An Introduction to Mathematical Cryptography

Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman

Mathematics 158 — Brown University — Fall 2004

Contents

Chapter 1

An Introduction to Cryptology

1.1 Simple substitution ciphers

As Julius (later to become Caesar) surveys the unfolding battle from his hilltop outpost, an exhausted and disheveled courier bursts into his presence and hands him a sheet of parchment containing gibberish:

```
j s j r d k f q q n s l g f h p g w j f p y m w t z l m n r r n s j s y q z h n z x
```

Within moments, Julius sends an order for a reserve unit of charioteers to speed around the left flank and exploit a momentary gap in the opponent's formation.

How did this string of seemingly random letters convey such important information? The trick is easy, once it is explained. Simply take each letter in the message and shift it five letters up the alphabet. Thus `j` in the *ciphertext* becomes `e` in the *plaintext*, because `e` is five letters, `efghij`, before `j`. Applying this procedure to the entire ciphertext yields

```
j s j r d k f q q n s l g f h p g w j f p y m w t z l m n r r n s j s y q z h n z x
e n e m y f a l l i n g b a c k b r e a k t h r o u g h i m m i n e n t l u c i u s
```

The second line is the decrypted plaintext, and breaking it into words and supplying the appropriate punctuation, Julius reads the message

```
Enemy falling back.  Breakthrough imminent.  Lucius.
```

There remains one minor quirk that must be addressed. What happens when Julius finds a letter such as `d`? There is no letter appearing five letters

`*** Cipherwheel rotated 5 steps ***`

Figure 1.1: A cipher wheel with an offset of five letters

before `d` in the alphabet. The answer is that he must wrap around to the bottom of the alphabet. Thus `d` is replaced by `y`, since `yzabcd`.

This wrap around effect may be conveniently visualized by placing the alphabet `abcd...xyz` around a circle, rather than in a line. If a second alphabet circle is then placed within the first circle and the inner circle is rotated five letters, as illustrated in Figure 1.1, the resulting arrangement can be used to easily encrypt and decrypt Julius' messages. To decrypt a letter, simply find it on the inner wheel and read the corresponding plaintext letter from the outer wheel. To encrypt, do it the other way, find the plaintext letter on the outer wheel and read off the ciphertext letter from the inner wheel. And more importantly, if you build a cipherwheel whose inner wheel spins, then you are no longer restricted to always shifting by exactly five letters. Cipher wheels of this sort have been used for centuries.[1]

Although the details of the preceding scene are entirely fictional, and in any case it is unlikely in the extreme that a message to a Roman general would have been written in English(!), it appears likely that Julius employed this early method of cryptography, which is sometimes called the *Caesar cipher* in his honor, and sometimes called a *shift cipher*, since each letter in the alphabet is shifted up or down. *Cryptography*, the methodology of concealing the content of messages, comes from the Greek root words kryptos, meaning hidden,[2] and graphikos, meaning writing. The modern scientific study of cryptography is often called *cryptology*.

In the Caesar cipher, each letter is replaced by one specific substitute letter, so the Caesar cipher is an an example of what is known as a simple substitution cipher. However, if Bob encrypts a message for Alice[3] using a

[1] A cipher wheel with mixed up alphabets and with encryption performed using different offsets for different parts of the message is featured in a 15th century monograph by Leon Batista Alberti [24].

[2] The word cryptic meaning hidden or occult appears in 1638, while crypto- as a prefix for concealed or secret makes its appearance in 1760. The term cryptogram is much later, first occurring in 1880.

[3] In cryptography, it is traditional for Bob and Alice to exchange confidential messages and for their adversary Eve, the eavesdropper, to intercept and attempt to read their messages. This makes the field of cryptology much more personal than other areas of

Caesar cipher and allows the encrypted message to fall into Eve's hands, it will take Eve very little time to decrypt it. All she needs to do is try each of the 26 possible shifts.

Bob can make his message harder to attack by using a more complicated replacement scheme. A *simple substitution cipher* is a cipher in which each letter is replaced by another letter (or some other type of symbol). To make it easier to distinguish the plaintext from the ciphertext, we write the plaintext using lower case letters and the ciphertext using upper case letters. Then a simple substitution cipher may be viewed as a rule or function

$$\{\texttt{a,b,c,d,e,...,x,y,z}\} \longrightarrow \{\texttt{A,B,C,D,E,...,X,Y,Z}\}.$$

In order for decryption to work, this encryption function must have the property that each plaintext letter goes to a different ciphertext letter. A convenient way to describe the encryption function is to create a table by writing the plaintext alphabet in the top row and putting each ciphertext letter below the corresponding plaintext letter.

Example 1.1. A simple substitution encryption table is given in Table 1.1. The ciphertext alphabet (the upper case letters in the bottom row) is a randomly chosen permutation of the 26 letters in the alphabet. In order to encrypt the plaintext message

```
Four score and seven years ago,
```

we run the words together, look up each plaintext letter in the encryption table, and write the corresponding ciphertext letter below.

f	o	u	r	s	c	o	r	e	a	n	d	s	e	v	e	n	y	e	a	r	s	a	g	o
N	U	R	B	K	S	U	B	V	C	G	Q	K	V	E	V	G	Z	V	C	B	K	C	F	U

Finally, it is customary to write the ciphertext in five letter blocks:

```
NURBK SUBVC GQKVE VGZVC BKCFU
```

Decryption is a similar process. Suppose that we receive the message

```
GVVQG VYKCM CQQBV KKWGF SCVKV B
```

mathematics and computer science whose denizens are X and Y!

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
C	I	S	Q	V	N	F	O	W	A	X	M	T	G	U	H	P	B	K	L	R	E	Y	D	Z	J

Table 1.1: Simple substitution encryption table

j	r	a	x	v	g	n	p	b	z	s	t	l	f	h	q	d	u	c	m	o	e	i	k	w	y
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Table 1.2: Simple substitution decryption table

and that we know that it was encrypted using Table 1.1. We can reverse the encryption process by find each ciphtertext letter in the second row of Table 1.1 and writing down the corresponding letter from the top row. However, since the letters in the second row of Table 1.1 are all mixed up, this is a somewhat inefficient process. It is better to make a decryption table in which the ciphertext letters in the lower row are listed in alphabetical order and the corresponding plaintext letters in the upper row are mixed up. We have done this in Table 1.2 Using this table, we easily decrypt the message.

G	V	V	Q	G	V	Y	K	C	M	C	Q	Q	B	V	K	K	W	G	F	S	C	V	K	V	B
n	e	e	d	n	e	w	s	a	l	a	d	d	r	e	s	s	i	n	g	c	a	e	s	e	r

Putting in the appropriate word breaks and some punctuation reveals an urgent request!

```
Need new salad dressing.  -Caesar
```

1.1.1 Cryptanalysis of simple substitution ciphers

How many different simple substitution ciphers exist? We can count them by enumerating the possible ciphertext values for each plaintext letter. First we assign the plaintext letter a to one of the 26 possible ciphertext letters A–Z. So there are 26 possibilities for a. Next, since we are not allowed to assign b to the same letter as a, we may assign b to any one of the remaining 25 ciphertext letters. So there are $26 \cdot 25 = 650$ possible ways to assign a and b. We've now used up two of the ciphertext letters, so we may assign c to any one of the remaining 24 ciphertext letters. And so on.... Thus the total number of ways to assign the 26 plaintext letters to the 26 ciphertext letters, using each ciphertext letter only once, is

$$26 \cdot 25 \cdot 24 \cdots 4 \cdot 3 \cdot 2 \cdot 1 = 26! = 403291461126605635584000000.$$

There are thus approximately $10^{26.6}$ different simple substitution ciphers.

Suppose that Eve intercepts one of Bob's messages and that she attempts to decrypt it by trying every possible simple substitution cipher. The process of decrypting a message without knowing the underlying secret (in this case, the encryption table) is called *cryptanalysis*. If Eve (or her computer) is able to check one million cipher alphabets per second, it would still take her more than 10^{13} years to try them all. But the age of the universe is estimated to be on the order of 10^{10} years, so Eve has almost no chance of decrypting Bob's message. Thus Bob's message is secure and he has nothing to worry about![4] Or does he?

It is time for an important lesson in the practical side of the science of cryptology:

> Your opponent always uses her best strategy to defeat you, not the strategy that you want her to use. Thus the security of an encryption system depends on the <u>best known</u> method to break it. As new and improved methods are developed, the level of security can only get worse, never better.

Despite the large number of possible simple substitution ciphers, they are actually quite easy to break, and indeed many newspapers and magazines feature them as a companion to the daily crossword puzzle. The reason that Eve can easily cryptanalyze a simple substitution cipher is due to the fact that the letters in the English language (or any other language) are not random. To take an extreme example, the letter `q` in English is virtually always followed by the letter `u`. More useful is the fact that certain letters such as `e` and `t` appear far more frequently than other letters such as `f` and `c`. Table 1.3 lists the letters with their typical frequency in English text. As you can see, the most frequent letter is `e`, followed by `t`, `a`, `o`, and `n`.

[4] The assertion that a large number of possible keys, in and of itself, makes a cryptosystem secure, has appeared many times in history and has equally often been shown to be fallacious.

By decreasing frequency			
E	13.11%	M	2.54%
T	10.47%	U	2.46%
A	8.15%	G	1.99%
O	8.00%	Y	1.98%
N	7.10%	P	1.98%
R	6.83%	W	1.54%
I	6.35%	B	1.44%
S	6.10%	V	0.92%
H	5.26%	K	0.42%
D	3.79%	X	0.17%
L	3.39%	J	0.13%
F	2.92%	Q	0.12%
C	2.76%	Z	0.08%

In alphabetical order			
A	8.15%	N	7.10%
B	1.44%	O	8.00%
C	2.76%	P	1.98%
D	3.79%	Q	0.12%
E	13.11%	R	6.83%
F	2.92%	S	6.10%
G	1.99%	T	10.47%
H	5.26%	U	2.46%
I	6.35%	V	0.92%
J	0.13%	W	1.54%
K	0.42%	X	0.17%
L	3.39%	Y	1.98%
M	2.54%	Z	0.08%

Table 1.3: Frequency of letters in English text

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY LYLYD
LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
```

Table 1.4: A simple substitution cipher to cryptanalyze

Thus if Eve counts the letters in Bob's encrypted message and makes a frequency table, it is likely the that most frequent letter will represent `e`, and that `t`, `a`, `o`, and `n` will appear among the next most frequent letters. In this way, Eve can try various possibilities and, after a certain amount of trial-and-error, decrypt Bob's message.

In the remainder of this section we illustrate how to cryptanalyze a simple substitution cipher by decrypting the following message given in Table 1.4

There are 298 letters in the ciphertext. The first step is to make a frequency table listing how often each ciphtertext letter appears. We have done so in Table 1.5.

	J	L	D	G	Y	S	O	N	M	P	E	V	Q	C	T	W	U	K	I	X	Z	B	A	F	R	H
Freq	32	28	27	24	23	22	19	18	17	15	12	12	8	8	7	6	6	5	4	3	1	1	0	0	0	0
%	11	9	9	8	8	7	6	6	6	5	4	4	3	3	2	2	2	2	1	1	0	0	0	0	0	0

Table 1.5: Frequency table for Table 1.4 — Ciphertext length: 298

th	he	an	re	er	in	on	at	nd	st	es	en	of	te	ed
168	132	92	91	88	86	71	68	61	53	52	51	49	46	46

(a) Most common English bigrams (frequency per 1000 words)

LO	OJ	GY	DN	VD	YL	DL	DM	SN	KD	LY	NG	OY	JD	SK	EP	JG	SV	JM	JQ
9	7	6 each		5 each					4 each										

(b) Most common bigrams appearing in the ciphertext in Table 1.4

Table 1.6: Bigram frequencies

The ciphertext letter J appears most frequently, so we make the provisional guess that it is equal to the plaintext letter e. The next most frequent ciphertext letters are L (28 times) and D (27 times), so we might guess from Table 1.3 that they represent t and a. However, the letter frequencies in a short message are unlikely to exactly match the percentages in Table 1.3. All we can say is that among the ciphertext letters L, D, G, Y and S are likely to appear several of the plaintext letters t, a, o, n, and r.

There are several ways to proceed. One method is to look at *bigrams*, which are pairs of consecutive letters. Table 1.6(a) lists the most frequently appearing bigrams in the English text and Table 1.6(b) lists the ciphertext bigrams that appear most frequently in our message. The ciphertext bigrams LO and OJ appear frequently. We have already guessed that $\texttt{J} = \texttt{e}$, and based on its frequency we suspect that L is likely to represent one of the letters t, a, o, n, or r. Since the two most frequent English bigrams are th and he, we make the tentative identifications

$$\texttt{LO} = \texttt{th} \qquad \text{and} \qquad \texttt{OJ} = \texttt{he}.$$

We substitute the guesses $\texttt{J} = \texttt{e}$, $\texttt{L} = \texttt{t}$, and $\texttt{O} = \texttt{h}$, into the ciphertext, writing the putative plaintext letter below the corresponding ciphertext letter.

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
the-- -te-- ----e ----- --e-t ---e- --e-- ----t --t-h -----

GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
---e- ----- --e-- --e-e t---t h---- ----- ---tt h---h t-h--

ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
----- ----- --e-e ----- ----- -e--- ----- ----- --t-- ----e

CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY LYLYD
----- ---t- -t--- ----- -h--- e---t ----e --t-t he--- t-t--

LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
te-th -t--t --the --e-- -e-th e---- e--e- ---h- -hheh -----

YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
--e-- tthe- the-- --ht- e---- ----e -h--- ---e- ----- -e-
```

At this point, we can look at the fragments of plaintext and attempt to guess some common English words. For example, in the second line we see the three blocks

```
VSGLL OSCIO LGOYG,
---tt h---h t-h--.
```

Looking at the fragment `th---ht`, the natural guess is that this is the word `thought`, which gives three more equivalences,

$$\mathtt{S} = \mathtt{o}, \qquad \mathtt{C} = \mathtt{u}, \qquad \mathtt{I} = \mathtt{g}.$$

This yields

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
the-- -te-- ----e ----- o-e-t ---e- --e-- -o--t --t-h o---u
GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
---eo --g-- --eo- --e-e to--t ho--- ----- -o-tt hough t-h--
ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
-o--- u--o- --e-e ----- ----- -e--- o---- --o-o --t-o --o-e
CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY LYLYD
u---- -o-t- -t--- g-ou- -h--- e-u-t ----e --tot heu-- t-t--
LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
te-th -tu-t --the --e-- -e-th e--o- e--e- ---h- -hheh -----
YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
--e-- tthe- the-- -ght- e---o ----e -h--- ---e- -o--- -e-
```

Now look at the three letters ght in the last line. They must be preceded by a vowel, and the only vowels left are a and i, so we guess that Y = i. Then we find the letters itio in the third line, and we guess that they are followed by an n, which gives N = n. (There is no reason that a letter cannot represent itself, although this is often forbidden in the puzzle ciphers that appear in newspapers.) We now have

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
the-- ite-- --i-e ----- o-ent ---e- --e-- ion-t -it-h o---u
GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
---eo --g-- n-eo- -ne-e to--t ho--- -n-in -o-tt hough t-hi-
ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
-on-- u-ion --e-e --in- ---i- -e--- o--n- --o-o -itio n-o-e
CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY LYLYD
u--i- -o-t- -t-in g-ou- -hi-- e-u-t ----e --tot heuni titi-
LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
te-th -tunt i-the --e-- ne-th e--o- e--e- ---hi -hheh -----
YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
i-e-- tthe- the-- ight- e---o n-i-e -hi-- --ne- -o--n -e-
```

So far, we have reconstructed the following plaintext/ciphertext pairs:

	J	L	D	G	Y	S	O	N	M	P	E	V	Q	C	T	W	U	K	I	X	Z	B	A	F	R	H
	e	t	-	-	i	o	h	n	-	-	-	-	-	u	-	-	-	-	g	-	-	-	-	-	-	-
Freq	32	28	27	24	23	22	19	18	17	15	12	12	8	8	7	6	6	5	4	3	1	1	0	0	0	0

Recall that the most common letters in English (Table 1.3) are, in order of decreasing frequency,

e, t, a, o, n, r, i, s, h.

We have already assigned ciphertext values to e, t, o, n, i, h, so we guess that D and G represent two of the three letters a, r, s. In the third line we notice that GYLYSN gives -ition, so clearly G must by s. Similarly, on the fifth line we have LJQLO DLCNL equal to th-th -tunt, so D must be a, not r. Subsutituting these new pairs G = s and D = a gives

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
the-- ite-- -ai-e ---a- o-ent a--e- --ess ionat -it-h o-a-u

GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
s--eo -ag-a n-eo- ane-e to-at ho-a- ansin -ostt hough tshis

ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
-on-- usion s-e-e asin- a--i- -eass o-an- --o-o sitio nso-e

CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY LYLYD
u--i- sosta -t-in g-ou- -his- esu-t sa--e a-tot heuni titia

LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
te-th atunt i-the --ea- ne-th e--o- esses ---hi -hheh a-a--

YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
i-e-a tthe- the-- ight- e---o nsi-e -hi-a sane- -o-an -e-
```

It is now easy to fill in additional pairs by inspection. For example, the missing letter in the fragment atunt i-the on the fifth line must be l, which gives P = l, the missing letter in the fragment -osition on the third line must be p, which gives W = p. Substituting these in, we find the fragment e-p-ession on the first line, which gives Z = x and M = r, and the fragment -on-lusion on the third line, which gives E = c. Then consi-er on the last line gives Q = d and the initial words the-riterclai-e- must be the phrase "the writer claimed," yielding U = w and V = m. This gives

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
thewr iterc laime d--am oment ar-ex press ionat witch o-amu

GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
scleo ragla nceo- ane-e to-at homam ansin mostt hough tshis

ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
concl usion swere asin- alli- leass oman- propo sitio nso-e

CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY LYLYD
uclid sosta rtlin gwoul dhisr esult sappe artot heuni titia

LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
tedth atunt ilthe -lear nedth eproc esses --whi chheh adarr

YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
i-eda tthem the-m ightw ellco nside rhima sanec roman cer
```

We leave to the reader the task of filling in the few remaining letters and putting in the appropriate word breaks, capitalization, and punctuation to recover the plaintext:

> The writer claimed by a momentary expression, a twitch of a muscle or a glance of an eye, to fathom a man's inmost thoughts. His conclusions were as infallible as so many propositions of Euclid. So startling would his results appear to the unititiated that until they learned the processes by which he had arrived at them they might well consider him as a necromancer.[5]

1.2 Divisibility and greatest common divisors [M]

Modern cryptology is built on the foundation of algebra and number theory. So before we explore the subject of cryptology, we need to develop some important tools. In the next four sections we begin this development by describing and proving fundamental results from number theory and combinatorics. If you have already studied number theory in another course, a brief review of this material will suffice. But if this material is new to you, then it is vital to study it closely and to work out the exercises provided at the end of the chapter.

[5] *A Study in Scarlet* (Chapter 2), Sir Arthur Conan Doyle

At the most basic level, *Number Theory* is the study of the natural numbers

$$1, 2, 3, 4, 5, 6, \ldots,$$

or slightly more generally, the study of the integers

$$\ldots, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, \ldots.$$

The set of integers is denoted by the symbol $\mathbb{Z}$. Integers can be added, subtracted, and multiplied in the usual way, and they satisfy all the usual rules of arithmetic (commutative law, associative law, distributive law, etc.). The set of integers with their addition and multiplication rules are an example of a *ring*.

If a and b are integers, then we can always add them, $a + b$, subtract them, $a - b$, and multiply them, $a \cdot b$. In each case, we get an integer as the result. But if we want to use only integers, then we may not be able to divide them. For example, we cannot divide 3 by 2, since there is no integer that is equal to $\frac{3}{2}$. This leads to the fundamental concept of divisibility.

Definition. Let a and b be integers with $b \neq 0$. We say that b *divides* a, or that a *is divisible by* b, if there is an integer c such that

$$a = bc.$$

We write $b|a$ to indicate that b divides a. If b does not divide a, then we write $b \nmid a$.

Example 1.2. We have $847|485331$, since $485331 = 847 \cdot 573$. On the other hand, $355 \nmid 259943$, since when we try to divide 259943 by 355, we get a remainder of 83. More precisely, $259943 = 355 \cdot 732 + 83$, so 259943 is not an exact multiple of 355.

Remark 1.3. Notice that every integer is divisible by 1. The integers that are divisible by 2 are the *even integers*, and the integers that are not divisible by 2 are the *odd integers*.

There are a number of elementary divisibility properties, some of which we list in the following proposition.

Proposition 1.4. *Let $a, b, c \in \mathbb{Z}$ be integers.*

(a) *If $a|b$ and $b|c$, then $a|c$.*

(b) *If $a|b$ and $b|a$, then $a = \pm b$.*

(c) *If $a|b$ and $a|c$, then $a|(b+c)$ and $a|(b-c)$.*

Proof. We leave the proof as an exercise for the reader. □

Definition. A *common divisor* of two integers a and b is a positive integer d that divides both of them. The *greatest common divisor* of a and b is, as its name suggests, the largest postive integer d such that $d|a$ and $d|b$. The greatest common divisor of a and b is denoted $\gcd(a,b)$. (If a and b are both 0, then $\gcd(a,b)$ is not defined.)

Example 1.5. The greatest common divisor of 12 and 18 is 6, since $6|12$ and $6|18$ and there is no larger number with this property. Similarly,

$$\gcd(748, 2024) = 44.$$

One way to check that this is correct is to make lists of all of the positive divisors of 748 and of 2024.

$$\begin{aligned} \text{Divisors of } 748 &= \{1,2,4,11,17,22,34,44,68,187,374,748\} \\ \text{Divisors of } 2024 &= \{1,2,4,8,11,22,23,44,46,88,92,184,253, \\ &\qquad\qquad\qquad\qquad\qquad\qquad 506,1012,2024\} \end{aligned}$$

Examining the two lists, we see that the largest common entry is 44. Even from this small example, it is clear that we need a more efficient way to compute greatest common divisors!

The key to an efficient algorithm for computing greatest common divisors is *division with remainder*, which is simply the method of "long division" that you learned in elementary school. Thus if a and b are positive integers and if you attempt to divide a by b, you will get a quotient q and a remainder r, where the remainder r is smaller than b. For example,

```
      ___13 R 9
17 ) 230
     17_
      60
     _51
       9
```

so

230 divided by 17 gives quotient 13 with a remainder of 9.

What does this last statement really mean? It can be rewritten as the equation

$$230 = 17 \cdot 13 + 9.$$

Definition. Let a and b be positive integers. Then *a divided by b has quotient q and remainder r* means that

$$a = b \cdot q + r \qquad \text{with } 0 \le r < b.$$

Suppose now that we want to find the greatest common divisor of a and b. We first divide a by b to get

$$a = b \cdot q + r. \tag{1.1}$$

Suppose that d is a common divisor of a and b. Then it is clear from the formula (1.1) that d is also a divisor of r. Similarly, if e is a common divisor of b and r, then (1.1) shows that e is a divisor of a. In other words, the common divisors of a and b are the same as the common divisors of b and r, hence

$$\gcd(a, b) = \gcd(b, r).$$

Now repeat the process by dividing b by r to get a certain quotient and remainder, say

$$b = r \cdot q' + r' \qquad \text{with } 0 \le r' < r.$$

Then the same reasoning shows that

$$\gcd(b, r) = \gcd(r, r').$$

Continuing this process, the remainders get smaller and smaller, until eventually we get a remainder of 0, at which point the final value $\gcd(s, 0) = s$ is equal to the gcd of a and b.

We illustrate with an example and then describe the general method, which goes by the name of the *Euclidean algorithm.*

Example 1.6. We compute $\gcd(2024, 748)$ using the Euclidean algorithm, which is nothing more than repeated division with remainder. Notice how the quotient and remainder on each line become the new a and b on the subsequent line.

$$\begin{aligned}
2024 &= 748 \cdot 2 + 528 \\
748 &= 528 \cdot 1 + 220 \\
528 &= 220 \cdot 2 + 88 \\
220 &= 88 \cdot 2 + 44 \qquad \longleftarrow \boxed{\gcd = 44} \\
88 &= 44 \cdot 2 + 0
\end{aligned}$$

Theorem 1.7 (The Euclidean Algorithm)**.** *Let a and b be positive integers. The following algorithm computes* $\gcd(a,b)$ *in a finite number of steps.*

(1) *If $a < b$, switch a and b. Let $r_0 = a$ and $r_1 = b$.*

(2) *Set $i = 1$.*

(3) *Divide r_{i-1} by r_i to get a quotient and remainder*

$$r_{i-1} = r_i \cdot q_i + r_{i+1} \qquad \textit{with} \quad 0 \le r_{i+1} < r_i.$$

(4) *If the remainder $r_{i+1} = 0$, then $r_i = \gcd(a,b)$. Return the value r_i.*

(5) *Otherwise $r_{i+1} > 0$, so set $i = i + 1$ and go to Step 3.*

The division step (*Step 3*) *is executed at most*

$$2\log_2(\min\{a,b\}) + 2 \quad \textit{times.}$$

Proof. The Euclidean algorithm consists of a sequence of divisions with remainder as illustrated in Figure 1.2 (remember that we set $r_0 = a$ and $r_1 = b$).

$$\boxed{\begin{array}{rll} a &= b \ \ \cdot q_1 + r_2 & \text{with } 0 \le r_2 < b, \\ b &= r_2 \cdot q_2 + r_3 & \text{with } 0 \le r_3 < r_2, \\ r_2 &= r_3 \cdot q_3 + r_4 & \text{with } 0 \le r_4 < r_3, \\ r_3 &= r_4 \cdot q_4 + r_5 & \text{with } 0 \le r_5 < r_4, \\ \vdots & \quad \vdots & \qquad\qquad \vdots \\ r_{t-2} &= r_{t-1} \cdot q_{t-1} + r_t & \text{with } 0 \le r_t < r_{t-1}, \\ r_{t-1} &= r_t \cdot q_t & \\ & \text{Then } r_t = \gcd(a,b). & \end{array}}$$

Figure 1.2: The Euclidean algorithm step-by-step

The r_i values are strictly decreasing, and as soon as they reach zero the algorithm terminates, which proves that the algorithm does finish in a finite number of steps. Further, at each iteration of Step 3 we have an equation of the form

$$r_{i-1} = r_i \cdot q_i + r_{i+1}.$$

This equation implies that any common divisor of r_{i-1} and r_i is also a divisor of r_{i+1}, and similarly it implies that any common divisor of r_i and r_{i+1} is also a divisor of r_{i-1}. Hence

$$\gcd(r_{i-1}, r_i) = \gcd(r_i, r_{i+1}) \qquad \text{for all } i = 1, 2, 3, \ldots. \tag{1.2}$$

However, as noted above, we eventually get to an r_i that is zero, say $r_{t+1} = 0$. Then $r_{t-1} = r_t \cdot q_t$, so

$$\gcd(r_{t-1}, r_t) = \gcd(r_t \cdot q_t, r_t) = r_t.$$

But (1.2) says that this is equal to $\gcd(r_0, r_1)$, i.e. to $\gcd(a, b)$, which completes the proof that the last nonzero remainder in the Euclidean algorithm is equal to the greatest common divisor of a and b.

It remains to estimate the efficiency of the algorithm. We noted above that since the r_i values are strictly decreasing, the algorithm terminates, and indeed since $r_1 = b$, it certainly terminates in at most b steps. However, this upper bound is far from the truth. We claim that after every two iterations of Step 3, the value of r_i is at least cut in half. In other words:

$$\textbf{Claim:} \quad r_{i+2} < \tfrac{1}{2} r_i \quad \text{for all } i = 0, 1, 2, \ldots.$$

We prove the claim by considering two cases.

Case I: $r_{i+1} \le \frac{1}{2} r_i$

We know that the r_i values are strictly decreasing, so

$$r_{i+2} < r_{i+1} \le \tfrac{1}{2} r_i.$$

Case II: $r_{i+1} > \frac{1}{2} r_i$

Consider what happens when we divide r_i by r_{i+1}. The value of r_{i+1} is so large that we get

$$r_i = r_{i+1} \cdot 1 + r_{i+2} \quad \text{with} \quad r_{i+2} = r_i - r_{i+1} < r_i - \tfrac{1}{2} r_i < \tfrac{1}{2} r_i.$$

We have now proven our claim that $r_{i+2} < \frac{1}{2} r_i$ for all i. Using this inequality repeatedly, we find that

$$r_{2k+1} < \frac{1}{2} r_{2k-1} < \frac{1}{4} r_{2k-3} < \frac{1}{8} r_{2k-5} < \frac{1}{16} r_{2k-7} < \cdots < \frac{1}{2^k} r_1 = \frac{1}{2^k} b.$$

Hence if $2^k \ge b$, then $r_{2k+1} < 1$, which forces r_{2k+1} to equal 0 and the algorithm to terminate. In other words, if $k \ge \log_2(b)$, then $r_{2k+1} = 0$, so the Euclidean algorithm terminates in at most $2k + 1$ iterations. In particular, it terminates in no more than $2\log_2(b) + 2$ iterations, which completes the proof of Theorem 1.7. □

Remark 1.8. We proved that the Euclidean algorithm applied to a and b (with $b < a$) requires no more than $2\log_2(b) + 2$ iterations to compute $\gcd(a, b)$. It is possible to improve this estimate somewhat. More precisely, one can show that the Euclidean algorithm takes no more than $1.45\log_2(b) + 1.68$ iterations, and that for a randomly chosen pair, it takes approximately $0.85\log_2(b) + 0.14$ iterations. (See [26].)

Remark 1.9. One way to compute quotients and remainders is by long division, as we did on page 13. If you have a calculator, you can use it to speed up the process. The first step is to divide a by b on your calculator, which will give a real number. Throw away the part after the decimal point to get the quotient q. Then the remainder r can be computed as

$$r = a - b \cdot q.$$

For example, let $a = 2387187$ and $b = 27573$. Then $a/b \approx 86.57697748$, so $q = 86$ and

$$r = a - b \cdot q = 2387187 - 27573 \cdot 86 = 15909.$$

If you just need the remainder, you can instead take the decimal part (also sometimes called the *fractional part*) of a/b and multiply it by b. Continuing with our example, the decimal part of $a/b \approx 85.57697748$ is 0.57697748, and multiplying by $b = 27573$ gives

$$27573 \cdot 0.57697748 = 15909.00005604.$$

Rounding this off gives $r = 15909$.

After performing the Euclidean algorithm on two numbers, we can work our way back down the process to obtain an extremely interesting formula. Before giving the general result, we illustrate with an example.

Example 1.10. Recall that in Example 1.6 we used the Euclidean to compute $\gcd(2024, 748)$ as follows:

$$\begin{aligned}
2024 &= 748 \cdot 2 + 528 \\
748 &= 528 \cdot 1 + 220 \\
528 &= 220 \cdot 2 + 88 \\
220 &= 88 \cdot 2 + 44 \qquad \leftarrow \boxed{\gcd = 44} \\
88 &= 44 \cdot 2 + 0
\end{aligned}$$

We let $a = 2024$ and $b = 748$, so the first line says that

$$528 = a - 2b.$$

We substitute that into the second line to get

$$b = (a - 2b) \cdot 1 + 220, \qquad \text{so} \qquad 220 = -a + 3b.$$

We next substitute the expressions $528 = a - 2b$ and $220 = -a + 3b$ into the third line to get

$$a - 2b = (-a + 3b) \cdot 2 + 88, \qquad \text{so} \qquad 88 = 3a - 8b.$$

Finally, we substitute the expressions $220 = -a + 3b$ and $88 = 3a - 8b$ into the penultimate line to get

$$-a + 3b = (3a - 8b) \cdot 2 + 44, \qquad \text{so} \qquad 44 = -7a + 19b.$$

In other words,

$$-7 \cdot 2024 + 19 \cdot 748 = 44 = \gcd(2024, 748),$$

so we have found a way to write $\gcd(a, b)$ as a linear combination of a and b using integer coefficients.

In general, it always possible to write $\gcd(a, b)$ as an integer linear combination of a and b.

Theorem 1.11 (Extended Euclidean Algorithm)**.** *Let a and b be positive integers. Then the equation*

$$au + bv = \gcd(a, b)$$

always has a solution in integers u and v. (*See Exercise 1.11 for an efficient algorithm to find a solution.*)

If (u_0, v_0) is any one solution, then every solution has the form

$$u = u_0 + \frac{b \cdot k}{\gcd(a, b)} \quad \textit{and} \quad v = v_0 - \frac{a \cdot k}{\gcd(a, b)} \qquad \textit{for some } k \in \mathbb{Z}.$$

Proof. Look back at Figure 1.2, which illustrates the Euclidean algorithm step-by-step. We can solve the first line for $r_2 = a - b \cdot q_1$ and substitute it into the second line to get

$$b = (a - b \cdot q_1) \cdot q_2 + r_3, \qquad \text{so} \qquad r_3 = -a \cdot q_1 + b \cdot (1 + q_1 q_2).$$

Next substitute the expressions for r_2 and r_3 into the third line to get

$$a - b \cdot q_1 = \bigl(-a \cdot q_1 + b \cdot (1 + q_1 q_2)\bigr) q_3 + r_4.$$

After rearranging the terms, this gives

$$r_4 = a \cdot (1 + q_1 q_3) - b \cdot (q_1 + q_3 + q_1 q_2 q_3).$$

The key point is that $r_4 = a \cdot u + b \cdot v$, where u and v are integers. It does not matter that the expressions for u and v in terms of q_1, q_2, q_3 are rather messy. Continuing in this fashion, at each stage we find that r_i is the sum of an integer multiple of a and an integer multiple of b. Eventually, we get to $r_t = a \cdot u + b \cdot v$ for some integers u and v. But $r_t = \gcd(a, b)$, which completes the proof of the first part of the theorem. We leave the second part as an exercise (Exercise 1.10). □

1.3 Modular arithmetic [M]

A convenient way to deal with the subject of divisibility is through the notion of congruences. You may have encountered "clock arithmetic" in grade school, where after you get to 12, the next number is 1.

Definition. Let $m \geq 1$ be an integer. We say that the integers a and b are *congruent modulo* m if their difference $a - b$ is divisible by m. We write

$$a \equiv b \pmod{m}$$

to indicate that a and b are congruent modulo m.

Example 1.12. We have

$$17 \equiv 7 \pmod{5}, \qquad \text{since 5 divides } 17 - 7 = 10.$$

On the other hand,

$$19 \not\equiv 6 \pmod{11}, \qquad \text{since 11 does not divide } 19 - 6 = 13.$$

Notice that the numbers satisfying

$$a \equiv 0 \pmod{m}$$

are the number that are divisible by m, i.e. the multiples of m.

The reason that congruence notation is so useful is that congruences behave much like equalities, as the following proposition indicates.

Proposition 1.13. *Let $m \geq 1$ be an integer.*
(a) *If $a_1 \equiv a_2$ (mod m) and $b_1 \equiv b_2$ (mod m), then*

$$a_1 \pm b_1 \equiv a_2 \pm b_2 \ (mod\ m) \qquad \textit{and} \qquad a_1 \cdot b_1 \equiv a_2 \cdot b_2 \ (mod\ m).$$

(b) *If $\gcd(a, m) = 1$, then there is an integer b satisfying*

$$a \cdot b \equiv 1 \pmod{m}.$$

We say that b is a (multiplicative) inverse of a modulo m. *Conversely, if $\gcd(a, m) > 1$, then a does not have an inverse modulo m.*

Proof. (a) We leave this as an exercise.
(b) Suppose first that $\gcd(a, m) = 1$. Then Theorem 1.11 tells us that we can find integers u and v satisfying $au + mv = 1$. This means that $au - 1 = -mv$ is divisible by m, so by definition, $au \equiv 1 \pmod{m}$. In other words, we can take $b = u$. Next suppose that a has an inverse modulo m, say $a \cdot b \equiv 1 \pmod{m}$. This means that $ab - 1 = cm$ for some integer c. It follows that $\gcd(a, m)$ divides $ab - cm = 1$, so $\gcd(a, m) = 1$. This completes the proof that a has an inverse modulo m if and only if $\gcd(a, m) = 1$. □

Remark 1.14. The statement of Proposition 1.13(b) says that if $\gcd(a, m) = 1$, then there exists an inverse b of a modulo m. In order to compute b, we use the Extended Euclidean Algorithm (Theorem 1.11), which tells us that there are integers u and v satisfying

$$au + mv = \gcd(a, m) = 1. \tag{1.3}$$

Further, we can easily compute u and v by either applying the Euclidean algorithm directly as we did in Example 1.10 or by using the somewhat more efficient method described in Exercise 1.11. Notice that if we reduce (1.3) modulo m, we get $au \equiv 1 \pmod{m}$, so the quantity u is the desired inverse of a modulo m.

$+$	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

$\cdot$	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Figure 1.3: Addition and multiplication tables modulo 5

If a divided by m has quotient q and remainder r, it can be written as

$$a = m \cdot q + r \qquad \text{with } 0 \le r < m.$$

This shows that $a \equiv r \pmod{m}$ for some integer r between 0 and $r - 1$, so if we want to work with integers modulo m, it is enough to use the integers $0 \le r < m$. We write

$$\mathbb{Z}/m\mathbb{Z} = \{0, 1, 2, \ldots, m-1\}$$

and call $\mathbb{Z}/m\mathbb{Z}$ the *ring of integers modulo* m, where addition and multiplication are always done modulo m. Figure 1.3 illustrates this by giving addition and multiplication tables modulo 5.

Remark 1.15. If you have studied ring theory, you will recognize that $\mathbb{Z}/m\mathbb{Z}$ is the quotient ring of $\mathbb{Z}$ by the principal ideal $m\mathbb{Z}$, and that the numbers $0, 1, \ldots, m-1$ are actually coset representatives for the cosets that comprise the elements of $\mathbb{Z}/m\mathbb{Z}$.

Proposition 1.13(b) tells us that $\gcd(a, m) = 1$ if and only if a has an inverse modulo m, so we define

$$\begin{aligned}(\mathbb{Z}/m\mathbb{Z})^* &= \{a \in \mathbb{Z}/m\mathbb{Z} : \gcd(a, m) = 1\} \\ &= \{a \in \mathbb{Z}/m\mathbb{Z} : a \text{ has an inverse modulo } m\}\end{aligned}$$

and we call $(\mathbb{Z}/m\mathbb{Z})^*$ the *group of units modulo* m. Notice that if a_1 and a_2 are units modulo m, then so is $a_1 a_2$, so we can multiply two units and get another unit. However, if we add two units, we may not get another one.

Example 1.16. The group of units modulo 24 is

$$(\mathbb{Z}/24\mathbb{Z})^* = \{1, 5, 7, 11, 13, 17, 19, 23\}.$$

The multiplication table for $(\mathbb{Z}/24\mathbb{Z})^*$ is illustrated in Figure 1.4.

·	1	5	7	11	13	17	19	23
1	1	5	7	11	13	17	19	23
5	5	1	11	7	17	13	23	19
7	7	11	1	5	19	23	13	17
11	11	7	5	1	23	19	17	13
13	13	17	19	23	1	5	7	11
17	17	13	23	19	5	1	11	7
19	19	23	13	17	7	11	1	5
23	23	19	17	13	11	7	5	1

Unit group modulo 24

·	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	4	6	1	3	5
3	3	6	2	5	1	4
4	4	1	5	2	6	3
5	5	3	1	6	4	2
6	6	5	4	3	2	1

Unit group modulo 7

Figure 1.4: The unit groups $(\mathbb{Z}/24\mathbb{Z})^*$ and $(\mathbb{Z}/7\mathbb{Z})^*$

Example 1.17. The group of units modulo 7 is

$$(\mathbb{Z}/7\mathbb{Z})^* = \{1, 2, 3, 4, 5, 6\},$$

since every number between 1 and 6 is relatively prime to 7. The multiplication table for $(\mathbb{Z}/7\mathbb{Z})^*$ is illustrated in Figure 1.4.

Definition. For reasons that will become apparent later, the number of elements in the unit group modulo m is an important quantity. We denote it by

$$\phi(m) = \#\,(\mathbb{Z}/m\mathbb{Z})^* = \#\{0 \le a < m : \gcd(a, m) = 1\}.$$

The function ϕ is called *Euler's phi function* (or sometimes *Euler's totient function*). Examples 1.16 and 1.17 say that $\phi(24) = 8$ and $\phi(7) = 6$.

1.3.1 Modular arithmetic and shift ciphers

Recall that the Caesar (or shift) cipher studied in Section 1.1 works by shifting each letter in the alphabet a fixed number of letters. We can describe a shift cipher mathematically by assigning a number to each letter as in Table 1.7.

Then a shift cipher with shift k takes a plaintext letter corresponding to the number p and assigns it to the ciphertext letter corresponding to the number $p + k \bmod 26$. Notice how the use of modular arithmetic, in this case

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Table 1.7: Assigning numbers to letters

modulo 26, simplifies the description of the shift cipher. The shift amount serves as both the encryption key and the decryption key. Encryption is given by the formula

$$(\text{Ciphertext Letter}) \equiv (\text{Plaintext Letter}) + (\text{Secret Key}) \pmod{26},$$

and decryption works by shifting in the opposite direction,

$$(\text{Plaintext Letter}) \equiv (\text{Ciphertext Letter}) - (\text{Secret Key}) \pmod{26}.$$

More succintly, if we let

$$p = \text{Plaintext Letter}, \qquad c = \text{Ciphertext Letter}, \qquad k = \text{Secret Key},$$

then

$$\underbrace{c \equiv p + k \pmod{26}}_{\text{Encryption}} \qquad \text{and} \qquad \underbrace{p \equiv c - k \pmod{26}}_{\text{Decryption}}.$$

1.3.2 The fast exponentiation algorithm

Many public key cryptosystems require Alice and Bob to compute large powers of a number g modulo another number p, where p may have hundreds of digits. The naive way to compute g^m is by repeated multiplication of g with itself,

$$g_1 \equiv g \pmod{p}, \qquad g_2 \equiv g \cdot g_1 \pmod{p}, \qquad g_3 \equiv g \cdot g_2 \pmod{p},$$
$$g_4 \equiv g \cdot g_3 \pmod{p}, \qquad g_5 \equiv g \cdot g_4 \pmod{p}, \ldots.$$

Then $g_m \equiv g^m \pmod{p}$, but if m is large, say $m \approx 2^{1000}$, then this naive algorithm takes longer than the estimated age of the universe! Clearly we need a better way to compute $g^m \pmod{p}$.

The idea is to use the binary expansion of the exponent m and to convert the calculation of g^m into a succession of squarings and mutiplications. An example makes the idea clear, after which we give a formal description of the method.

i	0	1	2	3	4	5	6	7
$3^{2^i} \pmod{1000}$	3	9	81	561	721	841	281	961

Table 1.8: Successive square powers of 3 modulo 1000

Example 1.18. Suppose that we want to compute $3^{218} \pmod{1000}$. The first step is to write 218 as a sum of powers of 2,

$$218 = 2 + 2^3 + 2^4 + 2^6 + 2^7.$$

Then 3^{218} becomes

$$3^{218} = 3^{2+2^3+2^4+2^6+2^7} = 3^2 \cdot 3^{2^3} \cdot 3^{2^4} \cdot 3^{2^6} \cdot 3^{2^7}. \tag{1.4}$$

Notice that it is relatively easy to compute the sequence of values

$$3, \quad 3^2, \quad 3^{2^2}, \quad 3^{2^3}, \quad 3^{2^4}, \ldots,$$

since each number in the sequence is the square of the preceding one. Further, since we only need these values modulo 1000, we never need to store more than three digits. Table 1.8 lists the powers of 3 modulo 1000 up to 3^{2^7}. Creating Table 1.8 required only 7 multiplications, despite the fact that of $3^{2^7} = 3^{128}$ has quite a large exponent, because each successive entry is equal to the square of the previous.

From Table 1.8, we pick out the powers 3^{2^i} needed to compute 3^{218} using (1.4). Thus

$$\begin{aligned} 3^{218} &= 3^2 \cdot 3^{2^3} \cdot 3^{2^4} \cdot 3^{2^6} \cdot 3^{2^7} \\ &\equiv 9 \cdot 561 \cdot 721 \cdot 281 \cdot 961 \pmod{1000} \\ &\equiv 489 \pmod{1000}. \end{aligned}$$

(Notice that in computing the product $9 \cdot 561 \cdot 721 \cdot 281 \cdot 961$, we may reduce modulo 1000 after each multipication, so we never need to deal with very large numbers.)

In Example 1.18, we computed $3^{218} \pmod{1000}$ using only 11 multiplications. This is a huge savings over the naive approach, and if the exponent had been larger, we would have saved even more.

The general approach, which is variously called the *Fast Powering Algorithm* or the *Square-and-Multiply Algorithm*, is clear from the example. The following steps compute $g^m \pmod{p}$ (where p need not be prime) in no more than $2\log(m)$ steps.[6]

Step 1. Compute the binary expansion of m as

$$m = m_0+m_1\cdot 2+m_2\cdot 2^2+m_3\cdot 2^3+\cdots+m_r\cdot 2^r \quad \text{with } m_0,\ldots,m_r \in \{0,1\}.$$

Step 2. Compute the powers g^{2^i} for $0 \le i \le r$ by successive squaring,

$$\begin{aligned}
a_0 &\equiv g && \pmod{p}\\
a_1 &\equiv a_0^2 \equiv g^2 && \pmod{p}\\
a_2 &\equiv a_1^2 \equiv g^{2^2} && \pmod{p}\\
a_3 &\equiv a_2^2 \equiv g^{2^3} && \pmod{p}\\
\vdots &\quad \vdots && \quad \vdots\\
a_r &\equiv a_{r-1}^2 \equiv g^{2^r} && \pmod{p}.
\end{aligned}$$

Each term is the square of the previous one, so this requires r multiplications.

Step 3. Compute $g^m \pmod{p}$ using the formula

$$\begin{aligned}
g^m &= g^{m_0+m_1\cdot 2+m_2\cdot 2^2+m_3\cdot 2^3+\cdots+m_r\cdot 2^r}\\
&= g^{m_0}\cdot (g^2)^{m_1}\cdot (g^{2^2})^{m_2}\cdot (g^{2^3})^{m_3}\cdots (g^{2^r})^{m_r}\\
&\equiv a_0^{m_0}\cdot a_1^{m_1}\cdot a_2^{m_2}\cdot a_3^{m_3}\cdots a_r^{m_r} \pmod{p}. \qquad (1.5)
\end{aligned}$$

Note that the quantities $a_0, a_1, \ldots, a_r$ were computed in Step 2. Thus the product (1.5) can be computed by looking up the values of the g_i's whose exponent m_i is 1 and then multiplying them together. This requires at most another r multiplications.

There are various ways in which the square-and-multiply algorithm can be made somewhat more efficient (see Exercise 1.19 for example), but in any case, since $m \ge 2^r$, it takes no more than $2\log(m)$ multiplications to compute 2^m. Thus even if m is very large, say $m \approx 2^{1000}$, a computer has no trouble performing the approximately 2000 multiplications needed to calculate 2^m modulo p.

[6] Here's one of those confusing log situations, in this statement the quantity $\log(m)$ means the usual logarithm to the base 2, not the discrete logarithm.

1.4 Prime numbers and finite fields [M]

In Section 1.3 we studied modular arithmetic and saw that it makes sense to add, subtract, and multiply integers modulo m. Division is more problematic, since we can only divide by a in $\mathbb{Z}/m\mathbb{Z}$ if $\gcd(a, m) = 1$. However, if the integer m is a prime, then this is true for every nonzero element of $\mathbb{Z}/m\mathbb{Z}$. We start with a brief discussion of prime numbers before returning to the ring $\mathbb{Z}/p\mathbb{Z}$ with p prime.

Definition. An integer p is called a *prime* if $p \geq 2$ and if the only numbers dividing p are 1 and p.

For example, the first ten primes are $2, 3, 5, 7, 11, 13, 17, 19, 23, 29$, while the 100,000th prime is 1,299,709 and the 1,000,000th prime is 15,485,863. There are infinitely many primes, a fact that was known in ancient Greece and appears as a theorem in Euclid's *Elements.* (See Exercise 1.21.)

A prime p is defined in terms of the numbers that divide p. So the following useful proposition describing a property of numbers that are divisible by p is not obvious and needs to be carefully proved. Notice that the proposition is certainly false for composite numbers. For example, 6 divides $3 \cdot 10$, but 6 divides neither 3 nor 10.

Proposition 1.19. *Let p be a prime number, and suppose that p divides the product ab of two positive integers a and b. Then p divides at least one of a or b.*

Proof. Let $g = \gcd(a, p)$. Then $g|p$, so either $g = 1$ or $g = p$. If $g = p$, then $p|a$ (since $g|a$), so we are done. Otherwise, $g = 1$ and Theorem 1.11 tells us that we can find integers u and v satisfying $au + pv = 1$. We multiply both sides of the equation by b to get

$$abu + pbv = b. \tag{1.6}$$

By assumption, p divides the product ab, and certainly p divides pbv, so p divides both terms on the lefthand side of (1.6). Hence it divides the righthand side, which shows that p divides b and completes the proof of Proposition 1.19. □

Theorem 1.20 (The Fundamental Theorem of Arithmetic). *Let $a \geq 2$ be an integer. Then a can be factored as a product of prime numbers*

$$a = p_1^{e_1} \cdot p_2^{e_2} \cdot p_3^{e_3} \cdots p_r^{e_r}.$$

Further, other than rearranging the order of the primes, this factorization into prime powers is unique.

Proof. It is not hard to prove that every $a \geq 2$ can be factored into a product of primes, but the fact that the factorization is unique is considerably subtler, although most people assume that the uniqueness is obvious. A proof of the fundamental theorem of arithmetic may be found in any number theory textbook, for example [14, 21, 22, 39, 44, 55]. In particular, see [55, Chapter 7] for a brief discussion of why the uniqueness is not trivially true. □

Let p be a prime and let $a \geq 1$ be an integer. The Fundamental Theorem of Arithmetic says that in the factorization of a into primes, the prime p appears to a particular power. We denote this power by $\operatorname{ord}_p(a)$. (If $a = 1$, we set $\operatorname{ord}_p(a) = 0$.) For example, $\operatorname{ord}_2(1728) = 6$ and $\operatorname{ord}_3(1738) = 3$, since $1728 = 2^6 \cdot 3^3$.

