnologia "Leonardo da Vinci", CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=47910919).

Essentially, the Enigma machine was an elaborate keyboard with sets of dials. A configuration of the dials would determine a method for scrambling the plaintext letters, in such a way that each successive plaintext letter is encrypted differently from the last. When a plaintext letter was typed into the keyboard, the corresponding ciphertext letter would light up elsewhere on the machine.

The breaking of this Enigma cipher had a significant impact on the course of World War II, and represented one of the first times a cipher was broken mechanically. If you are interested, there are plenty of books and movies that tell this story in a very interesting way – Simon Singh's book *The Code Book* and the movie *The Imitation Game* are two excellent examples.

Following the war, the DES (Data Encryption Standard) was widely adopted for encryption applications, though it is no longer used today, because the number of possible DES keys is considered too small to be secure. In 2001, a new encryption standard, the AES (Advanced Encryption Standard) was adopted to replace DES, and continues to be used today.

Looking Ahead: Public Key Cryptography

Unfortunately, the onetime pad and these later ciphers all suffer from the same problem. In order to encrypt messages, both the sender and receiver need to get their hands on the same key. This is not a problem if sender and receiver can physically meet to exchange the key before they communicate securely, but this becomes a huge problem if they live on opposite sides of the planet.

For example, if Nick and Shefaza called or e-mailed each other to exchange keys, anyone could tap

the line to learn the key as well, because the first contact between Nick and Shefaza would necessarily be unencrypted.

Nick and Shefaza could use a trusted courier to carry keys from one to the other, but this is way too expensive and troublesome for most applications. Without a satisfactory solution to this problem, making purchases over the internet would have been impossible. There is no way that a company can be expected to securely exchange keys with every potential customer before the customers send along their credit card details!

In the next Math Circles lesson (coming two weeks from now), we will explore how this problem was solved in the 1970s, resulting in a whole new version of cryptography, called *public key cryptography*.