Using the ord_p notation, the factorization of a can be succintly written as

$$a = \prod_{\text{primes } p} p^{\operatorname{ord}_p(a)}.$$

This product makes sense, since $\operatorname{ord}_p(a)$ is zero for all but finitely many primes.

Notice that ord_p is a function

$$\operatorname{ord}_p : \{1, 2, 3, \ldots\} \longrightarrow \{0, 1, 2, 3, \ldots\}. \tag{1.7}$$

This function has a number of interesting properties that you can explore in Exercise 1.23.

We now observe that if p is a prime, then every nonzero number modulo p has a multiplicative inverse modulo p. This means that when we do arithmetic modulo a prime p, we can add, subtract, multiply <u>and</u> divide by nonzero numbers, just as we do with real numbers.

Proposition 1.21. *Let p be a prime. Then every nonzero element a in $\mathbb{Z}/p\mathbb{Z}$ has a multiplicative inverse, that is, there is a number b so that*

$$ab \equiv 1 \pmod{p}.$$

We denote this value of b by $a^{-1} \bmod p$, or if p has already been specified, then simply by a^{-1}.

$1^1 \equiv 1$	$1^2 \equiv 1$	$1^3 \equiv 1$	$1^4 \equiv 1$	$1^5 \equiv 1$	$1^6 \equiv 1$
$2^1 \equiv 2$	$2^2 \equiv 4$	$2^3 \equiv 1$	$2^4 \equiv 2$	$2^5 \equiv 4$	$2^6 \equiv 1$
$3^1 \equiv 3$	$3^2 \equiv 2$	$3^3 \equiv 6$	$3^4 \equiv 4$	$3^5 \equiv 5$	$3^6 \equiv 1$
$4^1 \equiv 4$	$4^2 \equiv 2$	$4^3 \equiv 1$	$4^4 \equiv 4$	$4^5 \equiv 2$	$4^6 \equiv 1$
$5^1 \equiv 5$	$5^2 \equiv 4$	$5^3 \equiv 6$	$5^4 \equiv 2$	$5^5 \equiv 3$	$5^6 \equiv 1$
$6^1 \equiv 6$	$6^2 \equiv 1$	$6^3 \equiv 6$	$6^4 \equiv 1$	$6^5 \equiv 6$	$6^6 \equiv 1$

Table 1.9: Powers of numbers modulo 7

Proof. This proposition is the special case of Proposition 1.13(b) with $m = p$ taken to be a prime, since if $a \in \mathbb{Z}/p\mathbb{Z}$ is not zero, then $\gcd(a, p) = 1$. □

Definition. The set $\mathbb{Z}/p\mathbb{Z}$ of integers modulo p with its addition, subtraction, multiplication, and division rules is called a *field*. (You may recall from algebra that a field is a commutative ring in which every nonzero element has a multiplicative inverse.) You are certainly familiar with some other fields, for example the field of real numbers $\mathbb{R}$, the field of rational numbers (fractions) $\mathbb{Q}$, and the field of complex numbers $\mathbb{C}$. The field of integers modulo p has only finitely many elements, so it is called a *finite field* and is often denoted by $\mathbb{F}_p$. Of course, we also have the notation $\mathbb{Z}/p\mathbb{Z}$ from Section 1.3, so $\mathbb{F}_p$ and $\mathbb{Z}/p\mathbb{Z}$ are really just two different notations for the same object.[7] Finite fields are of fundamental importance throughout cryptography and, indeed, throughout all of mathematics.

The application of finite fields in cryptography often involves raising elements of $\mathbb{F}_p$ to high powers. As a practical matter, we know how to do this efficiently using the powering algorithm in Section 1.3.2. We now briefly consider the purely mathematical question of powers in $\mathbb{F}_p$ and prove a fundamental result due to Fermat.

We begin with a simple example. Table 1.9 lists the powers of $1, 2, 3, \ldots, 6$ modulo the prime 7. There are quite a few interesting patterns visible in Table 1.9, including in particular the fact that the righthand column consists

[7] Finite fields are also sometimes called *Galois fields*, after E. Galois, who studied them in the 19th century. Yet another notation for $\mathbb{F}_p$ is $\mathrm{GF}(p)$, in honor of Galois. And yet one more notation for $\mathbb{F}_p$ that you may run across is $\mathbb{Z}_p$, although $\mathbb{Z}_p$ is also commonly used to denote the p-adic integers.

entirely of ones. We can restate this observation by saying that

$$a^6 \equiv 1 \pmod{7} \qquad \text{for every } a = 1, 2, 3, \ldots, 6.$$

Of course, this cannot be true for all values of a, since if a is a multiple of 7, then so are all of its powers, so in that case $a^n \equiv 0 \pmod{7}$. On the other hand, if a is not divisible by 7, then a is congruent to one of the values $1, 2, 3, \ldots, 6$ modulo 7. Hence

$$a^6 \equiv \begin{cases} 1 & \text{if } 7 \nmid a, \\ 0 & \text{if } 7 \mid a. \end{cases}$$

Further experiments with other primes suggest that this example reflects a general fact.

Theorem 1.22 (Fermat's Little Theorem)**.** *Let p be a prime number and let a be any integer. Then*

$$a^{p-1} \equiv \begin{cases} 1 & \textit{if } p \nmid a, \\ 0 & \textit{if } p \mid a. \end{cases}$$

Proof. There are many proofs of Fermat's Little Theorem. If you have studied group theory, the quickest proof is to observe that the nonzero elements in $\mathbb{F}_p$ form a group $\mathbb{F}_p^*$ of order $p-1$, so by Lagrange's theorem, every element of $\mathbb{F}_p^*$ has order dividing $p-1$. For those who have not yet taken a course in group theory, we provide a direct proof.

If $p|a$, then it is clear that every power of a is divisible by p. So we only need to consider the case that $p \nmid a$. We now look at the list of numbers

$$a, \quad 2a, \quad 3a, \ldots, (p-1)a \qquad \text{reduced modulo } p. \tag{1.8}$$

There are $p-1$ numbers in this list, and we claim that they are all different. To see why, take any two of them, say $ja \bmod p$ and $ka \bmod p$, and suppose that they are the same. This means that

$$ja \equiv ka \pmod{p}, \qquad \text{and hence that} \qquad (j-k)a \equiv 0 \pmod{p}.$$

Thus p divides the product $(j-k)a$. Proposition 1.19 tells us that either p divides $j-k$ or p divides a. However, we have assumed that p does not divide a, so we conclude that p divides $j-k$. But both j and k are between 1 and $p-1$, so their difference $j-k$ is between $-(p-2)$ and $p-2$. There is only one number between $-(p-2)$ and $p-2$ that is divisible by p, and that number is zero! This proves that $j-k=0$, which means that $ja=ka$. We have thus shown that the $p-1$ numbers in the list (1.8) are all different. They are also nonzero, since $1,2,3,\ldots,p-1,a$ are not divisible by p.

To recapitulate, we have shown that the list of numbers (1.8) consists of $p-1$ distinct numbers between 1 and $p-1$. But there are only $p-1$ distinct numbers between 1 and $p-1$, so the list of numbers (1.8) must simply be the list of numbers $1,2,\ldots,p-1$ in some mixed up order.

Now consider what happens when we multiply together all of the numbers $a, 2a, 3a, \ldots, (p-1)a$ in the list (1.8) and reduce them modulo p. This is the same as multiplying together all of the numbers $1,2,3,\ldots,p-1$ modulo p, we get a congruence

$$a \cdot 2a \cdot 3a \cdots (p-1)a \equiv 1 \cdot 2 \cdot 3 \cdots (p-1) \pmod{p}.$$

There are $p-1$ copies of a appearing on the lefthand side. We factor these out and use factorial notation $(p-1)! = 1 \cdot 2 \cdots (p-1)$ to obtain

$$a^{p-1} \cdot (p-1)! \equiv (p-1)! \pmod{p}.$$

Finally, we are allowed to cancel $(p-1)!$ from both sides, since it is not divisible by p. (We are using the fact that $\mathbb{F}_p$ is a field, so we are allowed to divide by any nonzero number.) This yields

$$a^{p-1} \equiv 1 \pmod{p},$$

which completes the proof of Fermat's "Little" Theorem.[8] □

Example 1.23. The number $p = 15{,}485{,}863$ is prime, so Fermat's Little Theorem 1.22 tells us that

$$2^{15485862} \equiv 1 \pmod{15485863}.$$

Thus without doing any computing, we know that the number $2^{15485862}-1$, a number having more than two million digits, is a multiple of 15485863.

[8] You may wonder why Theorem 1.22 is called a "little" theorem. The reason is to distinguish it from Fermat's "Big" Theorem, which is the famous assertion that $x^n + y^n = z^n$ has no solutions in positive integers x, y, z if $n \geq 3$. It is unlikely that Fermat himself could prove this big theorem, but in 1996, more than three centuries after Fermat's era, Andrew Wiles finally found a proof.

Example 1.24. Consider the number $m = 15{,}485{,}207$. Using the powering algorithm, it is not hard to compute (on a computer)

$$2^{m-1} = 2^{15485206} \equiv 4136685 \pmod{15485207}.$$

We did not get the value 1, so it seems that Fermat's Little Theorem is not true for m. What does that tell us? Well, if m were prime, then Fermat's Little Theorem says that we would have gotten 1. Hence the fact that we did not get 1 proves that the number $m = 15{,}485{,}207$ is not prime. Think about this for a minute. By a simple computation, we have conclusively proven that m is not prime, yet we do not know any of its factors![9]

Fermat's Little Theorem describes a special property of the units (i.e. the nonzero elements) in a finite field. We conclude this section with a brief discussion of another property that is quite important both theoretically and practically.

Theorem 1.25 (Theorem of the Primitive Element)**.** *Let p be a prime number. Then there exists an element $g \in \mathbb{F}_p^*$ whose powers give every element of $\mathbb{F}_p^*$, i.e.*

$$\mathbb{F}_p^* = \{1, g, g^2, g^3, \ldots, g^{p-2}\}.$$

Elements with this property are called primtive roots of $\mathbb{F}_p$ *or* generators of $\mathbb{F}_p^*$.

Proof. See [55, Chapter 20] or one of the texts [14, 21, 22, 39, 44]. □

Example 1.26. The field $\mathbb{F}_{11}$ has 2 as a primitive root, since in $\mathbb{F}_{11}$,

$$\begin{array}{lllll} 2^0 = 1 & 2^1 = 2 & 2^2 = 4 & 2^3 = 8 & 2^4 = 5 \\ 2^5 = 10 & 2^6 = 9 & 2^7 = 7 & 2^8 = 3 & 2^9 = 6. \end{array}$$

On the other hand, 2 is not a primitive root for $\mathbb{F}_{17}$, since in $\mathbb{F}_{17}$,

$$\begin{array}{lllll} 2^0 = 1 & 2^1 = 2 & 2^2 = 4 & 2^3 = 8 & 2^4 = 16 \\ 2^5 = 15 & 2^6 = 13 & 2^7 = 9 & 2^8 = 1, & \end{array}$$

so we get back to 1 before obtaining all 16 nonzero values modulo 17. However, it turns out that 3 is a primitive root for 17, since in $\mathbb{F}_{17}$,

$$\begin{array}{llllll} 3^0 = 1 & 3^1 = 3 & 3^2 = 9 & 3^3 = 10 & 3^4 = 13 & 3^5 = 5 \\ 3^6 = 15 & 3^7 = 11 & 3^8 = 16 & 3^9 = 14 & 3^{10} = 8 & 3^{11} = 7 \\ 3^{12} = 4 & 3^{13} = 12 & 3^{14} = 2 & 3^{15} = 6. & & \end{array}$$

[9] The prime factorization of m is $m = 15485207 = 3853 \cdot 4019$.

Remark 1.27. If p is large, then the finite field $\mathbb{F}_p$ actually has quite a few primtive roots. The precise formula is that $\mathbb{F}_p$ has $\phi(p-1)$ primitive roots, where ϕ is Euler's phi function (see page 22). For example, you can check that the set of primitive roots for $\mathbb{F}_{29}$ is

$$\{2, 3, 8, 10, 11, 14, 15, 18, 19, 21, 26, 27\},$$

which agrees with the value $\phi(28) = 12$.

1.5 Cryptography by hand [H]

1.6 Symmetric and asymmetric ciphers

We have now seen several different examples of ciphers, all of which share a common protocol. Bob wants to send a secret message to Alice. He uses a secret key k in order to scramble his plaintext message m and turn it into a ciphertext c. Alice, upon receiving c, uses the secret key k to unscramble c and recreate m. This procedure works properly only if both Alice and Bob possess copies of the secret key k, and it provides security against their adversary Eve only if Eve does not know k, cannot guess k, and cannot recover m from c without knowing k.

In this section we formulate the notion of a cryptosystem in abstract mathematical terms. There are many reasons why this is desirable. For example, it allows us to highlight both the similarities and the differences between various ciphers, while also providing a means to rigorously analyze the security of a cryptosystem against various types of attacks.

1.6.1 Symmetric ciphers

Returning to Bob and Alice, we observe that they must share knowledge of the secret key k, and using that secret key, they can both encrypt and decrypt messages. For this reason, ciphers of this sort are known as *symmetric ciphers*. Mathematically, a symmetric cipher uses a key k chosen from a space (i.e. a set) of possible keys $\mathcal{K}$ to encrypt a plaintext message m chosen from a space of possible messages $\mathcal{M}$, and the result of the encryption process is a ciphertext c belonging to a space of possible ciphertexts $\mathcal{C}$.

Thus encryption may be viewed as a function

$$e : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$$

whose domain $\mathcal{K} \times \mathcal{M}$ is pairs (k, m) consisting of a key k and a plaintext m and whose range is the space of ciphertexts $\mathcal{C}$. Similarly, decryption is given by a function

$$d : \mathcal{K} \times \mathcal{C} \to \mathcal{M}.$$

Further, we want the decryption function to "undo" the results of encryption. Mathematically, this may be expressed as the formula

$$d\bigl(k, e(k, m)\bigr) = m \qquad \text{for all } k \in \mathcal{K} \text{ and all } m \in \mathcal{M}.$$

It is sometimes convenient to write the dependence on k as a subscript. Then for each key k, we get a pair of functions

$$e_k : \mathcal{M} \longrightarrow \mathcal{C} \qquad \text{and} \qquad d_k : \mathcal{C} \longrightarrow \mathcal{M}$$

satisfying the decryption property

$$d_k\bigl(e_k(m)\bigr) = m \qquad \text{for all } m \in \mathcal{M}.$$

In other words, for every key k, the function d_k is the inverse function of the function e_k. In particular, this means that e_k must be one-to-one, since if $e_k(m) = e_k(m')$, then

$$m = d_k\bigl(e_k(m)\bigr) = d_k\bigl(e_k(m')\bigr) = m'.$$

It is safest for Alice and Bob to assume that Eve knows the encryption method that is being employed. In mathematical terms, this means that Eve knows the functions e and d. What Eve does not know is the particular key k that Alice and Bob are using. For example, if Alice and Bob use a simple substitution cipher, they should assume that Eve is aware of this fact. This illustrates a basic principle of modern cryptography called *Kerckhoff's Principle*, which says that the security of a cryptosystem should depend only on the secrecy of the key, and not on the secrecy of the encryption algorithm itself.

In order for $(\mathcal{K}, \mathcal{M}, \mathcal{C}, e, d)$ to be a successful cipher, it must have the following properties:

1. For any key $k \in \mathcal{K}$ and plaintext $m \in \mathcal{M}$, it must be easy to compute the ciphertext $e_k(m)$.

2. For any key $k \in \mathcal{K}$ and ciphertext $c \in \mathcal{C}$, it must be easy to compute the plaintext $d_k(c)$.

3. Given one or more ciphertexts $c_1, c_2, \dots c_n \in \mathcal{C}$ encrypted using the key $k \in \mathcal{K}$, it must be very difficult to compute any of the corresponding plaintexts $d_k(c_1), \dots, d_k(c_n)$ without knowledge of k.

There is a fourth property that is desirable, although it is more difficult to achieve.

4. Given one or more pairs of plaintexts and their corresponding ciphertexts, $(m_1, c_1), (m_2, c_2), \dots, (m_n, c_n)$, it must be difficult to decrypt any ciphertext c that is not in the given list without knowing k. This is known as security against a *chosen plaintext attack*.

Example 1.28. The simple substituion cipher does not have Property 4, since even a single plaintext/ciphertext pair (m, c) reveals most of the encryption table. Thus simple substitution ciphers are vulnerable to chosen plaintext attacks.

In our list of four desirable properties for a cryptosystem, we have left open the question of what exactly is meant by the words "easy" and "hard." We defer a formal discussion of this profound question to Chapter 8 (see also Section 2.1). For now, we take "easy" to mean computable in less than a second on a typical desktop computer and "hard" to mean that all of the computing power in the world would require several years (at least) to perform the computation.

1.6.2 Encoding schemes

It is convenient to view keys and plaintexts and ciphertexts as numbers and to write those numbers in binary form. For example, we could take strings of 8 bits, which give numbers from 0 to 255, and use them to represent the letters of alphabet via

$$\texttt{a} = 00000000, \quad \texttt{b} = 00000001, \quad \texttt{c} = 00000010, \quad \dots \quad \texttt{z} = 00011010.$$

	32	00100000
(	40	00101000
)	41	00101001
,	44	00101100
.	46	00101110

A	65	01000001
B	66	01000010
C	67	01000011
D	68	01000100
⋮	⋮	⋮
X	88	01011000
Y	89	01011001
Z	90	01011010

a	97	01100001
b	98	01100010
c	99	01100011
d	100	01100100
⋮	⋮	⋮
x	120	01111000
y	121	01111001
z	122	01111010

Table 1.10: The ASCII encoding scheme

To distinguish lowercase from uppercase, we could let `A` = 00011011, `B` = 00011100, and so on. This encoding method allows up to 256 distinct symbols to be translated into binary form.

Your computer uses a method of this type, called the ASCII code, to store data, although for historical reasons the alphabetic characters are not assigned the lowest binary values. Part of the ASCII code is listed in Table 1.10. For example, the phrase “`Bed bug.`” (including spacing and punctuation) is encoded in ASCII as

B	e	d		b	u	g	.
66	101	100	32	98	117	103	46
01000010	01100101	01100100	00100000	01100010	01110101	01100111	00101110

Thus where you see the phrase “`Bed bug.`”, your computer sees the list of bits

$$0100001001100101011001000010000001100010011101010110011100101110.$$

Definition. An *encoding scheme* is a method of converting one sort of data into another sort of data, for example, for converting text into numbers. The distinction between an encoding scheme and an encryption scheme is one of intent. An encoding scheme is assumed to be entirely public knowledge and used by everyone for the same purposes. An encryption scheme is designed to hide information from anyone who does not possess the secret key. Thus an encoding scheme consists of an encoding function and its inverse decoding function, but for an encoding scheme, both functions are public knowledge and should be fast and easy to compute.

A plaintext or ciphertext may thus be viewed as a sequence of blocks, where each block consists of eight ones or zeros. The individual ones and zeros are call *bits*, and a block of eight bits is called a *byte*. For human comprehension, a byte is often written as a decimal number between 0 and 255, or as a two digit hexadecimal (base 16) number between 00 and FF. Computers often operate on more than one byte a time. For example, a 32 bit processor operates on 4 bytes at a time.

1.6.3 Symmetric encryption of encoded blocks

Using an encoding scheme as described in Section 1.6.2, it is convenient to view the elements of the plaintext space $\mathcal{M}$ as consisting of bit strings of a fixed length B, i.e. strings of exactly B ones and zeros. We call B the *block size* of the cipher. A general plaintext message then consists of a list of message blocks chosen from $\mathcal{M}$, and the encryption function transforms them into a list of ciphertext blocks in $\mathcal{C}$, where each block is a sequence of B bits. If the plaintext ends with a block of fewer than B bits, we pad the end of the block with zeros. Keep in mind that this encoding process, which converts the original plaintext message into a sequence of blocks of bits in $\mathcal{M}$, is public knowledge.

Encryption and decryption is done one block at a time, so it suffices to study the process for a single plaintext block, i.e. for a single $m \in \mathcal{M}$. The plaintext block m is a string of B bits, which for concreteness we identify with the corresponding number in binary form. In other words, we identify $\mathcal{M}$ with the set of integers m satisfying $0 \le m < 2^B$ via

$$\overbrace{m_{B-1}m_{B-2}\cdots m_2m_1m_0}^{\text{list of } B \text{ bits of } m} \longleftrightarrow \overbrace{m_{B-1}\cdot 2^{B-1} + \cdots + m_2\cdot 2^2 + m_1\cdot 2 + m_0}^{\text{integer between } 0 \text{ and } 2^B-1}.$$

Similarly, we identify the key space $\mathcal{K}$ and the ciphertext space $\mathcal{C}$ with sets of integers corresponding to bit strings of a certain blocksize. For notational convenience, we denote the blocksizes for keys, plaintexts, and ciphertexts by B_k, B_m, and B_c. They need not be the same. Thus we have identified $\mathcal{K}$, $\mathcal{M}$, and $\mathcal{C}$ with sets of positive integers

$$\begin{aligned}\mathcal{K} &= \{k \in \mathbb{Z}\ : 0 \le k < 2^{B_k}\},\\ \mathcal{M} &= \{m \in \mathbb{Z} : 0 \le m < 2^{B_m}\},\\ \mathcal{C} &= \{c \in \mathbb{Z}\ : 0 \le c < 2^{B_c}\}.\end{aligned}$$

An important question immediately arises.

How large should Alice and Bob make the set $\mathcal{K}$, or equivalently, how large should they choose the key blocksize B_k? If B_k is too small, then Eve can check every number from 0 to $2^{B_k} - 1$ until she finds Alice and Bob's key. More precisely, since Eve is assumed to know the decryption algorithm d (Kerckhoff's Principle), she takes each $k \in \mathcal{K}$ and uses it to compute $d_k(c)$. Assuming that Eve is able to distinguish between valid and invalid plaintexts, then eventually she will recover the message.

This attack is known as an *exhaustive search attack* (also sometimes referred to as a *brute force attack*), since Eve exhaustively searches through the keyspace. With current technology, an exhaustive search is considered to be infeasible if the space has at least 2^{80} elements. Thus Bob and Alice should definitely choose $B_k \geq 80$.

For many cryptosystems, especially the public key cryptosystems that form the core of this book, there are refinements on the exhaustive search attach that effectively replace the size of the space wth its square root. Some of these *meet-in-the-middle* or *collision attacks* are described in Section 3.4. If meet-in-the-middle attacks are available, then Alice and Bob should take $B_k \geq 160$.

1.6.4 Examples of symmetric ciphers

Before descending further into a morass of theory and notation, we pause to give a mathematical description of some elementary symmetric ciphers.

Let p be a large prime,[10] say $2^{159} < p < 2^{160}$. Alice and Bob take their keyspace $\mathcal{K}$, plaintext space $\mathcal{M}$, and ciphertext space $\mathcal{C}$ to be the same set,

$$\mathcal{K} = \mathcal{M} = \mathcal{C} = \{1, 2, 3, \ldots, p-1\}.$$

In fancier terminology, $\mathcal{K} = \mathcal{M} = \mathcal{C} = \mathbb{F}_p^*$ are all taken to equal the group of units in the finite field $\mathbb{F}_p$.

Alice and Bob randomly select a key $k \in \mathcal{K}$, i.e. they select an integer k satisfying $1 \leq k < p$. The encryption function e_k is defined by

$$e_k(m) \equiv k \cdot m \pmod{p}. \tag{1.9}$$

[10] There are in fact many primes in the interval $2^{159} < p < 2^{160}$. The prime number theorem implies that approximately 1% of the numbers in this interval are prime. Of course, there is also the question of identifying a number as prime or composite. There are efficient tests that do this, even for very large numbers, see Section 4.4.

Here we mean that $e_k(m)$ is set equal to the unique positive integer between 0 and p that is congruent to $k \cdot m$ modulo p. The corresponding decryption function d_k is defined by

$$d_k(c) \equiv k' \cdot c \pmod{p},$$

where k' is the inverse of k modulo p. It is important to note that although p is very large, the extended Euclidean algorithm (Remark 1.14) allows us to calculate k' in fewer than $2 \cdot \log_2 p$ steps. Thus finding k' from k counts as "easy" the world of cryptography.

It is clear that Eve will have a hard time guessing k, since there are approximately 2^{160} possibilities from which to choose. Is it also difficult for Eve to recover k if she knows the ciphertext c? The answer is yes, it is still difficult. Notice that the encryption function

$$e_k : \mathcal{M} \longrightarrow \mathcal{C}$$

is surjective (onto) for any choice of key k. Thus a given ciphertext c may represent any plaintext, provided that the plaintext is encrypted by an appropriate key. Mathematically, this may be rephrased by saying that given any ciphertext $c \in \mathcal{C}$ and any plaintext $m \in \mathcal{M}$, there exists a key k such that $e_k(m) = c$, specifically this is true for the key

$$k \equiv m^{-1} \cdot c \pmod{p}.$$

This shows that Alice and Bob's cipher has the Properties 1, 2, and 3 listed on page 33

It is interesting to observe that if Alice and Bob define their encryption function to be simply multiplication of integers $e_k(m) = k \cdot m$ with no reduction modulo p, then their cipher still has Properties 1 and 2, but Property 3 fails. If Eve tries to decrypt a single ciphertext $c = k \cdot m$, she still faces the (moderately) difficult task of factoring a 96 digit number. However, if she manages to aquire several ciphertexts $c_1, c_2, \ldots, c_n$ with their accompanying plaintexts, then there is a good chance that

$$\begin{aligned} \gcd(c_1, c_2, \ldots, c_n) &= \gcd(k \cdot m_1, k \cdot m_2, \ldots, k \cdot m_n) \\ &= k \cdot \gcd(m_1, m_2, \ldots, m_n) \end{aligned}$$

equals k itself or a small multiple of k. Note that it is an easy task to compute the greatest common divisor.

This observation provides our first indication of how reduction modulo an integer has a wonderful "mixing" effect that destroys properties such as divisibility. However, reduction is not by itself the ultimate solution. Consider the vulnerability of the cipher (1.9) to a chosen plaintext attack. If Eve can get her hands on both a ciphertext c and its corresponding plaintext m, then she easily recovers the key by computing

$$k \equiv m^{-1} \cdot c \pmod{p}.$$

Thus even a single plaintext/ciphertext pair suffices to reveal the key, so (1.9) does not have Property 4 on page 34.

There are many variants of this "multiplication-modulo-p" cipher. For example, since addition is more efficient than multiplication, there is an "addition-modulo-p" cipher given by

$$e_k(m) \equiv m + k \pmod{p} \qquad \text{and} \qquad d_k(c) \equiv c - k \pmod{p},$$

which is nothing other than the shift or Caesar cipher that we studied in Section 1.1. Another variant called an *affine cipher* is a combination of the shift cipher and the multiplication cipher. The key for an affine cipher consists of two integers $k = (k_1, k_2)$ and encryption and decryption are defined by

$$\begin{aligned} e_k(m) &= k_1 \cdot m + k_2 \pmod{p}, \\ d_k(c) &= k_1' \cdot (c - k_2) \pmod{p}, \end{aligned} \tag{1.10}$$

where k_1' is the inverse of k_1 modulo p.

The affine cipher has a further generalization called the *Hill Cipher*, in which the plaintext m, the ciphertext c, and the second part of the key k_2 are replaced by vectors consisting of lists of n numbers modulo p. The first part of the key k_1 is taken to be an n-by-n matrix with mod P integer entries. Encryption and decryption is again given by (1.10), but now multiplication $k_1 \cdot m$ is the product of a matrix and a vector, and k_1' is the inverse matrix of k_1 modulo p. Both the affine cipher and the Hill cipher are vulnerable to chosen plaintext attacks, see Exercises 1.30.

As noted earlier, addition is faster than multiplication. There is a basic computer operation that is even more efficient than addition. It is called *exclusive or* and is denoted by XOR or $\oplus$. At the lowest level, XOR takes two bits β and β' and yields

$$\beta \oplus \beta' = \begin{cases} 0 & \text{if } \beta \text{ and } \beta' \text{ are the same,} \\ 1 & \text{if } \beta \text{ and } \beta' \text{ are different.} \end{cases} \tag{1.11}$$

If you think of a bit as a number that is 0 or 1, then XOR is the same as addition modulo 2. More generally, the XOR of two bit strings is the result of performing XOR on each corresponding pair of bits. For example

$$10110 \oplus 11010 = (1 \oplus 1)(0 \oplus 1)(1 \oplus 0)(1 \oplus 1)(0 \oplus 0) = 01100.$$

Using this new operation, Alice and Bob have at their disposal yet another basic cipher defined by

$$e_k(m) = k \oplus m \qquad \text{and} \qquad d_k(c) = k \oplus c.$$

Here $\mathcal{K}$, $\mathcal{M}$, and $\mathcal{C}$ are the sets of all binary strings of length B, or equivalently, the set of all numbers between 0 and $2^B - 1$.

This cipher has the advantage of being highly efficient and completely symmetrical in the sense that e_k and d_k are the same function. If k is chosen randomly and is used only once, then this cipher is called Vernam's one-time pad. In Section 3.56 we show that the one-time pad is provably secure. Unfortunately, this requires that the key be as long as the plaintext, which can be cumbersome; and if k is used to encrypt more than one plaintext, then Eve may be able to exploit patterns in message sums

$$c' \oplus c' = (k \oplus m) \oplus (k \oplus m') = m \oplus m'$$

to extract information about k.

1.6.5 Random bit sequences and symmetric ciphers

At long last, we have arrived at the fundamental question regarding the creation of secure and efficient symmetric ciphers. Is it possible to use a single relatively short key k (say consisting of 160 random bits) to securely and efficiently send arbitrarily long messages? Here is one possible construction. Suppose that we could construct a function

$$R : \mathcal{K} \times \mathbb{Z} \longrightarrow \{0, 1\}$$

with the following properties:

1. For all $k \in \mathcal{K}$ and all $j \in \mathbb{Z}$, it is easy to compute $R(k, j)$.

2. Given an arbitrarily long sequence of integers $j_1, j_2, \ldots, j_n$ and given all of the values $R(k, j_1), R(k, j_2), \ldots, R(k, j_n)$, it is hard to determine k.

3. Given an arbitrarily long sequence of integers $j_1, j_2, \ldots, j_n$ and given all of the values $R(k, j_1), R(k, j_2), \ldots, R(k, j_n)$, it is hard to guess the value of $R(k, j)$ with better than a 50% chance of success for any value of j not already in the list.

If we could find a function R with these three properties, then we could use it to turn an initial key k into a sequence of bits

$$R(k, 1), R(k, 2), R(k, 3), R(k, 4), \ldots, \tag{1.12}$$

and then we could use this sequence of bits as the key for a one-time pad. The fundamental problem with this approach is that the sequence of bits is not truely random, since it is generated by the function R. Instead, we say that the sequence of bits (1.12) is a *pseudorandom sequence* and we call R a *pseudorandom number generator*.

Do pseudorandom number generators exist? If so, they would provide examples of the one-way functions defined by Diffie and Hellman in their groundbreaking paper [16], but despite more than a quarter century of work, no one has yet proven that one-way functions exist. We return to this fascinating subject in Sections 2.1 and 8.6. For now we content ourselves with a few brief remarks.

Although no one has yet conclusively proven that pseudorandom number generators exist, many candidates have been suggested, and some of these proposals have withstood the test of time. There are two basic approaches to constructing candidates for R, and these two methods provide a good illustration of the fundamental conflict in cryptography between security and efficiency.

The first approach is to repeatedly apply an ad hoc collection of mixing operations that are well suited to efficient computation and that appear to be very hard to untangle. This method is, disconcertingly, the basis for most practical symmetric ciphers, including the two mostly widely used today, namely DES and AES. We briefly describe how they work in Section 1.7.

The second approach is to construct R using a function whose efficient inversion is a well-known mathematical problem that is believed to be difficult. This approach provides a far more satisfactory theoretical underpinning for a symmetric cipher, but unfortunately all known constructions of this sort are far less efficient than the ad hoc constructions, and hence are less attractive for real-world applications.

1.6.6 Asymmetric ciphers make a first appearance

If Alice and Bob want to exchange messages using a symmetric cipher, they must first mutually agree on a secret key k. This is fine if they have the opportunity to meet in secret or if they are able to communicate once over a secure channel. But what if they do not have this opportunity and if every communication between them is monitored by their adversary Eve? Is it possible for Alice and Bob to exchange a secret key under these conditions?

Most people's first reaction is that it is not possible, since Eve sees every piece of information that Alice and Bob exchange. It was the brilliant insight of Diffie and Hellman[11] that under certain hypotheses, it is possible. The search for efficient (and provable) solutions to this problem, which is called *public key* (or *asymmetric*) *cryptography* form one of the most intereseting parts of mathematical cryptography and are the principal focus of this book.

We start by describing a non-mathematical way to visualize public key cryptography. Alice takes a safe whose door is open and she places the safe in a public location. Anyone who wants to may examine the safe, both inside and outside. Bob writes a message to Alice, places it in the safe, and slams the safe door shut. Now only someone who has the key to the safe, which presumably means only Alice, can retrieve and read Bob's message. In this scenario, Alice's public key is the open safe, the encryption algorithm is the process of closing the door of the safe, and the decryption algorithm is the process by which Alice uses the key to open the safe.

However, there is one way in which our analogy is misleading. In a true public key cryptosystem, Alice only needs to put one open safe into a public location, and everyone in the world can use it repeatedly to send encrypted messages to Alice. There is no need for Alice to open the safe and remove Bob's message before Carl uses it to send Alice another message.

We next give a mathematical formulation of an asymmetric cipher. As usual, there spaces of keys $\mathcal{K}$, plaintexts $\mathcal{M}$ and ciphertexts $\mathcal{C}$. However, an element k of the keyspace is really a pair of keys, $k = (k_{\mathsf{priv}}, k_{\mathsf{pub}})$, called the private key and the public key, respectively. For each public key k_{pub}, there is a corresponding encryption function

$$e_{k_{\mathsf{pub}}} : \mathcal{M} \longrightarrow \mathcal{C},$$

[11] The history is actually somewhat more complicated than this, see our brief discussion in Section 2.1 and the references listed there for further reading.

and for each private key k_{priv} a corresponding decryption function

$$d_{k_{\mathsf{priv}}} : \mathcal{C} \longrightarrow \mathcal{M}.$$

These have the property that if the pair $(k_{\mathsf{priv}}, k_{\mathsf{pub}})$ is in the keyspace $\mathcal{K}$, then

$$d_{k_{\mathsf{priv}}}\bigl(e_{k_{\mathsf{pub}}}(m)\bigr) = m \qquad \text{for all } m \in \mathcal{M}.$$

In order to qualify as an asymmetric cipher, it must be difficult for Eve to compute the decryption function $d_{k_{\mathsf{priv}}}(c)$, even if she knows the public key k_{pub}. Notice that under this assumption, Alice can send k_{pub} to Bob using an insecure communication channel and Bob can send back the ciphertext $e_{k_{\mathsf{pub}}}(m)$ without worrying that Eve will be able to decrypt the message. It is only knowledge of the private key k_{priv} that enables Alice to easily decrypt. The private key is sometimes called Alice's *trapdoor information*, because it provides a trapdoor (i.e. a shortcut) for computing the inverse function of $e_{k_{\mathsf{pub}}}$. The fact that the encryption and decryption keys k_{pub} and k_{priv} are different makes the cipher unsymmetrical, whence its moniker.

It is quite intriguing that Diffie and Hellman created this concept without finding a candidate for an actual pair of functions, although they did propose a similar method by which Alice and Bob can securely exchange a random piece of data whose value is not known initially to either one. We describe Diffie and Hellman's key exchange method in Section 2.3 and then go on to discuss a number of asymmetric ciphers such as ElGamal (Section 2.4), RSA (Section 4.2), ECC (Section 5.4), and NTRU (Section 7.6), whose security relies on the presumed difficulty of a variety of different mathematical problems.

1.7 Modern symmetric ciphers

Exercises

Section 1.1. Simple substitution ciphers

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
S	C	J	A	X	U	F	B	Q	K	T	P	R	W	E	Z	H	V	L	I	G	Y	D	N	M	O

Table 1.11: Simple substitution encryption table for exercise 1.3

1.1. Build a cipher wheel as illustrated in Figure 1.1, but with an inner wheel that rotates, and use it to complete the following tasks. (There is a cipher wheel that you can print and cut out available online from the NSA at `http://www.nsa.gov/kids/ciphers/images/image00004.gif`.)

(a) Encrypt the following plaintext using a rotation of 11 clockwise.

"A page of history is worth a volume of logic."

(b) Decrypt the following message that was encrypted with a rotation of 7 clockwise.

`AOLYLHYLUVZLJYLAZILAALYAOHUAOLZLJYLALZAOHALCLYFIVKFNBLZZLZ`

(c) Decrypt the following message that was encrypted by rotating 1 clockwise for the first letter, then 2 clockwise for the second letter, etc.

`XJHRFTNZHMZGAHIUETXZJNBWNUTRHEPOMDNBJMAUGORFAOIZOCC`

1.2. Decrypt each of the following Caesar encryptions by trying the various possible shifts until you obtain readable text.

(a) `LWKLQNWKDWLVKDOOQHYHUVHHDELOOERDUGORYHOBDVDWUHH`

(b) `UXENRBWXCUXENFQRLQJUCNABFQNWRCJUCNAJCRXWORWMB`

(c) `BGUTBMBGZTFHNLXMKTIPBMAVAXXLXTEPTRLEXTOXKHHFYHKMAXFHNLX`

1.3. For this exercise, use the simple substitution table given in Table 1.11 on page 44.

(a) Encrypt the plaintext message

`The gold is hidden in the garden.`

(b) Make a decryption table, that is, make a table in which the ciphertext alphabet is in order from `A` to `Z` and the plaintext alphabet is mixed up.

(c) Use your decryption table from (b) to decrypt the following message.

`IBXLX JVXIZ SLLDE VAQLL DEVAU QLB`

1.4. Each of the following messages has been encrypted using a simple substitution cipher. Decrypt them. For your convenience, we have given you a frequency table and a list of the most common bigrams that appear in the ciphertext. (If you do not want to recopy the ciphertexts by hand, they can be downloaded or printed from the web site listed in the introduction.)

(a) "A Piratical Treasure"

```
JNRZR BNIGI BJRGZ IZLQR OTDNJ GRIHT USDKR ZZWLG OIBTM NRGJN
IJTZJ LZISJ NRSBL QVRSI ORIQT QDEKJ JNRQW GLOFN IJTZX QLFQL
WBIMJ ITQXT HHTBL KUHQL JZKMM LZRNT OBIMI EURLW BLQZJ GKBJT
QDIQS LWJNR OLGRI EZJGK ZRBGS MJLDG IMNZT OIHRK MOSOT QHIJL
QBRJN IJJNT ZFIZL WIZTO MURZM RBTRZ ZKBNN LFRVR GIZFL KUHIM
MRIGJ LJNRB GKHRT QJRUU RBJLW JNRZI TULGI EZLUK JRUST QZLUK
EURFT JNLKJ JNRXR S
```

The ciphertext contains 316 letters. Here is a frequency table:

	R	J	I	L	Z	T	N	Q	B	G	K	U	M	O	S	H	W	F	E	D	X	V
Freq	33	30	27	25	24	20	19	16	15	15	13	12	12	10	9	8	7	6	5	5	3	2

The most frequent bigrams are: JN (11 times), NR (8 times), TQ (6 times), and LW, RB, RZ, and JL (5 times each).

(b) "A Botanical Code"

```
KZRNK GJKIP ZBOOB XLCRG BXFAU GJBNG RIXRU XAFGJ BXRME MNKNG
BURIX KJRXR SBUER ISATB UIBNN RTBUM NBIGK EBIGR OCUBR GLUBN
JBGRL SJGLN GJBOR ISLRS BAFFO AZBUN RFAUS AGGBI NGLXM IAZRX
RMNVL GEANG CJRUE KISRM BOOAZ GLOKW FAUKI NGRIC BEBRI NJAWB
OBNNO ATBZJ KOBRC JKIRR NGBUE BRINK XKBAF QBROA LNMRG MALUF
BBG
```

The ciphertext contains 253 letters. Here is a frequency table:

	B	R	G	N	A	I	U	K	O	J	L	X	M	F	S	E	Z	C	T	W	P	V	Q
Freq	32	28	22	20	16	16	14	13	12	11	10	10	8	8	7	7	6	5	3	2	1	1	1

The most frequent bigrams are: NG and RI (7 times each), BU (6 times), and BR (5 times).

(c) In order to make this one a bit more challenging, we have removed all occurences of the word "the" from the plaintext.

"A Brilliant Detective"

```
GSZES GNUBE SZGUG SNKGX CSUUE QNZOQ EOVJN VXKNG XGAHS AWSZZ
BOVUE SIXCQ NQESX NGEUG AHZQA QHNSP CIPQA OIDLV JXGAK CGJCG
SASUB FVQAV CIAWN VWOVP SNSXV JGPCV NODIX GJQAE VOOXC SXXCG
OGOVA XGNVU BAVKX QZVQD LVJXQ EXCQO VKCQG AMVAX VWXCG OOBOX
VZCSO SPPSN VAXUB DVVAX QJQAJ VSUXC SXXCV OVJCS NSJXV NOJQA
MVBSZ VOOSH VSAWX QHGMV GWVSX CSXXC VBSNV ZVNVN SAWQZ ORVXJ
CVOQE JCGUW NVA
```

The ciphertext contains 313 letters. Here is a frequency table:

	V	S	X	G	A	O	Q	C	N	J	U	Z	E	W	B	P	I	H	K	D	M	L	R	F
Freq	39	29	29	22	21	21	20	20	19	13	11	11	10	8	8	6	5	5	5	4	3	2	1	1

The most frequent bigrams are: XC (10 times), NV (7 times), and CS, OV, QA, and SX (6 times each).

Section 1.2. Divisibility and greatest common divisors [**M**]

1.5. Let $a, b, c \in \mathbb{Z}$ be integers. Use the definition of divisibility to directly prove the following properties of divisibility. (This is Proposition 1.4.)

(a) If $a|b$ and $b|c$, then $a|c$.
(b) If $a|b$ and $b|a$, then $a = \pm b$.
(c) If $a|b$ and $a|c$, then $a|b + c$ and $a|b - c$.

1.6. Use a calculator and the method described in Remark 1.9 to compute the following quotients and remainders.

(a) 34787 divided by 353.
(b) 238792 divided by 7843.
(c) 9829387493 divided by 873485.
(d) 1498387487 divided by 76348.

1.7. Use a calculator and the method described in Remark 1.9 to compute the following remainders, without bothering to compute the associated quotients.

(a) The remainder of 78745 divided by 127.
(b) The remainder of 2837647 divided by 4387.
(c) The remainder of 8739287463 divided by 18754.
(d) The remainder of 4536782793 divided by 9784537.

1.8. Use the Euclidean algorithm to compute the following greatest common divisors.

(a) $\gcd(291, 252)$.
(b) $\gcd(16261, 85652)$.
(c) $\gcd(139024789, 93278890)$.
(d) $\gcd(16534528044, 8332745927)$.

1.9. For each of the $\gcd(a, b)$ values in Exercise 1.8, use the Extending Euclidean Algorithm (Theorem 1.11) to find integers u and v so that $au + bv = \gcd(a, b)$.

1.10. Let a and b be positive integers.

(a) Suppose that there are integers u and v satisfying $au + bv = 1$. Prove that $\gcd(a, b) = 1$.
(b) Suppose that there are integers u and v satisfying $au + bv = 6$. Is it necessarily true that $\gcd(a, b) = 6$? If not, what are the possible values of $\gcd(a, b)$?
(c) Suppose that (u_1, v_1) and (u_2, v_2) are two solutions in integers to the equation $au + bv = 1$. Prove that a divides $v_2 - v_1$ and that b divides $u_2 - u_1$.

(d) More generally, let $g = \gcd(a, b)$ and let (u_0, v_0) be a solution in integers to $au + bv = g$. Prove that every other solution has the form $u = u_0 + kb/g$ and $v = v_0 - ka/g$ for some integer k. (This is the second part of Theorem 1.11.)

1.11. The method for solving $au + bv = \gcd(a, b)$ described in Section 1.2 is somewhat inefficient. This exercise describes a better way to compute u and y that is well-suited for computer implementation.

(a) Show that the following algorithm computes the greatest common divisor g of the positive integers a and b, together with a solution (u, v) in integers to the equation $au + bv = \gcd(a, b)$.

```
1. Set $u = 1$, $g = a$, $x = 0$, and $y = b$
2. If $y = 0$, set $v = (g - au)/b$ and return the values $(g, u, v)$
3. Divide $g$ by $y$ with remainder, $g = qy + t$, with $0 \le t < y$
4. Set $s = u - qx$
5. Set $(u, g) = (x, y)$
6. Set $(x, y) = (s, t)$
7. Go To Step (2)
```

(b) Implement the above algorithm on a computer using the computer language of your choice.

(c) Use your program to compute $g = \gcd(a, b)$ and integer solutions to the equation $au + bv = g$ for the following pairs (a, b).

(i) $(527, 1258)$
(ii) $(228, 1056)$
(iii) $(163961, 167181)$
(iv) $(3892394, 239847)$

(d) What happens to your program if $b = 0$? Fix the program so that it deals with this case correctly.

(e) It is often useful to have a solution with $u > 0$. Modify your program so that it returns a solution with $u > 0$ and u as small as possible. [*Hint.* If (u, v) is a solution, then so is $(u + b/g, v - a/g)$.] Redo (c) using your modified program.

1.12. Let $a_1, a_2, \ldots, a_k$ be integers with $\gcd(a_1, a_2, \ldots, a_k) = 1$, i.e. the largest positive integer dividing all of $a_1, \ldots, a_k$ is 1. Prove that the equation

$$a_1u_1 + a_2u_2 + \cdots + a_ku_k = 1.$$

has a solution in integers $u_1, u_2, \ldots, u_k$. (*Hint.* Repeatedly apply the Extended Euclidean Algorithm, Theorem 1.11. You may find it easier to prove a more general statement in which $\gcd(a_1, \ldots, a_k)$ is allowed to be larger than 1.)

Section 1.3. Modular arithmetic [**M**]

1.13. Write out the following tables for $\mathbb{Z}/m\mathbb{Z}$ and $(\mathbb{Z}/m\mathbb{Z})^*$, as we did in Figures 1.3 and 1.4.

(a) Make addition and multiplication tables for $\mathbb{Z}/3\mathbb{Z}$.

(b) Make addition and multiplication tables for $\mathbb{Z}/6\mathbb{Z}$.

(c) Make a multiplication table for the unit group $(\mathbb{Z}/9\mathbb{Z})^*$.

(d) Make a multiplication table for the unit group $(\mathbb{Z}/16\mathbb{Z})^*$.

1.14. Do the following modular computations. In each case, fill in the box with an integer between 0 and $m-1$, where m is the modulus.

(a) $347 + 513 \equiv \boxed{} \pmod{763}$.

(b) $3274 + 1238 + 7231 + 6437 \equiv \boxed{} \pmod{9254}$.

(c) $153 \cdot 287 \equiv \boxed{} \pmod{353}$.

(d) $357 \cdot 862 \cdot 193 \equiv \boxed{} \pmod{943}$.

(e) $5327 \cdot 6135 \cdot 7139 \cdot 2187 \cdot 5219 \cdot 1873 \equiv \boxed{} \pmod{8157}$.
(*Hint.* After each multiplication, reduce modulo 8157 before doing the next multiplication.)

(f) $137^2 \equiv \boxed{} \pmod{327}$.

(g) $373^6 \equiv \boxed{} \pmod{581}$.

(h) $23^3 \cdot 19^5 \cdot 11^4 \equiv \boxed{} \pmod{97}$.

1.15. Find all values of x between 0 and $m-1$ that are solutions of the following congruences. (*Hint.* If you can't figure out a clever way to find the solution(s), you can just substitute each value $x = 1$, $x = 2$,..., $x = m-1$ and see which ones work.)

(a) $x + 17 \equiv 23 \pmod{37}$.

(b) $x + 42 \equiv 19 \pmod{51}$.

(c) $x^2 \equiv 3 \pmod{11}$.

(d) $x^2 \equiv 2 \pmod{13}$.

(e) $x^2 \equiv 1 \pmod{8}$.

(f) $x^3 - x^2 + 2x - 2 \equiv 0 \pmod{11}$.

(g) $x \equiv 1 \pmod{5}$ and also $x \equiv 2 \pmod{7}$. (Find all solutions modulo 35, that is, find the solutions satisfying $0 \le x \le 34$.)

1.16. Let $m \in \mathbb{Z}$.

(a) Suppose that m is odd. What integer between 1 and $m-1$ is equal to $2^{-1} \bmod m$.

(b) More generally, suppose that $m \equiv 1 \pmod{b}$. What integer between 1 and $m-1$ is equal to $b^{-1} \bmod m$.

1.17. Let m be an odd integer and let a be any integer. Prove that $2m + a^2$ can never be a perfect square. (*Hint.* If a number is a perfect square, what are its possible values modulo 4?)

1.18. (a) Find a single value x that simultaneously solves the two congruences

$$x \equiv 3 \pmod{7} \qquad \text{and} \qquad x \equiv 4 \pmod{9}.$$

(*Hint.* Note that every solution of the first congruence looks like $x = 3+7y$ for some y. Substitute this into the second congruence and solve for y, then use that to get x.)

(b) Find a single value x that simultaneously solves the two congruences

$$x \equiv 13 \pmod{71} \qquad \text{and} \qquad x \equiv 41 \pmod{97}.$$

(c) Find a single value x that simultaneously solves the three congruences

$$x \equiv 4 \pmod{7}, \qquad x \equiv 5 \pmod{8} \qquad \text{and} \qquad x \equiv 11 \pmod{15}.$$

(d) Prove that if $\gcd(m, n) = 1$, then the pair of congruences

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

has a solution for any choice of a and b. Also give an example to show that the condition $\gcd(m, n) = 1$ is necessary.

1.19. Let N, g, and m be positive integers (N need not be prime). Prove that the following algorithm, which is a low-storage variant of the square-and-multiply algorithm described in Section 1.3.2, returns the value $g^m \pmod{N}$.

```
    Input. Postive integers $N$, $g$, and $m$.
1.  Set $a = g$ and $b = 1$.
2.  Loop while $m > 0$.
      3.  If $m \equiv 1 \pmod{2}$, set $b = b \cdot a \pmod{N}$.
      4.  Set $a = a^2 \pmod{N}$ and $m = \lfloor m/2 \rfloor$.
      5.  If $m > 0$, continue with loop at Step 2.
6.  Return the number $b$, which equals $g^m \pmod{N}$.
```

1.20. Use the square-and-multiply algorithm described in Section 1.3.2, or the more efficient version in Exercise 1.19, to compute the following powers.

(a) $17^{183} \pmod{256}$.

(b) $2^{477} \pmod{1000}$.

(c) $11^{507} \pmod{1237}$.

Section 1.4. Finite fields [**M**]

1.21. Let $\{p_1, p_2, \ldots, p_r\}$ be a set of prime numbers, and let

$$N = p_1 p_2 \cdots p_r + 1.$$

Prove that N is divisible by some prime not in the original set. Use this fact to deduce that there must be infinitely many prime numbers. (This proof for the infinitude of primes appears in Euclid's *Elements*. The study of prime numbers goes back a long way in history.)

1.22. Without using the fact that every integer has a unique factorization into primes, prove that if $\gcd(a,b) = 1$ and if $a|bc$, then $a|c$. (*Hint.* Use the fact that it is possible to find a solution to $au + bv = 1$.)

1.23. Compute the following ord_p values.

(a) Compute $\operatorname{ord}_2(2816)$.

(b) Compute $\operatorname{ord}_7(2222574487)$.

(c) Compute $\operatorname{ord}_p(46375)$ for each of $p = 3$, 5, 7, and 11.

1.24. Let p be a prime number. Prove that ord_p has the following properties.

(a) $\operatorname{ord}_p(ab) = \operatorname{ord}_p(a) + \operatorname{ord}_p(b)$. (Thus ord_p resembles the logarithm function, since it converts multiplication into addition!)

(b) $\operatorname{ord}_p(a+b) \ge \min\bigl\{\operatorname{ord}_p(a), \operatorname{ord}_p(b)\bigr\}$.

(c) If $\operatorname{ord}_p(a) \ne \operatorname{ord}_p(b)$, then $\operatorname{ord}_p(a+b) = \min\bigl\{\operatorname{ord}_p(a), \operatorname{ord}_p(b)\bigr\}$.

1.25. This exercise begins the study of squares and square roots modulo p.

(a) Let p be a prime number. Prove that an integer b has at most two square roots modulo p. In other words, prove that the congruence

$$X^2 \equiv b \pmod{p}$$

has at most two solutions modulo p.

(b) For each of the following values of p and b, find all of the square roots of b modulo p.

(i) $(p,b) = (7,2)$ (ii) $(p,b) = (11,5)$

(iii) $(p,b) = (11,7)$ (iv) $(p,b) = (37,3)$

(c) How many square roots does 29 have modulo 35? Why doesn't this contradict the assertion in (a)?

(d) Let p be an odd prime and let g be a primitive root modulo p. Then any number a is equal to some power of g modulo p, say $a \equiv g^k \pmod{p}$. Prove that a has a square root modulo p if and ony if k is even.

1.26. Compute the value of

$$2^{(p-1)/2} \pmod{p}$$

for every prime $3 \le p < 20$. Make a conjecture as to the possible values of $2^{(p-1)/2} \pmod{p}$ when p is prime and prove that your conjecture is correct.

1.27. Recall that g is called a primitive root modulo p if the powers of g give all nonzero elements of $\mathbb{F}_p$.

(a) For which of the following primes is 2 a primitive root modulo p?
(i) $p = 7$ (ii) $p = 13$ (iii) $p = 19$ (iv) $p = 23$

(b) For which of the following primes is 3 a primitive root modulo p?
(i) $p = 5$ (ii) $p = 7$ (iii) $p = 11$ (iv) $p = 17$

(c) Find a primitive root for each of the following primes.
(i) $p = 23$ (ii) $p = 29$ (iii) $p = 41$ (iv) $p = 43$

(d) Find all primitive roots modulo 11. Verify that there are exactly $\phi(10)$ of them, as asserted in Remark 1.27.

(e) Write a compute program to check for primitive roots and use it to find all primitive roots modulo 229. Verify that there are exactly $\phi(228)$ of them.

(f) Use your program from (e) to find all primes less than 100 for which 2 is a primitive root.

(g) Repeat the previous exercise to find all primes less than 100 for which 3 is a primitive root. Ditto to find the primes for which 4 is a primitive root.

1.28. Let p be a prime and let g be an integer not divisible by p.

(a) Suppose that $g^a \equiv 1 \pmod{p}$ and also that that $g^b \equiv 1 \pmod{p}$. Prove that $g^{\gcd(a,b)} \equiv 1 \pmod{p}$.

(b) Let n be the smallest positive integer such that

$$g^n \equiv 1 \pmod{p}.$$

Prove that n divides $p - 1$. (*Hint.* Use (a) and Fermat's Little Theorem 1.22.)

(c) Suppose that $q = \frac{1}{2}(p-1)$ is also prime and that g satisfies

$$g \not\equiv \pm 1 \pmod{p} \qquad \text{and} \qquad g^q \not\equiv 1 \pmod{p}.$$

Prove that g is a primitive root modulo p.

Section 1.5. Cryptography by hand [**H**]

Section 1.6. Symmetric ciphers and asymetric ciphers

1.29. Encode the following phrase (including capitalization, spacing and punctuation) into a string of bits using the ASCII encoding scheme given in Table 1.10:

```
Bad day, Dad.
```

1.30. Consider the affine cipher with key $k = (k_1, k_2)$ whose encryption and decryption functions are given by (1.10) on page 39.

(a) Let $p = 541$ and let the key be $k = (34, 71)$. Encrypt the message $m = 204$. Decrypt the ciphertext $c = 431$.

(b) Assuming that p is public knowledge, explain why the affine cipher is vulnerable to a chosen plaintext attack. How many plaintext/ciphertext pairs are likely to be needed in order to recover the private key?

(c) Alice and Bob decide to use the prime $p = 601$ for their affine cipher. The value of p is public knowledge, and Eve intercepts the ciphertexts $c_1 = 324$ and $c_2 = 381$ and also manages to find out that the corresponding plaintexts are $m_1 = 387$ and $m_2 = 491$. Determine the private key and then use it to encrypt the message $m_3 = 173$.

Suppose now that p is not public knowledge. Is the affine cipher still vulnerable to a chosen plaintext attack? If so, how many plaintext/ciphertext pairs are likely to be needed in order to recover the private key?

1.31. Let N be a large integer and let $\mathcal{K} = \mathcal{M} = \mathcal{C} = \mathbb{Z}/N\mathbb{Z}$. For each of the following functions

$$e : \mathcal{K} \times \mathcal{M} \longrightarrow \mathcal{C},$$

answer the following questions:

- Is e an encryption function?
- If e is an encryption function, what is its associated decryption function d?
- If e is not an encryption function, can you make it into an encryption function by using a smaller set of keys?

(a) $e_k(m) \equiv k - m \pmod{N}$.

(b) $e_k(m) \equiv k * m \pmod{N}$.

(c) $e_k(m) \equiv (k + m)^2 \pmod{N}$.

1.32. (a) Convert the 12 bit binary number 110101100101 into a decimal integer between 0 and $2^{12} - 1$.

(b) Convert the decimal integer $m = 37853$ into a binary number.

(c) Convert the decimal integer $m = 9487428$ into a binary number.

(d) Use exclusive or (XOR) to "add" the bit strings $11001010 \oplus 10011010$.

(e) Convert the decimal numbers 8734 and 5177 into binary numbers, combine them using XOR, and convert the result back into a decimal number.

1.33. Alice and Bob choose a key space $\mathcal{K}$ containing 2^{56} keys. Eve builds a special purpose computer that can check 10,000,000,000 keys per second.

(a) How many days does it take Eve to check half the keys in $\mathcal{K}$?

(b) Alice and Bob replace their key space with a larger key space containing 2^B different keys. How large should Alice and Bob choose B in order to force Eve's computer to spend 100 years checking half the keys? (Use the approximation that there are 365.25 days in a year.)

For many years the United States government recommended a symmetric cipher called DES that had 56 bit keys. During the 1990's, people built special purpose computers demonstrating that 56 bits provided insufficient security. A new symmetric cipher called AES, with 128 bit keys, was developed to replace DES.

1.34. Explain why the cipher

$$e_k(m) = k \oplus m \qquad \text{and} \qquad d_k(c) = k \oplus c.$$

defined by XOR of bit strings is not secure against a chosen plaintext attack. Demonstrate your attack by finding the private key used to encrypt the 16 bit ciphertext $c = 1001010001010111$ if you know that the corresponding plaintext is $m = 0010010000101100$.

1.35. Alice and Bob create a symmetric cipher as follows. Their private key k is a large integer and their messages (plaintexts) are d-digit integers

$$\mathcal{M} = \{m \in \mathbb{Z} : 0 \le m < 10^d\}.$$

In order to encrypt a message, Alice computes $\sqrt{k}$ to d decimal places, throws away the part to the left of the decimal point, and keeps the remaining d digits. Let α be this d-digit number. (For example, if $k = 23$ and $d = 6$, then $\sqrt{23} = 9.32737905\ldots$ and $\alpha = 327379$.)

Alice encrypts a message m as

$$c \equiv m + \alpha \pmod{10^d}.$$

Since Bob knows k, he can also find α, and then he decrypts c by computing $m \equiv c - \alpha \pmod{10^d}$.

(a) Alice and Bob choose the secret key $k = 11$ and use it to encrypt 6 digit integers (i.e. $d = 6$). Bob wants to send Alice the message $m = 328973$. What is the ciphertext that he sends?

(b) Alice and Bob use the secret key $k = 23$ and use it to encrypt 8 digit integers. Alice receives the ciphertext $c = 78183903$. What is the plaintext m?

(c) Show that the number α used for encryption and decryption is given by the formula

$$\alpha = \left\lfloor 10^d \left(\sqrt{k} - \lfloor \sqrt{k} \rfloor \right) \right\rfloor,$$

where $\lfloor t \rfloor$ is the greatest integer that is less than or equal to t.

(d) (Challenge Problem) If Eve steals a plaintext/ciphertext pair (m, c), then she clearly can recover the number α, since $\alpha \equiv c - m \pmod{10^d}$. If 10^d is large compared to k, can she also recover the number k? This might be useful, for example, if Alice and Bob use the second d digits of $\sqrt{k}$ to encrypt their next message.

Section 1.7. Modern symmetric ciphers

Chapter 2

Discrete Logarithms and Diffie-Hellman

2.1 The birth of public key cryptography [H]

In 1976, Whitfield Diffie and Martin Hellman published their now famous paper [16] entitled "New directions in cryptography." In their paper they formulated the concept of a public key encryption system and made several groundbreaking contributions to this new field. A short time earlier, Ralph Merkle had independently isolated one of the fundamental problems and invented a public key construction for an undergraduate project in a computer science class at Berkeley, but this was little understood at the time. Merkle's work "Secure communication over insecure channels" appeared in 1982 [33].

However, it turns out that the concept of public key encryption was orginally discovered by James Ellis while working at the British Government Communications Headquarters (GCHQ). Ellis' discoveries in 1969 were classified as secret material by the British government and were not declassified and released until after his death. It is now known that two other researchers at GCHQ, Williamson and Cocks, discovered the Diffie-Hellman key exchange algorithm and the RSA public key encryption system, respectively, before their rediscovery and public dissemination by Diffie, Hellman, Rivest, Shamir, and Adelman. We have given here only a bare outline of the published record of public key cryptography — there are a number of excellent historical sources, including [15, 18].

The Diffie-Hellman publication was an extremely important event—it set forth the basic definitions and goals of a new field of mathematics/computer science, a field whose existence was dependent on the then emerging age of the digital computer. Indeed, their paper begins with a call to arms:

We stand today on the brink of a revolution in cryptography.

An original or breakthrough scientific idea is often called revolutionary, but in this instance, as the authors were fully aware, the term revolutionary was relevant in another sense. Prior to the publication of "New Directions...," encryption research in the United States was the domain of the National Security Agency, and all information in this area was classified. Indeed, until the mid-1990's, the United States government treated cryptographic algorithms as munitions, which meant that their export was prosecutable as a treasonable offense. Eventually, the government realized the futility of trying to prevent free and open discussion about abstract cryptographic algorithms and the dubious legality of restricting domestic use of strong cryptographic methods. However, in order to maintain some control, the government continued to retrict export of high security cryptographic algorithms if they were "machine readable." Their object, to prevent widespread global dissemination of sophisticated cryptography programs to potential enemies of the United States, was laudable,[1] but there were two difficulties that rendered the government's policy unworkable.

First, the existence of optical scanners creates a very blurry line between "machine readable" and "human text." To protest the government's policy, people wrote a three line version of the RSA algorithm in a programming language called perl and printed it on tee shirts and soda cans, thereby making these products into munitions. In principle, wearing an "RSA enabled" tee shirt on a flight from New York to Europe subjected the wearer to a large fine and a ten year jail term. Even more amusing (or frightening, depending on your viewpoint), tattoos of the RSA perl code made people bodies into non-exportable munitions!

These and other more serious protests and legal challenges had some effect, but ultimately the government's policy was rendered moot by a simple reality. Public key algorithms are quite simple, and although it requires a certain expertise to implement them in a secure fashion, the world is full

[1] It is surely laudable to keep potential weapons out of the hands of one's enemies, but many have argued, with considerable justification, that the government also had the less benign objective of preventing everyone from using communication methods secure from United States prying.

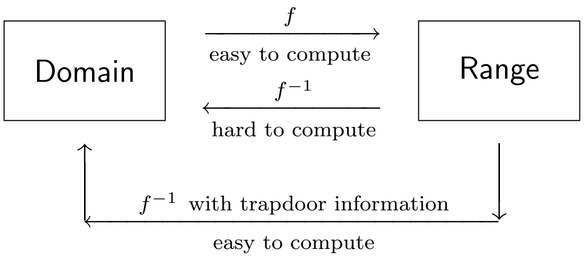

Figure 2.1: Illustration of a one-way trapdoor function

of excellent mathematicians and computer scientists and engineers. Thus government restrictions on the export of "strong crypto" simply encouraged the creation of cryptographic industries in other parts of the world. The government was able to slow the adoption of strong crypto for a few years, but it is now possible for anyone to purchase for a nominal sum cryptographic software that allows completely secure communications.[2]

The first important contribution of Diffie and Hellman in [16] was the definition of a public key cryptosystem (PKC) and it's associated components—one-way functions and trapdoor information. A *one-way function* is an invertible function that is easy to compute, but whose inverse is difficult to compute. What does it mean to be "difficult to compute"? A rigorous definition requires the introducion of notions from complexity theory, which we postpone until Chapter 8. Intuitively, the inverse is difficult to compute if any polynomial time algorithm that attempts to compute the inverse will almost certainly fail, where the phrase "almost certainly" must be defined probabilistically. Secure PKC's are built using one-way functions that have a *trapdoor*. The trapdoor is a piece of auxiliary information that allows the inverse to be easily computed. This idea is illustrated in Figure 2.1, although it must be noted that there is a vast chasm separating the abstract idea of a one-way trapdoor function and the actual construction of such a function.

In a public key cryptosystem, one has as usual a collection of keys $\mathcal{K}$, and for each $k \in \mathcal{K}$, an enryption algorithm e_k and corresponding decryption

[2] Of course, one never knows what cryptanalytic breakthroughs have been made by the scientists at the National Security Agency, since virtually all of its research is classified. The NSA is reputed to be the world's largest single employer of Ph.D.'s in mathematics. However, in contrast to the situation before the 1970's, there are now far more cryptographers employed in academia and in the business world than there are in government agencies.

algorithm d_k The encryption algorithm e_k, which must be easily computable, is public knowledge, although the key k itself is not. Further, the decryption algorithm d_k must be easily computable to someone who knows the key k, but should be very difficult to compute for someone who does not know the key k.

In common terminology, the key k is called the *private key* or the *trapdoor information*, and the encryption algorithm e_k is called the *public key.* More precisely, the private key is a number k_{priv} that is used to compute a second number k_{pub}. In order to encrypt, it suffices to know the public key k_{pub}, but efficient decryption requires knowledge of the private key k_{priv}. Obviously the function used to create k_{pub} from k_{priv} must be difficult to invert, since k_{pub} is public knowledge and k_{priv} allows efficient decryption.

It may come as a surprise to learn that no one knows whether one-way functions exist. In fact, a proof of the existence of one-way functions would simultaneously prove that $\mathcal{P} \neq \mathcal{N}P$, thereby solving the most famous open problem in complexity theory.[3] There are various candidates for one-way functions in the literature, some of which are used by modern public key encryption algorithms. The security of these systems rests on the *assumption* that inverting the function (or finding the private key from the public one) is a hard problem.

The situation is somewhat analogous to theories in physics that gain credibility over time, as they fail to be disproved and continue to explain or generate interesting phenomena. Diffie and Hellman made several suggestions in [16] for one-way functions, including knapsack systems and exponentiation mod q, but they did not produce an example of a PKC, mainly for lack of finding the right trap door information. They did, however, describe a public key method by which certain material could be shared over an insecure channel. Their method, which is now called Diffie-Hellman Key Exchange, is based on the assumption that the discrete logarithm problem (DLP) is difficult to solve. We discuss the DLP in Section 2.2 and then describe their key exchange method in Section 2.3 In their paper, Diffie and Hellman also defined a variety of cryptanalytic attacks and introduced the important concepts of digital signatures and one-way authentication, which we discuss in Chapter 8.

With the publication of [16] in 1976, the race was on to invent a practical

[3] The $\mathcal{P} = \mathcal{N}P$ problem is one of the so-called Millenium Prizes, each of which has a \$1,000,000 prize attached.

public key cryptosystem. Within two years, two papers describing public key cryptosystems were published: the RSA scheme of Rivest, Shamir and Adelman [43] and the knapsack scheme of Merkle and Hellman [34]. Of these two, only RSA has withstood the test of time, in the sense that its underlying hard problem of integer factorization is still sufficiently computationally difficult to allow RSA to function efficiently. By way of contrast, the knapsack system of Merkle and Hellman was shown to be insecure at practical computational levels [49]. However, the cryptanalysis of knapsack systems introduces important links to hard computational problems in the theory of integer lattices, which we explore further in Chapter 6.

2.2 The discrete logarithm problem

The discrete log problem is a mathematical problem that arises in many settings, ranging from the mod p version described in this section to the elliptic curve version studied later in Chapter 5 to other, more general, algebraic groups. The first published public key construction, due to Diffie and Hellman [16], is based on the discrete logarithm problem for a finite field $\mathbb{F}_p$, where recall from Section 1.4 that $\mathbb{F}_p$ is a field consisting of a prime number of element p. Recall also that $\mathbb{F}_p$ is an alternative notation for $\mathbb{Z}/p\mathbb{Z}$, so we will interchangeably refer to elements of $\mathbb{F}_p$ and integers modulo p.

Let p be a (large) prime and let g be a primitive root modulo p (Theorem 1.25). This means that every nonzero element of $\mathbb{F}_p$ is equal to some power of g. In particular, $g^{p-1} = 1$ and no smaller power of g is equal to 1. We also recall Fermat's Little Theorem 1.22, which says that

$$a^{p-1} = 1 \qquad \text{for every nonzero } a \in \mathbb{F}_p.$$

Definition. Let g be a primitive root for $\mathbb{F}_p$ and let h be a nonzero element of $\mathbb{F}_p$. The *Discrete Logarithm Problem* (DLP) is the problem of finding an exponent x so that

$$g^x \equiv h \pmod{p}.$$

The number x is called the *discrete logarithm of h to the base g* and is denoted by $\log_g(h)$.

Remark 2.1. An older term for the discrete logarithm is the *index*, denoted $\mathrm{ind}_g(h)$. This terminology is still the one most commonly used in number theory. It is also convenient if there is a danger of confusion between ordinary logarithms and discrete logarithms, since for example, the quantity $\log_2$ frequently occurs in both contexts.

Remark 2.2. The discrete logarithm $\log_g(h)$ is not actually a well-defined integer, since the congruence $g^x \equiv h \pmod{p}$ has many solutions. This follows from Fermat's Little Theorem 1.22, which says that $g^{p-1} \equiv 1 \pmod{p}$. Hence if x is one solution, then

$$g^{x+k(p-1)} = g^x \cdot (g^{p-1})^k \equiv h \cdot 1^k \equiv h \pmod{p},$$

so $x + k(p-1)$ is also a solution for every $k \in \mathbb{Z}$. Thus $\log_g(h)$ is only defined up to adding or subtracting multiples of $p-1$, i.e. $\log_g(h)$ is defined modulo $p-1$, and it is not hard to verify (see Exercise 2.3(a)) that $\log_g$ gives a well-defined function[4]

$$\log_g : \mathbb{F}_p^* \longrightarrow \frac{\mathbb{Z}}{(p-1)\mathbb{Z}}. \tag{2.1}$$

Sometimes, for concreteness, we refer to "the" discrete logarithm as the appropriate integer lying between 0 and $p-2$.

Remark 2.3. It is not hard to prove (see Exercise 2.3(b)) that

$$\log_g(ab) = \log_g(a) + \log_g(b) \qquad \text{for all } a, b \in \mathbb{F}_p^*.$$

Thus calling $\log_g$ a "logarithm" is reasonable, since it converts multiplication into addition in the same way as does the usual logarithm function. In fancier terminology, the discrete logarithm $\log_g$ is a group isomorphism from $\mathbb{F}_p^*$ to $\mathbb{Z}/(p-1)\mathbb{Z}$.

Example 2.4. The number $p = 56509$ is prime, and it turns out that $g = 2$ is a primitive root modulo p. How would we go about calculating the discrete logarithm of $h = 38679$? The only method that is immediately obvious is to compute

$$2^2,\ 2^3,\ 2^4,\ 2^5,\ 2^6,\ 2^7, \ldots \pmod{56509}$$

until we find some power that equals 38679. It would be difficult to do this by hand, but a computer quickly finds that $\log_p(h) = 11235$. You can verify this by calculating $2^{11235} \bmod 56509$ and checking that it is equal to 38679.

[4] If you have studied complex analysis, you may have noticed an analogy with the complex logarithm, which is not actually well-defined on $\mathbb{C}^*$. This is due to the fact that $e^{2\pi i} = 1$, so $\log(z)$ is only well-defined up to adding or subtracting multiples of $2\pi i$. The complex logarithm thus defines an isomorphism from $\mathbb{C}^*$ to the quotient group $\mathbb{C}/2\pi i\mathbb{Z}$, analogous to (2.1).

n	$g^n \bmod p$
1	627
2	732
3	697
4	395
5	182
6	253
7	543
8	760
9	374
10	189

n	$g^n \bmod p$
11	878
12	21
13	934
14	316
15	522
16	767
17	58
18	608
19	111
20	904

h	$\log_g(h)$
1	0
2	183
3	469
4	366
5	356
6	652
7	483
8	549
9	938
10	539

h	$\log_g(h)$
11	429
12	835
13	279
14	666
15	825
16	732
17	337
18	181
19	43
20	722

Table 2.1: Powers and discrete logarithms for $g = 627$ modulo $p = 941$

Remark 2.5. It must be emphasized that the discrete logarithm bears little resemblance to the continuous logarithm defined on the real or complex numbers. The terminology is still reasonable, because in both instances the process of exponentiation is inverted—but exponentiation modulo p varies in a very irregular way with the exponent, contrary to the behavior of its continuous counterpart. The random looking behavior of exponentiation modulo p is apparent from even a cursory glance at a table of values such as those in Table 2.1, where we list the first few powers and the first few discrete logarithms for $g = 627$ modulo $p = 941$.

Our statement of the discrete logarithm problem includes the assumption that the base g is a primitive root modulo p, but this is not necessary. In general, for any $g \in \mathbb{F}_p^*$ and any $h \in \mathbb{F}_p^*$, the discrete logarithm problem is the determination of an exponent x satisfying $g^x \equiv h \pmod{p}$, assuming that such an x exists.

More generally, rather than taking nonzero elements of a finite field $\mathbb{F}_p$ and multiplying them together or raising them to powers, we can take elements of any group and use the group law instead of multiplication. (If you are unfamiliar with the theory of groups, see Section 2.2.1 below for an overview.) This leads to the general discrete logarithm problem.

Definition. Let G be a group whose group law we denote by the symbol $\star$. The *Discrete Logarithm Problem* for G is to determine, for any two given elements g and h in G, an integer x so that

$$\underbrace{g \star g \star g \star \cdots \star g}^{x \text{ times}} = h.$$

2.2.1 A brief aside on the theory of groups

For readers unfamiliar with the theory of groups, we briefly introduce a few basic concepts which should help to place the study of discrete logarithms, both here and in Chapter 5, into a broader context.

We first observe that exponentiation of elements in $\mathbb{F}_p^*$ is simply repeated multiplication, and that multiplication in $\mathbb{F}_p^*$ has some important properties, including:

- There is an element $1 \in \mathbb{F}_p^*$ satisfying $1 \cdot a = a$ for every $a \in \mathbb{F}_p^*$.
- Every $a \in \mathbb{F}_p^*$ has an inverse $a^{-1} \in \mathbb{F}_p^*$ satisfying $a \cdot a^{-1} = a^{-1} \cdot a = 1$.
- Multiplication is associative, $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ for all $a, b, c \in \mathbb{F}_p^*$.
- Multiplication is commutative, $a \cdot b = b \cdot a$ for all $a, b \in \mathbb{F}_p^*$.

There are other objects that have an operation satisfying these properties. For example, if instead of multiplication on $\mathbb{F}_p^*$, we use addition on $\mathbb{F}_p$, then all four properties are still true, since

$$0 + a = 0, \qquad a + (-a) = 0, \qquad (a + b) + c = a + (b + c), \qquad a + b = b + a.$$

A group is simply an abstraction of this idea.

Definition. A *group* consists of a set G and a rule, which we denote by $\star$, for combining two elements $a, b \in G$ to obtain an element $a \star b \in G$. The composition rule $\star$ is required to have the following three properties:

[**Identity Law**] There is an $e \in G$ such that
$$e \star a = a \star e = a \text{ for every } a \in G.$$

[**Inverse Law**] For every $a \in G$ there is an $a^{-1} \in G$ satisfying
$$a \star a^{-1} = a^{-1} \star a = e.$$

[**Associative Law**] $a \star (b \star c) = (a \star b) \star c$ for all $a, b, c \in G$.

If in addition, the composition rule satisfies

[**Commutative Law**] $a \star b = b \star a$ for all $a, b \in G$,

then the group is called *commutative* or *abelian*.

Example 2.6. Groups are ubiquitous in mathematics and in the physical sciences. We present a few examples.

(a) $G = \mathbb{F}_p^*$ and $\star =$ multiplication. The identity element is $e = 1$. Proposition 1.21 tells us that inverses exist.

(b) $G = \mathbb{Z}$ and $\star =$ addition. The identity element is $e = 0$ and the inverse of a is $-a$. Similarly, $G = \mathbb{Z}/N\mathbb{Z}$ with $\star =$ addition is a group.

(c) Note that $G = \mathbb{Z}$ and $\star =$ multiplication is not a group, since some elements do not have inverses.

(d) An example of a noncommutative group is

$$G = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} : a, b, c, d \in \mathbb{R} \text{ and } ad - bc \neq 0 \right\}$$

with operation $\star =$ matrix multiplication. The identity element is $e = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$ and the inverse is given by the familiar formula

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \begin{pmatrix} \frac{d}{ad-bc} & \frac{-b}{ad-bc} \\ \frac{-c}{ad-bc} & \frac{a}{ad-bc} \end{pmatrix}$$

Notice that G is noncommutative, since for example, $\left(\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right)\left(\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right)$ is not equal to $\left(\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right)\left(\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right)$.

(e) More generally, we can use matrices of any size. This gives the group

$$\mathrm{GL}_n(\mathbb{R}) = \big\{ n\text{-by-}n \text{ matrices } A \text{ with real coefficients and } \det(A) \neq 0 \big\}$$

with operation $\star =$ matrix multiplication. We an also replace $\mathbb{R}$ by some other field, for example, the finite field $\mathbb{F}_p$, see exercise 2.6.

If g is an element of a group G and x is an integer, then exponentiation g^x means to apply the group operation x times to the element g

$$g^x = \underbrace{g \star g \star g \star \cdots \star g}_{x \text{ repetitions}}.$$

Thus in the group $\mathbb{F}_p^*$ with multiplication, exponentiation g^x has the usual meaning, multiply g with itself x times. But in the group $\mathbb{F}_p$ with addition, "exponentiation" g^x means to add g to itself x times, $g + g + \cdots + g$, so it would be more typical to write this as $x \cdot g$. But this is just a matter of notation, the key underlying concept is repeatedly applying the group operation to an element of the group.

*** put in a problem about $\mathrm{GL}_2(\mathbb{F}_p)$, including listing all of the elements in $\mathrm{GL}_2(\mathbb{F}_2)$ ***

As noted earlier, the discrete logarithm problem in G is the problem of finding an exponent x satisfying $g^x = h$. For $G = \mathbb{F}_p^*$, this is our original discrete logarithm problem, and it is believed to be quite difficult to solve if p is large.

How hard is the DLP for the group $G = \mathbb{F}_p$ with addition? We need to solve $x \cdot g \equiv h \pmod{p}$ for x. As we know from Remark 1.14 in Section 1.3, the extended Euclidean algorithm provides an extremely fast method of solution, indeed taking only $O(\log p)$ steps. Thus the DLP for $G = \mathbb{F}_p$ with addition is very easy to solve.

This is an important lesson to learn. Different discrete logarithm problems may display different levels of difficulty for their solution. In Chapter 5 we discuss the DLP for elliptic curve groups. The elliptic curve DLP is believed to be even more difficult than the DLP for $\mathbb{F}_p^*$.

We make one final observation. We have stated that the DLP on $\mathbb{F}_p^*$ is believed to be a difficult problem. However, this is not true for all primes p. There are some primes for which it is actually not that hard. For example, we will see in Section 2.6 that if the number $p-1$ factors into a product of small primes, then the DLP in $\mathbb{F}_p^*$ is comparatively easy.

2.3 Diffie-Hellman key exchange

The Diffie-Hellman key exchange protocol provides a solution to the following dilemma. Alice and Bob want to share a secret key for use in a symmetric key cipher, but their only means of communication is insecure. Every piece of information that they exchange is observed by their adversary Eve. How is it possible for Alice and Bob to share a key without also making it available to Eve?

At first glance, it might appear that Alice and Bob face an impossible task. It was a brilliant insight of Diffie and Hellman that the difficulty of the discrete logarithm problem for $\mathbb{F}_p^*$ provides a possible solution.

The first step is for Alice and Bob to agree on a large prime p and a nonzero integer g modulo p. The values of p and g are public knowledge, so

Eve knows them, too.

The next step is for Alice to pick an integer a that she does not reveal to anyone. At the same time, Bob also picks an integer b that he keeps secret. Bob and Alice then use their secret integers to compute

$$\underbrace{A \equiv g^a \pmod{p}}_{\text{Alice computes this}} \qquad \text{and} \qquad \underbrace{B \equiv g^b \pmod{p}}_{\text{Bob computes this}}.$$

They next exchange these computed values, Alice sends A to Bob, and Bob sends B to Alice. Note that Eve gets to see the values of A and B, since they are sent over the insecure communication channel.

Finally, Bob and Alice again use their secret integers to compute

$$\underbrace{A' \equiv B^a \pmod{p}}_{\text{Alice computes this}} \qquad \text{and} \qquad \underbrace{B' \equiv A^b \pmod{p}}_{\text{Bob computes this}}.$$

The values that they compute, A' and B' respectively, are actually the same, since

$$A' \equiv B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \equiv B' \pmod{p},$$

and this common value is their exchanged key. For your convenience, the Diffie-Hellman key exhange protocol is summarized in Table 2.2.

Example 2.7. *** put in a numerical DH example here, before discussing Eve's dilemma ***

Consider now Eve's dilemma. She knows the values of A and B, so she knows the values of g^a and g^b. She also knows the value of g, so if she can solve the DLP, then she can find a and b, after which it is easy for her to compute Alice and Bob's shared key g^{ab}. It thus appears that Alice and Bob are safe provided Eve is not able to solve the DLP.

However, this is not quite correct. It is true that one method of finding Alice and Bob's shared key is to solve the DLP, but that is not the exact problem that Eve needs to solve. The security of Alice and Bob's shared key rests on the difficulty of the following possibly easier, but closely related, problem.

Definition. Let p be a prime number and g an integer. The *Diffie-Hellman Problem* (DHP) is the problem of computing the value of $g^{ab} \pmod{p}$ from the known values of $g^a \pmod{p}$ and $g^b \pmod{p}$.

Public Parameter Creation	
A trusted party chooses and publishes a (large) prime p and an integer g having large order in $\mathbb{F}_p^*$.	
Private Computations	
Alice	**Bob**
Choose a secret integer a. Compute $A \equiv g^a \pmod{p}$.	Choose a secret integer b. Compute $B \equiv g^b \pmod{p}$.
Public Exchange of Values	
Alice sends A to Bob $\longrightarrow$	A
B $\longleftarrow$	Bob sends B to Alice
Further Private Computations	
Alice	**Bob**
Compute the number $B^a \pmod{p}$.	Compute the number $A^b \pmod{p}$.
The shared secret value is $B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \pmod{p}$.	

Table 2.2: Diffie-Hellman key exchange

It is clear that the DHP is no harder than the DLP. If Eve can solve the DLP, then she can deduce Alice and Bob's secret exponents a and b from the intercepted values $A = g^a$ and $B = g^b$, and then it is easy for her to compute their shared key g^{ab}. (In fact, Eve only needs to compute one of a or b.) The converse is trickier. Suppose that Eve has an algorithm that efficiently solves the DHP. Can she use it to also efficiently solve the DLP? The answer is not known.

Example 2.8. Alice and Bob agree to use the prime $p = 941$ and the primitive root $g = 627$. Alice sends to Bob the number $A = 390$ and Bob sends to Alice the number $B = 691$. Eve sees these two numbers, but as far as anyone knows, she needs to solve at least one of the two congruences

$$627^a \equiv 390 \pmod{941} \qquad \text{or} \qquad 627^b \equiv 691 \pmod{941}$$

in order to discover Alice and Bob's shared key. It turns out that $a = 347$ and $b = 781$, so their shared key is $627^{347\cdot 781} \equiv 470 \pmod{941}$. Of course, in this example the numbers are much too small to afford Alice and Bob any real security, since Eve can simply make a table of all of the powers of 627 modulo 941. In practice, Alice and Bob choose a prime p having approxmiately 1000 binary digits (i.e. $p \approx 2^{1000}$), in which case Eve faces a difficult task indeed.

2.4 The ElGamal public key cryptosystem

The first public key cryptosystem was the RSA system of Rivest, Shamir, and Adelman [43], which they published in 1978. RSA was, and still is, a fundamentally important discovery and we discuss it in great detail in Chapter 4. However, although RSA was historically first, the most natural development of a public key cryptosystem following the Diffie-Hellman paper [16] is a system described by Taher ElGamal in 1985 [17]. The ElGamal public key encryption algorithm is based on the discrete log problem and is quite similar in many ways to the Diffie-Hellman key exchange described in Section 2.3. In this section we describe the version of the ElGamal PKC that is based on the discrete logarithm problem for $\mathbb{F}_p^*$, but the construction works quite generally using the DLP in any group, and indeed in Chapter 5 we discuss a version of the ElGamal PKC based on elliptic curve groups.

The ElGamal PKC is our first example of a public key cyrptosystem, so we take it slowly and provide all of the details. Alice begins by making public some information consisting of a public key and an algorithm. The key is simply a number and the algorithm is the method by which Bob will encrypt his message using Alice's public key. At the same time, Alice keeps secret her private key, which is another number that allows her, and only her, to decrypt message that have been encrypted using her public key.

This is all somewhat vague and applies to any public key cryptosystem. For the ElGamal PKC, Alice needs a large prime number p for which the discrete logarithm problem in $\mathbb{F}_p^*$ is difficult and a primitive root g modulo p. She may choose g and p herself, or they may have been preselected by some trusted party such as an industry panel or government agency.

Alice next chooses a secret number a to act as her private key, and she computes the quantity

$$A \equiv g^a \pmod{p}.$$

Notice the resemblance to the Diffie-Hellman key exchange protocol. Alice publishes her public key A and she keeps her private key a secret.

Now Bob wants to encrypt a message using Alice's public key. He first converts his plaintext message into an integer m modulo p, that is, into a number $1 \leq m < p$. In theory, Bob could just represent strings of English letters as strings of integers and convert the integers into blocks of bits. And if his message is too long to fit into one number modulo p, he can use transmit it as a sequence of numbers modulo p. In practice, there are significant security problems with using a naive English-to-number conversion scheme of this sort, but for simplicity we assume that Bob simply wants to send a message consisting of a single number m modulo p.

Before encrypting his plaintext message m, Bob first randomly generates another number k modulo p. He uses k to encrypt one, only one message, and then he discards it. The number k is called an *ephemeral key*, since it exists only for the purposes of encrypting a single message.[5]

Bob uses his plaintext message m and his chosen random ephemeral key k to compute the two quantities

$$c_1 = g^k \pmod{p} \qquad \text{and} \qquad c_2 = mA^k \pmod{p},$$

where recall that g and p are public parameters and A is Alice's public key. Bob's ciphertext, i.e. his encryption of m, is the pair of numbers (c_1, c_2), which he sends to Alice.

How does Alice decrypt Bob's ciphertext (c_1, c_2)? Since Alice knows a, she can compute the quantity

$$x \equiv c_1^a \pmod{p},$$

and hence also $x^{-1} \pmod{p}$. Alice next multiplies c_2 by x^{-1}, and lo and behold, the resulting value is the plaintext m. To see why, we expand the value of $x^{-1} \cdot c_2$ and find that

$$\begin{aligned} x^{-1} \cdot c_2 &\equiv (c_1^a)^{-1} \cdot c_2 && \pmod{p}, && \text{since } x \equiv c_1^a, \\ &\equiv (g^{ak})^{-1} \cdot (mA^k) && \pmod{p}, && \text{since } c_1 \equiv g^k \text{ and } c_2 \equiv mA^k, \\ &\equiv (g^{ak})^{-1} \cdot (m(g^a)^k) && \pmod{p}, && \text{since } A \equiv g^a, \\ &\equiv m && \pmod{p}, && \text{since the } g^{ak} \text{ terms cancel out.} \end{aligned}$$

What is Eve's task in trying to decrypt the message? Eve knows the public paramters p and g, and she also knows the value of $A \equiv g^a \pmod{p}$,

[5] Most public key cryptosystems require the use of random numbers in order to operate securely. The generation of random or random-looking integers is actually a delicate process. We discuss the problem of generating pseudorandom numbers in Section 8.6, but for now we ignore this issue and assume that Bob has no trouble generating random numbers modulo p.

<table>
<tr><th colspan="2">Public Parameter Creation</th></tr>
<tr><td colspan="2">A trusted party chooses and publishes a (large)
prime p and a primitive root g modulo p.</td></tr>
<tr><th>Alice</th><th>Bob</th></tr>
<tr><th colspan="2">Key Creation</th></tr>
<tr><td>Choose a private key $1 \leq a \leq p-1$.
Compute $A = g^a \pmod{p}$.
Publish the public key A.</td><td></td></tr>
<tr><th colspan="2">Encryption</th></tr>
<tr><td></td><td>Choose plaintext m.
Choose random ephemeral key k.
Use Alice's public key A
to compute $c_1 = g^k \pmod{p}$.
and $c_2 = mA^k \pmod{p}$.
Send ciphtertext (c_1, c_2) to Alice.</td></tr>
<tr><th colspan="2">Decryption</th></tr>
<tr><td>Compute $(c_1^a)^{-1} \cdot c_2 \pmod{p}$.
This quantity is equal to m.</td><td></td></tr>
</table>

Table 2.3: ElGamal key creation, encryption, and decryption

since Alice's public key A is public knowledge. If Eve can solve the discrete logarithm problem, she can find a and decrypt the message. Otherwise it appears difficult for Eve to find the plaintext, although there are subtleties, some of which we discuss after doing an example with small numbers.

Example 2.9. Alice uses the prime $p = 467$ and the primitive root $g = 2$. She chooses $a = 153$ to be her private key and computes

$$A \equiv g^a \equiv 2^{153} \equiv 224 \pmod{467}$$

as her public key.

Bob decides to send Alice the message $m = 331$. He chooses an ephemeral key $k = 197$ at random and computes two quantities

$$c_1 \equiv 2^{197} \equiv 87 \pmod{467} \qquad \text{and} \qquad c_2 \equiv 331 \cdot 224^{197} \equiv 57 \pmod{467}.$$

The pair $(c_1, c_2) = (87, 57)$ is the ciphertext that Bob sends to Alice.

Alice, knowing $a = 153$, first computes

$$x \equiv c_1^a \equiv 87^{153} \equiv 367 \pmod{467}, \quad \text{and then} \quad x^{-1} \equiv 14 \pmod{467}.$$

Finally, she computes

$$c_2 x^{-1} \equiv 57 \cdot 14 \equiv 331 \pmod{467}$$

and recovers the plaintext message m.

Remark 2.10. *** discuss briefly the 2-to-1 expansion of DH

We now show that ElGamal encryption, as summarized in Table 2.3, is secure against a chosen ciphertext attack, in the sense that anyone who can decrypt arbitrary ciphertexts is also able to solve the Diffie-Hellman problem.

Proposition 2.11. *Fix a prime p and base g to use for ElGamal encryption. Suppose that Eve has access to an oracle that decrypts arbitrary ElGamal ciphertexts encrypted using arbitrary ElGamal public keys. Then she can use the oracle to solve the Diffie-Hellman problem described on page 65.*

Proof. The Diffie-Hellman problem says that Eve is given the two values

$$A \equiv g^a \pmod{p} \qquad \text{and} \qquad B \equiv g^b \pmod{p}$$

and that she is required to compute the value of $g^{ab} \pmod{p}$. Keep in mind that she knows the values of A and B, but she does not know either of the values a or b.

Eve tells the oracle that the ElGamal public key is the number A and that the ciphertext is (B, c_2), where she may choose any value that she wants for c_2. The oracle recovers the supposed plaintext m, where note that the plaintext satisfies

$$m \equiv (c_1^a)^{-1} \cdot c_2 \equiv (B^a)^{-1} \cdot c_2 \equiv (g^{ab})^{-1} \cdot c_2 \pmod{p}.$$

After the oracle tells Eve the value of m, she simply computes

$$m^{-1} \cdot c_2 \equiv g^{ab} \pmod{p}$$

to find the value of $g^{ab} \pmod{p}$. Thus Eve has solved the Diffie-Hellman problem. It is worth observing that although, with the oracle's help, she has computed $g^{ab} \pmod{p}$, she has done so without knowledge of a or b, so she has only solved the Diffie-Hellman problem, not the discrete logarithm problem. □

2.5 The Chinese Remainder Theorem [M]

The Chinese Remainder Theorem deals with the problem of solving systems of simultaneous congruences. The simplest situation is a system of two congruences

$$x \equiv a \pmod{m} \qquad \text{and} \qquad x \equiv b \pmod{n}, \tag{2.2}$$

in which case the Chinese Remainder Theorem states that if $\gcd(m, n) = 1$, then there is a unique solution modulo mn.

The first recorded instance of a problem of this type appears in a Chinese mathematical work from the late third or early fourth century and, somewhat surprisingly, actually deals with the harder problem of three simultaneous congruences.

> "We have a number of things, but we do not know exactly how many. If we count them by threes, we have two left over. If we count them by fives, we have three left over. If we count them by sevens, we have two left over. How many things are there?"
> *Sun Tzu Suan Ching* (Master Sun's Mathematical Manual)
> circa 300 AD, volume 3, problem 26.

The Chinese Remainder Theorem and its generalizations have many applications in number theory. In the next section we will put it to immediate use to solve certain instances of the discrete logarithm problem. We begin with an example of two simultaneous congruences. As you read this example, notice how it describes an algorithm that allows us to find the solution, and is not merely an abstract statement that a solution exists.

Example 2.12. We look for an integer x that simultaneously solves both of the congruences

$$x \equiv 1 \pmod{5} \qquad \text{and} \qquad x \equiv 9 \pmod{11}. \tag{2.3}$$

The first congruence tells us that $x \equiv 1 \pmod{5}$, so the full set of solutions to the first congruence is the collection of integers

$$x = 1 + 5y, \qquad y \in \mathbb{Z}. \tag{2.4}$$

Substituting (2.4) into the second congruence in (2.3) gives

$$1 + 5y \equiv 9 \pmod{11}, \quad \text{and hence} \quad 5y \equiv 8 \pmod{11}. \tag{2.5}$$

We solve for y by multiplying both sides of (2.5) by the inverse of 5 modulo 11. This inverse exists because $\gcd(5, 11) = 1$ and can be computed using the procedure described in Proposition 1.13 (see also Remark 1.14), or in this case, one can observe that $11 - 2 \cdot 5 = 1$, so the inverse of 5 modulo 11 is -2, which is the same as 9.

In any case, multiplying both sides of (2.5) by 9 yields

$$y \equiv 9 \cdot 8 \equiv 72 \equiv 6 \pmod{11}.$$

Finally, substituting this value of y into (2.4) gives the solution

$$x = 1 + 5 \cdot 6 = 31$$

to the original problem.

The procedure outlined in Example 2.12 yields a general formula for the solution of two simultaneous congruences (see Exercise 2.14), but it is much better to learn the method, rather than memorizing a formula. This is especially true because the Chinese Remainder Theorem applies to systems of arbitrarily many simultaneous congruences.

Theorem 2.13 (Chinese Remainder Theorem)**.** *Let $m_1, m_2, \ldots, m_k$ be pairwise relatively prime integers, i.e.* $\gcd(m_i, m_j) = 1$ *for all $i \neq j$, and let $a_1, a_2, \ldots, a_k$ be arbitrary integers. Then the system of simultaneous congruences*

$$x \equiv a_1 \ (mod\ m_1), \quad x \equiv a_2 \ (mod\ m_2), \quad \ldots \quad x \equiv a_k \ (mod\ m_k) \tag{2.6}$$

has a solution $x = c$. Further, if $x = c$ and $x = c'$ are both solutions, then

$$c \equiv c' \pmod{m_1 m_2 \cdots m_k}. \tag{2.7}$$

Proof. Suppose that for some value of i we have already managed to find a solution $x = c_i$ to the system of simultaneous congruences

$$x \equiv a_1 \ (\text{mod } m_1), \quad x \equiv a_2 \ (\text{mod } m_2), \quad \ldots \quad x \equiv a_i \ (\text{mod } m_i). \tag{2.8}$$

For example, if $i = 1$, then $c_1 = a_1$ works. We are going to explain how to find a solution to

$$x \equiv a_1 \ (\text{mod } m_1), \quad x \equiv a_2 \ (\text{mod } m_2), \quad \ldots \quad x \equiv a_{i+1} \ (\text{mod } m_{i+1}).$$

The idea is to look for a solution having the form

$$x = c_i + m_1 m_2 \cdots m_i y.$$

Notice that this value of x still satisfies all of the congruences (2.8), so all that we need to do is choose y so that it also satisfies $x \equiv a_{i+1} \ (\text{mod } m_{i+1})$. In other words, we need to find a value of y satisfying

$$c_i + m_1 m_2 \cdots m_i y \equiv a_{i+1} \pmod{m_{i+1}}.$$

Proposition 1.13(b) and the fact that $\gcd(m_{i+1}, m_1 m_2 \cdots m_i) = 1$ imply that we can always do this. This completes the proof of the existence of a solution. We leave to you the task of proving that different solutions satisfy (2.7). □

The proof of the Chinese Remainder Theorem 2.13 is easily converted into an algorithm for finding the solution to a system of simultaneous congruences. An example suffies to illustrate the general method.

Example 2.14. We solve the three simultaneous congruences

$$x \equiv 2 \ (\text{mod } 3), \quad x \equiv 3 \ (\text{mod } 7), \quad x \equiv 4 \ (\text{mod } 16). \tag{2.9}$$

The Chinese Remainder Theorem says that there is a unique solution modulo 336, since $336 = 3 \cdot 7 \cdot 16$. We start with the solution $x = 2$ to the first congruence $x \equiv 2 \pmod{3}$. We use it to form the general solution $x = 2 + 3y$ and substitute it into the second congruence to get

$$2 + 3y \equiv 3 \ (\text{mod } 7).$$

This simplifies to $3y \equiv 1 \pmod{7}$, and we multiply both sides by 5 (since 5 is the inverse of 3 modulo 7) to get $y \equiv 5 \pmod{7}$. This gives the value

$$x = 2 + 3y = 2 + 3 \cdot 5 = 17$$

as a solution to the first two congruences in (2.9).

The general solution to the first two congruences is thus $x = 17 + 21z$. We substitute this into the third congruence to obtain

$$17 + 21z \equiv 4 \pmod{16}.$$

This simplifies to $5z \equiv 3 \pmod{16}$. We multiply by 13, which is the inverse of 5 modulo 16, to obtain

$$z \equiv 3 \cdot 13 \equiv 39 \equiv 7 \pmod{16}.$$

Finally, we substitute this into $x = 17 + 21z$ to get the solution

$$x = 17 + 21 \cdot 7 = 164.$$

All other solutions are obtained by adding and subtracting multiples of 336 to this particular solution.

2.5.1 Solving congruences with composite moduli

It is usually easiest to solve a congruence with a composite modulus by first solving several congruences modulo primes (or prime powers) and then fitting together the solutions using the Chinese Remainder Theorem. We illustrate the principal in this section by discussing the problem of finding square roots modulo m. It turns out that it is relatively easy to compute square roots modulo a prime, and indeed for some primes, such as those congruent to 3 modulo 4, it is extremely easy as shown by the following proposition.

Proposition 2.15. *Let p be a prime satisfying $p \equiv 3 \pmod 4$. Let a be an integer that has a square root modulo p. Then*

$$b \equiv a^{\frac{p+1}{4}} \pmod p$$

is a square root of a modulo p.

Proof. Let g be a primitive root modulo p. Then a is equal to some power of g, and the fact that a has a square root modulo p means that a is an even power of g, say $a \equiv g^{2k} \pmod p$. (See Exercise 2.5.) Now we compute

$$\begin{aligned} b^2 &\equiv a^{\frac{p+1}{2}} && \pmod p && \text{definition of } b, \\ &\equiv (g^{2k})^{\frac{p+1}{2}} && \pmod p && \text{since } a \equiv g^{2k} \pmod p, \\ &\equiv g^{(p+1)k} && \pmod p && \\ &\equiv g^{2k+(p-1)k} && \pmod p && \\ &\equiv a \cdot (g^{p-1})^k && \pmod p && \text{since } a \equiv g^{2k} \pmod p, \\ &\equiv a && \pmod p && \text{since } g^{p-1} \equiv 1 \pmod p. \end{aligned}$$

Hence b is indeed a square root of a modulo p. □

Example 2.16. A square root of $a = 2201$ modulo the prime $p = 4127$ is

$$b \equiv a^{\frac{p+1}{4}} \equiv 2201^{\frac{4128}{4}} \equiv 2201^{1032} \equiv 3718 \pmod{4127}.$$

Suppose now that we want to compute a square root modulo m, where m is not necessarily a prime. An efficient method is to factor m, compute the square root modulo each of the prime (or prime power) factors, and then combine the solutions using the Chinese Remainder Theorem. An example makes the idea clear.

Example 2.17. We look for a solution to the congruence

$$x^2 \equiv 197 \pmod{437}. \tag{2.10}$$

The modulus factors as $437 \equiv 19 \cdot 23$, so we first solve the two congruences

$$y^2 \equiv 197 \equiv 7 \pmod{19} \qquad \text{and} \qquad z^2 \equiv 197 \equiv 13 \pmod{23}.$$

This can be done using Proposition 2.15 or by trial and error. In any case, we find that

$$y \equiv 8 \pmod{19} \qquad \text{and} \qquad z \equiv 6 \pmod{23}.$$

Next we use the Chinese Remainder Theorem to solve the simultaneous congruences

$$x \equiv 8 \pmod{19} \qquad \text{and} \qquad x \equiv 6 \pmod{23}. \tag{2.11}$$

We find that $x \equiv 236 \pmod{437}$, which gives the desired solution to (2.10).

Remark 2.18. The solution to Example 2.17 is not unique. In the first place, we can always take the negative,

$$-236 \equiv 201 \pmod{437}$$

to get a second square root of 197 modulo 437. If the modulus were prime, there would be only these two square roots (Exercise 1.25(a)). However, since $437 = 19 \cdot 23$ is composite, there are two others. In order to find them, we replace either 8 or 6 with its negative in (2.11). This leads to the values $x = 144$ and $x = 293$ as the other two square roots of 197 modulo 437.

Remark 2.19. It is clear from Example 2.17 (see also Exercises 2.17 and 2.18) that it is relatively easy to compute square roots modulo m if one knows how to factor m into a product of prime powers. However, suppose that m is so large that we are not able to factor it. It is then a very difficult problem to find square roots modulo m. Indeed, in a certain reasonably precise sense, it is just as difficult to compute square roots modulo m as it is to factor m. This provides a hard mathematical problem that has been used to construct public key cryptosystems.

2.6 The Pohlig-Hellman algorithm

In the next chapter we discuss collision methods that solve the discrete logarithm problem in $\mathbb{F}_p$ in approximately $\sqrt{p}$ steps. More precisely, we will prove the following result.

Proposition 2.20 (Collision Method)**.** *Let $f \in \mathbb{F}_p^*$ be an element of order N, i.e. $f^N = 1$ and no smaller power of h is equal to 1. Then the discrete logarithm problem*

$$f^x = b \tag{2.12}$$

can be solved in a small multiple of $\sqrt{N}$ steps

See Proposition 3.42 in Section 3.4.1 for a complete statement and proof. Proposition 2.20 applies equally well to every finite field $\mathbb{F}_p$, and there is no *a* *priori* reason to suspect that the discrete logarithm problem might be easier for some primes p than it is for others of a comparable size. However, it turns out that the DLP is in fact considerably easier to solve if $p-1$ happens to factor into a product of many small primes. This observation, which is due to Pohlig and Hellman [40], relies on the following idea.

Proposition 2.21. *Let $h \in \mathbb{F}_p^*$ be an element of order q^e, i.e. $h^{q^e} = 1$ in $\mathbb{F}_p$, and no smaller power of h is equal to 1. Then assuming that there is a solution, the discrete logarithm problem*

$$h^x = b \quad \text{in } \mathbb{F}_p, \tag{2.13}$$

can be solved in no more than $e\sqrt{q}$ steps.

Proof. Notice that if e is large, then the running time $e\sqrt{q}$ in Proposition 2.21 is much faster than the collision algorithm running time of $\sqrt{q^e}$ given by Proposition 2.20. We build up the value of x step-by-step by applying Proposition 2.20 to the element $h^{q^{e-1}}$ of order q.

We write the unkonwn exponent x in the form

$$x = x_0 + x_1 q + x_2 q^2 + \cdots + x_{e-1}q^{e-1} \tag{2.14}$$

and determine successively $x_0, x_1, x_2, \ldots$. We begin with the computation

$$\begin{aligned}
b^{q^{e-1}} &= (h^x)^{q^{e-1}} && \text{raising both sides of (2.13)} \\
& && \qquad \text{to the } q^{e-1} \text{ power} \\
&= \left(h^{x_0+x_1q+x_2q^2+\cdots+x_{e-1}q^{e-1}}\right)^{q^{e-1}} && \text{from (2.14)} \\
&= h^{x_0q^{e-1}} \cdot \left(h^{q^e}\right)^{x_1+x_2q+\cdots+x_{e-1}q^{e-2}} && \\
&= \left(h^{q^{e-1}}\right)^{x_0} && \text{since } h^{q^e} = 1.
\end{aligned}$$

Notice that the q^{th} power of $h^{q^{e-1}}$ is equal to 1, so $h^{q^{e-1}}$ is an element of order q in $\mathbb{F}_p$. Hence the equation

$$\left(h^{q^{e-1}}\right)^{x_0} = b^{q^{e-1}}$$

is a discrete logarithm problem whose base is an element of order q, so the collision method (Proposition 2.20) says that we can solve for x_0 in at most $\sqrt{q}$ steps. To summarize, we now know an exponent x_0 with the property that

$$h^{x_0q^{e-1}} = b^{q^{e-1}} \quad \text{in } \mathbb{F}_p.$$

We next do a similar computation, this time raising both sides of (2.13) to the q^{e-2} power, which yields

$$\begin{aligned}
b^{q^{e-2}} &= (h^x)^{q^{e-2}} \\
&= \left(h^{x_0+x_1q+x_2q^2+\cdots+x_{e-1}q^{e-1}}\right)^{q^{e-2}} \\
&= h^{x_0q^{e-2}} \cdot h^{x_1q^{e-1}} \cdot \left(h^{q^e}\right)^{x_2+x_3q+\cdots+x_{e-1}q^{e-3}} \\
&= h^{x_0q^{e-2}} \cdot h^{x_1q^{e-1}}
\end{aligned}$$

Keep in mind that we have already determined the value of x_0 and that the element $h^{q^{e-1}}$ has order q in $\mathbb{F}_p$. In order to find x_1, we must solve the discrete logarithm problem

$$\left(h^{q^{e-1}}\right)^{x_1} = \left(b \cdot h^{-x_0}\right)^{q^{e-2}} \quad \text{in } \mathbb{F}_p$$

for the unknown quantity x_1. Again the collision method (Proposition 2.20) gives a solution in $\sqrt{q}$ steps. Hence in $2\sqrt{q}$ steps, we have determined values for x_0 and x_1 sastifying

$$h^{(x_0+x_1q)q^{e-2}} = b^{q^{e-2}} \quad \text{in } \mathbb{F}_p.$$

Similarly, we find x_2 by solving the discrete logarithm problem

$$\left(h^{q^{e-1}}\right)^{x_2} = \left(b \cdot h^{-x_0-x_1q}\right)^{q^{e-3}},$$

and in general, after we have determined $x_0, \ldots, x_{i-1}$, then the value of x_i is obtained by solving

$$\left(h^{q^{e-i}}\right)^{x_i} = \left(b \cdot h^{-x_0-x_1q-\cdots-x_{i-1}q^{i-1}}\right)^{q^{e-i}} \quad \text{in } \mathbb{F}_p.$$

Each of these is a discrete logarithm problem whose base is of order q, so each of them can be solved in $\sqrt{q}$ steps. Hence after $e\sqrt{q}$ steps, we obtain an exponent $x = x_0 + x_1q + \cdots + x_{e-1}q^{e-1}$ satisfying $h^x = b$, thus solving the original discrete logarithm problem. □

Example 2.22. We do an example to clarify the algorithm described during the proof of Proposition 2.21. We solve

$$5448^x = 6909 \quad \text{in } \mathbb{F}_{11251}. \tag{2.15}$$

The prime $p = 11251$ has the property that $p - 1$ is divisible by 5^4, and it is easy to check that 5448 has order exactly 5^4 in $\mathbb{F}_p$. The first step is to solve

$$\bigl(5448^{5^3}\bigr)^{x_0} = 6909^{5^3}, \qquad \text{i.e.} \quad 11089^{x_0} = 11089.$$

This one is easy, the answer is $x_0 = 1$, so our inital value of x is $x = 1$.

The next step is to solve

$$\bigl(5448^{5^3}\bigr)^{x_1} = (6909 \cdot 5448^{-x_0})^{5^2} = (6909 \cdot 5448^{-1})^{5^2}, \qquad \text{i.e.} \quad 11089^{x_1} = 3742.$$

Note that we only need to check values of x_1 between 1 and 4, although if q were large, it would pay to use the collision algorithm to solve this discrete logarithm problem. In any case, the solution is $x_1 = 2$, so the value of x is now $x = 11 = 1 + 2 \cdot 5$.

Continuing on our merry way, we next solve

$$\bigl(5448^{5^3}\bigr)^{x_2} = (6909 \cdot 5448^{-x_0 - x_1 \cdot 5})^{5} = \bigl(6909 \cdot 5448^{-11}\bigr)^{5}, \qquad \text{i.e.} \quad 11089^{x_2} = 1.$$

Thus $x_2 = 0$, so the value of x remains at $x = 11$.

The final step is to solve

$$\bigl(5448^{5^3}\bigr)^{x_3} = 6909 \cdot 5448^{-x_0 - x_1 \cdot 5 - x_2 \cdot 5^2} = 6909 \cdot 5448^{-11}, \ \text{i.e.} \ 11089^{x_3} = 6320.$$

The solution is $x_3 = 4$, so the final answer is $x = 511 = 1 + 2 \cdot 5 + 4 \cdot 5^3$. As a check, we compute

$$5448^{511} = 6909 \quad \text{in } \mathbb{F}_{11251}. \quad \checkmark$$

The Pohlig-Hellman algorithm for solving the discrete logarithm problem combines the solution method for prime powers described in Proposition 2.21 with the Chinese Remainder Theorem 2.13.

Theorem 2.23 (Pohlig-Hellman algorithm)**.** *Let p be a prime and factor $p-1$ into a product of prime powers*

$$p - 1 = \prod_{i=1}^{k} q_i^{e_i}.$$

Then the discrete logarithm problem $g^x = a$ in $\mathbb{F}_p$ can be solved in approximately

$$\sum_{i=1}^{k} e_i \sqrt{q_i} \quad \textit{steps} \tag{2.16}$$

using the following algorithm:

(1) *For each* $1 \le i \le k$*, let*

$$h_i = g^{(p-1)/q_i^{e_i}} \qquad \textit{and} \qquad b_i = a^{(p-1)/q_i^{e_i}}$$

and use the algorithm from Proposition 2.21 to solve the discrete logarithm problem

$$h_i^y = b_i \quad \textit{in } \mathbb{F}_p. \tag{2.17}$$

Let $y = y_i$ *be a solution to* (2.17)*.*

(2) *Use the Chinese Remainder Theorem 2.13 to solve*

$$x \equiv y_1 \ (mod\, q_1^{e_1}), \quad x \equiv y_2 \ (mod\, q_2^{e_2}), \quad \dots \quad x \equiv y_k \ (mod\, q_k^{e_k}). \tag{2.18}$$

Proof. Proposition 2.21 says that it takes $e_i\sqrt{q_i}$ steps to solve (2.17), so summing over $i = 1, 2, \dots, k$ gives the running time estimate (2.16). The time for the Chinese Remainder Theorem is generally negligible by comparison, since it solves (2.18) in approximately

$$\log(p-1) = \sum e_i \log(q_i) \quad \text{steps.}$$

It remains to show that the algorithm actually gives a solution to $g^x = a$ in $\mathbb{F}_p$. Let x be a solution to the system of congruences (2.18). Then

$$\begin{aligned}
\big(g^x\big)^{(p-1)/q_i^{e_i}} &= \big(g^{y_i + q_i^{e_i} z_i}\big)^{(p-1)/q_i^{e_i}} && \text{since } x \equiv y_i \ (\text{mod } q_i^{e_i}),\\
&= \big(g^{(p-1)/q_i^{e_i}}\big)^{y_i} \cdot g^{(p-1)z_i} \\
&= \big(g^{(p-1)/q_i^{e_i}}\big)^{y_i} && \text{since } g^{p-1} = 1,\\
&= a^{(p-1)/q_i^{e_i}} && \text{from (2.17).}
\end{aligned}$$

In terms of discrete logarithms to the base g, we can rewrite this last formula as

$$\frac{p-1}{q_i^{e_i}} \cdot x \equiv \frac{p-1}{q_i^{e_i}} \cdot \log_g(a) \quad (\text{mod } p-1), \tag{2.19}$$

where remember that the discrete logarithm is only defined modulo $p-1$.

Next we observe that the numbers

$$\frac{p-1}{q_1^{e_1}}, \quad \frac{p-1}{q_2^{e_2}}, \quad \dots \quad \frac{p-1}{q_k^{e_k}}$$

have no nontrivial common factor, i.e. their greatest common divisor is 1. Repeated application of the Extended Euclidean Algorithm (Theorem 1.11, see also Exercise 1.12) says that we can find integers $c_1, c_2, \ldots, c_k$ so that

$$\frac{p-1}{q_1^{e_1}} \cdot c_1 + \frac{p-1}{q_2^{e_2}} \cdot c_2 + \cdots + \frac{p-1}{q_k^{e_k}} \cdot c_k = 1. \tag{2.20}$$

Now multiply both sides of (2.19) by c_i and sum (2.19) over $i = 1, 2, \ldots, k$. This gives

$$\sum_{i=1}^{k} \frac{p-1}{q_i^{e_i}} \cdot c_i \cdot x \equiv \sum_{i=1}^{k} \frac{p-1}{q_i^{e_i}} \cdot c_i \cdot \log_g(a) \pmod{p-1},$$

and then (2.20) tells us that

$$x = \log_g(a) \pmod{p-1}.$$

This completes the proof that x satisfies $g^x \equiv a$ in $\mathbb{F}_p$. □

Example 2.24. We illustrate the Pohlig-Hellman algorithm by solving the discrete logarithm problem

$$23^x = 9689 \quad \text{in } \mathbb{F}_{11251}. \tag{2.21}$$

The base 23 is a primitive root in $\mathbb{F}_{11251}$. Since $11250 = 2 \cdot 3^2 \cdot 5^4$ is a product of small primes, the Pohlig-Hellman algorithm should work well. In the notation of Theorem 2.23, we set

$$p = 11251, \qquad g = 23, \qquad a = 9689, \qquad p - 1 = 2 \cdot 3^2 \cdot 5^4.$$

The first step is solve three subsidiary discrete logarithm problems, as indicated in the following table.

q	e	$h = g^{(p-1)/q^e}$	$b = a^{(p-1)/q^e}$	y with $h^y = b$
2	1	11250	11250	1
3	2	5029	10724	4
5	4	5448	6909	511

Notice that the first problem is trivial, while the third one is the problem that we solved in Example 2.22. In any case, the individual problems in this step of the algorithm may be solved as described in the proof of Proposition 2.21.

The second step is to use the Chinese Remainder Theorem to solve the simultaneous congruences

$$x \equiv 1 \pmod{2}, \qquad x \equiv 4 \pmod{3^2}, \qquad x \equiv 511 \pmod{5^4}.$$

The smallest solution is $x = 4261$. We check our answer by computing

$$23^{4261} = 9689 \quad \text{in } \mathbb{F}_{11251}. \quad \checkmark$$

Exercises

Section 2.1. Diffie-Hellman and RSA [**H**]

2.1. Write a one page essay giving arguments, both pro and con, for the following assertion:

> If the government is able to convince a court that there is a valid reason for their request, then they should have access to any individual's private keys, in the same way that the government is allowed to conduct court authorized wiretaps in cases of suspected criminal activity or threats to national security.

Based on your arguments, would you support or oppose the government being given this power? The idea that all private keys should be stored at a secure central location and be accessible to government agencies (under suitably stringent legal conditions) is called *key escrow.*

2.2. Research and write a one to two page essay on the classification of cryptographic algorithms as munitions under ITAR (International Traffic in Arms Regulations). How does that act define "export"? What are the potential fines and jail terms for those convicted of violating the Arms Export Control Act? Would teaching non-classified cryptographic algorithms to a college class that includes non-US citizens be considered a form of export? How has US government's policy changed from the early-1990's to the present?

Section 2.2. The discrete logarithm problem

2.3. Let g be a primitive root for $\mathbb{F}_p$.

(a) Suppose that $x = a$ and $x = b$ are both integer solutions to the congruence $g^x \equiv h \pmod{p}$. Prove that $a \equiv b \pmod{p-1}$. Explain why this implies that the map (2.1) on page 60 is well-defined.

(b) Prove that $\log_g(ab) = \log_g(a) + \log_g(b)$ for all $a, b \in \mathbb{F}_p^*$.
(c) Prove that $\log_g(a^n) = n \log_g(a)$ for all $a \in \mathbb{F}_p^*$ and $n \in \mathbb{Z}$.

2.4. Compute the following discrete logarithms.
(a) $\log_2(13)$ for the prime 23, i.e. $p = 23$, $g = 2$, and you must solve the congruence $2^x \equiv 13 \pmod{23}$.
(b) $\log_{10}(22)$ for the prime 47.
(c) $\log_{627}(608)$ for the prime $p = 941$. (*Hint.* Look in the second column of Table 2.1 on page 61.)

2.5. Let p be an odd prime and let g be a primitive root modulo p. Prove that a has a square root modulo p if and only if its discrete logarithm $\log_g(a)$ modulo p is even.

2.6. *** prove $\mathrm{GL}_2(\mathbb{F}_p)$ is a group, show it is noncommutative, describe $\mathrm{GL}_2(\mathbb{F}_2)$ completely, show that $\mathrm{GL}_n(\mathbb{F}_p)$ is a group (challenge: how many elements?), maybe also ask about $\mathrm{SL}_2(\mathbb{R})$ and $\mathrm{SL}_2(\mathbb{F}_p)$?

Section 2.3. Diffie-Hellman key exchange

2.7. Alice and Bob agree to use the prime $p = 1373$ and the base $g = 2$ for a Diffie-Hellman key exchange. Alice sends Bob the value $A = 974$. Bob asks your assistance, so you tell him to use the secret exponent $b = 871$. What value B should Bob send to Alice, and what is their secret shared value? Can you figure out Alice's secret exponent?

2.8. Bob and Alice engage in the following communications. The prime $p = 32611$ is publicly known, but all of the other numbers that follow are private. Alice takes her secret number $m = 11111$, raises it to the power $a = 3589$ and sends the number $m^a \pmod{p} = 15950$ to Bob. Bob raises this number to the power $b = 4037$, reduces it modulo p, and sends the number 15422 back to Alice. Alice then computes $15422^{15619} \equiv 27257 \pmod{32611}$ and sends 27257 to Bob. Bob then computes $27257^3 1883 \equiv 11111 \pmod{32611}$. Thus Bob now knows Alice's secret number.
(a) Explain why this process works. In particular, Alice uses the numbers $a = 3589$ and 15619 as exponents. How are they related? And similarly how are Bob's exponents $b = 4037$ and 31883 related?
(b) Formulate a general version of this process, i.e. using variables, and show that it works in general. This method is called *Shamir's Secret Sharing Scheme*. It has the disadvantage that it requires Alice and Bob to exchange several messages.

(c) Does this seem to be a secure method for Alice and Bob to exchange information?

(d) If Eve can solve the discrete logarithm problem modulo p, can she break this system?

2.9. Let p be a prime and let g be an integer. The *Diffie-Hellman Decision Problem* is as follows. Supoose that you are given three numbers A, B, and C, and suppose that A and B are equal to

$$A \equiv g^a \pmod{p} \qquad \text{and} \qquad B \equiv g^b \pmod{p},$$

but that you do not necessarily know the values of the exponents a and b. Determine whether or not C is equal to $g^{ab} \pmod{p}$. Notice that this is different from the Diffie-Hellman problem described on page 65. The Diffie-Hellman problem asks you to actually compute the value of g^{ab}.

(a) Prove that an algorithm that solves the Diffie-Hellman problem can be used to solve the Diffie-Hellman decision problem.

(b) Do you think that the Diffie-Hellman decision problem is hard or easy? Why?

Section 2.4. The ElGamal public key cryptosystem

2.10. Alice and Bob agree to use the prime $p = 1373$ and the base $g = 2$ for communications using the ElGamal public key cryptosystem.

(a) Alice chooses $a = 947$ as her private key. What is the value of her public key A?

(b) Bob chooses $b = 716$ as his private key, so his public key is

$$B \equiv 2^{716} \equiv 469 \pmod{1373}.$$

Alice encrypts the message $m = 583$ using the ephemeral key $k = 877$. What is the ciphertext (c_1, c_2) that Alice sends to Bob?

(c) Alice decides to choose a new private key $a = 299$ with associated public key $A \equiv 2^{299} \equiv 34 \pmod{1373}$. Bob encrypts a message using Alice's public key and sends her the ciphertext $(c_1, c_2) = (661, 1325)$. Decrypt the message.

(d) Now Bob chooses a new private key and publishes the associated public key $B = 893$. Alice encrypts a message using this public key and sends the ciphertext $(c_1, c_2) = (693, 793)$ to Bob. Eve intercepts the transmission. Help Eve by solving the discrete logarithm problem $2^b \equiv 893 \pmod{1373}$ and using the value of b to decrypt the message.

2.11. Suppose that an oracle offers to solve the Diffie-Hellman problem for you. (See page 65 for a description of the Diffie-Hellman problem.) Explain how you can use the oracle to decrypt messages that have been encrypted using the ElGamal public key cryptosystem.

Section 2.5. The Chinese Remainder Theorem [**M**]

2.12. Solve (if possible) each of the following simultaneous systems of congruences.

(a) $x \equiv 3 \pmod{7}$ and $x \equiv 4 \pmod{9}$.

(b) $x \equiv 137 \pmod{423}$ and $x \equiv 87 \pmod{191}$.

(c) $x \equiv 133 \pmod{451}$ and $x \equiv 237 \pmod{697}$.

(d) $x \equiv 5 \pmod{9}$, $x \equiv 6 \pmod{10}$, and $x \equiv 7 \pmod{11}$.

(e) $x \equiv 37 \pmod{43}$, $x \equiv 22 \pmod{49}$, and $x \equiv 18 \pmod{71}$.

2.13. Solve the 1700 year old Chinese remainder problem from the *Sun Tzu Suan Ching* stated on page 71.

2.14. Let a, b, m, n be integers with $\gcd(m, n) = 1$. Let

$$c \equiv (b - a) \cdot m^{-1} \pmod{n}.$$

Prove that $x = a + cn$ is a solution to

$$x \equiv a \pmod{m} \qquad \text{and} \qquad x \equiv b \pmod{n}, \tag{2.22}$$

and that every solution to (2.22) has the form $x = a + cn + ymn$ for some $y \in \mathbb{Z}$.

2.15. Let $x = c$ and $x = c'$ be two solutions of the system of simultaneous congruences (2.6) in the Chinese Remainder Theorem 2.13. Prove that

$$c \equiv c' \pmod{m_1 m_2 \cdots m_k}.$$

2.16. If you have studied ring theory, this exercise sketches a short, albeit nonconstructive proof of the Chinese Remainder Theorem. Let $m_1, \ldots, m_k$ be integers and let $m = m_1 m_2 \cdots m_k$ be their product.

(a) Prove that the map

$$\begin{array}{ccc} \dfrac{\mathbb{Z}}{m\mathbb{Z}} & \longrightarrow & \dfrac{\mathbb{Z}}{m_1\mathbb{Z}} \times \dfrac{\mathbb{Z}}{m_2\mathbb{Z}} \times \dfrac{\mathbb{Z}}{m_k\mathbb{Z}} \\ a \bmod m & \longrightarrow & (a \bmod m_1, a \bmod m_2, \ldots, a \bmod m_k) \end{array} \tag{2.23}$$

is a well-defined homomorphism of rings. (*Hint.* First define a homomorphism from $\mathbb{Z}$ to the righthand side of (2.23), and then show that $m\mathbb{Z}$ is in the kernel.)

(b) Assume that $m_1, \ldots, m_k$ are pairwise relatively prime. Prove that the map (2.23) is one-to-one. (*Hint.* What is the kernel?)

(c) Continuing with the assumption that $m_1, \ldots, m_k$ are pairwise relatively prime, prove that the map (2.23) is onto. (*Hint.* Use (b) and count the size of both sides.)

(d) Explain why (b) and (c) are another way of stating the Chinese Remainder Theorem 2.13.

2.17. Use the method described in Section 2.5.1 to find square roots modulo composite moduli.

(a) Find a square root of 340 modulo 437. (Note $437 = 19 \cdot 23$.)

(b) Find a square root of 253 modulo 3143.

(c) Find four square roots of 2833 modulo 4189. (The modulus factors as $4189 = 59 \cdot 71$. Note that your four square roots should be distinct modulo 4189.)

(d) Find eight square roots of 813 modulo 868.

2.18. Let p be an odd prime and let b be a square root of a modulo p. This exercise investigates the square root of a modulo powers of p.

(a) Prove that for some choice of k, the number $b + kp$ is a square root of a modulo p^2, i.e. $(b + kp)^2 \equiv a \pmod{p^2}$.

(b) The number $b = 537$ is a square root of $a = 476$ modulo the the prime $p = 1291$. Use the idea in (a) to compute the square root of 476 modulo p^2.

(c) Suppose that c is a square root of a modulo p^n. Prove that for some choice of j, the number $b + jp$ is a square root of a modulo p^{n+1}.

(d) Explain why (c) implies the following statement: If p is an odd prime and if a has a square root modulo p, then a has a square root modulo p^n for every power of p. Is this true if $p = 2$?

(e) Use the method in (c) to compute the square root of 3 modulo 13^3, given that $9^2 \equiv 3 \pmod{13}$.

Section 2.6. The Pohlig-Hellman algorithm

2.19. Use the Pohlig-Hellman Algorithm (Theorem 2.23) to solve the discrete logarithm problem

$$g^x = a \quad \text{in } \mathbb{F}_p$$

in each of the following cases.

(a) $p = 433, \quad g = 7, \quad a = 166.$

(b) $p = 746497, \quad g = 10, \quad a = 243278.$

(c) $p = 41022299, \quad g = 2, \quad a = 39183497.$ (*Hint.* $p = 2 \cdot 29^5 + 1$.)

(d) $p = 1291799, \quad g = 17, \quad a = 192988.$ (*Hint.* $p - 1$ has a factor of 709.)

Chapter 3

Probability Theory and Information Theory

3.1 Basic principles of counting [M]

As I was going to St. Ives,
I met a man with seven wives,
Each wife had seven sacks,
Each sack had seven cats.
Each cat had seven kits.
Kits, cats, sacks, and wives,
How many were there going to St. Ives?

The answer to this ancient riddle is that there is only one person going to St. Ives, namely the narrator, since all of the other people and animals and objects that he meets in the rhyme are not traveling *to* St. Ives, they are traveling *away* from St. Ives! However, if we are in a pedantic, rather than a clever, frame of mind, we might instead ask the natural question: How many people, animals, and objects does the narrator meet?

The answer is

$$2801 = \underbrace{1}_{\text{man}} + \underbrace{7}_{\text{wives}} + \underbrace{7^2}_{\text{sacks}} + \underbrace{7^3}_{\text{cats}} + \underbrace{7^4}_{\text{kits}}.$$

The computation of this number employs basic counting principles that are fundamental for the probability calcuations important in cryptology and in many other areas of mathematics. We have already seen an example in

Section 1.1.1, where we computed the number of different simple substitution ciphers.

A cipher is said to be *combinatorially secure* if it is not practically feasible to break the system by exhaustively checking every possible key.[1] This depends to some extent on how long it takes to check each key, but more importantly, it depends on the number of keys. In this section we develop some basic counting techniques that are used in a variety of ways to analyze the security of cryptographic constructions.

Example 3.1 (A Basic Counting Principle). Two students (Alice and Bob) need to sign up for a course, and they may each choose from among any of 20 different classes. How many possibilities are there?

We need to count the number of pairs (x, y), where x is either "Alice" or "Bob" and y is a class. The total number is obtained by letting x vary over the 2 possibilities and letting y vary over the 20 possibilities and counting up the total number of pairs

$$(A, 1), (A, 2), \ldots, (A, 20), (B, 1), (B, 2), \ldots, (B, 20).$$

The answer is that there are 40 possibilities, which we compute as

$$40 = \underbrace{2}_{\text{people}} \cdot \underbrace{20}_{\text{classes}} .$$

Suppose next that Alice an Bob are joined by a third student, Carl, that each of them needs to sign up for a lunch hour (lunch is offered at either 11:00am or 1:00pm) and needs to choose among the five cafeterias on campus. How many ways can this be done. The calculation is similar, we are counting triples (u, v, w), where u is one of the three people, v is one of the two times, and w is one of the five cafeterias. Hence the total number of configurations is

$$30 = \underbrace{3}_{\text{people}} \cdot \underbrace{2}_{\text{times}} \cdot \underbrace{5}_{\text{cafeterias}} .$$

In this example, we counted the number of ways of assigning a student (Alice or Bob) to the variable x. It is convenient to view this assignment as the outcome of an experiment. That is, we perform an experiment, the outcome is one of Alice or Bob and we assign that outcome to x. Similarly,

[1] Sometimes the length of the search can be signficantly shortened by matching pieces of keys taken from two or more lists, see Section 3.4.

we perform a second independent experiment whose possible outcomes are any one of the twenty classes and we assign that outcome to y. The total number of outcomes of the two experiments is the product of the number of outcomes for each one individually. This leads to:

Basic Counting Principle
If two experiments are performed, one of which has n outcomes and the other of which has m outcomes, then there are nm possible outcomes of performing both experiments.

More generally, if k independent experiments are performed and if the number of possible outcomes of the i^{th} experiment is n_i, then the total number of outcomes for all of the experiments is the product $n_1 n_2 \cdots n_k$. It is easy to derive this result by writing x_i for the outcome of the i^{th} experiment. Then the outcome of all k experiments is the value of the k-tuple $(x_1, x_2, \ldots, x_n)$, and the total number of possible k-tuples is the product $n_1 n_2 \cdots n_k$.

The basic counting principle can be used to solve the pedantic version of the St. Ives problem. For example, the number of cats traveling from St. Ives is

$$\#\text{ of cats} = 343 = 7^3 = \underbrace{1}_{\text{man}} \cdot \underbrace{7}_{\text{wives}} \cdot \underbrace{7}_{\text{sacks}} \cdot \underbrace{7}_{\text{cats}}.$$

The earliest published version of the St. Ives riddle dates to around 1730, but similar problems date back to antiquity (Exercise 3.1).

3.1.1 Permutations

The numbers $1, 2, \ldots, 10$ are typically listed in increasing order, but suppose instead we allow the order to be mixed. Then how many different ways are there to list these ten integers? Each possible configuration is called a permutation of $1, 2, \ldots, 10$. The problem of counting the number of possible permutations of a given list of objects occurs in many forms and contexts within mathematics.

Each permutation of $1, 2, \ldots, 10$ is a sequence of all ten distinct integers in some order. For example, here is a random choice: $8, 6, 10, 3, 9, 2, 4, 7, 5, 1$. How can we create all of the possibilities? It helps to create them one place at a time. We start by assigning a number to the first position. There are ten choices. Next we assign a number to the second position, but for the second position there are only nine choices, since we already used up one of the integers in the first position. (Remember that we are not allowed to use an integer twice.) Then there are eight integers left as possibilities for the third position. And so on. Hence the total number of ways to arrange $1, 2, \ldots, 10$ is $10 \cdot 9 \cdot 8 \cdots 2 \cdot 1 = 10!$, which is larger than three million.

Notice how we are using the basic counting principle. The only subtlety is noticing that the outcome of the first experiment reduces the number of possible outcomes of the second experiment, the results of the first two experiments further reduces the numer of possible outcomes of the third experiment, and so on.

Definition. Let S be a set containing n distinct objects. A *permuation of S* is an ordered list of the objects in S. A permutation of the set $\{1, 2, \ldots, n\}$ is simply called a permuation of n.

Proposition 3.2. *Let S be a set containing n distinct objects. Then there are exactly $n!$ different permuations of S.*

Remark 3.3 (Permuations and Simple Substitution Ciphers). By definition, a permutation of a set $\{a_1, a_2, \ldots, a_n\}$ is a list consisting of the a_i's in some order. Another way to describe a permutation is with a bijective (i.e. one-to-one and onto) function

$$\pi : \{1, 2, \ldots, n\} \longrightarrow \{1, 2, \ldots, n\}.$$

The function π determines the permuation

$$(a_{\pi(1)}, a_{\pi(2)}, \ldots, a_{\pi(n)}),$$

and given a permutation, it is easy to write down the corresponding function.

Now suppose that we take the set of letters $\{\mathtt{A}, \mathtt{B}, \mathtt{C}, \ldots, \mathtt{Z}\}$. A permutation π of this set is just another name for a simple substitution cipher, since π tells us that A gets sent to the $\pi(1)^{\text{st}}$ letter, and B gets sent to the $\pi(2)^{\text{nd}}$, and so on. In order to decrypt, we use the inverse function π^{-1}.

Example 3.4. Sometimes one needs to count the number of possible permuations of n objects when some of the objects are indistinguishable. For example, there are 6 permuations of three distinct objects A, B, C,

$$ABC, \quad CAB, \quad BCA, \quad ACB, \quad BAC, \quad \text{and} \quad CBA,$$

but if two of them are indistinguishable, say A, A, B, then there are only 3 different arrangements,

$$AAB, \quad ABA, \quad \text{and} \quad BAA.$$

We do an example to illustrate how to count the number of permuations in this type of situation without making a list of all of the possibilities. We count the number of different letter arrangement of the five letters A, A, A, B, B. If the five letters were distinguishable, say they were labeled as A_1, A_2, A_3, B_1, B_2, then there would be 5! permutations. However, permuations such as

$$A_1A_2B_1B_2A_3 \qquad \text{and} A_2A_3B_2B_1A_1$$

are supposed to be the same, but we have counted them as distinct in arriving at the number 5!. How many different arrangements have been multiply counted? Once we know where the three A's have been placed, they can be rearranged in 3! different ways to give identical configurations, and similarly the two B's may be switched. Thus we need to divide 5! by the 3! ways of permuting the three A's and by the 2! ways of permuting the two B's. Hence there are $\frac{5!}{3!\cdot 2!} = 10$ different letter arrangement of the five letters A, A, A, B, B.

3.1.2 Combinations

A permutation is a way of arranging a set of objects into a list. A combination is similar, except that no the order no longer matters. We start with an example that is typical of problems involving combinations.

Example 3.5. Five people (Alice, Bob, Carl, Dave, and Eve[2]) are ordering a meal at a Chinese restaurant. The menu contains twenty different items. Each person gets to choose one dish, no dish may be ordered twice, and they plan to share the food. How many different meals are possible?

Aice orders first and she has 20 choices for her dish. Then Bob orders from the remaining 19 dishes, and then Carl chooses from the remaining 18 dishes, and so on. It thus appears that there are $20 \cdot 19 \cdot 18 \cdot 17 \cdot 16 = 1860480$ possible meals. However, the order in which the dishes are ordered is immaterial.

[2] You may wonder why Alice and Bob, those intrepid exchangers of encrypted secret messages, are sitting down for a meal with their cryptographic adversary Eve. In the real world, this happens all the time, especially at cryptography conferences!

If Alice orders fried rice and Bob orders egg rolls, or if Alice orders egg rolls and Bob orders fried rice, the meal is the same. Unfortunately, we did not take this into account when we arrived at the number 1860480.

Let's number the dishes $D_1, D_2, \ldots, D_{20}$. Then for example, we have counted the two possible dinner orders

$$D_1, D_5, D_7, D_{18}, D_{20} \qquad \text{and} \qquad D_5, D_{18}, D_{20}, D_7, D_1$$

as two different orders, although they are really the same meal. In order to correct this overcount, note that every permutation of the same five dishes was counted separately, but we really want to count them as one meal. Thus we should divide by the number of ways to permute the five distinct dishes in the order, i.e. we should divide by 5!. Hence the total number of different meals is

$$\frac{20 \cdot 19 \cdot 18 \cdot 17 \cdot 16}{5!} = 15504.$$

It is often convenient to rewrite this quantity entirely in terms of factorials by multiplying the numerator and the denominator by 15! to get

$$\frac{20 \cdot 19 \cdot 18 \cdot 17 \cdot 16}{5!} = \frac{(20 \cdot 19 \cdot 18 \cdot 17 \cdot 16) \cdot (15 \cdot 14 \cdots 3 \cdot 2 \cdot 1)}{5! \cdot 15!} = \frac{20!}{5! \cdot 15!}.$$

Definition. Let S be a set containing n distinct objects. A *combination* of r objects of S is a subset consisting of exactly r distinct elements of S, where the order of the objects in the subset does not matter.

Proposition 3.6. *The number of possible combinations of r objects chosen from a set of n objects is equal to*

$$\binom{n}{r} = \frac{n!}{(n-r)!r!}.$$

Remark 3.7. The symbol $\binom{n}{r}$ is called a *combinatorial symbol* or a *binomial coefficient*. It is read as "n choose r". Note that by convention, zero factorial (0!) is set equal to 1, so $\binom{n}{0} = \frac{n!}{n! \cdot 0!} = 1$. This makes sense, since there is one way to choose zero objects from a set.

Example 3.8. Returning to the five people ordering a meal at the Chinese restaurant, suppose that they want the order to consist of two vegatarian dishes and three meat dishes, and suppose that the menu contains 5 vegatarian choices and 15 meat choices. Now how many possible meals can they order? There are $\binom{5}{2}$ possibilities for the two vegetarian dishes and there are $\binom{15}{3}$ choices for the three meat dishes. Hence by our basic counting principle, there are

$$\binom{5}{2} \cdot \binom{15}{3} = 10 \cdot 455 = 4550$$

possible meals.

3.1.3 The binomial theorem

You may have seen the combinatorial numbers $\binom{n}{r}$ appearing in the Binomial Theorem,[3] which gives a closed formula for the n^{th} power of the sum of two numbers.

Theorem 3.9 (The Binomial Theorem)**.**

$$(x+y)^n = \sum_{j=0}^{n} \binom{n}{j} x^j y^{n-j} \tag{3.1}$$

Proof. Let's start with a particular case, say $n = 3$. If we multiply out the product

$$(x+y)^3 = (x+y) \cdot (x+y) \cdot (x+y), \tag{3.2}$$

the result is a sum terms x^3, x^2y, xy^2, and y^3. There is only one x^3 term, since to get x^3 we must take x from each of the three factors in (3.2). How many copies of x^2y are there? We can get x^2y in several ways. For example, we could take x from the first two factors and y from the last factor. Or we could take x from the first and third factors and take y from the second factor. Thus we get x^2y by choosing two of the three factors in (3.2) to give x (note that the order doesn't matter) and then the remaining factor gives y.

[3] The Binomial Theorem's fame extends beyond mathematics. Moriarty, Sherlock Holmes' arch enemy, "wrote a treatise upon the Binomial Theorem," on the strength of which he won a mathematical professorship. And intrepid Major General Stanley, that very "Model of a Modern Major General," proudly informs the Pirate King and his cutthroat band:

About binomial theorem I'm teeming with a lot o' news—
With many cheerful facts about the square of the hypotenuse.
(*The Pirates of Penzance*, W.S. Gilbert and A. Sullivan, 1879)

There are thus $\binom{3}{2} = 3$ ways to get x^2y. Similarly, there are $\binom{3}{1} = 3$ ways to get xy^2 and only one way to get y^3. Hence

$$(x+y)^3 = \binom{3}{3}x^3 + \binom{3}{2}x^2y + \binom{3}{1}xy^2 + \binom{3}{0}y^3 = x^3 + 3x^2y + 3xy^2 + y^3.$$

The general case is exactly the same. The product

$$(x+y)^n = (x+y)\cdot(x+y)\cdot(x+y)\cdots(x+y), \tag{3.3}$$

when multiplied out, is a sum of terms $x^n, x^{n-1}y, \ldots, xy^{n-1}, y^n$. We get copies of x^jy^{n-j} by choosing x from any j of the factors in (3.3) and then taking y from the other $n-j$ factors. Thus we get $\binom{n}{j}$ copies of x^jy^{n-j}. Summing over the possible values of j gives (3.1), which completes the proof of the Binomial Theorem. □

Example 3.10. We use the binomial theorem to compute

$$\begin{aligned}(2t+3)^4 &= \binom{4}{4}(2t)^4 + \binom{4}{3}(2t)^3\cdot 3 + \binom{4}{2}(2t)^2\cdot 3^2 + \binom{4}{1}2t\cdot 3^3 + \binom{4}{0}3^4\\ &= 16t^4 + 4\cdot 8t^3\cdot 3 + 6\cdot 4t^2\cdot 9 + 4\cdot 2t\cdot 27 + 81\\ &= 16t^4 + 96t^3 + 216t^2 + 216t + 81.\end{aligned}$$

3.2 The Vigenère cipher

The simple substitution ciphers that we studied in Section 1.1 are examples of *monoalphabetic ciphers*, since every plaintext letter is encrypted using a single cipher alphabet. As cryptanalytic methods became more sophisticated in Renaissance Italy, correspondingly more sophisticated ciphers were invented. Consider how much more difficult a task is faced by the cryptanalyst if every plaintext letter is encrypted using a different ciphertext alphabet. This ideal resurfaces in modern cryptology in the form of the one-time pad, which we discuss in Section 3.6, but in this section we discuss a less complicated *polyalphabetic cipher* called the Vigenère cipher[4] dating back to the 16th century.

[4] This cipher is named after Blaise de Vigenère (1523–1596), whose 1586 book *Traicté des Chiffres* describes the known ciphers of his time. These include polyalphabetic ciphers such as the "Vigenére cipher," which according to [24] Vigenère did not invent, and an ingenious autokey system (see Exercise 3.17), which he did.

The Vigenère cipher works by using different shift ciphers to encrypt different letters. In order to decide how far to shift each letter, Bob and Alice first agree on a keyword or phrase. Bob then uses the letters of the keyword, one by one, to determine how far to shift each successive plaintext letter. If the keyword letter is `a`, there is no shift, if the keyword letter is `b`, he shifts by 1, if the keyword letter is `c`, he shifts by 2, and so on.

A simple example illustrates the process. Suppose that the keyword is `dog` and the plaintext is `yellow`. The first letter of the keyword is `d`, which gives a shift of 3, so Bob shifts the first plaintext letter `y` down by 3, which gives the ciphertext letter `b`. (Remember that `a` follows `z`.) The second letter of the keyword is `o`, which gives a shift of 14, so Bob shifts the second plaintext letter `e` down by 14, which gives the ciphertext letter `s`. The third letter of the keyword is `g`, which gives a shift of 6, so Bob shifts the third plaintext letter `l` down by 6, which gives the ciphertext letter `r`.

Bob has run out of keyword letters, so what does he do now? He simply starts again with the first letter of the keyword. The first letter of the keyword is `d`, which again gives a shift of 3, so Bob shifts the fourth plaintext letter `l` down by 3, which gives the ciphertext letter `o`. Then the second keyword letter `o` tells him to shift the fifth plaintext letter `o` down by 14, giving the ciphertext letter `c`, and finally the third keyword letter `g` tells him to shift the sixth plaintext letter `w` down by 6, giving the ciphertext letter `c`.

In conclusion, Bob has encrypted the plaintext `yellow` using the keyword `dog` and obtained the ciphertext `bsrocc`. Even this simple example illustrates two important characteristics of the Vigenère cipher. First, the repeated letters `ll` in the plaintext lead to non-identical letters `ro` in the ciphertext, and second, the repeated letters `cc` in the ciphertext correspond to different letters `ow` of the plaintext. Thus a straigtforward frequency analysis as we used to cryptanalyze simple substitution ciphers (Section 1.1.1) is not going to work for the Vigenère.

A useful tool for doing Vigenère encryption and decryption, at least if no computer is available (as was typically the case in the 16th century!), is the so-called *Vigenère tableaux* illustrated in Figure 3.1. The Vigenère tableaux consists of 26 alphabets arranged in a square, with each alphabet shifted one further than the alphabet to its left. In order to use a given keyword letter to encrypt a given plaintext letter, Bob finds the plaintext letter in the top row and the keyword letter in the first column. He then looks for the letter in the tableaux lying below the plaintext letter and to the right of the keyword letter.

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a
c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b
d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c
e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d
f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f
h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g
i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h
j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i
k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j
l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k
m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l
n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m
o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n
p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r
t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s
u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t
v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u
w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x
z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y

- Find the plaintext letter in the top row.
- Find the keyword letter in the first column.
- The ciphertext letter lies below the plaintext letter and to the right of the keyword letter.

Table 3.1: The Vigenère Tableaux

For example, if the keyword letter is `d` and the plaintext letter is `y`, Bob looks in the fourth row (which is the one that starts with `d`) and in the next to last column (which is the one headed by `y`). This row and column intersect at the letter `b`, so the corresponding ciphertext letter is `b`.

Decryption is just as easy. Alice uses the row containing the keyword letter and looks in that row for the ciphertext letter. Then the top of that column is the plaintext letter. For example, if the keyword letter is `g` and the ciphertext letter is `r`, Alice looks in the row starting with `g` until she finds `r` and then she moves to the top of that column to find the plaintext letter `l`.

Example 3.11. We illustrate the use of the Vigenère tableaux by encrypting the plaintext message

`The rain in Spain stays mainly in the plain.`

using the keyword `flamingo`. Since the key word has 8 letters, the first step is to split the plaintext into 8 letter units,

`theraini | nspainst | aysmainl | yinthepl | ain`

Next we write the keyword beneath each block of plaintext, where for convenience we label lines as $\mathcal{P}$, $\mathcal{K}$, and $\mathcal{C}$ to indicate, respectively, the plaintext, the keyword, and the ciphertext.

$\mathcal{P}$	t h e r a i n i	n s p a i n s t	a y s m a i n l	y i n t h e p l	a i n
$\mathcal{K}$	f l a m i n g o	f l a m i n g o	f l a m i n g o	f l a m i n g o	f l a

Finally, we encrypt each letter using the Vigenère tableaux. The initial plaintext letter `t` and initial keyword letter `f` combine in the Vigenère tableaux to yield the ciphertext letter `y`, the second plaintext letter `h` and second keyword letter `l` combine in the Vigenère tableaux to yield the ciphertext letter `s`, and so on. Continuing in this fashion, we complete the encryption process.

$\mathcal{P}$	t h e r a i n i	n s p a i n s t	a y s m a i n l	y i n t h e p l	a i n
$\mathcal{K}$	f l a m i n g o	f l a m i n g o	f l a m i n g o	f l a m i n g o	f l a
$\mathcal{C}$	y s e d i v t w	s d p m q a y h	f j s y i v t z	d t n f p r v z	f t n

Splitting the ciphertext into convenient blocks of five letters each, we are ready to transmit our encrypted message

`ysedi vtwsd pmqay hfjsy ivtzd tnfpr vzftn`

Remark 3.12. As we already pointed out, the same plaintext letter in a Vigenère cipher is represented in the ciphertext by many different letters. However, if the keyword is short, there will be a tendency for repetitive parts of the plaintext to end up aligned at the same point in the keyword, in which case they will be identically enciphered. This occurs in Example 3.11, where the `ain` in `rain` and in `mainly` are encrypted using the same three keyword letters `ing`, so they yield the same ciphertext letters `ivt`. This repetition in the ciphertext, which appears separated by 16 letters, suggests that the keyword has length dividing 16. Of course, not every occurence of `ain` in the plaintext yields the same ciphertext. It is only when two occurences line up with the same part of the keyword that repetition occurs.

In the next section we develop the idea of using ciphertext repetitions to guess the length of the keyword, but here we simply want to make the point that short keywords are less secure than long keywords.[5] On the other hand, Bob and Alice find it easier to remember a short keyword than a long one. We thus see the beginnings of the eternal struggle in practical (as opposed to purely theoretical) cryptogaphy, namely the battle between

$$\textbf{Efficiency} \longleftarrow \text{versus} \longrightarrow \textbf{Security}$$

As a further illustration of this dichotomy, we consider ways in which Bob and Alice might make their Vigenère-type cipher more secure. They can certainly make Eve's job harder by mixing up the letters in the first row of their Vigenère tableaux and then rotating this "mixed alphabet" in the subsequent rows. Unfortunately, a mixed alphabet makes encryption and decryption more cumbersome, plus it means that Bob and Alice must remember (or write down for safekeeping!) not only their keyword, but also the mixed alphabet. And if they want to be even more secure, they can use different randomly mixed alphabets in every row of their Vigenère tableaux. But if they do that, then they will certainly need to keep a written copy of the tableaux, which is a terrible security risk.

3.2.1 Cryptanalysis of the Vigenère cipher — theory

At various times in history it has been claimed that Vigenère-type ciphers, especially with mixed alphabets, are "unbreakable." In fact, nothing could be

[5] More typically one uses a key phrase consisting of several words, but for simplicity we use the term "keyword" to cover both single keywords and longer key phrases.

further from the truth. If Eve knows Bob and Alice, she may be able to guess part of the keyword and proceed from there. (How many people do you know who use some variation of their name and birthday as an internet password!) But even without lucky guesses, elementary statistical methods developed in the 19^{th} century allow for a straightforward cryptanalysis of Vigenère-type ciphers. In the interests of simplicity, we stick with the original Vigenère, i.e. we do not allow mixed alphabets in the tableaux.

You may wonder why we take the time to cryptanalyze the Vigenère cipher, since no one these days uses the Vigenère for secure communications. The answer is that our exposition is principally designed to introduce you to the use of statistical tools in cryptanalysis. This builds on and extends the elementary application of frequency tables as we used them in Section 1.1.1 to cryptanalyze simple substitution ciphers. In this section we describe the theoretical tools used to cryptanalyze the Vigenère, and in the next section we apply those tools to decrypt a sample ciphertext. If at any point the theory in this section is confusing, it may help to turn to Section 3.2.2 and see how the theory is applied in practice.

The first goal in cryptanalyzing a Vigenère cipher is to find the length of the keyword, which is sometimes called the *blocksize* or the *period*. We already saw in Remark 3.12 how this might be accomplished by looking for repeated fragments in the ciphertext. The point is that certain plaintext fragments such at `the` occur quite frequently, while other plaintext fragments such as `ugw` occur infrequently or not at all. Among the many occurences of the letters `the` in the plaintext, a certain percentage of them will line up with exactly the same part of the keyword.

This leads to the *Kasiski method*, first described by a German military officer named Friedrich Kasiski in his book *Die Geheimschriften und die Dechiffrir-kunst*[6] published in 1863. One looks for repeated fragments within the ciphertext and compiles a list of the distances that separate the repetitions. The blocksize is likely to divide many of these distances. Of course, a certain number of repetitions will occur by pure chance, but these are random, while the ones coming from repeated plaintext fragments are always divisible by the blocksize. It is generally not hard to pick out the blocksize from this data.

There is another method of guessing the blocksize that works with individual letters, rather than with fragments consisting of several letters. The

[6] *Cryptography and the Art of Decryption*

underlying idea goes all the way back to the frequency table of English letters (Table 1.3), which shows that some letters are more likely to occur than others. Suppose now that you are presented with a ciphertext encrypted using a Vigenère cipher and that you guess that it was encrypted using a keyword of length 5. This means that every fifth letter was encrypted using the same rotation, so if you pull out every fifth letter and form them into a string, this entire string was encrypted using a single substitution cipher. Hence the strings letter frequencies should look more-or-less like they do in English, with some letters must more frequent and some much less frequent. And the same will be true of the string consisting of the 2nd, 7th, 12th,. . . letters of the ciphertext, and so on. On the other hand, if you guessed wrong and the blocksize is not five, then the string consisting of every fifth letter should be more-or-less random, so its letter frequencies should look different than the frequencies in English.

How can we quantify the following two statements so as to be able to distinguish between them?

$$\text{“String 1 has letter frequencies similar to those in Table 1.3.”} \tag{3.4}$$

$$\text{“String 2 has letter frequencies that look more-or-less random.”} \tag{3.5}$$

One method is to use the following device.

Definition. Let $\mathbf{s} = c_1c_2c_3\cdots c_n$ be a string of n alphabetic characters. The *Index of Coincidence* of $\mathbf{s}$, denoted $\operatorname{IndCo}(\mathbf{s})$, is the probability that two randomly chosen characters in the string $\mathbf{s}$ are identical.

We are going to derive a formula for the index of coincidence. It is convenient to identify the letters `a`,. . . ,`z` with the numbers $0, 1, \ldots, 25$ as in Table 1.7. For each value $i = 0, 1, 2, \ldots, 25$, let F_i be the frequency with which letter i appears in the string $\mathbf{s}$. For example, if the letter `h` appears 23 times in the string $\mathbf{s}$, then $F_7 = 23$, since `h` $= 7$ in our labeling of the alphabet.

For each i, there are $\binom{F_i}{2} = \frac{F_i(F_i-1)}{2}$ ways to select two instances of the i^{th} letter of the alphabet from $\mathbf{s}$, so the total number of ways to get a repeated letter is the sum of $\frac{F_i(F_i-1)}{2}$ for $i = 0, 1, \ldots, 25$. On the other hand, there are a total of $\binom{n}{2} = \frac{n(n-1)}{2}$ ways to select two arbitrary characters from $\mathbf{s}$. Hence the probability of selecting two identical letters is

$$\operatorname{IndCo}(\mathbf{s}) = \frac{1}{n(n-1)} \sum_{i=0}^{25} F_i(F_i - 1). \tag{3.6}$$

Example 3.13. Let $\mathbf{s}$ be the string

$$\mathbf{s} = \text{“A bird in hand is worth two in the bush.”}$$

Ignoring the spaces between words, $\mathbf{s}$ consists of 30 characters. The following table counts the frequencies of each letter that appears at least once:

	A	B	D	E	H	I	N	O	R	S	T	U	W
i	0	1	3	4	7	8	13	14	17	18	19	20	22
F_i	2	2	2	1	4	4	3	2	2	2	3	1	2

Then the index of coincidence of $\mathbf{s}$, as given by (3.6), is

$$\mathrm{IndCo}(\mathbf{s}) = \frac{1}{30 \cdot 29}(2{\cdot}1+2{\cdot}1+2{\cdot}1+4{\cdot}3+4{\cdot}3+3{\cdot}2+\cdots+3{\cdot}2+2{\cdot}1) \approx 0.0575.$$

We return to our two statements (3.4) and (3.5). Suppose first that the string $\mathbf{s}$ consists of random characters. Then the probability that $c_i = c_j$ is exactly $\frac{1}{26}$, so we would expect $\mathrm{IndCo}(\mathbf{s}) \approx \frac{1}{26} \approx 0.0385$. On the other hand, if $\mathbf{s}$ consists of English text, then we would expect the relative frequencies to be as in Table 1.3. So for example, if $\mathbf{s}$ consists of 10,000 characters, we would expect approximately 815 A's, approximately 144 B's, approximately 276 C's, and so on. Thus the index of coincedence for a string of English text should be approximately

$$\frac{815 \cdot 814 + 144 \cdot 143 + 276 \cdot 275 + \cdots + 8 \cdot 7}{10000 \cdot 9999} \approx 0.0685.$$

The disparity between 0.0385 and 0.0685, as small as it may seem, provides the means to distinguish between Statement 3.4 and Statement 3.5. More precisely:

If $\mathrm{IndCo}(\mathbf{s}) \approx 0.68$, then $\mathbf{s}$ looks like simple substitution English. (3.7)

If $\mathrm{IndCo}(\mathbf{s}) \approx 0.38$, then $\mathbf{s}$ looks like random letters. (3.8)

Of course, the value of $\mathrm{IndCo}(\mathbf{s})$ will tend to fluctuate, especially if $\mathbf{s}$ is fairly short. But the moral of (3.7) and (3.8) is that larger values of $\mathrm{IndCo}(\mathbf{s})$ make it more likely that $\mathbf{s}$ is English encrypted with some sort of simple subsitution, while smaller values of $\mathrm{IndCo}(\mathbf{s})$ make it more likely that $\mathbf{s}$ is random.

Now suppose that Eve intercepts a message $\mathbf{s}$ that she believes was encrypted using a Vigenère cipher and wants to check whether the keyword has length k. Her first step is to break the string $\mathbf{s}$ into k pieces $\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_k$, where $\mathbf{s}_1$ consists of every k^{th} letter starting from the first letter, $\mathbf{s}_2$ consists of every k^{th} letter starting from the second letter, and so on. In mathematical terms, if we write $\mathbf{s} = c_1c_2c_3 \cdots c_n$, then

$$\mathbf{s}_i = c_i c_{i+k} c_{i+2k} c_{i+3k} \cdots .$$

Notice that if Eve's guess is correct and the keyword has length k, then each $\mathbf{s}_i$ consists of characters that were encrypted using the same shift amount, so although they do not decrypt to form actual words (remember that $\mathbf{s}_i$ is every k^{th} letter of the text), the pattern of their letter frequencies will look like English. On the other hand, if Eve's gues is incorrect, then the $\mathbf{s}_i$ strings will be more-or-less random.

Thus Eve merely needs to compute $\operatorname{IndCo}(\mathbf{s}_i)$ for $i = 1, 2, \ldots, k$ and see whether these numbers are closer to 0.068 or closer to 0.038. If the former, she accepts that the keysize is k, if the latter, she tries a different value for k. In practice, Eve makes a table of the k different values of $\operatorname{IndCo}(\mathbf{s}_i)$ for each $k = 3, 4, 5, \ldots$. The value of k that gives the largest average value is quite likely to be the blocksize.

So let's assume now that Eve has used the Kasiski test or the index of coincidence test to determine that the keyword has length k That's a good start, but she's still quite distant from the plaintext. The next step is to compare the strings $\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_k$ to one another. The tool she uses to compare different strings is called the mutual index of coincidence.

Definition. Let

$$\mathbf{s} = c_1c_2c_3 \cdots c_n \qquad \text{and} \qquad \mathbf{t} = d_1d_2d_3 \cdots d_m$$

be strings of alphabetic characters. The *Mutual Index of Coincidence* of $\mathbf{s}$ and $\mathbf{t}$, denoted $\operatorname{MutIndCo}(\mathbf{s}, \mathbf{t})$, is the probability that a randomly chosen character from $\mathbf{s}$ and a randomly chosen character from $\mathbf{t}$ will be the same.

If we let $F_i(\mathbf{s})$ denote the number of times the i^{th} letter appears in the string $\mathbf{s}$, and similarly for $F_i(\mathbf{t})$, then the probability of choosing the i^{th} letter from both is the product of the probabilities $\frac{F_i\mathbf{s}}{n}$ and $\frac{F_i\mathbf{t}}{m}$. Then we get a formula for the the mutual index of coincidence of $\mathbf{s}$ and $\mathbf{t}$ by adding these probabilities over all possible letters,

$$\text{MutIndCo}(\mathbf{s}, \mathbf{t}) = \frac{1}{nm} \sum_{i=1}^{26} F_i(\mathbf{s}) F_i(\mathbf{t}). \tag{3.9}$$

Example 3.14. Let $\mathbf{s}$ be the string (3.13) in Example 3.13 and let $\mathbf{t}$ be the string

$$\mathbf{t} = \text{“A stitch in time saves nine.”}$$

Then using formula (3.9) to compute the mutual index of coincidence of $\mathbf{s}$ and $\mathbf{t}$ yields $\text{MutIndCo}(\mathbf{s}, \mathbf{t}) = 0.0773$.

The characterization of the index of coincidence embodied in the two statements (3.7) and (3.8) holds equally well for the mutual index of coincidence. If the two strings $\mathbf{s}$ and $\mathbf{t}$ are English text encrypted using the same simple substitution table, then $\text{MutIndCo}(\mathbf{s}, \mathbf{t})$ tends to be large, because of the uneven frequency with which letters appear. On the other hand, if $\mathbf{s}$ and $\mathbf{t}$ are random strings (which is how they to appear to one another if they have been encrypted using different substitution tables), then $\text{MutIndCo}(\mathbf{s}, \mathbf{t})$ will be much smaller.

We return now to Eve's attack on a Vigenère cipher. She knows the blocksize k and has split the ciphertext into k blocks $\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_k$ as usual. The characters in each block have been encrypted using the same shift amount, say

$$\beta_i = \text{Amount that block } \mathbf{s}_i \text{ has been shifted.}$$

Eve's next step is to compare $\mathbf{s}_i$ with the string obtained by shifting the characters in $\mathbf{s}_j$ by different amounts. As a notational convenience, we write

$$\mathbf{s}_j + \sigma = \begin{pmatrix} \text{The string } \mathbf{s}_j \text{ with every character} \\ \text{shifted } \sigma \text{ spots down the alphabet.} \end{pmatrix}$$

Suppose that σ happens to equal $\beta_i - \beta_j$. Then $\mathbf{s}_j + \sigma$ has been shifted a total of $\beta_j + \sigma = \beta_i$ from the plaintext, so $\mathbf{s}_j + \sigma$ and $\mathbf{s}_i$ have been encrypted using the same shift amount. Hence as noted above, their mutual index of coincidence will be fairly large. On the other hand, if σ is not equal to $\beta_i - \beta_j$, then $\mathbf{s}_j + \sigma$ and $\mathbf{s}_i$ have been encrypted using different shift amounts, so $\text{MutIndCo}(\mathbf{s}, \mathbf{t})$ will tend to be small.

Turning this idea on its head, Eve computes all of the mutual indices of coincidence

$$\text{MutIndCo}(\mathbf{s}_i, \mathbf{s}_j + \sigma) \qquad \text{for } 1 \le i < j \le k \text{ and } 0 \le \sigma \le 25.$$

Scanning the list of values, she picks out the ones that are large, say larger than 0.065. Each large value of $\mathrm{MutIndCo}(\mathbf{s}_i, \mathbf{s}_j + \sigma)$ makes it likely that

$$\beta_i - \beta_j \equiv \sigma \pmod{26}. \tag{3.10}$$

(Note that (3.10) is only a congruence modulo 26, since a shift of 26 is the same as a shift of 0.) The leads to a system of equations of the form (3.10) for the variables $\beta_1, \ldots, \beta_k$. In practice, some of these equations will be spurious, but after a certain amount of trial-and-error, Eve will end up with values $\gamma_2, \ldots, \gamma_k$ satisfying

$$\beta_2 = \beta_1 + \gamma_2, \quad \beta_3 = \beta_1 + \gamma_3, \quad \beta_4 = \beta_1 + \gamma_4, \ldots, \quad \beta_k = \beta_1 + \gamma_k.$$

Thus if the keyword happens to start with `A`, then the second letter of the keyword would be `A` shifted by γ_2, the third letter of the keyword would be `A` shifted by γ_3, and so on. Similarly, if the keyword happens to start with `B`, then its second letter would be `B` shifted by γ_2, its third letter would be `A` shifted by γ_3, etc. So all that Eve needs to do is try each of the 26 possible keyword starting letters and decrypt the message using each of the 26 corresponding keywords. Looking at the first few characters of the 26 putative plaintexts, it is easy for her to pick out the correct one.

Remark 3.15. We make one final remark before doing an example. We noted earlier that among the many occurences of the letters `the` in the plaintext, a certain percentage of them will line up with exactly the same part of the keyword. It turns out that these repeated encryptions occur much more frequently than one might guess. This is an example of the "Birthday Paradox," which says that the probablity of getting a match (e.g. of trigrams or birthdays or colors) is quite high. We dicuss the birthday paradox and some of its many applications to cryptography in Section 3.4. In particular, see Example **??** for its relevance to the Kasiski test.

3.2.2 Cryptanalysis of the Vigenère cipher — practice

In this section we illustrate how to cryptanalyze a Vigenére ciphertext by decrypting the message given in Table 3.2.

We begin by applying the Kasiski test. A list of repeated trigrams is given in Table 3.3, together with their location within the ciphertext and the number of letters that separates them. Most of the differences in the last

```
zpgdl rjlaj kpylx zpyyg lrjgd lrzhz qyjzq repvm swrzy rigzh
zvreg kwivs saolt nliuw oldie aqewf iiykh bjowr hdogc qhkwa
jyagg emisr zqoqh oavlk bjofr ylvps rtgiu avmsw lzgms evwpc
dmjsv jqbrn klpcf iowhv kxjbj pmfkr qthtk ozrgq ihbmq sbivd
ardym qmpbu nivxm tzwqv gefjh ucbor vwpcd xuwft qmoow jipds
fluqm oeavl jgqea lrkti wvext vkrrg xani
```

Table 3.2: A Vigenère ciphertext to cryptanalyze

Trigram	Appears at places	Difference
avl	117 and 258	$141 = 3 \cdot 47$
bjo	86 and 121	$35 = 5 \cdot 7$
dlr	4 and 25	$21 = 3 \cdot 7$
gdl	3 and 24	$16 = 2^4$
lrj	5 and 21	$98 = 2 \cdot 7^2$
msw	40 and 138	$84 = 2^2 \cdot 3 \cdot 7$
pcd	149 and 233	$13 = 13$
qmo	241 and 254	$98 = 2 \cdot 7^2$
vms	39 and 137	$84 = 2^2 \cdot 3 \cdot 7$
vwp	147 and 231	$84 = 2^2 \cdot 3 \cdot 7$
wpc	148 and 232	$21 = 3 \cdot 7$
zhz	28 and 49	$21 = 3 \cdot 7$

Table 3.3: Repeated trigrams in the ciphertext Table 3.2

column are divisible by 7, and 7 is the largest number with this property, so we guess that the blocksize is 7.

Although the Kasiski test shows that the period is probably 7, we also apply the Index of Coincidence test in order to to illustrate how it works. Table 3.4 lists the indices of coincidence for various choices of block size and the average index of coincidence for each block size. We see from Table 3.4 that block size 7 has far higher average index of coincidence than the other potential block sizes, which confirms the conclusion from the Kasiski test.

Now that Eve knows that the block size is 7, she compares the blocks with one another as described in Section 3.2.1. She first breaks the ciphertext into 7 blocks by taking every 7th letter. (Notice how the ciphertext appears down

Block Size	Average Index	Individual Indices of Coincidence
4	0.038	0.034, 0.042, 0.039, 0.035
5	0.037	0.038, 0.039, 0.043, 0.027, 0.036
6	0.036	0.038, 0.038, 0.039, 0.038, 0.032, 0.033
7	0.062	0.062, 0.057, 0.065, 0.059, 0.060, 0.064, 0.064
8	0.038	0.037, 0.029, 0.038, 0.030, 0.034, 0.057, 0.040, 0.039
9	0.037	0.032, 0.036, 0.028, 0.030, 0.026, 0.032, 0.045, 0.047, 0.056

Table 3.4: Indices of coincidence of Table 3.2 for various block sizes

Blocks		Shift Amount												
i	j	0	1	2	3	4	5	6	7	8	9	10	11	12
1	2	.025	.034	.045	.049	.025	.032	.037	.042	.049	.031	.032	.037	.043
1	3	.023	**.067**	.055	.022	.034	.049	.036	.040	.040	.046	.025	.031	.046
1	4	.032	.041	.027	.040	.045	.037	.045	.028	.049	.042	.042	.030	.039
1	5	.043	.021	.031	.052	.027	.049	.037	.050	.033	.033	.035	.044	.030
1	6	.037	.036	.030	.037	.037	.055	.046	.038	.035	.031	.032	.037	.032
1	7	.054	.063	.034	.030	.034	.040	.035	.032	.042	.025	.019	.061	.054
2	3	.041	.029	.036	.041	.045	.038	.060	.031	.020	.045	.056	.029	.030
2	4	.028	.043	.042	.032	.032	.047	.035	.048	.037	.040	.028	.051	.037
2	5	.047	.037	.032	.044	.059	.029	.017	.044	.060	.034	.037	.046	.039
2	6	.033	.035	.052	.040	.032	.031	.031	.029	.055	.052	.043	.028	.023
2	7	.038	.037	.035	.046	.046	.054	.037	.018	.029	.052	.041	.026	.037
3	4	.029	.039	.033	.048	.044	.043	.030	.051	.033	.034	.034	.040	.038
3	5	.021	.041	.041	.037	.051	.035	.036	.038	.025	.043	.034	.039	.036
3	6	.037	.034	.042	.034	.051	.029	.027	.041	.034	.040	.037	.046	.036
3	7	.046	.023	.028	.040	.031	.040	.045	.039	.020	.030	**.069**	.042	.037
4	5	.041	.033	.041	.038	.036	.031	.056	.032	.026	.034	.049	.029	.054
4	6	.035	.037	.032	.039	.041	.033	.032	.039	.042	.031	.049	.039	.058
4	7	.031	.032	.046	.038	.039	.042	.033	.056	.046	.027	.027	.036	.036
5	6	.048	.036	.026	.031	.033	.039	.037	.027	.037	.045	.032	.040	.041
5	7	.030	.051	.043	.031	.034	.041	.048	.032	.053	.037	.024	.029	.045
6	7	.032	.033	.030	.038	.032	.035	.047	.050	.049	.033	.057	.050	.021

Blocks		Shift Amount												
i	j	13	14	15	16	17	18	19	20	21	22	23	24	25
1	2	.034	.052	.037	.030	.037	.054	.021	.018	.052	.052	.043	.042	.046
1	3	.031	.037	.038	.050	.039	.040	.026	.037	.044	.043	.023	.045	.032
1	4	.039	.040	.032	.041	.028	.019	**.071**	.038	.040	.034	.045	.026	.052
1	5	.042	.032	.038	.037	.032	.045	.045	.033	.041	.043	.035	.028	.063
1	6	.040	.030	.028	**.071**	.051	.033	.036	.047	.029	.037	.046	.041	.027
1	7	.040	.032	.049	.037	.035	.035	.039	.023	.043	.035	.041	.042	.027
2	3	.054	.040	.028	.031	.039	.033	.052	.046	.037	.026	.028	.036	.048
2	4	.047	.034	.027	.038	.047	.042	.026	.038	.029	.046	.040	.061	.025
2	5	.034	.026	.035	.038	.048	.035	.033	.032	.040	.041	.045	.033	.036
2	6	.033	.034	.036	.036	.048	.040	.041	.049	.058	.028	.021	.043	.049
2	7	.042	.037	.041	.059	.031	.027	.043	.046	.028	.021	.044	.048	.040
3	4	.037	.045	.033	.028	.029	**.073**	.026	.040	.040	.026	.043	.042	.043
3	5	.035	.029	.036	.044	.055	.034	.033	.046	.041	.024	.041	**.067**	.037
3	6	.023	.043	**.074**	.047	.033	.043	.030	.026	.042	.045	.032	.035	.040
3	7	.035	.035	.035	.028	.048	.033	.035	.041	.038	.052	.038	.029	.062
4	5	.032	.041	.036	.032	.046	.035	.039	.042	.038	.034	.043	.036	.048
4	6	.034	.034	.036	.029	.043	.037	.039	.036	.039	.033	**.066**	.037	.028
4	7	.043	.032	.039	.034	.029	**.071**	.037	.039	.030	.044	.037	.030	.041
5	6	.052	.035	.019	.036	.063	.045	.030	.039	.049	.029	.036	.052	.041
5	7	.040	.031	.034	.052	.026	.034	.051	.044	.041	.039	.034	.046	.029
6	7	.029	.035	.039	.032	.028	.039	.026	.036	**.069**	.052	.035	.034	.038

Table 3.5: Mutual indices of coincidence of Table 3.2 for shifted blocks

the columns.)

$$\begin{aligned}
\mathbf{s}_1 &= \texttt{zlxrhrrhwloehdweoklilwvlhphqbynwhwfjulrxx}\\
\mathbf{s}_2 &= \texttt{pazjzezzitlwboamqbvuzpjpvmtiimiquptiqjkta}\\
\mathbf{s}_3 &= \texttt{gjpgqpyvvndfjgjihjpagcqckfkhvqvvccqpmgtvn}\\
\mathbf{s}_4 &= \texttt{dkydyvrrsliiocysoosvmdbfxkobdmxgbdmdoqiki}\\
\mathbf{s}_5 &= \texttt{lpyljmiesieiwqarafrmsmrijrzmapmeoxoseewr}\\
\mathbf{s}_6 &= \texttt{rygrzsggauayrhgzvrtsejnobqrqrbtfruofaavr}\\
\mathbf{s}_7 &= \texttt{jllzqwzkowqkhkgqlygwvskwjtgsduzjvwwlvleg}
\end{aligned}$$

She then compares the i^{th} block $\mathbf{s}_i$ to the j^{th} block shifted by σ, which we denote by $\mathbf{s}_j + \sigma$, taking successively $\sigma = 0, 1, 2, \ldots, 25$. Table 3.5 gives complete list of the 546 mutual indices of coincidence

$$\text{MutIndCo}(\mathbf{s}_i, \mathbf{s}_j + \sigma) \qquad \text{for } 1 \le i < j \le 7 \text{ and } 0 \le \sigma \le 25.$$

In Table 3.5, the entry in the row labeled (i, j) and the column labeled σ is equal to

$$\text{MutIndCo}(\mathbf{s}_i, \mathbf{s}_j + \sigma) = \text{MutIndCo}(\text{Block } \mathbf{s}_i, \text{Block } \mathbf{s}_j \text{ shifted by } \sigma). \quad (3.11)$$

If this quantity is large, it suggests that $\mathbf{s}_j$ has been shifted σ further than $\mathbf{s}_i$. As in Section 3.2.1 we let

$$\beta_i = \text{Amount that the block } \mathbf{s}_i \text{ has been shifted.}$$

Then a large value for (3.11) makes it likely that

$$\beta_i - \beta_j = \sigma. \quad (3.12)$$

We have underlined the large values (those greater than 0.065) in Table 3.5 and compiled them, with the associated shift relation (3.12), in Table 3.6.

Eve's next step is to solve the system of linear equations appearing in the final column of Table 3.6, keeping in mind that all values are modulo 26, since a shift of 26 is the same as no shift at all. Notice that there are 10 equations for the six variables $\beta_1, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7$ (unfortunately, β_2 does not appear, we'll deal with it later). In general, a system of 10 equations in 6 variables has no solutions,[7] but in this case a little bit of algebra shows that not only

[7] We were a little lucky in that every relation in Table 3.6 is correct. Sometimes there are erroneous relations, but it is not hard to eliminate them with some trial-and-error.

i	j	Shift	MutIndCo	Shift Relation
1	3	1	0.067	$\beta_1 - \beta_3 = 1$
3	7	10	0.069	$\beta_3 - \beta_7 = 10$
1	4	19	0.071	$\beta_1 - \beta_4 = 19$
1	6	16	0.071	$\beta_1 - \beta_6 = 16$
3	4	18	0.073	$\beta_3 - \beta_4 = 18$
3	5	24	0.067	$\beta_3 - \beta_5 = 24$
3	6	15	0.074	$\beta_3 - \beta_6 = 15$
4	6	23	0.066	$\beta_4 - \beta_6 = 23$
4	7	18	0.071	$\beta_4 - \beta_7 = 18$
6	7	21	0.069	$\beta_6 - \beta_7 = 21$

Table 3.6: Large indices of coincidence and shift relations

is there a solution, there is actually one solution for each value of β_1. In other words, the full set of solutions is obtained by expressing each of the variables $\beta_3, \ldots, \beta_7$ in terms of β_1:

$$\beta_3 = \beta_1 + 25, \quad \beta_4 = \beta_1 + 7, \quad \beta_5 = \beta_1 + 1, \quad \beta_6 = \beta_1 + 10, \quad \beta_7 = \beta_1 + 15. \tag{3.13}$$

What should Eve do about β_2? She could just ignore it for now, but instead she picks out the largest values in Table 3.5 that relate to block 2 and uses those. The largest such values are $(i, j) = (2, 3)$ with shift 6 and index 0.60 and $(i, j) = (2, 4)$ with shift 24 and index 0.061, which give the relations

$$\beta_2 - \beta_3 = 6 \qquad \text{and} \qquad \beta_2 - \beta_4 = 24.$$

Substituting in from (3.13), these both yield $\beta_2 = \beta_1 + 5$, and the fact that they give the same value gives Eve confidence that they are correct.

To summarize, Eve now knows that however much the first block $\mathbf{s}_1$ is rotated, blocks $\mathbf{s}_2, \mathbf{s}_3, \ldots, \mathbf{s}_7$ are rotated, respectively, 5, 25, 7, 10, 1, and 15 steps further than $\mathbf{s}_1$. So for example, if $\mathbf{s}_1$ is not rotated at all (i.e. if $\beta_1 = 0$ and the first letter of the keyword is `A`), then the full keyword is `AFZHBKP`. Eve uses the keyword `AFZHBKP` to decrypt the first few blocks of the ciphertext, finding the "plaintext"

`zkhwkhulvkdoowxuqrxwwrehwkhkhurripbrzqolihruzkhwkh`.

Shift	Keyword	Decrypted Text
0	AFZHBKP	zkhwkhulvkdoowxuqrxwwrehwkhkhurripbrzqolih
1	BGAICLQ	yjgvjgtkujcnnvwtpqwvvqdgvjgjgtqqhoaqypnkhg
2	CHBJDMR	xifuifsjtibmmuvsopvuupcfuififsppgnzpxomjgf
3	DICKENS	whetherishallturnouttobetheheroofmyownlife
4	EJDLFOT	vgdsgdqhrgzkkstqmntssnadsgdgdqnnelxnvmkhed
5	FKEMGPU	ufcrfcpgqfyjjrsplmsrrmzcrfcfcpmmdkwmuljgdc
6	GLFNHQV	tebqebofpexiiqroklrqqlybqebebollcjvltkifcb
7	HMGOIRW	sdapdaneodwhhpqnjkqppkxapdadankkbiuksjheba
8	INHPJSX	rczoczmdncvggopmijpoojwzoczczmjjahtjrigdaz
⋮	⋮	⋮

Table 3.7: Decryption of Table 3.2 using shifts of the keyword AFZHBKP

That doesn't look good! So next she tries $\beta_1 = 1$ and a keyword starting with the letter B. Continuing in this fashion, she needs only check the 26 possibilities for β_1. The results are listed in Table 3.7.

Taking $\beta_1 = 3$ yields the keyword DICKENS and an acceptable plaintext. Completing the decryption using this keyword and supplying the appropriate word breaks, puntuation, and capitalization, Eve recovers the full plaintext:

> Whether I shall turn out to be the hero of my own life, or whether that station will be held by anybody else, these pages must show. To begin my life with the beginning of my life, I record that I was born (as I have been informed and believe) on a Friday, at twelve o'clock at night. It was remarked that the clock began to strike, and I began to cry, simultaneously.[8]

3.3 Probabilty theory [M]

3.3.1 Basic concepts of probability theory

In this section we introduce the basic ideas of elementary probability theory in the discrete setting. A *probability space* is a finite collection Ω consisting of all possible outcomes of an experiment and a method for assigning a probability

[8] *David Copperfield*, 1850, Charles Dickens

to each possible outcome. In other words, a probability space is a finite set of outcomes Ω and a function

$$\mathrm{Pr} : \Omega \longrightarrow \mathbb{R}.$$

Of course, we want the function Pr to satisfy our intution that

$$\mathrm{Pr}(\omega) = \text{“probability that event } \omega \text{ occurred.”}$$

Example 3.16. Consider the toss of a single coin. There are two outcomes, heads and tails, so we can let Ω be the set $\{H, T\}$. Assuming that it is a fair coin, each outcome is equally likely, so $\mathrm{Pr}(H) = \mathrm{Pr}(T) = \frac{1}{2}$.

Example 3.17. Consider the roll of two die. The sample space Ω consists a set of 36 pairs of numbers

$$\Omega = \bigl\{(n, m) : n, m \in \mathbb{Z} \text{ with } 1 \le n, m \le 6\bigr\}.$$

Again each possible outcome is equally likely. For example, the probability of rolling $(6, 6)$ is the same as the probability of rolling $(3, 4)$. Indeed, $\mathrm{Pr}\bigl((n, m)\bigr) = \frac{1}{36}$ for any choice of (n, m).

We are typically more interested in computing the probability of *compound events*, which are subsets of the sample space that may include more than one outcome. For example, in the roll of two dice in Example 3.17, we might be interested in the probability of rolling at least one six. This compound event is the subset of Ω consisting of all outcomes that include the number six, which is the set

$$\bigl\{(1,6), (2,6), (3,6), (4,6), (5,6), (6,6), (6,1), (6,2), (6,3), (6,4), (6,5)\bigr\}.$$

Suppose that we know the probability of each particular outcome. How then do we compute the probability of compound events or of events consisting of repeated trials of an experiment? The answer leads inexorably to the idea of “independence of events,” which is the concept that gives probability theory much of its complexity and richness.

The formal theory of probability is an axiomatic theory: one starts with a small list of basic axioms and derives from these useful formulas for computing probabilities of compound events. We will be content with an informal presentation of the theory, but for those who are interested in a more rigorous axiomatic treatment of probability theory, see for example [45, §2.3].

We begin with some definitions. A *sample space* (or *set of outcomes*) is a finite[9] set Ω. Each outcome $\omega \in \Omega$ is assigned a probability $\Pr(\omega)$, where we require that the probability function

$$\Pr : \Omega \longrightarrow \mathbb{R}$$

satisfying the following two properties:

$$\text{(a)} \quad 0 \le \Pr(\omega) \le 1 \quad \text{for all } \omega \in \Omega \qquad \text{and} \qquad \text{(b)} \quad \sum_{\omega \in \Omega} \Pr(\omega) = 1. \tag{3.14}$$

Notice that (3.14)(a) correspondds to our intuition that every outcome has a probability between 0 (if it never occurs) and 1 (if it always occurs) and that (3.14)(b) says that Ω contains all possible outcomes for the experiment.

An *event* is any subset of Ω, and we assign a probability to an event $E \subset \Omega$ by setting

$$\Pr(E) = \sum_{\omega \in E} \Pr(\omega). \tag{3.15}$$

In particular, $\Pr(\emptyset) = 0$ by convention and $\Pr(\Omega) = 1$ from (3.14)(b).

We say that two events E and F are *disjoint* if $E \cap F = \emptyset$. It is clear that

$$\Pr(E \cup F) = \Pr(E) + \Pr(F) \qquad \text{if } E \text{ and } F \text{ are disjoint,}$$

since then $E \cup F$ is the collection of all outcomes in either E or F. When E and F and are not disjoint, the probability of the event $E \cup F$ is not the sum of $\Pr(E)$ and $\Pr(F)$, since the outcomes common to both E and F should not be counted twice. Thus we need to subtract the outcomes common to E and F, which gives the useful formula

$$\Pr(E \cup F) = \Pr(E) + \Pr(F) - \Pr(E \cap F). \tag{3.16}$$

(See Exercise 3.18.)

The *complement* of an event E is the event E^c consisting of all outcomes that are not in E, i.e.

$$E^c = \{\omega \in \Omega : \omega \notin E\}.$$

[9] General (continuous) probability theory also deals with infinite sample spaces Ω, in which case only certain subsets of Ω are allowed to be events and are assigned probabilities. There are also further restrictions on the probability function $\Pr : \Omega \to \mathbb{R}$. For our study of cryptography in this book, it suffices to use discrete (finite) sample spaces.

The probability of the complementary event is given by

$$\Pr(E^c) = 1 - \Pr(E). \tag{3.17}$$

It is sometimes easier to first compute the probability of the complement of the event of interest and then use (3.17) to find $\Pr(E)$.

Example 3.18. We continue with Example 3.17 in which Ω consists of the possible outcomes of rolling two dice. Let E be the event consisting of at least one six being rolled. Then

$$E = \bigl\{(1,6),(6,1),(2,6),(6,2),(3,6),(6,3),(4,6),(6,4),(5,6),(6,5),(6,6)\bigr\}$$

consists of 11 outcomes, each of which individually has probability $\frac{1}{36}$, so

$$\Pr(E) = \sum_{\omega \in E} \Pr(\omega) = \frac{11}{36}.$$

Next consider the event F consisting of not rolling any number higher than two. Thus $F = \bigl\{(1,1),(1,2),(2,1),(2,2)\bigr\}$ is disjoint from E, so the probability of either rolling a six or rolling no numbers higher than two is

$$\Pr(E \cup F) = \Pr(E) + \Pr(F) = \frac{11}{36} + \frac{4}{36} = \frac{15}{36}.$$

For nondisjoint events, the computation is more complicated, since we need to avoid double counting outcomes. Consider the event G consisting of rolling doubles, i.e. $G = \bigl\{(1,1),(2,2),(3,3),(4,4),(5,5),(6,6)\bigr\}$. Then E and G both contain the outcome $(6,6)$, so their union $E \cup G$ only contains 16 outcomes, not 17. Thus the probability of rolling either a six or doubles is $\frac{16}{36}$. We can also compute this probability using formula (3.16),

$$\Pr(E \cup G) = \Pr(E) + \Pr(G) - \Pr(E \cap G) = \frac{11}{36} + \frac{6}{36} - \frac{1}{36} = \frac{16}{36} = \frac{4}{9}.$$

Similarly, the probabilty of not rolling a six is $\Pr(E^c) = 1 - \Pr(E) = \frac{15}{36}$.

To conclude this example, let H be the event that the sum of the two dice is at least 4. We could compute $\Pr(H)$ directly, but it is easier to compute the probability of H^c. Indeed, there are only three outcomes that give a sum smaller than 4, $H^c = \bigl\{(1,1),(1,2),(2,1)\bigr\}$, so $\Pr(H^c) = \frac{3}{36} = \frac{1}{12}$. Hence the probability of summing to at least 4 is $\Pr(H) = 1 - \Pr(H^c) = \frac{11}{12}$.

Let E and F be events. The event consisting of both E and F is the intersection $E \cap F$ and we can ask for the probability

$$\Pr(E \text{ and } F) = \Pr(E \cap F).$$

As the following example makes clear, the probability of the intersection of two events is not a simple function of the probabilities of the individual events.

Example 3.19. Consider the experiment consisting of drawing two cards from a deck of cards, where the second card is drawn without replacing the first one. Let E and F be the following events:

$$\begin{aligned} E &= \{\text{the first card drawn is a king}\}, \\ F &= \{\text{the second card drawn is a king}\}. \end{aligned}$$

Clearly $\Pr(E) = \frac{1}{13}$ and $\Pr(F) = \frac{1}{13}$. However, it is also clear that the probability that F occurs depends on whether E occurs. More precisely, if E occurs, then there are only 3 kings left in the remaining 51 cards, so $\Pr(F) = \frac{3}{51}$; but if E does not occur, then there are 4 kings left and $\Pr(F) = \frac{4}{51}$. Thus the probability of both E and F occuring, i.e. the probability of drawing two consecutive kings, is smaller than simply multiplying $\Pr(E)$ times $\Pr(F)$, because the occurence of the event E makes the event F less likely.

Let

$$G = \{\text{the second card drawn is an ace}\}.$$

Then the occurence of E makes G more likely, since if the first card is known to be a king, then there are still four aces left. Thus if we know that E occurs, then the probability of G increases from $\frac{4}{52}$ to $\frac{4}{51}$.

Notice, however, that if we change the experiment and require that the first card be replaced in the deck before the second card is drawn, then whether or not E occurs has no effect at all on F. Thus under this new scenario, the probability that E and F both occur is simply the product

$$\Pr(E)\Pr(F) = (\frac{1}{13})^2 \approx 0.006.$$

We learn two things from the simple discussion in Example 3.19. First, it is often easier to compute the probability of an event F if we know that some other event E has occurred. Second, we gain an intuition that leads to the mathematical definition independent events.

Definition. Events E and F are said to be *independent* if

$$\Pr(E \text{ and } F) = \Pr(E)\Pr(F),$$

where recall that the probability of "E and F" is defined to be the probability of the intersection $\Pr(E \cap F)$.

Example 3.20. A coin is tossed 10 times and the results recorded. What are the probabilities of the following events?

$$E_1 = \{\text{the first five tosses are all heads}\}.$$
$$E_2 = \{\text{the first five tosses are heads and the rest are tails}\}.$$
$$E_3 = \{\text{exactly five of the ten tosses are heads}\}.$$

The result of any one toss is independent of the result of any other toss, so we can compute the probability of getting H on the first five tosses by multiplying together the probability of getting H on any one of these tosses. Assuming that it is a fair coin, this gives the answer to our first question,

$$\Pr(E_1) = \left(\frac{1}{2}\right)^5 = \frac{1}{32},$$

since E_1 puts no restriction on the results of the subsequent five tosses.

In order to compute the probability of E_2, note that we are now asking for the probability that our sequence of tosses is exactly $\mathsf{HHHHHTTTTT}$. Again using the independence of the individual tosses, we see that

$$\Pr(E_2) = \left(\frac{1}{2}\right)^{10} = \frac{1}{1024}.$$

The computation of $\Pr(E_3)$ is a little trickier, because it asks for exactly five H's to occur, but places no restriction on when they occur . If we were to specify exactly when the five H's and the five T's occur, then the probability would be $\frac{1}{2^{10}}$, just as it was for E_2. So all that we need to do is to count how many ways we can distribute five H's and the five T's into ten spots, or equivalently, how many different sequences can we form consisting of five H's and five T's. This is simply the number of ways of chooosing five locations from ten possible locations, which is given by the combinatorial symbol $\binom{10}{5}$. Hence

$$\Pr(E_3) = \binom{10}{5} \cdot \frac{1}{2^{10}} = \frac{252}{1024} = \frac{63}{256} \approx 0.246.$$

Thus there is just under a 25% chance of getting exactly five heads in ten tosses of a coin.

3.3.2 Bayes' formula

As we saw in Section 3.3.1, there is a connection between the probability of two events E and F occuring simultaneously and the probability of one occurring if we know that the other one has occurred. The former quantity is simply $\Pr(E \cap F)$, while the latter quantity is called the *conditional probability* of F on E and is denoted by

$$\Pr(F|E) = \Pr(F \text{ given that } E \text{ has occurred}).$$

They are related by the formula

$$\Pr(F|E) = \frac{\Pr(F \cap E)}{\Pr(E)}. \tag{3.18}$$

The intuition behind (3.18), which is usually taken as the definition of the conditional probability $\Pr(F|E)$, is simple. On the lefthand side, we are assuming that E occurs, so our sample space or universe is now E instead of Ω. We are asking for the probability that the event F occurs in this smaller universe of outcomes, so we need to compute the proportion of the event F encompassed by the event E, divided by the total size of the event E itself.

Formula (3.18) immediately implies that

$$\Pr(F|E)\Pr(E) = \Pr(F \cap E) = \Pr(E \cap F) = \Pr(E|F)\Pr(F).$$

Dividing both sides by $\Pr(F)$ gives a useful formula:

$$\Pr(E|F) = \frac{\Pr(F|E)\Pr(E)}{\Pr(F)}. \tag{3.19}$$

Sometimes it is easier to compute the probability of an event by dividing it into a union of disjoint events, as in the following proposition.

Proposition 3.21. *Le E and F be events.*

(a)

$$\Pr(E) = \Pr(E|F)\Pr(F) + \Pr(E|F^c)\Pr(F^c). \tag{3.20}$$

(b)

$$\text{(Bayes' Formula)} \qquad \Pr(E|F) = \frac{\Pr(F|E)\Pr(E)}{\Pr(F|E)\Pr(E) + \Pr(F|E^c)\Pr(E^c)} \tag{3.21}$$

Proof. We give the elementary proof of (a) to illustrate the manipulation of basic probability formulas.

$$\begin{aligned}
&\Pr(E|F)\Pr(F) + \Pr(E|F^c)\Pr(F^c)\\
&\qquad = \Pr(E\cap F) + \Pr(E \cap F^c) && \text{from (3.18),}\\
&\qquad = \Pr\bigl((E\cap F)\cup(E\cap F^c)\bigr) && \text{since } E\cap F \text{ and } E\cap F^c \text{ are disjoint,}\\
&\qquad = \Pr(E) && \text{since } F\cup F^c = \Omega.
\end{aligned}$$

This completes the proof of (a). In order to prove (b), which goes by the name of Bayes' formula, we reverse the roles of E and F in (a) to get

$$\Pr(F) = \Pr(F|E)\Pr(E) + \Pr(F|E^c)\Pr(E^c), \tag{3.22}$$

and then substitute (3.22) into the denominator of (3.19) to obtain (3.21). □

Here are some examples that illustrate the power of Bayes' formula and the use of conditional probabilities.

Example 3.22. We are given two urns[10] containing gold and silver coins. Urn #1 contains 10 gold coins and 5 silver ones. Urn #2 contains 2 gold coins and 8 silver ones. An urn is chosen at random, and then a coin is picked at random. What is the probability of choosing a gold coin?

Let

$$E = \{\text{a gold coin is chosen}\}.$$

The probability of E depends first on which urn was chosen, and then on which coin is chosen in that urn. It is thus natural to break up E according to the outcome of the event

$$F = \{\text{Urn \#1 is chosen}\}.$$

Notice that F^c is then the probability that Urn #2 is chosen. The decomposition formula (3.20) says that

$$\Pr(E) = \Pr(E|F)\Pr(F) + \Pr(E|F^c)\Pr(F^c).$$

[10] The authors of [20, chapter 1] explain the preponderance of urns in the field of probability theory as being connected with the French phrase *aller aux urnes* (to vote).

It is relatively easy to compute the quantities appearing on the righthand side,

$$\Pr(E|F) = \frac{10}{15} = \frac{2}{3}, \quad \Pr(E|F^c) = \frac{2}{10} = \frac{1}{5}, \quad \text{and} \quad \Pr(F) = \Pr(F^c) = \frac{1}{2}.$$

Therefore

$$\Pr(E) = \Pr(E|F)\Pr(F) + \Pr(E|F^c)\Pr(F^c) = \frac{2}{3} \cdot \frac{1}{2} + \frac{1}{5} \cdot \frac{1}{2} = \frac{13}{30} \approx 0.433.$$

Example 3.23. Suppose that an urn contains 100 balls, of which 21 are white and the rest are black. If we pick 10 balls at random (without replacement), what is the probability that exactly 3 of them are white?

The total number of ways of selecting 10 balls from among 100 is $\binom{100}{10}$. Similarly, there are $\binom{21}{3}$ ways to select 3 white balls from among the 21 that are white, and there are $\binom{79}{7}$ to pick the other 7 balls from among the 79 that are black. There are thus $\binom{21}{3}\binom{79}{7}$ ways to select exactly 3 white balls and exactly 7 black balls. Hence the probability of picking exactly 3 white balls is

$$\Pr(\text{exactly 3 white balls}) = \frac{\binom{21}{3}\binom{79}{7}}{\binom{100}{10}} = \frac{20271005}{91015876} \approx 0.223.$$

Example 3.24 (The Prisoner Paradox). The prisoner paradox is a classical problem about conditional probability, a version of which appears in the popular literature as the “Monty Hall problem.” Three prisoners, Alice, Bob, and Carl, are informed by their jailer that the next day, one of them will be released from prison, but that the other two will have to serve life sentences. The jailer says that he will not tell any prisoner what will happen to him or her. But Alice, who reasons that her chances of going free are now $\frac{1}{3}$, asks the jailer to give her the name of one prisoner who will not go free. The jailer tells Alice that Bob will remain in jail. Now what are Alice's chances of going free? Has the probability changed? Alice could argue that she now has a $\frac{1}{2}$ chance of going free, since Bob will definitely remain behind. On the other hand, it also seems reasonable to argue that, since one of them had to stay behind, this new information could not possibly change the odds for Alice.

In fact, either answer could be correct, depending on the strategy that the jailer follows in deciding which name to give to Alice. If the jailer picks a name at random whenever both Bob and Carl are possible choices, then Alice's chances of freedom have not changed. However, if the jailer names Bob whenever possible, and otherwise names Carl, then the new information does indeed change Alice's probability of release to $\frac{1}{2}$. See Exercise 3.24 for a description of the Monty Hall problem and other fun applications of these ideas.

3.3.3 Monte Carlo Algorithms

There are many algorithms whose output is not guaranteed to be correct. For example, in Section 4.4 we describe the Miller-Rabin algorithm (Table 4.2), which is used to check whether or not a given large number is prime. In practice, one runs the algorithm many times to obtain an output that is "probably" correct. When applying these so-called *Monte Carlo* or *probabilistic algorithms*, it is important to be able to compute a confidence level, which is the probability that the output is indeed correct. In this section we describe how to do such a computation.

The basic scenario consists of a large (possibly infinite) set of integers $\mathcal{S}$ and an interesting property, call it A. For example, $\mathcal{S}$ could be the set of all integers, or for concreteness take $\mathcal{S}$ to be the set of all integers between 2^{1024} and 2^{1025}, and A could be the property of being composite.

In the preceding example, we are looking for numbers that do not have property A. More generally, suppose that we are given an integer m in $\mathcal{S}$ and that we want to know if m has property A. Usually we know the (approximate) percentage of the integers in $\mathcal{S}$ that have property A, for example 99% of elements might have property A and the other 1% do not, but it may be quite difficult to determine with certainty that any particular $m \in \mathcal{S}$ does not have property A. So instead we settle for a faster algorithm that is not absolutely certain to be correct..

A *Monte Carlo* algorithm for property A takes as its input both a number $m \in \mathcal{S}$ to be tested and a randomly chosen number r and returns as output either Yes or No as follows:

(1) If the algorithm returns Yes, then m definitely has property A.

(2) If m have property A, then the algorithm returns Yes for at least 50% of the choices for r.[11]

[11] More generally, the success rate in a Monte Carlo algorithm need not be 50%, but may instead be any positive probability that is not too small. See Exercise 3.25

Now suppose that we run the algorithm N times on the integer $m \in \mathcal{S}$, using N different randomly chosen values for r. If even a single trial returns Yes, then we know that m has property A. But suppose instead that all N trials return the answer No. How confident can we be that our integer does not have property A?

Consider the following two events:

$$E = \{\text{an integer in } \mathcal{S} \text{ does not have property } A\}.$$
$$F = \{\text{the algorithm returns } \mathsf{No}\ N \text{ times in a row}\}.$$

We are interested in the conditional probability $\Pr(E|F)$, that is, the probability that m does not have property A despite the fact that the algorithm returned No N times. We can compute this probability using Bayes' formula (3.21)

$$\Pr(E|F) = \frac{\Pr(F|E)\Pr(E)}{\Pr(F|E)\Pr(E) + \Pr(F|E^c)\Pr(E^c)}.$$

We are given that 1% of the elements in $\mathcal{S}$ have property A, so

$$\Pr(E) = 0.99 \qquad \text{and} \qquad \Pr(E^c) = 0.01.$$

Next consider $\Pr(F|E)$. If m does not have property A, which is our assumption on this conditional probability, then the algorithm always returns No, since Property (1) of the Monte Carlo method tells us that a Yes output forces m to have property A. In symbols, Property (1) says that

$$\Pr(\mathsf{No}|\text{not } A) = \Pr(A|\mathsf{Yes}) = 1.$$

It follows that $\Pr(F|E) = \Pr(\mathsf{No}|\text{not } A)^N = 1$.

Finally, we must compute the value of $\Pr(F|E^c)$. Since the algorithm is run N independent times, we have

$$\begin{aligned}\Pr(F|E^c) &= \Pr(\text{Output is } \mathsf{No}|m \text{ has property } A)^N \\ &= \bigl(1 - \Pr(\text{Output is } \mathsf{Yes}|m \text{ has property } A)\bigr)^N \\ &\le \left(1 - \frac{1}{2}\right)^N \qquad \text{from Property (2) of the Monte Carlo method,} \\ &= \left(\frac{1}{2}\right)^N.\end{aligned}$$

Substituting these values into Bayes' formula, we find that if the algorithm returns No N times in a row, then the probabilty that the integer m does not have property A is

$$\Pr(E|F) \geq \frac{1 \cdot (0.99)}{1 \cdot (0.99) + 2^{-N} \cdot (0.01)} = \frac{99}{99 + 2^{-N}}.$$

Notice that if N is large, the lower bound is very close to 1.

For example, if we run the algorithm 100 times and get 100 No answers, then the probability that m does not have property A is at least

$$\frac{99}{99 + 2^{-100}} \approx 1 - 10^{-32.1}.$$

So for most practical purposes, one may assume that m does not have property A.

3.3.4 Random variables

We are generally more interested in the consequences of an experiment, for example the net loss or gain from a game of chance, than in the experiment itself. Mathematically, this means that we are interested in functions that are defined on events and that take values in some (finite) set. A function

$$X : \Omega \longrightarrow \mathbb{R}$$

whose domain is the sample space Ω and that takes values in the real numbers is called a *random variable*. Note that since our sample spaces are finite, a random variable takes on only finitely many values. In some cases, we will talk about random variables whose values are not real numbers, but in practice they could be represented as real numbers.

Random variables are useful for defining events. For example, if x is a real number, the following are three interesting events:

$$\{\omega \in \Omega : X(\omega) \leq x\}, \qquad \{\omega \in \Omega : X(\omega) = x\}, \qquad \{\omega \in \Omega : X(\omega) > x\}.$$

Definition. Let $X : \Omega \to \mathbb{R}$ be a random variable. The *probability density function of* X, denoted by $f_X(x)$, is defined to be

$$f_X(x) = \Pr(X = x).$$

In other words, $f_X(x)$ is the probability that X takes on the value x. Sometimes we write $f(x)$, if random variable is clear.

Remark 3.25. In the theory of probability, it is more typical to use the distribution function $\Pr(X \le x)$ of a random variable, rather than its density fucntion $\Pr(X = x)$. Indeed, this is essential when studying continuous probability theory. However, since our random variables are all finite and discrete, the two notions are essentially interchangeable, so for simplicity we stick to density functions.

Example 3.26. An urn contains N balls of which m are white and $N - m$ are black. From this collection, n balls are chosen at random without replacement. Let X denote the number of white balls chosen. Then X is a random variable taking on the integer values

$$0 \le X(\omega) \le \min\{m, n\}.$$

In the case that $n \le m$, an argument similar to the one that we gave in Example 3.23 shows that the density function of X is given by the formula

$$f_X(i) = \Pr(X = i) = \frac{\binom{m}{i}\binom{N-m}{n-i}}{\binom{N}{n}}. \tag{3.23}$$

This is called the *hypergeometric density function.*

There are a number of standard density functions, such as the hypergeometric density (3.23), that occur frequently in discrete probability calculations. We briefly describe a few other common density functions.

Example 3.27. Let S be a set containing N elements, for example S could be the set $S = \{0, 1, \ldots, N - 1\}$. Let X be a random variable satisfying

$$f_X(j) = \Pr(X = j) = \frac{1}{N} \qquad \text{if } j \in S,$$

and with $f_X(j) = 0$ if $j \notin S$. This random variable X is said to be *uniformly distributed* or to have *uniform density.*

Example 3.28. Suppose an experiment (such as a coin toss) has two outcomes, success or failure. Let p denote the probability of success. The experiment is performed n times and the random variable X records the number of successes. If we fix our attention on k particular experiments, the probability that those, and only those, experiments succeed is $p^k(1 - p)^{n-k}$. There are $\binom{n}{k}$ ways for us to choose k particular experiments, so the probability of exactly k successes (i.e. the density function of X evaluated at k) is

$$f_X(k) = \Pr(X = k) = \binom{n}{k} p^k (1-p)^{n-k}. \tag{3.24}$$

This is called the *binomial density function.*

Example 3.29. We repeatedly toss an unfair coin, where the probability of getting heads is some number $0 < p < 1$. Let X be the random variable giving the total number of coin tosses required to before heads appears for the first time. It is possible for X to take on any positive integer value, and the density function of X is given by the formula

$$f_X(n) = \Pr(X = n) = (1-p)^{n-1} p \qquad \text{for } n = 1, 2, 3, \ldots. \tag{3.25}$$

A random variable with the density function (3.25) is said to have a *geometric density*, because the sequence of probabilities $f_X(1), f_X(2), f_X(3), \ldots$ form a geometric progression.[12] Later in Example 3.35 we compute the expected value of this X by summing a geometric series.

Definition. If X and Y are two random variables, then their *joint density function* $f_{X,Y}(x, y)$, is the probability that X takes on the value x and Y takes on the value y,

$$f_{X,Y}(x, y) = \Pr(X = x \text{ and } Y = y).$$

The *conditional density function*, denoted $f_{X|Y}(x|y)$, is the probability that X takes on the value x given that Y takes on the value y,

$$f_{X|Y}(x|y) = \Pr(X = x | Y = y).$$

We say that X and Y are *independent* if

$$f_{X,Y}(x, y) = f_X(x) f_Y(y) \quad \text{for all } x \text{ and } y.$$

This corresponds to the events $\{X = x\}$ and $\{Y = y\}$ being independent in the earlier sense of independence defined on page 114. If there is no chance for confusion, we sometimes write $f(x, y)$ and $f(x|y)$ for $f_{X,Y}(x, y)$ and $f_{X|Y}(x|y)$, respectively.

[12] A sequence $a_1, a_2, a_3, \ldots$ is called a *geometric progression* if all of the ratios a_{n+1}/a_n are the same. Similarly, the sequence is an *arithmetic progression* if all of the difference $a_{n+1} - a_n$ are the same.

Example 3.30. An urn contains four gold coins and three silver coins. A coin is drawn at random, examined, and returned to the urn, and then a second coin is randomly drawn and examined. Let X be the number of gold coins drawn and let Y be the number of silver ones. To find the joint density function $f_{X,Y}(x,y)$, we need to compute the probability of the event $\{X = x \text{ and } Y = y\}$. To help explain the calculation, we define two additional random variables. Let

$$F = \begin{cases} 1 & \text{if first pick is gold,} \\ 0 & \text{if first pick is silver,} \end{cases} \quad \text{and} \quad S = \begin{cases} 1 & \text{if second pick is gold,} \\ 0 & \text{if second pick is silver.} \end{cases}$$

Notice that $X = F + S$ and $Y = 2 - X = 2 - F - S$. Further, F and S are independent and $\Pr(F = 1) = \Pr(S = 1) = \frac{4}{7}$. Then for example, we can compute $f_{X,Y}(1,1)$ as follows:

$$\begin{aligned} f_{X,Y}(1,1) &= \Pr(X = 1 \text{ and } Y = 1) \\ &= \Pr(F = 1 \text{ and } S = 0) + \Pr(F = 0 \text{ and } S = 1) \\ &= \Pr(F = 1) \cdot \Pr(S = 0) + \Pr(F = 0) \cdot \Pr(S = 1) \\ &= \frac{4}{7} \cdot \frac{3}{7} + \frac{3}{7} \cdot \frac{4}{7} = \frac{24}{49} \approx 0.4898. \end{aligned}$$

This compuation was easy, since X and Y are independent. How do our computations change if the first coin is not replaced before the second coin is selected? Then the probability of getting a silver coin on the second pick depends on whether the first pick was gold or silver. More precisely, the earlier computation changes to

$$\begin{aligned} f_{X,Y}(1,1) &= \Pr(X = 1 \text{ and } Y = 1) \\ &= \Pr(F = 1 \text{ and } S = 0) + \Pr(F = 0 \text{ and } S = 1) \\ &= \Pr(S = 0 | F = 1) \Pr(F = 1) + \Pr(S = 1 | F = 0) \Pr(F = 0) \\ &= \frac{3}{6} \cdot \frac{4}{7} + \frac{2}{6} \cdot \frac{3}{7} = \frac{3}{7} \approx 0.4286. \end{aligned}$$

Thus the chances of getting exactly one gold coin and exactly one silver coin is considerably less if the coins are not replaced after each pick.

For convenience, we restate a version of Bayes' formula using conditional densities.

Theorem 3.31 (Bayes' Formula)**.** *Let X and Y be two random variables with $p_Y(y) > 0$. Then*

$$f_{X|Y}(x|y) = \frac{f_X(x) f_{Y|X}(y|x)}{f_Y(y)}.$$

In particular,

$$X \text{ and } Y \text{ are independent} \iff f_{X|Y}(x|y) = f_X(x) \quad \text{for all } x \text{ and } y.$$

Example 3.32. Let X and Y be independent random variables taking on values $+1$ and -1 with probability $\frac{1}{2}$ each, and let $Z = XY$. Then Z also take on the values $+1$ and -1, and we have

$$f_Z(1) = \sum_{x\in\{-1,+1\}} \sum_{y\in\{-1,+1\}} \Pr(Z = 1|X = x \text{ and } Y = y) \cdot f_{X,Y}(x,y). \quad (3.26)$$

If $(X,Y) = (+1,-1)$ or $(X,Y) = (-1,1)$, then $Z \neq 1$, so only the two terms with $(x,y) = (1,1)$ and $(x,y) = (-1,-1)$ appear in the sum (3.26). For these two terms, we have $\Pr(Z = 1|X = x \text{ and } Y = y) = 1$, so

$$f_Z(1) = \Pr(X = 1 \text{ and } Y = 1) + \Pr(X = -1 \text{ and } Y = -1) = \frac{1}{2}\cdot\frac{1}{2} + \frac{1}{2}\cdot\frac{1}{2} = \frac{1}{2}.$$

It follows that $f_Z(-1) = 1 - f_Z(1)$ is also equal to $\frac{1}{2}$.

Next we compute the joint probabibility density of Z and X. For example,

$$\begin{aligned} f_{Z,X}(1,1) &= \Pr(Z = 1 \text{ and } X = 1) \\ &= \Pr(X = 1 \text{ and } Y = 1) \\ &= \frac{1}{4} \qquad \text{since } X \text{ and } Y \text{ are independent,} \\ &= f_Z(1) f_X(1). \end{aligned}$$

Similar computations show that

$$f_{Z,X}(z,x) = f_Z(z) f_X(x) \qquad \text{for all } z, x \in \{-1,+1\},$$

so by Theorem 3.31, Z and X are independent. The argument works equally well for Z and Y, so Z and Y are also independent. Thus among the three random variables X, Y, and Z, any pair of them are independent. Yet we would not want to call the three of them together an indepedent family, since the value of Z is determined by the values of X and Y. The prompts the following definition.

Definition. A family of two or more random variables $\{X_1, X_2, \ldots, X_n\}$ is *independent* if the events

$$\{X_1 = x_1 \text{ and } X_2 = x_2 \text{ and } \cdots \text{ and } X_n = x_n\}$$

are independent for every choice of $x_1, x_2, \ldots, x_n$.

Notice that the random variables X, Y and $Z = XY$ in Example 3.32 are not an independent family, since

$$\Pr(Z = 1 \text{ and } X = 1 \text{ and } Y = -1) = 0,$$

while

$$\Pr(Z = 1) \cdot \Pr(X = 1) \cdot \Pr(Y = -1) = \frac{1}{8}.$$

3.3.5 Expected value

The expected value of a random variable X, denoted $E(X)$, is the average of its values weighted by their probability of occurence. It thus provides a rough first indication as to the behavior of X.

Definition. Let X be a random variable that takes on the values $x_1, \ldots, x_n$. The *expected value* (or *mean*) of X is the quantity

$$E(X) = \sum_{i=1}^{n} x_i \cdot f_X(x_i) = \sum_{i=1}^{n} x_i \cdot \Pr(X = x_i).$$

Example 3.33. Let X be the random variable whose value is the sum of the numbers appearing on two tossed dice. The possible values of X are the integers between 2 and 12, so

$$E(X) = \sum_{i=2}^{12} i \cdot \Pr(X = i).$$

There are 36 ways for the two dice to fall, as indicated in Table 3.8(a). We read off from that table the number of ways that the sum can equal i for each value of i between 2 and 12 and compile the results in Table 3.8(b). The probability that $X = i$ is $\frac{1}{36}$ times the total number of ways that two dice can sum to i, so we can use Table 3.8(b) to compute

$$\begin{aligned} E(X) = 2 \cdot \frac{1}{36} + 3 \cdot \frac{2}{36} + 4 \cdot \frac{3}{36} + 5 \cdot \frac{4}{36} + 6 \cdot \frac{5}{36} + 7 \cdot \frac{6}{36} \\ + 8 \cdot \frac{5}{36} + 9 \cdot \frac{4}{36} + 10 \cdot \frac{3}{36} + 11 \cdot \frac{2}{36} + 12 \cdot \frac{1}{36} = 7. \end{aligned}$$

	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

(a) Sum of two dice

Sum	# of ways
2 or 12	1
3 or 11	2
4 or 10	3
5 or 9	4
6 or 8	5
7	6

(b) Number of ways to make a sum

Table 3.8: Outcome of rolling two dice

This answers makes sense, since the middle value is 7, and for any integer j, the value of X is just as likely to be $7+j$ as it is to be $7-j$.

The name "expected" value is somewhat misleading, since the fact that the expectation $E(X)$ is a weighted average means that it may take on a value that is not actually attained, as the next example shows.

Example 3.34. Suppose that we choose an integer at random from among the integers $\{1,2,3,4,5,6\}$ and let X be the value of our choice. Then $\Pr(X=i)=\frac{1}{6}$ for each $1 \le i \le 6$, i.e. X is uniformly distributed, so the expected value of X is

$$E(X) = \frac{1}{6}(1+2+3+4+5+6) = \frac{7}{2}.$$

Thus the expectation of X is a value that X does not actually attain. More generally, the expected value of a random variable uniformly distributed on $\{1,2,\ldots,N\}$ is $(N+1)/2$.

Example 3.35. We return to our coin tossing experiment (Example 3.29), where the probability of getting H on any one coin toss is equal to p. Let X be the random variable that is equal to n if H appears for the first time at the n^{th} coin toss. Then X has a geometric density and its density function $f_X(n)$ is given by the formula (3.25). We compute $E(X)$, which is the expected number of tosses before getting the first H to appear.

$$\begin{aligned}E(X) = \sum_{n=1}^{\infty} np(1-p)^{n-1} &= -p \sum_{n=1}^{\infty} \frac{d}{dp}\bigl((1-p)^n\bigr) \\ &= -p\frac{d}{dp}\Bigl(\sum_{n=1}^{\infty}(1-p)^n\Bigr) = -p\frac{d}{dp}\left(\frac{1}{p}\right) = \frac{p}{p^2} = \frac{1}{p}.\end{aligned}$$

This answer passes the "plausibility test," since the smaller the value of p, the more tosses we expect to need before obtaining our first H. Notice that the computation of $E(X)$ involves a very useful trick with derivatives followed by the summation of a geometric series. See Exercise 3.28 for further applications of this method.

3.4 Collision algorithms

A simple, yet surprisingly powerful, search method is based on the observation that it is usually much easier to find matching objects (i.e. to find a collision) than it is to find a particular object. The following elementary result illustrates this idea. For cryptographic applications, you should view the urn as containing pieces of keys or plaintexts that you are trying to fit together.

Theorem 3.36 (Collision Theorem)**.** *An urn contains N balls, of which n are red and $N-n$ are blue. Bob randomly selects a ball from the urn, replaces it in the urn, randomly selects a second ball, replaces it, and so on. He does this until he has looked at a total of m ball.*

(a) *The probability that Bob selects at least one red ball is*

$$\Pr(\textit{at least one red}) = 1 - \left(1 - \frac{n}{N}\right)^m. \tag{3.27}$$

(b) *A lower bound for the probability* (3.27) *is*

$$\Pr(\textit{at least one red}) \geq 1 - e^{-mn/N}. \tag{3.28}$$

If N is large and mn is not too large (e.g. $mn < 100N$), then (3.28) *is almost an equality.*

Proof. Each time Bob selects a ball, his probability of choosing a red one is $\frac{n}{N}$, so you might think that since he chooses m balls, his probability of getting a red one is $\frac{mn}{N}$. However, a small amount of thought shows that this must be incorrect. For example, if m is large, this would lead to a probability that is larger than 1. The difficulty is that we are overcounting the times that Bob happens to select more than one red ball. The correct way to calculate is to compute the probability that Bob chooses only blue balls and then subtract this complementary probability from 1. Thus

$$\begin{aligned}\Pr\left(\begin{array}{c}\text{at least one red}\\ \text{ball in } m \text{ attempts}\end{array}\right) &= 1 - \Pr(\text{all } m \text{ choices are blue})\\ &= 1 - \prod_{i=1}^{m} \Pr(i^{\text{th}} \text{ choice is blue})\\ &= 1 - \prod_{i=1}^{m} \frac{N-n}{N}\\ &= 1 - \left(1 - \frac{n}{N}\right)^m\end{aligned}$$

This completes the proof of (a).

For (b), we use the inequality

$$e^{-x} \geq 1 - x \qquad \text{for all } x \in \mathbb{R}.$$

(See Exercise 3.32(a).) Setting $x = n/m$, this shows that

$$1 - \left(1 - \frac{n}{N}\right)^m \geq 1 - (e^{-n/N})^m = 1 - e^{-mn/N},$$

which proves the important inequality in (b). We leave it to the reader (Exercise 3.32(b)) to prove that the inequality is close to being an equality if mn is not too large compared to N. □

Example 3.37. A deck of cards is shuffled and eight cards are dealt face up. Bob then takes a second deck of cards and choose eight cards are random, replacing the card after each choice. What is Bob's probability of matching one of the cards from the first deck?

We view the eight dealt cards from the first deck as "marking" those same cards in the second deck. So our "urn" is the second deck, the "red balls" are the eight marked cards in the second deck, and the "blue balls" are the other 48 cards in the second decks. Theorem 3.36 tells us that

$$\Pr(\text{a match}) = 1 - \left(1 - \frac{8}{52}\right)^{8} \approx 73.7\%.$$

The approximation (3.28) gives a lower bound of 70.8%.

Similarly, if instead Bob deals ten cards from the first deck and only chooses five cards from the second deck, then

$$\Pr(\text{a match}) = 1 - \left(1 - \frac{10}{52}\right)^{5} \approx 65.6\%.$$

Example 3.38. A set contains 10^{10} elements. Bob randomly selects two lists of 100,000 elements each from the set. What is the (approximate) probability that there will be a match between the two lists? Formula (3.27) says that

$$\Pr(\text{a match}) = 1 - \left(1 - \frac{100{,}000}{10^{10}}\right)^{100{,}000} \approx 65.2\%.$$

More precisely, formula (3.27) gives the value 63.21224%, while the approximate lower bound in formula (3.28) gives 63.21206%. As you can see, the approximation is quite accurate.

It is interesting to observe that if Bob doubles the number of elements in each list to 200,000, then his probability of getting a match increases quite substantially to 98.2%. And if he triples the number of elements in each list to 300,000, then the probability of a match is 99.988%. This rapid increase reflects that fact that the exponential function in (3.28) decreases very rapidly as soon as mn becomes larger than N.

Example 3.39. A set contains N objects. Bob randomly chooses n of them, makes a list of his choices, replaces them, and then chooses another n of them. How large should he choose n to give himself a 50% chance of getting a match? How about if he wants a 99.99% chance of getting a match?

For the first question, Bob uses (3.28) to set

$$\Pr(\text{match}) \approx 1 - e^{-n^2/N} = \frac{1}{2}.$$

It is easy to solve this for n,

$$e^{-n^2/N} = \frac{1}{2}, \quad \text{so} \quad -\frac{n^2}{N} = \ln\left(\frac{1}{2}\right), \quad \text{so} \quad n = \sqrt{N \cdot \ln(2)} \approx 0.83\sqrt{N}.$$

Thus it is enough to create lists that are a bit shorter than $\sqrt{N}$ in length.

The second question is similar, but now Bob solves

$$\Pr(\text{match}) \approx 1 - e^{-n^2/N} = 0.9999 = 1 - 10^{-4}.$$

The solution is

$$n = \sqrt{N \cdot \ln(10^4)} \approx 3 \cdot \sqrt{N}.$$

Remark 3.40. Algorithms that rely on finding matching elements from within one or more lists go by a variety of names, including *collision algorithm*, *meet-in-the-middle algorithm*, *birthday paradox algorithm*, and *square root algorithm*. The last, of course, refers to the fact that in many instances, the running time of a collision algorithm is approximately the square root of the running time required by an exhaustive search. See Exercise 3.30 for the connection with birthdays When one of these algorithms is used to break a cryptosystem, the word algorithm is often replaced by the word attack, so for example, cryptanalysts refer to a *meet-in-the-middle attack* or a *square root attack*.

Remark 3.41. Collision algorithms tend to take approximately $\sqrt{N}$ steps in order to find a collision amongst N objects. A drawback of these algorithms is that they require creation of one or more lists of size approximately $\sqrt{N}$. When N is large, providing storage for $\sqrt{N}$ numbers may be more of an obstacle than doing the compuation. In Section 3.5 we describe a collision method due to Pollard that, at the cost of a small amount of extra computation, requires essentially no storage.

3.4.1 A discrete logarithm collision algorithm

There are many applications of the Collision Theorem 3.36 to cryptography. Typically these involve searching a space of keys or texts. In this section we illustrate one such application by showing how the discrete logarithm problem in $\mathbb{F}_p$ can be solved in approximately $\sqrt{p}$ steps. Later we describe another, faster, method called the index calculus to solve the discrete logarithm problem (Section 4.8), but for the discrete logarithm problem in other groups, such as the elliptic curve groups studied in Chapter 5, collision algorithms remain the fastest known method of solution.

Proposition 3.42. *Let $h \in \mathbb{F}_p^*$ be an element of order N, i.e. $h^N = 1$ and no smaller power of h is equal to 1. (Note the N divides $p-1$, see Exercise 1.28(b).) Then assuming that it has a solution, the discrete logarithm* *problem*

$$h^x = b \tag{3.29}$$

can be solved in a small multiple of $\sqrt{N}$ steps, where each step consists of raising h to a power. (Note that since $h^N = 1$, the powering algorithm from Section 1.3.2 lets us raise h to any power in fewer than $2 \log N$ multiplications in $\mathbb{F}_p$.)

Proof. The idea is to write x as $x = y - z$ and look for a solution to

$$h^y = b \cdot h^z.$$

We begin by choosing some random exponents $y_1, y_2, \ldots, y_n$ between 1 and N and computing the values of

$$h^{y_1}, h^{y_2}, h^{y_3}, \ldots, h^{y_n} \quad \text{in } \mathbb{F}_p. \tag{3.30}$$

Note that all of the values (3.30) are in the set

$$S = \{1, h, h^2, h^3, \ldots, h^{N-1}\},$$

so (3.30) is a selection of (approximately) n elements of S. In terms of the Collision Theorem 3.36, we view S as an urn containing N balls and the list (3.30) as coloring n of those balls red.

Next we choose some more random exponents $z_1, z_2, \ldots, z_n$ between 1 and k and compute the quantities

$$b \cdot h^{z_1}, b \cdot h^{z_2}, b \cdot h^{z_3}, \ldots, b \cdot h^{z_n} \quad \text{in } \mathbb{F}_p. \tag{3.31}$$

Since we are assuming that (3.29) has a solution, i.e. b is equal to some power of h, it follows that each of the values $b \cdot h^z$ is also in the set S. Thus the list (3.31) may be viewed as selecting n elements from the urn, and we would like to know the probability of selecting at least one red ball, i.e. the probability that at least one element in the list 3.31 matches an element in the list 3.30. The Collision Theorem 3.36 says that

$$\Pr\left(\begin{array}{l}\text{at least one match}\\ \text{between (3.30) and (3.31)}\end{array}\right) \approx \left(1 - \frac{n}{N}\right)^n \approx 1 - e^{-n^2/N}.$$

Thus if we choose (say) $n \approx 3\sqrt{N}$, then our probability of getting a match is approximately 99.98%, so we are almost guaranteed a match. (If that is not good enough, take $n \approx 5\sqrt{N}$ to get a probability of success greater than $1 - 10^{-10}$.) Notice that as soon as we find a match between the two lists, say $h^y = b \cdot h^z$, then we have solved the discrete logarithm problem (3.29) by setting $x = y - z$.[13]

How long does it take us to find this solution. Each of the lists (3.30) and (3.31) has n elements, so it takes approximately $2n$ steps to assemble each list. More precisely, each element in each list requires us to compute h^i for some value of i between 1 and N, and it takes approximately $2\log(i)$ multiplications to compute h^i using the powering algorithm (Section 1.3.2). Thus it takes approximately $4n\log(N)$ multiplications to assemble the two lists. (Here log is log base 2.) In addition, it takes about $\log(n)$ steps to check if an element of the second list is in the first list (e.g. sort the first list), so $n\log(n)$ comparisons altogether. Hence the total computation time is approximately

$$4n\log(N) + n\log(n) = n\log(N^4 n) \text{ steps.}$$

Taking $n \approx 3\sqrt{N}$, which as we have seen gives us a 99.98% chance of success, we find that

$$\text{Compuation Time} \approx 13.5 \cdot \sqrt{N} \cdot \log(1.3 \cdot N). \qquad \square$$

Remark 3.43. Proposition 3.42 gives an algorithm that, with very high probability, solves the discrete logarithm problem in $O\big(\sqrt{N} \cdot \log(N)\big)$ steps. A variant due to Shanks called the *Babystep-Giantstep Method* guarantees to find a solution in approximately the same number of steps. The idea is to take $n > \sqrt{N}$ and to compute the two lists

$$\begin{aligned} &\text{List 1:} \quad 1, h, h^2, h^3, \ldots, h^n, \\ &\text{List 2:} \quad b, b \cdot h^{-n}, b \cdot h^{-2n}, h^{-3n}, \ldots, h^{-n^2}. \end{aligned}$$

Note that in creating List 2, we first compute the quantity $H = h^{-n}$ and then compile List 2 by computing $b, b \cdot H, b \cdot H^2, \ldots, b \cdot H^n$. Thus creating the two lists takes approximately $2n$ multiplications.[14] Next we look for a match

[13] If this value of x happens to be negative and you want a positive solution, you can always use the fact that $h^N = 1$ to replace it with $x = y - z + N$.

[14] Multiplication by h is a "baby step" and multiplication by H is a "giant step," whence Shanks' name for the algorithm.

between Lists 1 and 2. This can be done in a small multiple of $\log(n)$ steps using standard sorting and searching algorithms. We claim that there is always a match.

To see this, let x be the unknown solution to $h^x = b$ and write x as

$$x = nq + r \quad \text{with } 0 \le r < n.$$

We know that $1 \le x < N$, so

$$q = \frac{x - r}{n} < \frac{N}{n} < n \quad \text{since } n > \sqrt{N}.$$

Hence we can rewrite the equation $h^x = b$ as

$$h^r = b \cdot h^{-qn} \qquad \text{with } 0 \le r < n \text{ and } 0 \le q < n.$$

Thus h^r is in List 1 and $b \cdot h^{-qn}$ is in List 2, which shows that Lists 1 and 2 have a common element.

To recapitulate, taking any $n > \sqrt{N}$, we have shown that in a small multiple of $n \log(n)$ steps we are guaranteed to find the match $h^r = b \cdot h^{-qn}$. Then $x = r + qn$ solves the original discrete logarithm problem.

Example 3.44. We do an example with small numbers to illustrate the use of collisions. We solve the discrete logarithm problem

$$2^x = 390 \quad \text{in the finite field } \mathbb{F}_{659}.$$

The number 2 has order 658 moduluo 659, so it is a primitive root. We choose random exponents t and compute the values of h^t and $a \cdot h^t$ until we get a match. The results are compiled in Table 3.9. We see that

$$2^{83} = 390 \cdot 2^{564} = 422 \quad \text{in } \mathbb{F}_{659}.$$

Hence using two lists of length 18 we have solved a discrete logarithm problem in $\mathbb{F}_{659}$. (We had a 39% chance of getting a match with lists of length 18, so we were a little bit lucky.) The solution is

$$2^{83} \cdot 2^{-564} = 2^{-481} = 2^{177} = 390 \quad \text{in } \mathbb{F}_{659}.$$

t	h^t	$a \cdot h^t$	t	h^t	$a \cdot h^t$	t	h^t	$a \cdot h^t$
564	410	**422**	53	10	605	513	164	37
469	357	181	332	651	175	71	597	203
276	593	620	178	121	401	314	554	567
601	416	126	477	450	206	581	47	537
9	512	3	503	116	428	371	334	437
350	445	233	198	426	72	83	**422**	489

Table 3.9: Solving $2^x = 390$ in $\mathbb{F}_{659}$ with random exponent collisions

k	h^k	$a \cdot H^k$	k	h^k	$a \cdot H^k$	k	h^k	$a \cdot H^k$	k	h^k	$a \cdot H^k$
1	9704	347	9	15774	16564	17	10137	10230	25	4970	12260
2	6181	13357	10	12918	11741	18	17264	3957	26	9183	6578
3	5763	12423	11	16360	16367	19	4230	9195	27	10596	7705
4	1128	13153	12	13259	7315	20	9880	13628	28	2427	1425
5	8431	7928	13	4125	2549	21	9963	10126	29	6902	6594
6	16568	1139	14	16911	10221	22	15501	5416	30	11969	12831
7	**14567**	6259	15	4351	16289	23	6854	13640	31	6045	4754
8	2987	12013	16	1612	4062	24	15680	5276	32	7583	**14567**

Table 3.10: Babystep-giantstep to solve $9704^x \equiv 13896 \pmod{17389}$

Example 3.45. We illustrate Shanks' babystep-giantstep method by using it to solve the discrete logarithm problem

$$h^x = a \quad \text{in } \mathbb{F}_p \text{ with} \quad h = 9704, \quad a = 13896, \quad \text{and} \quad p = 17389.$$

The number 9704 has order 1242 in $\mathbb{F}_{17389}$. Set $N = \lceil \sqrt{1242}\,\rceil = 36$ and $H = h^{-N} = 9704^{-36} = 2494$. Table 3.10 lists the values of h^k and $a \cdot H^k$ for $k = 1, 2, \ldots$. From the table we see that

$$9704^7 = 13896 \cdot 2494^{32} = 14567 \quad \text{in } \mathbb{F}_{17389}.$$

Using the fact that $2494 = 9704^{-36}$, we compute

$$13896 = 9704^7 \cdot 2494^{-32} = 9704^7 \cdot (9704^{36})^{32} = 9704^{1159} \quad \text{in } \mathbb{F}_{17389}.$$

Hence $x = 1159$ solves the problem $9704^x = 13896$ in $\mathbb{F}_{17389}$.

Remark 3.46. We have presented the collision algorithm (Proposition 3.42) and the babystep-giantstep method (Remark 3.43) as methods to solve the discrete logarithm problem in a finite field $\mathbb{F}_p$. It is worth noting these methods work for the discrete logarithm problem in any group. The key point is that both algorithms only require multiplication of elements. For $\mathbb{F}_p$, there is a faster method for solving the discrete logarithm method, called the *Index Calculus*, which makes use of the fact that $\mathbb{F}_p$ has two operations, multiplication and addition. (See Section 4.8.)

3.5 Pollard's ρ method

As we noted in Remark 3.41, collision algorithms tend to require quite a lot of storage. A beautiful idea of Pollard often allows one to use almost no storage, at the cost of a small amount of extra computation. We explain the basic idea of Pollard's method and then illustrate it by solving the discrete logarithm problem.

3.5.1 Abstract formulation of Pollard's ρ method

We begin with an abstract setting. Let S be a finite set and let

$$f : S \longrightarrow S$$

be a function that does a good job at mixing up the elements of S. Suppose that we start with some element $x \in S$ and we repeatedly apply f to create a sequence of elements

$$x_0 = x, \quad x_1 = f(x_0), \quad x_2 = f(x_1), \quad x_3 = f(x_2), \quad x_4 = f(x_3), \ldots.$$

In other words,

$$x_i = (\underbrace{f \circ f \circ f \circ \cdots \circ f}_{i \text{ iterations of } f})(x).$$

The map f from S to itself is an example of a *discrete dynamical system.* The sequence

$$x_0, x_1, x_2, x_3, x_4, \ldots \tag{3.32}$$

is called the (*forward*) *orbit of* x by the map f and is denoted $O_f^+(x)$.

Eventually there must be some element of S that appears twice in the orbit $O_f^+(x)$, since S is finite. We can illustrate the orbit as shown in Figure 3.1. For a while the points $x_0, x_1, x_2, x_3, \ldots$ travel along a "path" without repeating until eventually they loop around to give a repeated element. Then they continue moving around the loop. As illustrated, we let T be the number of

Figure 3.1: Pollard's ρ method

elements in the "tail" before getting to the loop and M be the number of elements in the loop. In other words, T and M are defined by the conditions

$$T = \left(\begin{array}{l} \text{largest integer so that } x_{T-1} \\ \text{appears only once in } O_f^+(x) \end{array} \right) \qquad M = \left(\begin{array}{l} \text{smallest integer so} \\ \text{that } x_{T+M} = x_T \end{array} \right)$$

Remark 3.47. Look again at the illustration in Figure 3.1. It may remind you of a certain Greek letter. For this reason, collision algorithms based on following the orbit of an element in a discrete dynamical system are called *ρ algorithms*. The first ρ algorithm was invented by Pollard in 1974.

Suppose that S contains N elements. Later in Theorem 3.48 we sketch a proof that the quantity $T + M$ is usually no more than a small multiple of $\sqrt{N}$. Since $x_T = x_{T+M}$ by definition, this means that we obtain a collision in $O(\sqrt{N}\,)$ steps. However, since we don't know the values of T and M, it appears that we need to make a list of $x_0, x_1, x_2, x_3, \ldots, x_{T+M}$ in order to detect the collision.

Pollard's clever idea is that it is possible to detect a collision in $O(\sqrt{N}\,)$ steps without storing all of the values. There are various ways to accomplish this. We describe one such method which, although not of optimal efficiency, has the advantage of being easy to explain. (For more efficient methods, see [8] [11, §8.5], or [36].) The idea is to compute not only the sequence x_i, but also a second sequence y_i defined by

$$y_0 = x \qquad \text{and} \qquad y_{i+1} = f\bigl(f(y_i)\bigr) \quad \text{for } i = 0, 1, 2, 3, \ldots.$$

In other words, every time that we apply f to generate the next element of the x_i sequence, we apply f twice to generate the next element of the y_i sequence. It is clear that

$$y_i = x_{2i}.$$

How long will it take us to find an index i with $x_{2i} = x_i$? In general, for $j > i$ we have

$$x_j = x_i \qquad \text{if and only if} \qquad i \geq T \text{ and } j \equiv i \pmod{M}.$$

This is clear from the ρ-shaped picture in Figure 3.1, since we get $x_j = x_i$ precisely when we are past x_T (i.e. when $i \geq T$)and x_j has gone around the loop past x_i an integral number of times (i.e. when $j - i$ is a multiple of M).

Thus $x_{2i} = x_i$ if and only if $i \geq T$ and $2i \equiv i \pmod{M}$. The latter condition is equivalent to $M|i$, so we get $x_{2i} = x_i$ exactly when i is equal to the first multple of M that is larger than T. Since one of the numbers T, $T+1$,..., $T+M-1$ is divisible by M, this proves that

$$x_{2i} = x_i \quad \text{for some } 1 \leq i < T + M.$$

We show in the next theorem that the average value of $T + M$ is approximately $1.25 \cdot \sqrt{N}$, so we probably get a collision in $O(\sqrt{N}$ steps, which is same running time as the collision algorithm described in Section 3.4.1, but notice that we only need to store two numbers, namely the current values of the x_i sequence and the y_i sequence.

Theorem 3.48 (Pollard's ρ Method—Abstract Version). *Let S be a finite set containing N elements, let $f : S \to S$ be a map, and let $x \in S$ be an initial point.*

(a) *Suppose that the forward orbit $O_f^+(x) = \{x_0, x_1, x_2, \ldots\}$ of x has a tail of length T and a loop of length M, as illustrated in Figure 3.1. Then*

$$x_{2i} = x_i \quad \textit{for some } 1 \leq i < T + M. \tag{3.33}$$

(b) *If the map f is sufficiently random, then the expected value of $T + M$ is*

$$E(T + M) \approx 1.2533 \cdot \sqrt{N}.$$

Hence if N is large, then we expect to find a collision (3.33) *in $O(\sqrt{N}\,)$ steps, where a "step" is one evaluation of the function f.*

Proof. (a) We proved this earlier in this section.
(b) We sketch the proof of (b) because it is an instructive blend of probability theory and analysis of algorithms. However, the reader desiring a rigorous proof will need to fill in some details. Suppose that we compute $x_0, x_1, x_2, \ldots, x_{k-1}$. What is the probability that we do not get any matches? If we assume that the successive x_i's are randomly chosen from

the set S, then we can compute this probability as

$$\begin{aligned}\Pr\left(\begin{array}{c}x_0, x_1, \ldots, x_{k-1}\\ \text{are all different}\end{array}\right) &= \prod_{i=1}^{k-1} \Pr\left(\begin{array}{c}x_i \ne x_j \text{ for}\\ \text{all } 0 \le j < i\end{array} \;\middle|\; \begin{array}{c}x_0, x_1, \ldots, x_{i-1}\\ \text{are all different}\end{array}\right)\\ &= \prod_{i=1}^{k-1}\left(\frac{N-i}{N}\right) \qquad (3.34)\\ &= \prod_{i=1}^{k-1}\left(1-\frac{i}{N}\right). \qquad (3.35)\end{aligned}$$

Note that the probability formula (3.34) comes from the fact that if the first i choices $x_0, x_1, \ldots, x_{i-1}$ are distinct, then among the N possible choices for x_i, exacly $N-i$ of them are different from the previously chosen values. Hence the probability of getting a new value, assuming that the earlier were distinct, is $\frac{N-i}{N}$.

We can approximate the product (3.35) using the estimate

$$1 - t \approx e^{-t}, \qquad \text{valid for small values of } t.$$

(Compare with the proof of Theorem 3.36(b), and see also Exercise 3.32.) In practice, k will be approximately $\sqrt{N}$ and N will be large, so $\frac{i}{N}$ will indeed be small for $1 \le i < k$. Hence

$$\Pr\left(\begin{array}{c}x_0, x_1, \ldots, x_{k-1}\\ \text{are all different}\end{array}\right) \approx \prod_{i=1}^{k-1} e^{-i/N} = e^{-(1+2+\cdots+(k-1))/N} \approx e^{-k^2/2N}. \qquad (3.36)$$

For the last approximation we are using the fact that

$$1 + 2 + \cdots + (k-1) = \frac{k^2-k}{2} \approx \frac{k^2}{2} \quad \text{when } k \text{ is large.}$$

We now know the probability that $x_0, x_1, \ldots, x_{k-1}$ are all distinct. Assuming that they are distinct, what is the probability that the next choice x_k gives a match? There are k elements for it to match among the N possible elements, so this conditional probability is

$$\Pr\bigl(x_k \text{ is a match} \bigm| x_0, \ldots, x_{k-1} \text{ are distinct}\bigr) = \frac{k}{N}. \qquad (3.37)$$

Hence

$$\begin{aligned}\Pr\bigl(x_k \text{ is the first match}\bigr) \qquad&\\ &= \Pr\bigl(x_k \text{ is a match AND } x_0,\dots,x_{k-1} \text{ are distinct}\bigr)\\ &= \Pr\bigl(x_k \text{ is a match} \bigm| x_0,\dots,x_{k-1} \text{ are distinct}\bigr)\\ &\qquad\qquad\qquad\qquad \cdot \Pr\bigl(x_0,\dots,x_{k-1} \text{ are distinct}\bigr)\\ &\approx \frac{k}{N}\cdot e^{-k^2/2N} \qquad\qquad \text{from (3.36) and (3.37).}\end{aligned}$$

The expected number of steps before finding the first match is then given by the formula

$$E(\text{first match}) = \sum_{k\ge 1} k\cdot \Pr\bigl(x_k \text{ is the first match}\bigr) \approx \sum_{k\ge 1}\frac{k^2}{N}\cdot e^{-k^2/2N}. \qquad (3.38)$$

We want to know what this series looks like as a function of N. The following estimate derived using elementary calculus is helpful in estimating series of this sort.

Lemma 3.49. *Let $F(t)$ be a "nicely behaved" real valued function*[15] *with the property that $\int_0^\infty F(t)\,dt$ converges. Then for large values of n we have*

$$\sum_{k=1}^{\infty} F\left(\frac{k}{n}\right) \approx n\cdot\int_0^\infty F(t)\,dt. \qquad (3.39)$$

Proof. We start with the definite integral of $F(t)$ over an interval $0 \le t \le A$. By definition, this integral is equal to a limit of Riemann sums,

$$\int_0^A F(t)\,dt = \lim_{n\to\infty}\sum_{k=1}^{An} F\left(\frac{k}{n}\right)\cdot\frac{1}{n},$$

where in the sum we have broken the interval $[0, A]$ into An pieces. In particular, if n is large, then

$$n\cdot\int_0^A F(t)\,dt \approx \sum_{k=1}^{An} F\left(\frac{k}{n}\right).$$

[15] For example, it should suffice that F have a continuous derivative.

Now letting $A \to \infty$ yields (3.39). (We do not claim that this is a rigorous arguemnt. Our aim is merely to convey the underlying idea. The interested reader may supply the details needed to complete the argument and to obtain explicit upper and lower bounds.)

□

We use Lemma 3.49 to estimate

$$\begin{aligned} E(\text{first match}) &\approx \sum_{k\geq 1} \frac{k^2}{N} \cdot e^{-k^2/2N} && \text{from (3.38),} \\ &= \sum_{k\geq 1} F\left(\frac{k}{\sqrt{N}}\right) && \text{letting } F(t) = t^2 e^{-t^2/2}, \\ &\approx \sqrt{N} \cdot \int_0^\infty t^2 e^{-t^2/2}\, dt && \text{from (3.39) with } n = \sqrt{N}, \\ &\approx 1.2533 \cdot \sqrt{N} && \text{by numerical integration.} \end{aligned}$$

We can easily estimate it using a numerical method such as Simpson's rule. This completes the proof of (b), and combining (a) and (b) gives the final statement of Theorem 3.48. □

Remark 3.50. It is instructive to check numerically the accuracy of the estimates used in the proof of Theorem 3.48. In that proof, we claimed that for large values of N, the expected number of steps before finding a match is given by each of the following three formulas:

$E_1 = \sum_{k\geq 1} \frac{k^2}{N} \prod_{i=1}^{k-1} \left(1 - \frac{i}{N}\right)$	$E_2 = \sum_{k\geq 1} \frac{k^2}{N} e^{-k^2/2N}$	$E_3 = \sqrt{N} \int_0^\infty t^2 e^{-t^2/2}\, dt$

More precisely, E_1 is the exact formula, but hard to compute exactly if N is very large, while E_2 and E_3 are approximations. We have computed the values of E_1, E_2 and E_3 for some moderate sized values of N and compiled the results in Table 3.11. As you can see, E_2 and E_3 are quite close to one another, and once N gets reasonably large, they also provide a good approximation for E_1. Hence for very large values of N, say $2^{80} < N < 2^{160}$, it is quite reasonable to estimate E_1 using E_3.

N	E_1	E_2	E_3	E_1/E_3
100	12.210	12.533	12.533	0.97421
500	27.696	28.025	28.025	0.98827
1000	39.303	39.633	39.633	0.99167
5000	88.291	88.623	88.623	0.99626
10000	124.999	125.331	125.331	0.99735
20000	176.913	177.245	177.245	0.99812
50000	279.917	280.250	280.250	0.99881

Table 3.11: Expected number of steps until a ρ collision

3.5.2 Discrete logarithms via Pollard's ρ method

In this section we describe how to use Pollard's ρ method to solve the discrete logarithm problem

$$g^t = a \quad \text{in } \mathbb{F}_p.$$

The idea is to find a collision between $g^i a^j$ and $g^k a^\ell$ for some known exponents i, j, k, ℓ. Then $g^{i-j} = a^{\ell-j}$, and taking roots in $\mathbb{F}_p$ will more-or-less solve the problem of expressing a as a power of g.

The difficulty is finding a function $f : \mathbb{F}_p \to \mathbb{F}_p$ that is complicated enough to mix up the elements of $\mathbb{F}_p$, yet simple enough to keep track of its orbits. Pollard **??** suggests using the function

$$f(x) = \begin{cases} gx & \text{if } 0 \le x < p/3, \\ x^2 & \text{if } p/3 \le x < 2p/3, \\ ax & \text{if } 2p/3 \le x < p. \end{cases} \tag{3.40}$$

Note that x must be reduced modulo p into the range $0 \le x < p$ before doing the comparisons.

Remark 3.51. No one has proven that the function $f(x)$ given by (3.40) is sufficiently random to guarantee that Theorem 3.48 is true for f, but experimentally f works fairly well. On the other hand, Teske [59, 60] has shown that f is not sufficiently random to give optimal results and she gives examples of somewhat more complicated functions that work better in practice.

Consider what happens when we repeatedly apply the function f given by (3.40) to the starting point $x_0 = 1$. At each step we either multiply by g, multiply by a, or square the previous value. At each stage we end up with a power of g multiplied by a power of a, so say after i steps we have

$$x_i = (\underbrace{f \circ f \circ f \circ \cdots \circ f}_{i \text{ iterations of } f})(1) = g^{\alpha_i} \cdot a^{\beta_i}.$$

We cannot predict the values of α_i and β_i, but we can compute them at the same time that we are computing the x_i's using the definition (3.40) of f. Clearly $\alpha_0 = \beta_0 = 0$, and then subsequent values are given by

$$\alpha_{i+1} = \begin{cases} \alpha_i + 1 & \text{if } 0 \le x < p/3, \\ 2\alpha_i & \text{if } p/3 \le x < 2p/3, \\ \alpha_i & \text{if } 2p/3 \le x < p. \end{cases} \qquad \text{and} \qquad \beta_{i+1} = \begin{cases} \beta_i & \text{if } 0 \le x < p/3, \\ 2\beta_i & \text{if } p/3 \le x < 2p/3, \\ \beta_i + 1 & \text{if } 2p/3 \le x < p. \end{cases}$$

When computing α_i and β_i, it suffices to keep track of their values modulo $p-1$, since $g^{p-1} = 1$ and $a^{p-1} = 1$. This is important, otherwise the values of α_i and β_i would become prohibitively large.

In a similar fashion we compute the sequence given by

$$y_0 = 1 \qquad \text{and} \qquad y_{i+1} = f\bigl(f(y_i)\bigr).$$

Then

$$y_i = x_{2i} = g^{\gamma_i} \cdot a^{\delta_i},$$

where the exponents γ_i and δ_i can be computed by two repetitions of the same recursions used for α_i and β_i. Of course, the first time we use y_i to determine which case to use and the second time we use $f(y_i)$ to decide.

Applying the above procedure, we eventually find a collision in the x and the y sequences, say $y_i = x_i$. This means that

$$g^{\alpha_i} \cdot a^{\beta_i} = g^{\gamma_i} \cdot a^{\delta_i}.$$

So if we let

$$u \equiv \alpha_i - \gamma_i \pmod{p-1} \qquad \text{and} \qquad v \equiv \delta_i - \beta_i \pmod{p-1},$$

then $g^u = a^v$ in $\mathbb{F}_p$. Equivalently,

$$v \cdot \log_g(a) \equiv u \pmod{p-1}. \tag{3.41}$$

If $\gcd(v, p-1) = 1$, then we can multiply both sides of (3.41) by the inverse of v modulo $p-1$ to solve the discrete logarithm problem.

More generally, let $d = \gcd(v, p-1)$ and use the Extended Euclidean algorithm 1.11 to find an integer s so that

$$s \cdot v \equiv d \pmod{p-1}.$$

Multiplying both sides of (3.41) by s yields

$$d \cdot \log_g(a) \equiv w \pmod{p-1}, \tag{3.42}$$

where $w \equiv s \cdot u \pmod{p-1}$. In this congruence we know all of the quantities except for $\log_g(a)$. The fact that d divides $p-1$ will force d to divide w, so w/d is one solution to (3.42), but there are others. The full set of solutions to (3.42) is obtained by starting with w/d and adding multiples of $(p-1)/d$,

$$\log_g(a) \in \left\{ \frac{w}{d} + k \cdot \frac{p-1}{d} : k = 0, 1, 2, \ldots, d-1 \right\}.$$

In practice, d will tend to be fairly small,[16] so it suffices to check each of the d possibilities for $\log_g(a)$ until finding the correct value.

Example 3.52. We illustrate Pollard's ρ method by solving the discrete logarithm problem

$$19^t \equiv 24717 \pmod{48611}.$$

The first step is to compute the x and y sequences until finding a match $y_i = x_i$, while also computing the exponent sequences $\alpha, \beta, \gamma, \delta$. The initial stages of this process and the final few steps before finding a collision are given in Table 3.12.

From the table we see that $x_{688} = x_{344} = 23809$ in $\mathbb{F}_{48611}$. The associated exponent values are

$$\alpha_{344} = 15568, \qquad \beta_{344} = 628, \qquad \gamma_{344} = 13628, \qquad \delta_{344} = 15128,$$

so we know that

$$19^{15568} \cdot 24717^{628} = 19^{13628} \cdot 24717^{15128} \quad \text{in } \mathbb{F}_{48611}.$$

[16] For most cryptographic applications, the prime p is chosen so that $p-1$ has very few small prime factors, since otherwise the Pohlig-Hellman algorithm 2.23 may be applicable. And it is unlikely that d will be divisible by a large prime factor of $p-1$.

i	x_i	y_i	α_i	β_i	γ_i	δ_i
0	1	1	0	0	0	0
1	19	361	1	0	2	0
2	361	33464	2	0	4	1
3	6859	13882	3	0	5	2
4	33464	14974	4	1	13	4
5	13523	22469	4	2	28	10
6	13882	12845	5	2	112	40
7	11022	14714	12	4	228	80
8	14974	1020	13	4	458	162
9	18931	8800	14	5	919	324
$\vdots$						
338	4320	31385	8020	30457	2490	8590
339	39886	14053	8021	30458	4981	17180
340	31182	38751	8021	30459	9965	34362
341	8284	4320	16042	12309	19930	20117
342	11563	31182	16043	12309	19931	20119
343	1326	11563	32088	24619	39863	40239
344	23809	23809	15568	628	13628	15128

Table 3.12: Pollard ρ computations to solve $19^t \equiv 24717$ in $\mathbb{F}_{48611}$

(We should probably check this equality before proceeding to make sure that we haven't made an arithmetic error.) Moving the powers of 19 to one side and the powers of 24717 to the other side yields

$$19^{1940} = 24717^{14500} \quad \text{in } \mathbb{F}_{48611}. \tag{3.43}$$

We next observe that

$$\gcd(14500, 48610) = 10 \qquad \text{and} \qquad 1874 \cdot 14500 \equiv 10 \pmod{48610}.$$

Raising both sides of (3.43) to the 1874th power yields

$$19^{1940 \cdot 1874} = 19^{3635560} = 19^{38420} = 24717^{10} \quad \text{in } \mathbb{F}_{48611}.$$

Hence

$$10 \cdot \log_{19}(24717) \equiv 38420 \pmod{48610},$$

which means that

$$\log_{19}(24717) \equiv 3842 \pmod{4861}.$$

The possible values for the discrete logarithm are obtained by adding multiples of 4861 to 3842, so $\log_{19}(24717)$ is one of the numbers in the set

$$\{3842, 8703, 13564, 18425, 23286, 28147, 33008, 37869, 42730, 47591\}.$$

To complete the solution, we compute 19 raised to each of these 10 values until we find the one that is equal to 24717:

$$\begin{aligned} 19^{3842} &= 16580, & 19^{8703} &= 29850, & 19^{13564} &= 23894, & 19^{18425} &= 20794, \\ 19^{23286} &= 10170, & 19^{28147} &= 32031, & 19^{33008} &= 18761, & 19^{37869} &= \boxed{24717}. \end{aligned}$$

This gives the solution $\log_{19}(24717) = 37869$. We check our answer

$$19^{37869} = 24717 \quad \text{in } \mathbb{F}_{48611}. \quad \checkmark$$

3.6 Information theory [M]

In 1948 and 1949, Claude Shannon published two papers [50, 51] that form the mathematical foundation of modern cryptography. In these papers he defines the concept of perfect (or unconditional) secrecy, introduces the idea of entropy of natural language and statistical analysis, provides the first proofs of security using probability theory, and gives precise connections between provable security and the size of the key, plaintext and ciphertext spaces.

In public key cryptography, one is interested in how computationally difficult it is to break the system. The issue of security is thus a relative one—a given cryptosystem is hard to break if one assumes that some underlying problem is hard to solve. It requires some care to formulate these concepts properly. In this section we briefly introduce Shannon's ideas and explain their relevance to symmetric key systems. In [51], Shannon develops a theory of the security of cryptosystems assuming no bounds on the computational resources that may be brought to bear against it. For example, the symmetric ciphers introduced in Chapter 1, such as the simple substitution cipher (Section 1.1) and the Vigénere cipher (Section 3.2) are not computationally secure. With unlimited resources—indeed with very limited resources—an adversary can easily break these ciphers. If we seek unconditional security, we must either seek new algorithms or modify the implementation of known algorithms. In fact, Shannon shows that perfectly secure cryptosystems must have at least as many keys as plaintexts and that every key must be used with equal probability. This means that most practical cryptosystems are not unconditionally secure. In [50] Shannon develops a mathematical theory that measures the amount of information (entropy) revealed by a random variable. When these random variables represent the different possible plaintexts, ciphertexts, or keys of a particular cipher used to encrypt some natural language, we obtain a framework for the rigorous mathematical study of the security of cryptosystems.

3.6.1 Perfect secrecy

A cryptosystem has perfect security if the interception of a ciphertext (the encryption of a message) gives the cryptanalyst no new information about this message or any future encrypted messages. To formalize this concept, we introduce random variables M, C and K representing the (finite number of) possible messages, ciphertexts and keys. In other words, M is a random variable whose values are the possible messages (plaintexts), C is a random variable whose values are the possible ciphertexts, and K is a random variable whose values are the possible keys used for encryption and decryption. We let f_M, f_C, and f_K be the associated density functions.

We also have the joint densities and the conditional densities of all pairs of these random variables, such as $f_{C|M}$ and $f_(M|C)$ and so forth. We will let the variable names simplify the notation. For example, we write $f(c|m)$ for $f_{C|M}(c|m)$, the conditional probability density of the random variables C and M, i.e.

$$f(c|m) = \Pr(C = c \text{ given that } M = m).$$

Similarly, we write $f(m)$ for $f_M(m)$, the probability that $M = m$.

Definition. A cryptosystem is *perfectly secure* if

$$f(m|c) = f(m) \qquad \text{for all } m \in \mathcal{M} \text{ and all } c \in \mathcal{C}. \tag{3.44}$$

What does (3.44) mean? It says that the probabilty of any particular plaintext, $\Pr(M = m)$, is independent of the ciphertext. Intuitively, this means that the ciphertext reveals no knowledge of the plaintext.

Bayes' Formula (Theorem 3.31) says that $f(m|c)f(c) = f(c|m)f(m)$, which implies that perfect secrecy is equivalent to the condition

$$f(c|m) = f(c). \tag{3.45}$$

Formula (3.45) says that the appearance of any particular ciphertext is equally likely, independent of the plaintext.

If we know f_K and f_M, then f_C is determined. To see this, we note that if a given key k is used, then the probability that the ciphertext is c is equal to the probability that the decryption of c is the plaintext. This allows us to compute the total probability $f_C(c)$ by summing over all possible keys and using the decomposition formula (3.20) of Proposition 3.21, or more precisely, its generalization described in Exercise 3.21. Thus

$$f_C(c) = \sum_{k \in \mathcal{K}} f_K(k) f_M\bigl(d_k(c)\bigr),$$

where recall that $\mathcal{K}$ denotes the set of all possible keys and d_k denotes the decryption function $d_k : \mathcal{C} \to \mathcal{M}$ for the key k.

The Shift Cipher described in Section 1.1, with 26 equally probable keys and implemented to encrypt every character with a different randomly chosen key, is perfectly secure. (See Exercise 3.39.)

Recall that an encryption function is one-to-one, meaning that each message gives rise to a unique ciphertext. This implies that there are at least as many ciphertexts as plaintexts (messages). But perfect secrecy has other implications for the relative size of the key, message and ciphertext space. Let's first explore an example of a (trivial) cryptosystem which is not perfectly secure.

Example 3.53. Suppose that a cryptosystem has two keys k_1 and k_2, three messages m_1, m_2 and m_3, and three ciphertexts c_1, c_2, and c_3. Assume that the message random variable has density

$$p(m_1) = p(m_2) = \frac{1}{4} \qquad \text{and} \qquad p(m_3) = \frac{3}{4}. \tag{3.46}$$

Table 3.13 describes how the different keys act on the messages to produce ciphertexts.

For example, the encryption of the plaintext m_1 with the key k_1 is the ciphertext c_2. Under the assumption that the keys are used with equal prob-

	m_1	m_2	m_3
k_1	c_2	c_1	c_3
k_2	c_1	c_3	c_2

Table 3.13: Encryption of messages with keys k_1 and k_2.

ability, we compute the probability that the ciphertext is c_1.

$$\begin{aligned} f(c_1) &= f(k_1)f_M(d_{k_1}(c_1)) + f(k_2)f(d_{k_2}(c_1)) \\ &= f(k_1)f(m_2) + f(k_2)f(m_1) \\ &= \frac{1}{2}\cdot\frac{1}{4} + \frac{1}{2}\cdot\frac{1}{4} = \frac{1}{4}. \end{aligned}$$

Now observe that $f(c_1|m_3) = 0$, so this cryptosystem is not perfectly secure. It is intuitively obvious that viewing a ciphertext leaks information about the plaintext: when we see c_1, we know that the message was either m_1 or m_2, it cannot be m_3. Exercise 3.38 asks you to compute some other densities associated to this system.

As we have noted, the number of ciphertexts must be at least as large as the number of plaintexts, since otherwise decryption is not possible. It turns out that one consequence of perfect secrecy is that the number of possible keys must also be at least as large as the number of possible ciphertexts.

Proposition 3.54. *If a cryptosystem has perfect security, then $\#\mathcal{K} \geq \#\mathcal{M}$.*

Proof. We start by fixing some ciphertext $c \in \mathcal{C}$ with $f(c) > 0$. Perfect security tells us that

$$f(c|m) = f(c) > 0 \qquad \text{for all } m \in \mathcal{M}.$$

This says that there is a positive probability that m encrypts to c, so in particular there is at least one key k satisfying $e_k(m) = c$. Further, if we start with a different plaintext m', then we get a different key k', since otherwise $e_k(m) = c = e_k(m')$ would contradict the injectivity of e_k.

To recapitulate, we have shown that for every $m \in \mathcal{M}$, the set

$$\{k \in \mathcal{K} : e_k(m) = c\}$$

is nonempty, and further that for different m's, these sets are disjoint. Thus each plaintext $m \in \mathcal{M}$ is matched with one or more keys distinct keys, and different m's get matched with different keys, which shows that there must be at least as many keys as there are plaintexts. $\square$

Given the restriction on the relative sizes of the key, ciphertext, and plaintext spaces in perfectly secure systems, namely

$$\#\mathcal{K} \geq \#\mathcal{M} \qquad \text{and} \qquad \#\mathcal{C} \geq \#\mathcal{M},$$

it is most efficient to assume that the the key space, the plaintext space, and the ciphertext space are all of equal size. Assuming this, Shannon proves a theorem characterizing perfect secrecy.

Theorem 3.55. *Suppose that a cryptosystem satisfies*

$$\#\mathcal{K} = \#\mathcal{M} = \#\mathcal{C},$$

i.e. the number of keys, plaintexts, and ciphertexts are all equal. Then the system has perfect secrecy if and only if the following two conditions hold:
(a) *Each key $k \in \mathcal{K}$ is used with equal probability.*
(b) *For a given message $m \in \mathcal{M}$ and ciphertext $c \in \mathcal{C}$, there is exactly one key $k \in \mathcal{K}$ that encrypts m to c.*

Proof. Suppose first that a cryptosystem has perfect security. We start by verifying (b). For any plaintext $m \in \mathcal{M}$ and ciphertext $c \in \mathcal{C}$, consider the (possibly empty) set of keys

$$\mathcal{S}_{m,c} = \bigl\{k \in \mathcal{K} : e_k(m) = c\bigr\}$$

that encrypt m to c. We are going to prove that if the cryptosystem has perfect security, then in fact $\#\mathcal{S}_{m,c} = 1$ for every $m \in \mathcal{M}$ and every $c \in \mathcal{C}$, which is equivalent to statement (b) of the theorem. We do this in three steps.

Claim 1. If $m \neq m'$, then $\mathcal{S}_{m,c} \cap \mathcal{S}_{m',c} = \emptyset$.
Suppose that $k \in \mathcal{S}_{m,c} \cap \mathcal{S}_{m',c}$. Then $e_k(m) = c = e_k(m')$, which implies that $m = m'$ since the encryption function e_k is injective. This proves Claim 1.

Claim 2. If the cryptosystem has perfect security, then $\mathcal{S}_{m,c}$ is nonempty for every m and c.

We use the perfect security assumption in the form $f(m,c) = f(m)f(c)$. We know that every $m \in \mathcal{M}$ is a valid plaintext for at least one key, so $f(m) > 0$, and similarly, every $c \in \mathcal{C}$ appears as the encryption of at least one plaintext using some key, so $f(c) > 0$. Hence perfect security implies that

$$f(m,c) > 0 \qquad \text{for all } m \in \mathcal{M} \text{ and all } c \in \mathcal{C}. \tag{3.47}$$

However, $f(m,c) > 0$ is simply another way of saying that c is a possible encryption of m. Hence there must be at least one key $k \in \mathcal{K}$ satisfying $e_k(m) = c$, i.e. there is some key $k \in \mathcal{S}_{m,c}$. This completes the proof of Claim 2.

Claim 3. If the cryptosystem has perfect security, then $\#\mathcal{S}_{m,c} = 1$.
Fix a ciphertext $c \in \mathcal{C}$. Then

$$\begin{aligned}
\#\mathcal{K} &\geq \#\biggl(\bigcup_{m\in\mathcal{M}} \mathcal{S}_{m,c}\biggr) && \text{since } \mathcal{K} \text{ contains every } \mathcal{S}_{m,c}, \\
&= \sum_{m\in\mathcal{M}} \#\mathcal{S}_{m,c} && \text{since the } \mathcal{S}_{m,c} \text{ are disjoint from Claim 1,} \\
&\geq \#\mathcal{M} && \text{since } \#\mathcal{S}_{m,c} \geq 1 \text{ from Claim 2,} \\
&= \#\mathcal{K} && \text{since } \#\mathcal{K} = \#\mathcal{M} \text{ by assumption.}
\end{aligned}$$

Thus all of these inequalities are equalities, so in particular,

$$\sum_{m\in\mathcal{M}} \#\mathcal{S}_{m,c} = \#\mathcal{M}.$$

Then the fact (Claim 2) that every $\#\mathcal{S}_{m,c} \geq 1$ implies that $\#\mathcal{S}_{m,c}$ must equal 1. This completes the proof of Claim 3.

As noted above, Claim 3 is equivalent to statement (b) of the Theorem. We turn now to statement (a). Consider the set of triples

$$(k, m, c) \in \mathcal{K} \times \mathcal{M} \times \mathcal{C} \quad \text{satisfying} \quad e_k(m) = c.$$

Clearly k and m determine a unique value for c, and (b) says that m and c determine a unique value for k. It is also not hard (using the assumption that $\#\mathcal{M} = \#\mathcal{C}$) to show that c and k determine a unique value for m, see Exercise 3.40.

For any triple (k, m, c) satisfying $e_k(m) = c$, we compute

$$\begin{aligned} f(m) &= f(m|c) && \text{by perfect security,} \\ &= \frac{f(m,c)}{f(c)} && \text{definition of conditional probability,} \\ &= \frac{f(m,k)}{f(c)} && \text{since any two of } m,\ k,\ c \text{ determines the third,} \\ &= \frac{f(m)f(k)}{f(c)} && \text{since } M \text{ and } K \text{ are independent}^{17} \end{aligned}$$

Canceling $f(m)$ from both sides, we have shown that

$$f(k) = f(c) \qquad \text{for every } k \in \mathcal{K} \text{ and every } c \in \mathcal{C}. \tag{3.48}$$

Note that our proof shows that (3.48) is true for every k and every c, because Exercise 3.40 tells us that for every (k, c) there is a (unique) m satisfying $e_k(m) = c$.

We sum (3.48) over all $c \in \mathcal{C}$ and divide by $\#\mathcal{C}$ to obtain

$$f(k) = \frac{1}{\#\mathcal{C}} \sum_{c \in \mathcal{C}} f(c) = \frac{1}{\#\mathcal{C}}.$$

This shows that $f(k)$ is constant, independent of the choice of $k \in \mathcal{K}$, which is precisely the assertion of (a). At the same time we have proven the useful fact that $f(c)$ is constant, i.e. every ciphertext is used with equal probability.

In the other direction, if a cryptosystem has properties (a) and (b), then the steps outlined to prove perfect security of the shift cipher in Exercise 3.39 can be applied in this more general setting. We leave the details to the reader. □

Example 3.56 (The one-time pad). Vernam's one-time pad, patented in 1917, is an extremely simple, perfectly secure, albeit very inefficient, cryptosystem. The key k, which consists of a string of binary digits $k_0k_1 \cdots k_N$, is used to encrypt a binary plaintext string $m = m_0m_1 \cdots m_N$ using bit-by-bit XOR, which we denote by $\oplus$. (See (1.11) on page 39 for a description of the XOR operation.) Thus the ciphertext $c = c_0c_1 \cdots c_N$ is given by

$$c_i = k_i \oplus m_i \quad \text{for } i = 0, 1, \ldots, N.$$

Each key is *used only once* and then discarded, whence the name of the system. Since every key is used with equal probability, and since there is exactly one key that encrypts a given m to a given c, namely the key $m \oplus c$, Theorem 3.55 shows that Vernam's one-time pad has perfect security.

However, notice that in order to use a Vernam one-time pad to exchange N bits of information, Bob and Alice must already know N bits of shared secret information to use as the key. This makes one-time pads too inefficient for most large scale communication networks, but there are situations where they have been used, for example for communications between diplomatic offices or between spies and their home bases.

It is also worth noting that a one-time pad only remains completely secure as long as its keys are never reused. When a key pad is used more than once, either due to error or to the difficulty of providing enough key material, then the cryptosystem may be vulnerable to cryptoanalysis. This occurred in the real world when the Soviet Union reused some one-time pads during World War II and the United States mounted a massive cryptanaysis effort called the VERONA project that successfully decrypted a number of documents.

3.6.2 Entropy

In efficient cryptosystems, a single key must be used to encrypt many different plaintexts, so perfect security is not possible. At best, we can hope to build cryptosystems that are compuationally secure. Unfortunately, anything less than perfect security leaves open the possibility that a list of ciphertexts will reveal signficant information about the key. To study this phenomenon, Shannon introduced the concept of *entropy* in order to quantify the uncertainty of the outcome of an experiment.

Here an experiment consists of certain events, each of which occurs with a given probability. Thus the outcome of is a random variable X, and we want to measure the uncertainty of the value of X. Obviously the uncertainty depends on the probabilities of the various possible values of X. For concreteness, say that X takes on the finitely many values say $x_1, x_2, \ldots, x_n$ and let $p_1, p_2, \ldots, p_n$ denote the respective probabilities of these values, i.e. $f_X(x_i) = p_i$.

The entropy of X, denoted $H(X)$, is a number that will depend only on the probabilities $p_1, \ldots, p_n$ of the possible outcomes of X, so we write[18]

$$H(X) = H(p_1, \ldots, p_n),$$

We would like to capture the idea that H is the expected value of a random variable that measures the uncertainty that the outcome x_i has occurred, or to put it another way, that measures the amount of information about X revealed by the outcome of an experiment. What properties should H possess?

Property H$_1$. The function H should be continuous in the variables p_i. This reflects the intuition that a small change in p_i should produce a small change in the amount of information revealed by X.

Property H$_2$. Let X_n be the random variable that is uniformly distributed on a set $\{x_1, \ldots, x_n\}$, i.e. the random variable X_n has n possible outcomes, each occurring with probability $\frac{1}{n}$. Then $H(X_n)$ is a monotonically increasing function of n. This reflects the intuition that if all events are equally likely, then the uncertainty increases as the number of events increases.

Property H$_3$. If an outcome of X is thought of as a choice, and if that choice can be broken down into two successive choices, then the original H should be a weighted sum of the individual values of H. (See Example 3.57.) In particular, this implies that $H(X_{n^r}) = r \cdot H(X_n)$, where as above, X_n is uniformly distributed on n objects.

Example 3.57. Properties $\mathbf{H}_1$ and $\mathbf{H}_2$ are not complicated, but the statement of Property $\mathbf{H}_3$ is less clear, so we illustrate with an example. Suppose that X is a random variable with three outcomes $\{x_1, x_2, x_3\}$ having respective probabilities $\{\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\}$. Thus the three outcomes for X are illustrated by the branched tree in Figure 3.2(a).

Now suppose that X is written as two successive choices, first deciding between x_1 and y_1, and then, if the first choice was y_1, making a second choice between x_2 and x_3. So we have two random variables Y and Z, where Y is given by

$$\Pr(Y = x_1) = \frac{1}{2} \qquad \text{and} \qquad \Pr(Y = y_1) = \frac{1}{2},$$

[18] Although this notation is useful, it is important to remember that the domain of H is the set of random variables, not the set of n-tuples for some fixed value of n. Thus the domain of H is itself a set of functions.

(a) Three outcomes of a choice (b) Splitting into two choices

Figure 3.2: Splitting X into Y followed by Z

while Z depends on Y and is given by

$$\begin{aligned} \Pr(Z = x_2 | Y = x_1) &= 0, & \Pr(Z = x_3 | Y = x_1) &= 0, \\ \Pr(Z = x_2 | Y = x_2) &= \frac{2}{3}, & \Pr(Z = x_3 | Y = x_2) &= \frac{1}{3}. \end{aligned}$$

Then X is the composition of first using Y to make a choice, and then using Z to make a choice, as illustrated by Figure 3.2(b).

Property $\mathbf{H}_3$ of entropy, as applied to this example, says that

$$H\left(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\right) = H\left(\frac{1}{2}, \frac{1}{2}\right) + \underset{\uparrow}{\frac{1}{2}} H\left(\frac{2}{3}, \frac{1}{3}\right)$$

This $\frac{1}{2}$ reflects the fact that the second choice only occurs $\frac{1}{2}$ of the time.

Theorem 3.58. *Every function having Properties* $\mathbf{H}_1$*,* $\mathbf{H}_2$ *and* $\mathbf{H}_3$ *is a constant multiple of the function*

$$H(p_1, \dots, p_n) = -\sum_{i=1}^{n} p_i \log p_i, \tag{3.49}$$

where log *denotes the logarithm to the base* 2*, and if* $p = 0$*, then we set* $p \log p = 0$.[19]

To illustrate the notion of uncertainly, consider what happens when one of the probabilities $p_i = 1$ and the other probabilities are zero. In this case, the formula (3.49) for entropy gives $H(p_1, \ldots, p_n) = 0$, which makes sense, since there is no uncertainty about the outcome of an experiment having this probability density.

It turns out that the other extreme, namely maximal uncertainly, occurs when all of the probabilities p_i are equal. In order to prove this, we use an important inequality from real analysis known as Jensen's inequality.

Definition. A function F on the real line is called *concave* on an interval I if the following inequality is true for for all $0 \le \alpha \le 1$ and all s and t in I:

$$(1-\alpha)F(s) + \alpha F(t) \le F\big((1-\alpha)s + \alpha t\big). \tag{3.50}$$

This definition may seem mysterious, but it has a simple geometric interpretation. Notice that if we fix s and t and let a vary from 0 to 1, then the points $(1-a)s + at$ trace out the interval from s to t on the real line. So the inequality 3.50 is the geometric statement that the line segment connecting any two points on the graph of F lies below the graph of F. For example, the function $F(t) = 1 - t^2$ is concave. If the function F has a second derivative, then the second derivative test that you learned in calculus can be used to test for concavity (Exercise 3.45).

Theorem 3.59 (Jensen's Inequality)**.** *Suppose that* F *is concave on an interval* I*, and let* $\alpha_1, \alpha_2, \ldots, \alpha_n$ *be nonnegative numbers satisfying* $\alpha_1 + \alpha_2 + \cdots + \alpha_n = 1$*. Then*

$$\sum_{i=1}^{n} \alpha_i F(t_i) \le F\Big(\sum_{i=1}^{n} \alpha_i t_i\Big) \qquad \textit{for all } t_1, t_2, \ldots, t_n \in I. \tag{3.51}$$

Further, equality holds in (3.51) *if and only if* $t_1 = t_2 = \cdots = t_n$*.*

Proof. Notice that for $n = 2$, the desired inequality (3.51) is exactly the definition of concavity (3.50). The general case can then be proven by induction, see Exercise 3.46. □

[19] This convention makes sense, since we want H to be continuous in the p_i's, and it's true that $\lim_{p \to 0} p \log p = 0$.

Corollary 3.60. *Let X be an experiment* (*a random variable*)*. Then*

(a) $H(X) \le \log n$.

(b) $H(X) = \log n$ *if and only if every outcome* (*i.e. every individual event x_i*) *occurs with the same probability $\frac{1}{n}$.*

Proof. Let X take on the values $x_1, \ldots, x_n$ with probabilities $p_1, \ldots, p_n$. Then $p_1 + \cdots + p_n = 1$, so we may apply Jensen's Inequality to the concave function $F(t) = \log t$ (see Exercise 3.45), taking $\alpha_i = p_i$ and $t_i = \frac{1}{p_i}$. The lefthand side of (3.51) is exactly the formula for entropy (3.49), so we find that

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i = \sum_{i=1}^{n} p_i \log \frac{1}{p_i} \le \log\left(\sum_{i=1}^{n} p_i \frac{1}{p_i}\right) \le \log n.$$

This proves (a). Further, equality occurs if and only if $p_1 = p_2 = \cdots = p_n$, i.e. if all of the probabilities satisfy $p_i = \frac{1}{n}$. This proves (b). □

Notice that Corollay 3.60 says that entropy is maximized when all of the probabilities are equal. This conforms to our intuitive understanding that uncertainty is maximized when every outcome is equally likely.

The theory of entropy is applied to cryptography by computing the entropy of the random variables such as K, M, and C associated to the cryptosystem and comparing the actual values with the maximum possible values. Clearly the more entropy the better, since increased uncertainty makes the cryptanalysts job harder.

For instance, consider the key random variable K associated to a shift cipher. It has 26 possible values, since the shift may be any integer between 0 and 25, and each shift amount equally probable, so K has maximal entropy $H(K) = \log 26$.

Example 3.61. We consider the system with two keys described in Example 3.53 on page 147. Each key is equally likely, so $H(K) = \log 2 = 1$. Similarly, we can use the plaintext probabilities for this system as given by (3.46) to compute the entropy of M,

$$H(M) = -\frac{1}{4}\log\left(\frac{1}{4}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right) - \frac{1}{2}\log\left(\frac{1}{2}\right) = \frac{3}{2} \approx 0.585.$$

Notice that $H(M)$ is considerably smaller than $\log 3 \approx 1.585$, which would be the maximal possible entropy for M in a cryptosystem with three plaintexts.

Suppose that a signal is sent over a noisy channel, that is, the signal may be distorted during transmission. Shannon [50] defines the conditional entropy of the original signal given the received signal. He uses this quantity, which he calls the *equivocation*, to measure the amount of uncertainty in transmissions across a noisy channel. Shannon [51] later observes that a noisy communication channel is also a model for a secrecy system: the original signal (the plaintext) is "distorted" by applying the encryption process, and the received signal (the ciphertext) is thus a noisy version of the original signal. In this way, the notion of equivocation can be applied to cryptography.

Definition. Let X and Y be random variables. The *equivocation*, or *conditional entropy*, $H(X|Y)$ is defined to be

$$H(X|Y) = -\sum_{x,y} f_Y(y) f_{X|Y}(x|y) \log f_{X|Y}(x|y),$$

where the sum is over all possible values x taken on by X and all possible values y taken on by Y.

When $X = K$ is the key random variable and $Y = C$ is the ciphertext random variable, the quantity $H(K|C)$ is called the *key equivocation*. It measures the total amount of information about the key revealed by the ciphertext, or more precisely, it is the expected value of the conditional entropy $H(K|c)$ of K given a single observation c of C. This quantity can be found by computing all of the conditional probabilities $f(k|c)$ of the cryptosystem, or by using the formula

$$H(K|C) = H(K) + H(M) - H(C), \tag{3.52}$$

whose proof we omit.

Example 3.62. We compute the key equivocation of the cryptosystem described in Examples 3.53 and 3.61. We already computed $H(K) = 1$ and $H(M) = \frac{3}{2}$, so it remains to compute $H(C)$. To do this, we need the values of $f(c)$ for each ciphertext $c \in \mathcal{C}$. We already computed $f(c_1) = \frac{1}{4}$, and a similar computation using (3.46) and Table 3.13 yields

$$\begin{aligned} f(c_2) &= f(k_1)f(m_1) + f(k_2)f(m_3) = \left(\frac{1}{2}\right)\left(\frac{1}{4}\right) + \left(\frac{1}{2}\right)\left(\frac{1}{2}\right) = \frac{3}{8}, \\ f(c_3) &= f(k_1)f(m_3) + f(k_2)f(m_2) = \left(\frac{1}{2}\right)\left(\frac{1}{2}\right) + \left(\frac{1}{2}\right)\left(\frac{1}{4}\right) = \frac{3}{8}. \end{aligned}$$

Therefore,

$$H(C) = -\frac{1}{4}\log\left(\frac{1}{4}\right) - 2\cdot\frac{3}{8}\log\left(\frac{3}{8}\right) = \frac{1}{2} + \frac{3}{4}\log\left(\frac{8}{3}\right) \approx 2.39,$$

and hence using (3.52), we find that

$$H(K|C) = H(K) + H(M) - H(C) \approx 1 + 1.5 - 2.39 \approx 0.11.$$

This is quite low, which confirms our intution that in this cryptosystem, an average ciphertext reveals a significant amount of information about the key.

3.6.3 Redundancy and the entropy of natural language

In modern ciphers, the plaintext typically looks like a random string of bits. But if the plaintext is instead a natural language, then nearby letters or bits are heavily dependent on one another, rather than looking random. For example, correlations between successive letters (digrams or trigrams) can aid the cryptanalyst, as we saw when cryptanalyzing a simple substitution cipher in Section 1.1. In this section we use the notion of entropy to quantify the redundancy inherent in a natural language.

We start by approximating the entropy of a single letter in English text. Let L denote the random variable whose values are the letters of the English language $\mathtt{E}$ with their associated probability distribution as given in Table 1.3 on page 6. For example, the table says that

$$f_L(\mathtt{A}) = 0.0815, \quad f_L(\mathtt{B}) = 0.0144, \quad f_L(\mathtt{C}) = 0.0276, \quad \ldots \quad f_L(\mathtt{Z}) = 0.0008.$$

We can use the values in Table 1.3 to compute the entropy of a single letter in English text,

$$H(L) = 0.0815\log(0.0815) + \cdots + 0.0008\log(0.0008) \approx 4.132.$$

If every letter were equally likely, the entropy would be $\log 26 \approx 4.7$. The fact that the entropy is only 4.132 shows that some letters in English are more prevelant than others.

We want to use the concept of entropy to measure the average number of bits of information conveyed per letter of a language. The value of $H(L)$ that we computed does reveal some redundancy, it says that a letter conveys only 4.132 bits of information on average, although it takes 4.7 bits on average to specify a letter in the English alphabet.

The fact that natural languages contain redundancy is obvious. For example, most people have no trouble reading the following sentence, despite the fact that we have removed almost 40% of the letters:

Th prblms o crptgry nd scrcy sysms frnsh n ntrstng aplcatn o comm thry.

However, the entropy $H(L)$ of a single letter does not taken into account correlations between nearby letters, so it alone cannot give us a good approximation for the redundancy of the English language E. As a first step, we might take into account the correlations between pairs of letters (bigrams). Let L^2 denote the random variable whose values are pairs of English letters as they appear in typical English text. Some bigrams appear fairly frequently, for example $f_{L^2}(\mathtt{TH}) = .00315$ and $f_{L^2}(\mathtt{AN}) = .00172$. Others, such as JX and DX, never occur. Just as Table 1.3 was created experimentally by counting the letters in a long sample text, we can create a frequency table of bigrams and use it to obtain an experimental value for L^2. This leads to a value of $H(L^2) \approx 7.12$, so on average, each letter of E has entropy equal to half this value, namely 3.56. Continuing, we could experimentally compute the entropy of L^3, which is the random variable whose values are trigrams (triples of letters), and then $\frac{1}{3}H(L^3)$ would be an even better approximation to the entropy of E. However, we would need to analyze a lot of text to get a reliable estimate for trigram frequencies, and the problem becomes even harder as we look at L^4, L^5, L^6 and so on. However, this idea leads to the following important concept.

Definition. Let L be a language (e.g. English or French or C++), and for each $n \geq 1$, let L^n denote the random variables whose values are strings of n consecutive characters of L. The *entropy* of L is defined to be the quantity

$$H(\mathsf{L}) = \lim_{n\to\infty} \frac{H(L^n)}{n}.$$

Although it is not possible to precisely determine the entropy of the English language E, experimentally it appears that

$$1.0 \leq H(\mathsf{E}) \leq 1.5.$$

This means that despite the fact that it requires almost five bits to represent each of the 26 letters used in English, each letter conveys less than one and a half bits of information. Thus English is approximately 70% redundant![20]

3.6.4 The algebra of secrecy systems

We make only a few brief remarks about the algebra of cryptosystems. In [51], Shannon considers ways of building new cryptosystems from individual ones through algebraic combinations. The new systems are described in terms of linear combinations and products of the orginal encryption transformations.

Example 3.63 (Summation Systems). If R and T are two secrecy systems, then Shannon defines a "weighted sum" of R and T as $S = pR + qT$, where $p+q = 1$. More precisely, one first makes a choice of R and T, with probability p of choosing R and q of choosing T. Imagine flipping a (possibly unfair) coin resulting in a choice of system. This choice is part of the key of S, which must then also specify which key of T or R is to be used. This notion of summation extends to the sum of any number of secrecy systems. The systems R and T need to have the same message space, but they need not act on messages in a similar way. The system R could be a subtstitution cipher, and T could be a transposition cipher. Suppose T_i encrypts a letter of the alphabet by tranposition by i places. Then the system which encrypts by choosing a transposition at random, then encrypting according to that rule can be represented as $\sum_{i=1}^{26} p_i T_i$ where each $p_i = 1/n$.

Example 3.64 (Product Systems). One can also define the product of two cryptosystems if the ciphertexts of the first system are plaintexts for the second system. Thus let $e : \mathcal{M} \to \mathcal{C}$ be the encryption function for the first system and let $e' : \mathcal{M} \to \mathcal{C}'$ be the encryption function for the second system. Suppose further that $\mathcal{C} = \mathcal{M}'$, or more generally, that $\mathcal{C} \subseteq \mathcal{M}'$. Then the product system $e'e$ is defined to be the composition of e and e',

$$(e'e) : \mathcal{M} \xrightarrow{e} \mathcal{C} \subseteq \mathcal{M}' \xrightarrow{e'} \mathcal{C}'.$$

Product ciphers provide a means to strengthen security. They were used in the development of DES [38], the first national standard for symmetric

[20] This does not mean that one can remove 70% of the letters and still have an intelligible message. What it means is that in principle, it is possible to take a long message that requires 4.7 bits to specify each letter and to compress it into a form that takes only 30% as many bits.

encryption. DES features several rounds of S-box encryption, so it is a multiple product of a cipher with itself. Further, each round consists of the composition of several different transformations. The use of product ciphers continues to be of importance in the development of new symmetric ciphers, including AES, the Advanced Encryption Standard [37].

Exercises

Section 3.1. Basic principles of counting

3.1. The Rhind papyrus is an ancient Egyptian mathematical manuscript that is more than 3500 years old. Problem 79 of the Rhind papyrus poses a problem that can be paraphrased as follows: There are seven houses; in each house lives seven cats; each cat kills seven mice; each mouse has eaten seven spelt[21]; each spelt would have produced seven hekat[22] of barley. What is the sum of all of the named items? Solve this 3500 year old problem.

3.2. (a) How many n-tuples $(x_1, x_2, \dots, x_n)$ are there if the coordinates are required to be integers satisfying $0 \le x_i < q$?

(b) Same question as (a), except now there are separate bounds $0 \le x_i < q_i$ for each coordinate.

(c) How many n-by-n matrices are there if the entries of the matrix are integers satisfying $0 \le x < q$?

(d) Same question as (a), except now the order of the coordinates does not matter. So for example, $(0, 0, 1, 3)$ and $(1, 0, 3, 0)$ are considered the same. (This one is rather tricky.)

(e) Twelve students are each taking four classes, for each class they need two looseleaf notebooks, for each notebook they need 100 sheets of paper, and each sheet of paper has 32 lines on it. Altogether, how many students, classes, notebooks, sheets, and lines are there? (Bonus. Make this or a similar problem of your own devising into a rhyme like the St. Ives riddle.)

3.3. (a) List all of the permuations of the set $\{A, B, C\}$.

(b) List all of the permuations of the set $\{1, 2, 3, 4\}$.

(c) How many permuations are there of the set $\{1, 2, \dots, 20\}$?

(d) Seven students are to be assigned to seven dormitory rooms, each student receiving their own room. How many ways can this be done?

[21] A *spelt* is a grain of barley.

[22] A *hekat* is $\frac{1}{30}$ of a cubic cubit, which is approximately 4.8 liters.

(e) How many different words can be formed with the four symbols A, A, B, C?

3.4. (a) List the 24 possible permuations of the letters A_1, A_2, B_1, B_2. If A_1 is indistinguishable from A_2 and B_1 is indistinguishable from B_2, show how the permuations become grouped into 6 distinct letter arrangements, each containing 4 of the original 24 permutations.

(b) Using the seven symbols A, A, A, A, B, B, B, how many different words can be formed?

(c) Using the nine symbols $A, A, A, A, B, B, B, C, C$, how many different words can be formed?

3.5. (a) There are 100 students eligible for an award, and the winner gets to choose from among 5 different possible prizes. How many possible outcomes are there?

(b) Same as in (a), but this time there is a first place winner, a second place winner, and a third place winner, each of whom gets to select a prize. However, there is only one of each prize. How many possible outcomes are there?

(c) Same as in (b), except that there are multiple copies of each prize, so each of the three winners may choose any of the prizes. Now how many possible outcomes are there? Is this larger or smaller than your answer from (b)?

(d) Same as in (c), except that rather than specified a first, second, and third place winner, we just choose three winning students without differentiating between them. Now how many possible outcomes are there? Compare the size of your answers to (b), (c) and (d).

3.6. Use the binomial theorem to compute each of the following quantities.

(a) $(5z + 2)^3$

(b) $(2a - 3b)^4$

(c) $(x - 2)^5$

3.7. The binomial coefficients satisfy many interesting identities. Give three proofs of the identity

$$\binom{n}{j} = \binom{n-1}{j-1} + \binom{n-1}{j}.$$

(a) For Proof #1, use the definition of $\binom{n}{j}$ as $\frac{n!}{(n-j)!j!}$.

(b) For Proof #2, use the Binomial Theorem 3.9 and compare the coefficients of $x^j y^{n-j}$ on the two sides of the identity

$$(x + y)^n = (x + y)(x + y)^{n-1}.$$

(c) For Proof #3, argue directly that choosing j objects from a set of n objects can be decomposed into either choosing $j-1$ objects from $n-1$ objects or choosing j objects from $n-1$ objects.

3.8. This exercise sketches another proof of Fermat's Little Theorem. Let p be a prime number.

(a) If $1 \le j \le p-1$, prove that the binomial coefficient $\binom{p}{j}$ is divisible by p.

(b) Use (a) and the Binomial Theorem 3.9 to prove that

$$(a+b)^p \equiv a^p + b^p \pmod{p} \qquad \text{for all } a, b \in \mathbb{Z}.$$

(c) Use (b) with $b = 1$ and induction on a to prove that $a^p \equiv a \pmod{p}$ for all $a \ge 0$.

(d) Use (c) to deduce that $a^{p-1} \equiv 1 \pmod{p}$ for all a with $\gcd(p, a) = 1$.

Section 3.2. The Vigenère cipher

3.9. Encrypt each of the following Vigenère plaintexts using the given keyword and the Vigenère tableaux (Table 3.1).

(a) Keyword: `hamlet`
Plaintext: `To be, or not to be, that is the question.`

(b) Keyword: `fortune`
Plaintext: `The treasure is buried under the big W.`

3.10. Decrypt each of the following Vigenère ciphertexts using the given keyword and the Vigenère tableaux (Table 3.1).

(a) Keyword: `condiment`
Ciphertext:
```
rsghz bmcxt dvfsq hnigq xrnbm
pdnsq smbtr ku
```

(b) Keyword: `rabbithole`
Ciphtertext:
```
khfeq ymsci etcsi gjvpw ffbsq
moapx zcsfx epsox yenpk daicx
cebsm ttptx zooeq laflg kipoc
zswqm taujw ghboh vrjtq hu
```

3.11. Explain how a cipher wheel (with rotating inner wheel) can be used in place of a Vigeǹere tableaux (Table 3.1) to perform Vigenère encryption and decryption. Illustrate by describing the sequence of rotations used to perform a Vigenère encryption with the keyword `mouse`.

3.12. Let

$$\mathbf{s} = \text{“I am the very model of a modern major general.”}$$

$$\mathbf{t} = \text{“I have information vegatable, animal, and mineral.”}$$

(a) Make frequency tables for **s** and **t**.
(b) Compute IndCo(**s**) and IndCo(**t**).
(c) Compute MutIndCo($\mathbf{s}, \mathbf{t}$).

3.13. The following strings are blocks from a Vigenère encryption. It turns out that the keyword contains a repeated letter, so two of these blocks were encrypted with the same shift. Compute MutIndCo($\mathbf{s}_i, \mathbf{s}_j$) for $1 \le i < j \le 3$ and use these values to deduce which two strings were encrypted using the same shift.

$$\mathrm{s}_1 = \texttt{iwseesetftuonhdptbunnybioeatneghictdnsevi}$$
$$\mathrm{s}_2 = \texttt{qibfhroeqeickxmirbqlflgkrqkejbejpepldfjbk}$$
$$\mathrm{s}_3 = \texttt{iesnnciiheptevaireittuevmhooottrtaaflnatg}$$

3.14. Figure 3.3 is a Vigenère ciphertext in which we have marked some of the repeated trigrams for you. How long do you think that the keyword is? Why?

Bonus: Complete the cryptanalysis and recover the plaintext.

```
nhqrk  vvvfe  fwgjo  mzjgc  kocgk  lejrj  wossy  wgvkk  hnesg  kwebi
bkkcj  vqazx  wnvll  zetjc  zwgqz  zwhah  kwdxj  fgnyw  gdfgh  bitig
mrkwn  nsuhy  iecru  ljjvs  qlvvw  zzxyv  woenx  ujgyr  kqbfj  lvjzx
dxjfg  nywus  rwoar  xhvvx  ssmja  vkrwt  uhktm  malcz  ygrsz  xwnvl
lzavs  hyigh  rvwpn  ljazl  nispv  jahym  ntewj  jvrzg  qvzcr  estul
fkwis  tfylk  ysnir  rddpb  svsux  zjgqk  xouhs  zzrjj  kyiwc  zckov
qyhdv  rhhny  wqhyi  rjdqm  iwutf  nkzgd  vvibg  oenwb  kolca  mskle
cuwwz  rgusl  zgfhy  etfre  ijjvy  ghfau  wvwtn  xlljv  vywyj  apgzw
trggr  dxfgs  ceyts  tiiih  vjjvt  tcxfj  hciiv  voaro  lrxij  vjnok
mvrgw  kmirt  twfer  oimsb  qgrgc
```

Figure 3.3: A Vigenère ciphertext for Exercise 3.14

3.15. We applied a Kasiski test to the following Vigenére ciphertext and found that the the blocksize (i.e. keyword size) is probably 5.

```
togmg  gbymk  kcqiv  dmlxk  kbyif  vcuek  cuuis  vvxqs  pwwej  koqgg
phumt  whlsf  yovww  knhhm  rcqfq  vvhkw  psued  ugrsf  ctwij  khvfa
thkef  fwptj  ggviv  cgdra  pgwvm  osqxg  hkdvt  whuev  kcwyj  psgsn
gfwsl  jsfse  ooqhw  tofsh  aciin  gfbif  gabgj  adwsy  topml  ecqzw
asgvs  fwrqs  fsfvq  rhdrs  nmvmk  cbhrv  kblxk  gzi
```

Blocks		Shift Amount												
i	j	0	1	2	3	4	5	6	7	8	9	10	11	12
1	2	.044	.047	.021	.054	.046	.038	.022	.034	.057	.035	.040	.023	.038
1	3	.038	.031	.027	.037	.045	.036	.034	.032	.039	.039	.047	.038	.050
1	4	.025	.039	.053	.043	.023	.035	.032	.043	.029	.040	.041	.050	.027
1	5	.050	.050	.025	.031	.038	.045	.037	.028	.032	.038	.063	.033	.034
2	3	.035	.037	.039	.031	.031	.035	.047	.048	.034	.031	.031	.067	.053
2	4	.040	.033	.046	.031	.033	.023	.052	.027	.031	.039	.078	.034	.029
2	5	.042	.040	.042	.029	.033	.035	.035	.038	.037	.057	.039	.038	.040
3	4	.032	.033	.035	.049	.053	.027	.030	.022	.047	.036	.040	.036	.052
3	5	.043	.043	.040	.034	.033	.034	.043	.035	.026	.030	.050	.068	.044
4	5	.045	.033	.044	.046	.021	.032	.030	.038	.047	.040	.025	.037	.068

Blocks		Shift Amount												
i	j	13	14	15	16	17	18	19	20	21	22	23	24	25
1	2	.040	.063	.033	.025	.032	.055	.038	.030	.032	.045	.035	.030	.044
1	3	.026	.046	.042	.053	.027	.024	.040	.047	.048	.018	.037	.034	.066
1	4	.042	.050	.042	.031	.024	.052	.027	.051	.020	.037	.042	.069	.031
1	5	.030	.048	.039	.030	.034	.038	.042	.035	.036	.043	.055	.030	.035
2	3	.039	.015	.030	.045	.049	.037	.023	.036	.030	.049	.039	.050	.037
2	4	.027	.048	.050	.037	.032	.021	.035	.043	.047	.041	.047	.042	.035
2	5	.033	.035	.039	.033	.037	.047	.037	.028	.034	.066	.054	.032	.022
3	4	.040	.048	.041	.044	.033	.028	.039	.027	.036	.017	.038	.051	.065
3	5	.039	.029	.045	.040	.033	.028	.031	.037	.038	.036	.033	.051	.036
4	5	.049	.033	.029	.043	.028	.033	.020	.040	.040	.041	.039	.039	.059

Table 3.14: Mutual indices of coincidence for Exercise 3.15

We then performed a mutual index of coincidence test to each shift of each pair of blocks and listed the results for you in Table 3.14. (This is the same type of table as Table 3.5 in the text, except that we haven't underlined the large values.) Use Table 3.14 to guess the relative rotations of the blocks, as we did in Table 3.6. This will give you a rotated version of the keyword. Try rotating it, as we did in Table 3.7, to find the correct keyword and decrypt the text.

3.16. Here is a Vigenére ciphertext for you to analyze from scratch. It is probably easiest to do so by writing a computer program, but you are welcome to try to decrypt it with just paper and pencil.

```
mgodt beida psgls akowu hxukc iawlr csoyh prtrt udrqh cengx
uuqtu habxw dgkie ktsnp sekld zlvnh wefss glzrn peaoy lbyig
uaafv eqgjo ewabz saawl rzjpv feyky gylwu btlyd kroec bpfvt
psgki puxfb uxfuq cvymy okagl sactt uwlrx psgiy ytpsf rjfuw
igxhr oyazd rakce dxeyr pdobr buehr uwcue ekfic zehrq ijezr
xsyor tcylf egcy
```

(a) Make a list of matching trigrams as we did in Table 3.3. Use the Kasiski test on matching trigrams to find the likely blocksize, i.e. the length of the keyword.

(b) Make a table of indices of coincidence for various block sizes, as we did in Table 3.4. Use your results to guess the probable blocksize.

(c) Using the probable blocksize from (a) or (b), make a table of mutual indices of coincidence between rotated blocks, as we did in Table 3.5. Pick the largest indices from your table and use them to guess the relative rotations of the blocks, as we did in Table 3.6.

(d) Use your results from (c) to guess a rotated version of the keyword, and then try the different rotations as we did in Table 3.7 to find the correct keyword and decrypt the text.

3.17. The *autokey cipher* is similar to the Vigenére cipher, except that rather than repeating the key, it simply uses the key to encrypt the first few letters and then uses the plaintext itself (shifted over) to continue the encryption. For example, in order to encrypt the message "`The autokey cipher is cool`" using the keyword `random`, we proceed as follows:

Plaintext	`t h e a u t o k e y c i p h e r i s c o o l`
Key	`r a n d o m t h e a u t o k e y c i p h e r`
Ciphertext	`k h r d i f h r i y w b d r i p k a r v s c`

The autokey cipher has the advantage that different messages are encrypted using different keys (except for the first few letters). Futher, since the key does not repeat, there is no blocksize, so the autokey is not directly susceptible to a Kasiski or index of coincidence analysis. A disadvantage of the autokey is that a single mistake in decryption renders the remainder of the message unintelligible. According to [24], the beautiful invention of the autokey cipher by Vigenére (published in 1586) was ignored and forgotten before being reinvented in the 1800's.

(a) Encrypt the following message using the autokey cipher:
Keyword: `LEAR`
Plaintext: `Come not between the dragon and his wrath.`

(b) Decrypt the following message using the autokey cipher:
Keyword: `CORDELIA`
Ciphertext: `pckkm yowvz ejwzk knyzv vurux cstri tgac`

(c) Eve intercepts an autokey ciphertext and manages to steal the accompanying plaintext:
Plaintext `ifmusicbethefoodofloveplayon`
Ciphertext `azdzwqvjjfbwnqphhmptjsszfjci`
Help Eve to figure out the keyword that was used for encryption. Describe your method in sufficient generality to show that the autokey cipher is susceptible to chosen plaintext attacks.

(d) Bonus Problem: Try to formulate a statistical or algebraic attack on the autokey cipher, assuming that you are given a large amount of ciphertext to analyze.

Section 3.3. Probabilty theory [**M**]

3.18. Use the definition (3.15) of the probabilty of an event to prove the following basic facts about probability theory.

(a) Let E and F be disjoint events. Then

$$\Pr(E \cup F) = \Pr(E) + \Pr(F) \qquad \text{if } E \text{ and } F \text{ are disjoint.}$$

(b) Let E and F be events that need not be disjoint. Then

$$\Pr(E \cup F) = \Pr(E) + \Pr(F) - \Pr(E \cap F).$$

(c) Let E be an event. Then $\Pr(E^c) = 1 - \Pr(E)$.

(d) Let E_1, E_2, E_3 be events. Prove that

$$\begin{aligned}\Pr(E_1 \cup E_2 \cup E_3) = \Pr(E_1) + \Pr(E_2) + \Pr(E_3) - \Pr(E_1 \cap E_2) \\ - \Pr(E_1 \cap E_3) - \Pr(E_2 \cap E_3) + \Pr(E_1 \cap E_2 \cap E_3).\end{aligned}$$

3.19. We continue with the coin tossing scenario from Example 3.20, so our experiment consists of tossing a fair coin ten time. Compute the probabilities of the following events.

(a) The first and last tosses are both heads.

(b) Either the first toss or the last toss (or both) are heads.

(c) Either the first toss or the last toss (but not both) are heads.

(d) There are exactly k heads and $10 - k$ tails. (Compute the probability for each value of k between 0 and 10. *Hint.* To save time, note that the probability of exactly k heads is the same as the probability of exactly k tails.)

(e) There are an even number of heads.

(f) There are an odd number of heads.

3.20. Someone offers to make the following bet with you. They will toss a fair coin 14 times. If exactly 7 heads come up, they will give you $4, otherwise you must give them $1. Would you take this bet? If so, and if you repeated the bet 10000 times, how much money would you expect to win or lose?

3.21. Let E and F be events.

(a) Prove that $\Pr(E|E) = 1$. Explain in words why this is reasonable.

(b) If E and F are disjoint, prove that $\Pr(F|E) = 0$. Explain in words why this is reasonable.

(c) Let $F_1, \ldots, F_n$ be events satisfying $F_i \cap F_j = \emptyset$ for all $i \neq j$. We say that $F_1, \ldots, F_n$ are *pairwise disjoint*. Prove then that

$$\Pr\left(\bigcup_{i=1}^{n} F_i\right) = \sum_{i=1}^{n} \Pr(F_i).$$

(d) Let $F_1, \ldots, F_n$ be pairwise disjoint as in (c), and assume further that $F_1 \cup \cdots \cup F_n = \Omega$. Prove the following general version of the formula 3.20 in Proposition 3.21(a):

$$\Pr(E) = \sum_{i=1}^{n} \Pr(E|F_i)\Pr(F_i).$$

(e) Prove a general version of Bayes' formula:

$$\Pr(F_i|E) = \frac{\Pr(E|F_i)\Pr(F_i)}{\Pr(E|F_1)\Pr(F_1) + \Pr(E|F_2)\Pr(F_2) + \cdots \Pr(E|F_n)\Pr(F_n)}$$

3.22. There are two urns containing pens and pencils. Urn #1 contains 3 pens and 7 pencils and Urn #2 contains 8 pens and 4 pencils.

(a) An urn is chosed at random and an object is drawn. What is the probability that it is a pencil?

(b) An urn is chosed at random and an object is drawn. If the object drawn is a pencil, what is the probability that it came from Urn #1?

(c) If an urn is chosen at random and two objects are drawn simultaneously, what is the probability that both are pencils?

3.23. An urn contain 20 silver coins and 10 gold coins. You are the sixth person in line to draw a coin at random from the urn. What is the probability that you draw a gold coin? If you draw a gold coin, what is the probability that the five people ahead of you all drew silver coins?

3.24. (*The Monty Hall Problem*) Monty Hall gives Dan, a contestant, the choice of three boxes. One box contains a valuable prize and the other two contain nothing. Dan chooses a box, but does not yet open it. Monty Hall then opens one of the other boxes, shows that it is empty, and offers Dan the option of keeping his original box or of switching it for the remaining box. The Monty Hall Problem is to figure out Dan's best stratgy: "To hold or to switch?"

The answer may depend on the strategy that Monty Hall employs in deciding which box to open when he has a choice, i.e. when Dan initially chooses the prize box and the other two boxes are empty. Here are some possible strategies:

(a) Suppose that when he has a choice, Monty Hall randomly opens one of the two empty boxes. Should Dan hold or switch, and what is his probability of winning?

(b) Suppose that the boxes are labeled 0, 1, and 2, and that if Dan chooses Box n and if the other two boxes are empty, then Monty Hall opens Box $n+1$. (If $n = 2$, then he opens Box 0.) Should Dan hold or switch, and what is his probability of winning?

(c) Again assume that the boxes are labeled 0, 1, and 2, but now suppose that Monty Hall always opens the lowest numbered empty box. What is Dan's best strategy and what is his probability of winning? (You may assume that the prize is placed in each box with equal probability.)

(d) With the same assumptions as in (c), suppose that Dan employs his best strategy and that Monty Hall knows that Dan is empolying this strategy. Can Monty Hall hurt Dan's chances of winning by placing the prize in one box more often than the others? But if he does so and if Dan knows, can Dan do better by changing his strategy?

(e) Suppose that we give Monty Hall another option, namely he can force Dan to keep the box that Dan initally chose. Now what is Dan's best strategy to win the prize and what is Monty Hall's best strategy to stop Dan?

3.25. Let $\mathcal{S}$ be a set, let A be a property of interest, and suppose that for $m \in \mathcal{S}$, we have $\Pr(m \text{ has property } A) = \delta$. Suppose further that a Monte Carlo algorithm applied to m and a random number r satisfies:

(1) If the algorithm returns Yes, then m definitely has property A.

(2) If m has property A, then $\Pr(\text{algorithm returns Yes}) \geq p$.

Suppose that we run the algorithm N times on the number m, and that every single time the algorithm returns No. Derive a lower bound, in terms of δ, p, and N, for the probability that m does not have property A. (This generalizes the example with $\delta = 0.01$ and $p = \frac{1}{2}$ that we did in Section 3.3.3. Be careful to distinguish p from $1 - p$ in your calculations.)

3.26. We continue with the setup described in Exercise 3.25.

(a) Suppose that $\delta = \frac{9}{10}$ and $p = \frac{3}{4}$. If we run the algorithm 25 times on the input m and always get back No, what is the probability that m does not have property A?

(b) Same question as (a), but this time we run the algorithm 100 times.

(c) Suppose that $\delta = \frac{99}{100}$ and $p = \frac{1}{2}$. How many times should we run the algorithm on m to be 99% confident that m does not have property A, assuming that every output is No?

(d) Same question as (c), except now we want to be 99.9999% confident.

3.27. Let $f_X(k)$ be the binomial density function (3.24). Prove directly (using the binomial theorem) that $\sum_{k=0}^{n} f_X(k) = 1$.

3.28. In Example 3.35 we used a differentiation trick to compute the value of the infinite series $\sum_{n=1}^{\infty} np(1-p)^{n-1}$. This exercise further develops this useful technique. The starting point is the formula for the geometric series

$$\sum_{n=0}^{\infty} x^n = \frac{1}{1-x} \qquad \text{for } |x| < 1. \tag{3.53}$$

(a) Prove that

$$\sum_{n=1}^{\infty} n x^{n-1} = \frac{1}{(1-x)^2} \tag{3.54}$$

by differentiating both sides of (3.53) with respect to x. For which x does the lefthand side of (3.54) converge? (*Hint.* Use the ratio test.)

(b) Differentiate again to prove that

$$\sum_{n=2}^{\infty} n(n-1) x^{n-2} = \frac{2}{(1-x)^3} \tag{3.55}$$

(c) More generally, prove that for every $k \geq 0$,

$$\sum_{n=0}^{\infty} \binom{n+k}{k} x^n = \frac{1}{(1-x)^{k+1}}. \tag{3.56}$$

(*Hint.* Use induction on k.)

(d) Prove that

$$\sum_{n=0}^{\infty} n^2 x^n = \frac{2x^2}{(1-x)^3} \tag{3.57}$$

(*Hint.* Multiply (3.55) by x and (3.56) by x^2 and then add them together.)

(e) Find a formula for

$$\sum_{n=0}^{\infty} n^3 x^n. \tag{3.58}$$

(f) Prove that for every value of k there is a polynomial $F_k(x)$ so that

$$\sum_{n=0}^{\infty} n^k x^n = \frac{F_k(x)}{(1-x)^{k+1}}. \tag{3.59}$$

(*Hint.* Use induction on k and the formula (3.56).) What are the polynomials $F_0(x)$, $F_1(x)$ and $F_2(x)$?

(g) Prove that the polynomial $F_k(x)$ in (f) has degree k.

3.29. Compute the expectation of each of the following random variables.

(a) X is a random variable whose values are uniformly distributed on the set $\{0, 1, 2, \ldots, N-1\}$. (See Example 3.27.)

(b) X is a random variable whose values are uniformly distributed on the set $\{1, 2, \ldots, N\}$.

(c) X is a random variable whose values are uniformly distributed on the set $\{1, 3, 7, 11, 19, 23\}$.

(d) X is a random variable with a binomial density function (3.24).

Section 3.4. Collision algorithms and the birthday paradox

3.30. (a) In a group of 23 strangers, what is the probability that at least two of them have the same birthday? (*Hint.* Do a calculation similar to the proof of (3.27) in the Collision Theorem 3.36, but note that the formula is a bit different because the birthdays are being selected from a single list of 365 days.) The surprising answer to this question is known as the *Birthday Paradox*.

(b) Same question, but now there are 40 people.

(c) Suppose that there are N days in a year (where N could be any number) and that there are n people. Develop a general formula, analogous to (3.27), for the probability that two of them have the same birthday.

(d) Find a lower bound of the form

$$\Pr(\text{at least one match}) \ge 1 - e^{\text{-(some function of } n \text{ and } N)},$$

for the probability in (c), analogous to the estimate (3.28).

3.31. A deck of cards is shuffled and the top 8 cards are turned over.

(a) What is the probability that the king of hearts is visible?

(b) A second deck is shuffled and its top 8 cards are turned over. What is the probabilty that a visible card from the first deck matches a visible card from the second deck?

3.32. (a) Prove that

$$e^{-x} \ge 1 - x \quad \text{for all values of } x.$$

(*Hint.* Look at the graphs of e^{-x} and $1-x$, or use calculus to compute the minimum of the function $f(x) = e^{-x} - (1-x)$.)

(b) Prove that for all $a > 1$, the inequality

$$e^{-ax} \le (1-x)^a + \frac{1}{2}ax^2 \quad \text{is value for all } 0 \le x \le 1.$$

i	h^i	$a \cdot h^i$	i	h^i	$a \cdot h^i$	i	h^i	$a \cdot h^i$	i	h^i	$a \cdot h^i$
116	96	444	519	291	28	791	496	672	406	801	562
497	326	494	286	239	193	385	437	95	745	194	289
225	757	764	298	358	642	178	527	714	234	304	595
233	517	465	500	789	101	471	117	237	556	252	760
677	787	700	272	24	111	42	448	450	326	649	670
622	523	290	307	748	621	258	413	795	399	263	304

Table 3.15: Data for Exercise 3.33, $h = 10$, $a = 106$, $p = 811$.

(c) We used the inequality in (a) during the proof of the lower bound (3.28) in the Collision Theorem 3.36. Use (b) to prove that

$$\Pr(\text{at least one red}) \le 1 - e^{-mn/N} + \frac{mn^2}{2N^2}.$$

Thus if N is large and m and n are not much larger than $\sqrt{N}$, then the estimate

$$\Pr(\text{at least one red}) \approx 1 - e^{-mn/N}$$

is quite accurate. (*Hint.* Use (b) with $a = m$ and $x = n/N$.)

3.33. Solve the discrete logarithm problem $10^x = 106$ in the finite field $\mathbb{F}_{811}$ by finding a collision amongst the random powers 10^i and $106 \cdot 10^k$ that are listed in Table 3.15.

3.34. Write a computer program to solve the discrete logarithm problem using Shanks' babystep-giantstep method (Remark 3.43) and use it to solve the following problems:

(a) $11^x = 21$ in $\mathbb{F}_{71}$.

(b) $156^x = 116$ in $\mathbb{F}_{593}$.

(c) $650^x = 2213$ in $\mathbb{F}_{3571}$.

3.35. Table 3.16 gives some of the computations for the solution of the discrete logarithm problem

$$11^t = 41387 \quad \text{in } \mathbb{F}_{81799} \tag{3.60}$$

using Pollard's ρ method. (It is similar to Table 3.12 in Example 3.52.) Use the data in Table 3.16 to solve (3.60).

i	x_i	y_i	α_i	β_i	γ_i	δ_i
0	1	1	0	0	0	0
1	11	121	1	0	2	0
2	121	14641	2	0	4	0
3	1331	42876	3	0	12	2
4	14641	7150	4	0	25	4
			⋮			
151	4862	33573	40876	45662	29798	73363
152	23112	53431	81754	9527	37394	48058
153	8835	23112	81755	9527	67780	28637
154	15386	15386	81756	9527	67782	28637

Table 3.16: Computations to solve $11^t = 41387$ in $\mathbb{F}_{81799}$ for Exercise 3.35

3.36. Table 3.17 gives some of the computations for the solution of the discrete logarithm problem

$$7^t = 3018 \quad \text{in } \mathbb{F}_{7963} \tag{3.61}$$

using Pollard's ρ method. (It is similar to Table 3.12 in Example 3.52.) Extend Table 3.17 until you find a collision (we promise that it won't take too long) and then solve (3.61).

i	x_i	y_i	α_i	β_i	γ_i	δ_i
0	1	1	0	0	0	0
1	7	49	1	0	2	0
2	49	2401	2	0	4	0
3	343	2475	3	0	6	1
			⋮			
57	902	45	5659	3901	3139	3900
58	193	2205	5660	3902	3141	3900
59	1351	939	5661	3902	3142	3902
60	1494	2404	5662	3902	3144	3903
61	2495	193	5663	3902	3146	3904

Table 3.17: Computations to solve $7^t = 3018$ in $\mathbb{F}_{7963}$ for Exercise 3.36

3.37. Write a computer program implementing Pollard's ρ method for solving the discrete logarithm problem and use it to solve each of the following:

(a) $2^t = 2495 \quad \text{in } \mathbb{F}_{5011}$.
(b) $17^t = 14226 \quad \text{in } \mathbb{F}_{17959}$.
(c) $29^t = 5953042 \quad \text{in } \mathbb{F}_{15239131}$.

Section 3.6. Information theory [**M**]

3.38. 3.A Consider the cipher that has three keys, three plaintexts, and four ciphertexts that are combined using the following encryption table (which is similar to Table 3.13 used in Example 3.53 on page 148).

	m_1	m_2	m_3
k_1	c_2	c_4	c_1
k_2	c_1	c_3	c_2
k_3	c_3	c_1	c_2

Suppsoe further that the plaintexts and keys are used with the following probabilities:

$$f(m_1) = f(m_2) = \frac{2}{5}, \qquad f(m_3) = \frac{1}{5}, \qquad f(k_1) = f(k_2) = f(k_3) = \frac{1}{3}.$$

(a) Compute $f(c_1)$, $f(c_2)$, $f(c_3)$, and $f(c_4)$.
(b) Compute $f(c_1|m_1)$, $f(c_1|m_2)$, and $f(c_1|m_3)$. Is this cryptosystem perfectly secure?
(c) Compute $f(c_2|m_1)$, and $f(c_3|m_1)$.
(d) Compute $f(k_1|c_3)$ and $f(k_2|c_3)$.

3.39. 3.B Suppose that a shift cipher is employed so that each key (i.e. each shift amount from 0 to 25) is used with equal probability and so that a new key is chosen to encrypt each successive letter. Show that this cryptosystem is perfectly secure by filling in the details of the following steps.

(a) Show that $\sum_{k\in\mathcal{K}} f_M(d_k(c)) = 1$ for every ciphertext $c \in \mathcal{C}$.
(b) Compute the ciphertext density function f_C using the formula

$$f_C(c) = \sum_{k\in\mathcal{K}} f_K(k) f_M(d_k(c)).$$

(c) Compare $f_C(c)$ to $f_{C|M}(c|m)$.

3.40. Suppose that a cryptosystem has the same number of plaintexts as it does ciphertexts ($\#\mathcal{M} = \#\mathcal{C}$). Prove that for any given key $k \in \mathcal{K}$ and any given ciphertext $c \in \mathcal{C}$, there is a unique plaintext $m \in \mathcal{M}$ that encrypts to c using the key k. (We used this fact during the proof of Theorem 3.55. Notice that is does not require the cryptosystem to be perfectly secure; all that is needed is that $\#\mathcal{M} = \#\mathcal{C}$.)

3.41. Let $\mathcal{S}_{m,c} = \bigl\{k \in \mathcal{K} : e_k(m) = c\bigr\}$ be the set used during the proof of Theorem 3.55. Prove that if $c \neq c'$, then $\mathcal{S}_{m,c} \cap \mathcal{S}_{m,c'} = \emptyset$. (Prove this for any cryptosystem, it is not necessary to assume perfect security.)

3.42. Suppose that a cryptosystem satisfies $\#\mathcal{K} = \#\mathcal{M} = \#\mathcal{C}$ and that it has perfect security. Prove that every ciphertext is used with equal probability and that every plaintext is used with equal probability. (*Hint.* We proved one of these during the course of proving Theorem 3.55.)

3.43. Prove the "only if" part of Theorem 3.55, i.e. prove that if a cryptosystem with an equal number of keys, plaintexts, and ciphertexts satisfies condition (a) and (b) of Theorem 3.55, then it has perfect security.

3.44. Let X be an experiment (random variable) with outcomes $x_1, \ldots, x_n$ occurring with probabilities $p_1, \ldots, p_n$, and similarly let Y be an experiment with outcomes $y_1, \ldots, y_m$ occurring with probabilities $q_1, \ldots, q_m$. Consider the experiment Z consisting of first performing X and then performing Y. Thus the outcomes of Z are the mn pairs (x_i, y_j) occurring with probabilities $p_i q_j$. Use the formula for entropy (3.49) to prove that

$$H(Z) = H(X) + H(Y).$$

Thus entropy is additive on independent compound events, which is a special case of Property $\mathbf{H}_3$ on page 153.

3.45. Let $F(t)$ be a twice differentiable function with the property that $F''(t) < 0$ for all x in its domain. Prove that F is concave in the sense of (3.50). Conclude in particular that the function $F(t) = \log t$ is concave for all $t > 0$.

3.46. Use induction to prove Jensen's inequality (Theorem 3.59).

3.47. Let X and Y be independent random variables. Prove that the equivocation $H(X|Y)$ is equal to the entropy $H(X)$. Is the converse true?

3.48. Suppose a cryptosystem has two keys, $Kcal = \{k_1, k_2\}$, each of which is equally likely to be used, and suppose that it has three plaintexts $\mathcal{M} = \{m_1, m_2, m_3\}$ that occur with probabilities $f(m_1) = \frac{1}{2}$, $f(m_2) = \frac{1}{4}$, and $f(m_3) = \frac{1}{4}$.

(a) Create an encryption function for this cipher, similar to Example3.53, so that there are three ciphertexts $\mathcal{C} = \{c_1, c_2, c_3\}$ and so that the ciphertext c_1 occurs with probability $\frac{1}{2}$. (There is more than one correct answer to this problem.)

(b) Compute the entropies $H(K)$, $H(M)$, and $H(C)$ of your encryption scheme in (a).

(c) Compute the key equivocation $H(K|C)$.

(d) Use your answer in (c) to explain why each ciphertext leaks information.

3.49. Suppose that the key equivocation of a certain cryptosystem vanishes, i.e. $H(K|C) = 0$. Prove that even a single observed ciphertext uniquely determines which key was used.

3.50. Write a computer program that reads a text file and performs the following tasks:

[1] Convert all alphabetic characters to lower case and convert all strings of consecutive non-alphabetic characters to a single space. (The reason for leaving in a space is because when you count bigrams and trigrams, you will want to know where words begin and end.)

[2] Count the frequency of each letter `a`-to-`z`, print a frequency table, and use your frequency table to estimate the entropy of a single letter in English, as we did in Section 3.6.3 using Table 1.3.

[3] Count the frequency of each bigram `aa`, `ab`,…,`zz`, being careful to only include bigrams that appear within words. (As an alternative, also allow bigrams that either start or end with a space, in which case there are $27^2 - 1 = 728$ possible bigrams.) Print a frequency table of the 25 most common bigrams and their probabilities, and use your full frequency table to estimate the entropy of bigrams in English. In the notation of Section 3.6.3, this is the quantity $H(L^2)$. Compare $\frac{1}{2}H(L^2)$ with the value of $H(L)$ from step [1].

[4] Repeat (b), but this time with trigrams. Compare $\frac{1}{3}H(L^3)$ with the values of $H(L)$ and $\frac{1}{2}H(L^2)$ from [2] and [3]. (Note that for this part, you will need a lot of text in order to get some reasonable frequencies.)

Try running your program on some long blocks of text. For example, the following non-copyrighted material is available in the form of ordinary text files from Project Gutenberg at `http://www.gutenberg.net/`. To what extent are the letter frequencies similar and to what extent do they differ in these different texts?

(a) *Alice's Adventures in Wonderland* by Lewis Carroll,
`http://www.gutenberg.net/etext/11`

(b) *Relativity : the Special and General Theory* by Albert Einstein,
`http://www.gutenberg.net/etext/5001`

(c) The Old Testament (translated from the original Hebrew, of course!), `http://www.gutenberg.net/etext/1609`

(d) *20000 Lieues Sous Les Mers* (20000 Leagues Under the Sea) by Jules Verne, `http://www.gutenberg.net/etext/5097`. Note that this one is a little trickier, since first you need to convert all of the letters to their unaccented form.

Chapter 4

Integer Factorization and RSA

4.1 Euler's formula and roots modulo pq [M]

The Diffie-Hellman key exchange method and the ElGamal public key cryptosystem studied in Sections 2.3 and 2.4 rely on the fact that it is easy to compute powers $a^n \bmod p$, but difficult to recover the exponent n if you only know the values of a and $a^n \bmod p$. An essential fact in analyzing the security of these methods was Fermat's Little Theorem 1.22,

$$a^{p-1} \equiv 1 \pmod{p} \qquad \text{for all } a \not\equiv 0 \pmod{p}.$$

Fermat's Little Theorem expresses a beautiful property of prime numbers. It is natural to ask what happens if we replace p with a number m that is not prime. Is it still true that $a^{m-1} \equiv 1 \pmod{m}$? A few computations such as Example 1.24 in Section 1.4 will convince you that the answer is no. In this section we investigate the correct generalization of Fermat's Little Theorem when $m = pq$ is a product of two distinct primes, since this is the case that is most important for cryptographic applications. We leave the general case for you to do in Exercises 4.3 and 4.4.

As usual, we begin with an example. What do powers modulo 15 look like? If we make a table of squares and cubes modulo 15, they do not look very interesting, but many fourth powers are equal to 1 modulo 15. More precisely, we find that

$$\begin{aligned} a^4 &\equiv 1 \pmod{15} \qquad \text{for } a = 1, 2, 4, 7, 8, 11, 13, \text{ and } 14; \\ a^4 &\not\equiv 1 \pmod{15} \qquad \text{for } a = 3, 5, 6, 9, 10, \text{ and } 12. \end{aligned}$$

What distinguishes the list of numbers $1, 2, 4, 7, 8, 11, 13, 14$ whose fourth power is 1 modulo 15 from the list of numbers $3, 5, 6, 9, 10, 12, 15$ whose fourth power is not 1 modulo 15? A moments reflection shows that each of the numbers $3, 5, 6, 9, 10, 12, 15$ has a nontrivial factor in common with the modulus 15, while the numbers $1, 2, 4, 7, 8, 11, 13, 14$ are relatively prime to 15. This suggests that some version of Fermat's Little Theorem should be true if the number a is relatively prime to the modulus m, but that the correct exponent to use is not $m - 1$.

For $m = 15$, we found that the right exponent is 4. Why does 4 work? We could simply check each value of a, but a more enlightening argument would be better. In order to show that $a^4 \equiv 1 \pmod{15}$, it is enough to check the two congruences

$$a^4 \equiv 1 \pmod 3 \qquad \text{and} \qquad a^4 \equiv 1 \pmod 5. \tag{4.1}$$

These follow directly from the definition, since the two congruences (4.1) say that

$$3 \text{ divides } a^4 - 1 \quad \text{and} \quad 5 \text{ divides } a^4 - 1,$$

which in turn imply that 15 divides $a^4 - 1$.

The two congruences in (4.1) are modulo primes, so we can use Fermat's Little Theorem to check that they are true. Thus

$$\begin{aligned} &a^4 = (a^2)^2 = (a^{(3-1)})^2 \equiv 1^2 \equiv 1 \pmod 3, \quad \text{and} \\ &a^4 = a^{5-1} \equiv 1 \pmod 5. \end{aligned}$$

If you think about these two congruences, you will see that the crucial property of the exponent 4 is that it is a multiple of $p - 1$ for both $p = 3$ and $p = 5$. With this observation, we are ready to state the fundamental formula that underlies the RSA public key cryptosystem.

Theorem 4.1 (Euler's Formula for pq). *Let p and q be distinct primes and let*

$$G = \gcd(p - 1, q - 1).$$

Then

$$a^{(p-1)(q-1)/G} \equiv 1 \pmod{pq} \qquad \textit{for all } a \textit{ satisfying } \gcd(a, pq) = 1.$$

In particular, if p and q are odd primes, then

$$a^{(p-1)(q-1)/2} \equiv 1 \pmod{pq} \qquad \textit{for all } a \textit{ satisfying } \gcd(a, pq) = 1.$$

Proof. By assumption we know that p does not divide a and that G divides $q-1$, so we can compute

$$\begin{aligned} a^{(p-1)(q-1)/G} &= \left(a^{(p-1)}\right)^{(q-1)/G} && \text{allowed since } (q-1)/G \text{ is an integer,} \\ &\equiv 1^{(q-1)/G} \pmod{p} && \text{since } a^{p-1} \equiv 1 \pmod{p} \\ & && \qquad\text{from Fermat's Little Theorem,} \\ &\equiv 1 \pmod{p} && \text{since 1 to any power is 1!} \end{aligned}$$

The exact same computation, reversing the roles of p and q, shows that

$$a^{(p-1)(q-1)/G} \equiv 1 \pmod{q}.$$

This proves that $a^{(p-1)(q-1)/G} - 1$ is divisible by both p and by q, hence it is divisible by pq, which completes the proof of Theorem 4.1. □

Diffie-Hellman key exchange and the ElGamal public key cryptosystem (Sections 2.3 and 2.4) rely for their security on the difficulty of solving equations of the form

$$a^x \equiv b \pmod{p},$$

where a, b, and p are known quantities, p is a prime, and x is the unknown variable. The RSA public key cryptosystem, which we study in the next section, relies on the difficulty of solving equations of the form

$$x^e \equiv c \pmod{N},$$

where now the quantities e, c, and N are known and x is the unknown. In other words, the security of RSA relies on the assumption that it is difficult to take e^{th} roots modulo N.

Is this a reasonable assumption? If the modulus N is prime, then it turns out that it is is comparatively easy to compute e^{th} roots modulo N.

Proposition 4.2. *Let p be a prime and let $e \geq 1$ be an integer satisfying $\gcd(e, p-1) = 1$. Proposition 1.13 tells us that e has an inverse modulo $p-1$, say*

$$de \equiv 1 \pmod{p-1}.$$

Then the congruence

$$x^e \equiv c \pmod{p}$$

has the solution $x \equiv c^d \pmod{p}$.

Proof. If $c \equiv 0 \pmod{p}$, then $x = 0$ is a solution and we are done. So we assume that $c \not\equiv 0 \pmod{p}$. The proof is then an easy application of Fermat's Little Theorem 1.22. The congruence $de \equiv 1 \pmod{p-1}$ means that there is an integer k so that

$$de = 1 + k(p-1).$$

Now we check that c^d is a solution to $x^e \equiv c \pmod{p}$.

$$\begin{aligned} (c^d)^e &= c^{de} && \text{law of exponents,} \\ &= c^{1+k(p-1)} && \text{since } de = 1 + k(p-1), \\ &= c \cdot (c^{p-1})^k && \text{law of exponents again,} \\ &\equiv c \cdot 1^k \pmod{p} && \text{from Fermat's Little Theorem 1.22,} \\ &= c. \end{aligned}$$

This completes the proof that $x = c^d$ is a solution to $x^e \equiv c \pmod{p}$. □

Example 4.3. We solve the congruence

$$x^{1583} \equiv 4714 \pmod{7919},$$

where the modulus $p = 7919$ is prime. Proposition 4.2 says that first we need to solve the congruence

$$1583d \equiv 1 \pmod{7918}.$$

The solution, using the Extended Euclidean Algorithm (Theorem 1.11, see also Remark 1.14 and Exercise 1.11), is $d \equiv 5277 \pmod{7918}$. Then Proposition 4.2 tells us that

$$x \equiv 4714^{5277} \equiv 6059 \pmod{7919}$$

is a solution to $x^{1583} \equiv 4714 \pmod{7919}$.

Proposition 4.2 shows that it is easy to take e^{th} roots if the modulus is a prime p. The situation for a composite modulus N looks similar, but there is a crucial difference. If we know how to factor N, then it is again easy to compute e^{th} roots. The following proposition explains how to do this if $N = pq$ is a product of two primes. The general case is left for you to do in Exercise 4.5.

Proposition 4.4. *Let p and q be distinct primes and let $e \geq 1$ satisfy*

$$\gcd\bigl(e,(p-1)(q-1)\bigr) = 1.$$

Proposition 1.13 tells us that e has an inverse modulo $(p-1)(q-1)$, say

$$de \equiv 1 \pmod{(p-1)(q-1)}.$$

Then the congruence

$$x^e \equiv c \pmod{pq} \tag{4.2}$$

has the unique solution $x \equiv c^d \pmod{pq}$.

Proof. We assume that $\gcd(c, pq) = 1$, see Exercise 4.2 for the other cases. The proof of Proposition 4.4 is almost identical to the proof of Proposition 4.2 except that instead of Fermat's Little Theorem, we use Euler's formula (Theorem 4.1). The congruence $de \equiv 1 \pmod{(p-1)(q-1)}$ means that there is an integer k so that

$$de = 1 + k(p-1)(q-1).$$

Now we check that c^d is a solution to $x^e \equiv c \pmod{pq}$.

$$\begin{aligned}
(c^d)^e &= c^{de} && \text{law of exponents,}\\
&\equiv c^{1+k(p-1)(q-1)} \pmod{pq} && \text{since } de = 1 + k(p-1)(q-1),\\
&\equiv c \cdot (c^{(p-1)(q-1)})^k \pmod{pq} && \text{law of exponents again,}\\
&\equiv c \cdot 1^k \pmod{pq} && \text{from Euler's formula (Theorem 4.1),}\\
&= c.
\end{aligned}$$

This completes the proof that $x = c^d$ is a solution to the congruence (4.2). It remains to show that the solution is unique. Suppose that $x = m$ is a solution to (4.2). Then

$$\begin{aligned}
m &\equiv m^{de-k(p-1)(q-1)} \pmod{pq} && \text{since } de = 1 + k(p-1)(q-1),\\
&\equiv (m^e)^d \cdot (m^{(p-1)(q-1)})^{-k} \pmod{pq}\\
&\equiv c^d \cdot 1^{-k} \pmod{pq} && \text{since } m \text{ is a solution to (4.2) and}\\
& && \text{using Euler's formula (Theorem 4.1),}\\
&\equiv c^d \pmod{pq}.
\end{aligned}$$

Thus every solution to (4.2) is equal to $c^d \pmod{pq}$, so this is the unique solution. □

Remark 4.5. Proposition 4.4 gives an algorithm for solving $x^e \equiv c \pmod{pq}$ that involves first solving $de \equiv 1 \pmod{(p-1)(q-1)}$ and then computing $c^d \bmod pq$. We can make the computation faster by noticing that a smaller value of d often suffices. Let $G = \gcd(p-1, q-1)$ and suppose that we solve the following congruence for d:

$$de \equiv 1 \quad \left(\bmod \ \frac{(p-1)(q-1)}{G}\right).$$

Euler's formula (Theorem 4.1) says that $a^{(p-1)(q-1)/G} \equiv 1 \pmod{pq}$. Hence just as in the proof of Proposition 4.4, if we write $de = 1+k(p-1)(q-1)/G$, then

$$(c^d)^e = c^{de} = c^{1+k(p-1)(q-1)/G} = c \cdot (c^{(p-1)(q-1)/G})^k \equiv c \pmod{pq}.$$

Thus using this smaller value of d, we still find that $c^d \bmod pq$ is a solution to $x^e \equiv c \pmod{pq}$.

Example 4.6. We solve the congruence

$$x^{17389} \equiv 43927 \pmod{64349},$$

where the modulus $N = 64349 = 229 \cdot 281$ is a product of the two primes $p = 229$ and $q = 281$. The first step is to solve the congruence

$$17389d \equiv 1 \pmod{63840},$$

where $63840 = (p-1)(q-1) = 228 \cdot 280$. The solution, using the method described in Remark 1.14 or Exercise 1.11, is $d \equiv 53509 \pmod{63840}$. Then Proposition 4.4 tells us that

$$x \equiv 43927^{53509} \equiv 14458 \pmod{64349}$$

is a solution to $x^{17389} \equiv 43927 \pmod{64349}$.

We can save ourselves a little bit of work by using the idea described in Remark 4.5. We have

$$G = \gcd(p-1, q-1) = \gcd(228, 280) = 4,$$

so $(p-1)(q-1)/G = (228)(280)/4 = 15960$, which means that we can find a value of d by solving the congruence

$$17389d \equiv 1 \pmod{15960}.$$

The solution is $d \equiv 5629 \pmod{15960}$, and then

$$x \equiv 43927^{5629} \equiv 14458 \pmod{64349}$$

is a solution to $x^{17389} \equiv 43927 \pmod{64349}$. Notice that we obtained the same solution, as we should, but that we only needed to raise 43927 to the 5629th power, while using Proposition 4.4 directly required us to raise 43927 to the 53509th power. This saves some time, although not quite as much as it looks, since recall that computing $c^d \bmod N$ takes time $O(\log d)$. Thus the faster method takes about 80% as long as the slower method, since $\log(5629)/\log(53509) \approx 0.793$.

Example 4.7. Alice challenges Eve to solve the congruence

$$x^{9843} \equiv 134872 \pmod{30069476293}.$$

The modulus 30069476293 is not prime, since (cf. Example 1.24)

$$2^{30069476293-1} \equiv 18152503626 \not\equiv 1 \pmod{30069476293}.$$

It happens that 30069476293 is a product of two primes, but since Eve does not know the prime factors, she cannot use Proposition 4.4 to solve Alice's challenge. After accepting Eve's concession of defeat, Alice informs her that 30069476293 is equal to $104729 \cdot 287117$. With this new knowledge, Alice's challenge becomes easy. Eve computes $104728 \cdot 287116 = 30069084448$, solves the congruence $9843d \equiv 1 \pmod{30069084448}$ to find $d \equiv 18472798299 \pmod{30069084448}$, and computes the solution

$$x \equiv 134872^{18472798299} \equiv 25470280263 \pmod{30069476293}.$$

4.2 The RSA public key cryptosystem

Bob and Alice, as usual, have the problem of exchanging sensitive information over an insecure communication line. We have seen in Chapter 2 (Sections 2.3 and 2.4) various ways in which Bob and Alice can accomplish this task, based on the difficulty of solving the discrete logarithm problem. In this section we describe the RSA public key cryptosystem, the first invented and certainly best known such system. RSA is named after its (public) inventors, Ron Rivest, Adi Shamir, and Leonard Adleman.

The security of RSA depends on the following dichotomy.

- **Setup.** Let p and q be large primes, let $N = pq$, and let e and c be integers.
- **Problem.** Solve the congruence $x^e \equiv c \pmod{N}$ for x.
- **Easy.** Bob, who knows the values of p and q, can easily solve for x as described in Proposition 4.4.
- **Hard.** Eve, who does not know the values of p and q, cannnot easily find x.
- **Dichotomy.** Thus solving $x^e \equiv c \pmod{N}$ is easy for people possessing certain information, but is apparently hard for everyone else.

The *RSA public key cryptosystem* is summarized in Table 4.1. Bob's secret key is a pair of large primes p and q, and his public key is a pair (N, e) consisting of the product $N = pq$ and an encryption exponent e that is relatively prime to $(p-1)(q-1)$. Alice takes her plaintext and converts it into an integer m between 1 and N. She encrypts m by computing the quantity $c \equiv m^e \pmod{N}$. The integer c is her ciphertext, which she sends to Bob. It is then a simple matter for Bob to solve the congruence $x^e \equiv c \pmod{N}$ to recover Alice's message m, because Bob knows the factorization $N = pq$. Eve, on the other hand, can intercept the ciphertext c, but since she does not know how to factor N, she is not able to solve $x^e \equiv c \pmod{N}$. See Example 4.11 on page 188 for an illustration of RSA key creation, encryption and decryption. Of course, this example is not secure, since the numbers are so small that it is easy for Eve to factor the modulus N. Secure implementations of RSA use moduli N with hundreds of digits.

Remark 4.8. The quantities N and e that form Bob's public key are called, respectively, the *modulus* and the *encryption exponent*. The number d that Bob uses to decrypt Alice's message, that is, the number d satisfying

$$ed \equiv 1 \pmod{(p-1)(q-1)} \tag{4.3}$$

is called the *decryption exponent*. Clearly encryption is more efficient if e is chosen to be a small number, and similarly decryption is more efficient if d is chosen to be a small number. It is not possible to choose both of them small, since once one of them is selected, the other is determined by the congruence (4.3).

Bob	**Alice**
Key Creation	
Choose secret primes p and q. Choose encryption exponent e with $\gcd(e,(p-1)(q-1))=1$ Publish $N=pq$ and e.	
Encryption	
	Choose plaintext m. Use Bob's public key (N,e) to compute $c \equiv m^e \pmod{N}$. Send ciphtertext c to Bob.
Decryption	
Compute d satisfying $ed \equiv 1 \pmod{(p-1)(q-1)}$. Compute $m' \equiv c^d \pmod{N}$. Then m' equals the plaintext m.	

Table 4.1: RSA key creation, encryption, and decryption

If Bob chooses $e = 1$, then the plaintext and the ciphertext will be identical, so that would not be a good idea! And Bob cannot take $e = 2$, since he needs e to be relatively prime to $(p-1)(q-1)$. Thus the smallest possible value for e is $e = 3$. As far as is known, taking $e = 3$ is as secure as taking a larger value of e, although see [7]. People who want fast encryption but are worried that $e = 3$ is too small often take $e = 2^{16} + 1 = 65537$, since it only takes 16 squarings and one multiplication to compute m^{65537} via the square-and-multiply algorithm described in Section 1.3.2.

A natural alternative might be to take d small, but this turns out to be quite problematic. Using method of lattice reduction, any choice of d smaller than $N^{1/4}$ leads to an insecure implementation of RSA. See [4, 6, 5, 61].

Remark 4.9. Bob's public key includes the number N, which is a product of two secret primes p and q. Proposition 4.4 says that Eve can decrypt messages sent to Bob if she knows the value of $(p-1)(q-1)$, i.e. if she knows this value, then she can easily solve $x^e \equiv c \pmod{N}$.

Expanding $(p-1)(q-1)$ gives

$$(p-1)(q-1) = pq - p - q + 1 = N - (p+q) + 1.$$

Since Bob has published the value of N, Eve already knows N, so in order to compute the value of $(p-1)(q-1)$, Eve really only needs to determine the value of the sum $p+q$.

However, it turns out that if she knows both of the quantities pq and $p+q$, then it is very easy for her to find the values of p and q. In other words, once Bob publishes the value of N, it is no easier for Eve to find the value of $(p-1)(q-1)$ than it is for her to find p and q.

The way in which Eve would use the values of pq and $(p-1)(q-1)$ to compute p and q is via the factorization

$$(X-p)(X-q) = X^2 - (p+q)X + pq.$$

Thus Eve simply writes down the the polynomial on the right and uses the quadratic formula to factor it and find its roots.

We illustrate with an example. Suppose that Eve knows that

$$N = pq = 66240912547 \quad \text{and} \quad (p-1)(q-1) = 66240396760.$$

Then

$$p+q = N+1-(p-1)(q-1) = 515788,$$

so she writes down and factors the quadratic polynomial

$$\begin{aligned} X^2 - (p+q)X + N &= X^2 - 515788X + 66240912547 \\ &= (X-241511)(X-274277). \end{aligned}$$

This gives her the factorization $N = 66240912547 = 241511 \cdot 274277$.

Remark 4.10. One final, but very important, observation. We have shown that it is no easier for Eve to determine $(p-1)(q-1)$ than it is for her to factor N. But this does not prove that that Eve must factor N in order to decrypt Bob's messages. The point is that what Eve really needs to do is solve congruences of the form $x^e \equiv c \pmod{N}$, and it is conceivable that there is some way to solve these congruences without knowing the value of $(p-1)(q-1)$ and without using Proposition 4.4. No one knows if there is such an alternative method, although see [7] for a suggestion that it may be easier to take roots modulo N than it is to factor N.

Example 4.11. We illustrate the RSA public key cryptosystem with a numerical example using small numbers.

Key Creation

- Bob chooses two secret primes $p = 1223$ and $q = 1987$. Bob computes his public modulus
$$N = p \cdot q = 1223 \cdot 1987 = 2430101.$$
- Bob chooses a public encryption exponent $e = 948047$ with the property that
$$\gcd(e, (p-1)(q-1)) = \gcd(948047, 2426892) = 1.$$

Encryption

- Alice converts her plaintext into an integer
$$m = 1070777 \qquad \text{satisfying} \quad 1 \le m \le N - 1.$$
- Alice uses Bob's public key $(N, e) = (2430101, 948047)$ to compute
$$c \equiv m^e \pmod{N}, \quad c \equiv 1070777^{948047} \equiv 1473513 \pmod{2430101}.$$
- Alice sends the ciphertext $c = 1473513$ to Bob.

Decryption

- Bob knows $(p-1)(q-1) = 1222 \cdot 1986 = 2426892$, so he can solve
$$ed \equiv 1 \pmod{(p-1)(q-1)}, \qquad 948047 \cdot d \equiv 1 \pmod{2426892},$$
for d and get $d = 1051235$.
- Bob takes the ciphertext $c = 1473513$ and computes
$$c^d \pmod{N}, \qquad 1473513^{1051235} \equiv 1070777 \pmod{2430101}.$$
The value that he computes is Alice's message $m = 1070777$.

4.3 Implementation and security issues

Our principal focus in this book is the mathematics of the hard problems underlying modern cryptography, but we would be remiss without at least briefly mentioning some of the security issues related to implementation. You should be aware that we do not even scratch the surface of this vast and fascinating subject, but simply describe some examples to make you aware that there is far more to creating a secure communications system than simply using a cipher based on an intractable mathematical problem.

Example 4.12 (Woman-in-the-Middle Attack). Suppose that Eve is not merely an eavesdropper, but that she has full control over Alice and Bob's communications network. In this case, she can institute what is known as a *Man-in-the-Middle* attack. We describe this attack for Diffie-Hellman key exchange, but it exists for most public key constructions. (See Exercise 4.10.)

Recall that in Diffie-Hellman key exchange (Table 2.2), Alice sends Bob the value $A = g^a$ and Bob sends Alice the value $B = g^b$, where the computations take place in the finite field $\mathbb{F}_p$. What Eve does is to choose her own secret exponent e and compute the value $E = g^e$. She then intercepts Alice and Bob's communications and, instead of sending A to Bob and sending B to Alice, she sends both of them the number E. Notice that Eve has exchanged the value A^e with Alice and the value B^e with Bob, while Alice and Bob believe that they have exchanged values with each other. The Man-in-the-Middle attack is illustrated in Figure 4.1.

Suppose that Alice and Bob subsequently use their supposed secret shared value as the key for a symmetric cipher and send each other messages. For example, Alice encrypts a plaintext message m using E^a as the symmetric cipher key. Eve intercepts this message and is able to decrypt it using A^e as the symmetric cipher key, so she has read Alice's message. She then reencrypts it using B^e as the symmetric cipher key and sends it to Bob. Since Bob is then able to decrypt it using E^b as the symmetric cipher key, he is unaware that there is a breach in security.

Notice the insidious nature of this attack. Eve does not solve the underlying hard problem (in this case, the discrete logarithm problem or the Diffie-Hellman problem), yet she is able to read Alice and Bob's communications, and they are not aware of her success.

Example 4.13. Suppose that Eve is able to convince Alice to decrypt "random" RSA messages. This is a plausible scenario, since it provides a method

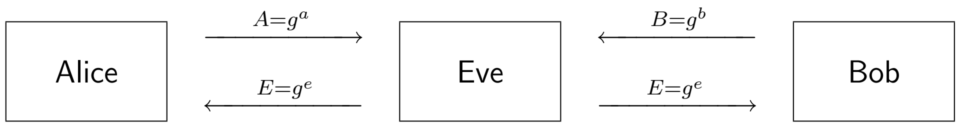

Figure 4.1: "Man-in-the-Middle" attack against Diffie-Hellman key exchange

for Alice to prove that she knows how to decrypt messages that were encrypted using the public key (N, e), i.e. it enables Alice to prove that she owns the public key (N, e). Now suppose that Eve has a copy of a ciphertext c that Bob encrypted using Alice's public modulus N and exponent e. Eve chooses a random value k and sends Alice the "message" $c' \equiv k^e \cdot c \pmod{N}$. Alice decrypts c' and returns the resulting m' to Eve. Note that

$$m' \equiv (c')^d \equiv (k^e \cdot c)^d \equiv (k^e \cdot m^e)^d \equiv k \cdot m \pmod{N}.$$

Thus Eve knows the quantity $k \cdot m \pmod{N}$, and since she knows k, she immediately recovers Bob's plaintext m.

There are two important observations to make. First, Eve has decrypted Bob's message without knowing or gaining knowledge of how to factor N, so the difficulty of the underlying mathematical problem is irrelevant. Second, Alice has no way to tell that Eve's message is in any way related to Bob's message, since Alice only sees the values $k^e \cdot c \pmod{N}$ and $k \cdot m \pmod{N}$, which to her look completely random.

Example 4.14. Suppose that Alice publishes two different exponents e_1 and e_2 for use with her public modulus N and that Bob encrypts a single plaintext m using both of Alice's exponents. If Eve intercepts the ciphertexts $c_1 \equiv m^{e_1} \pmod{N}$ and $c_2 \equiv m^{e_2} \pmod{N}$, she can take a solution to the equation

$$e_1 \cdot u + e_2 \cdot v = \gcd(e_1, e_2)$$

and use it to compute

$$c_1^u \cdot c_2^v \equiv (m^{e_1})^u \cdot (m^{e_2})^v \equiv m^{e_1 \cdot u + e_2 \cdot v} \equiv m^{\gcd(e_1, e_2)} \pmod{N}.$$

If it happens that $\gcd(e_1, e_2) = 1$, Eve has recovered the plaintext. More generally, if Bob encrypts a single message using several exponents $e_1, e_2, \ldots, e_r$, then Eve can recover the plaintext if $\gcd(e_1, e_2, \ldots, e_r) = 1$.

4.4 Primality testing [M]

Bob has finished reading Sections 4.2 and 4.3 and is now ready to communicate with Alice using his RSA public/private key pair. Or is he? In order to create an RSA key pair, Bob needs to choose two *very large* primes p and q. It's not enough for him to choose two very large, but possibly composite, numbers p and q. In the first place, if p and q are not prime, Bob will need to know how to factor them in order to decrypt Alice's message. But even worse, if p and q have some small prime factors, then Eve may well be able to factor pq and break Bob's system.

Bob is thus faced with the task of finding large prime numbers. More precisely, he needs a way of distinguishing between prime numbers and composite numbers, since if he knows how to do this, then he can choose large random numbers until he hits one that is prime. We discuss later (Section 4.4.1) the likelihood that a randomly chosen number is prime, but for now it is enough to know that he has a reasonably good chance of success. Hence what Bob really needs is an efficient way to tell whether or not a very large number is prime.

For example, suppose that Bob chooses the rather large number

$$n = 31987937737479355332620068643713101490952335301$$

and he wants to know if n is prime. First Bob searches for small factors, but he finds that n is not divisible by any primes smaller than 1000000. So he begins to suspect that maybe n is prime. Next he computes the quantity $2^{n-1} \bmod n$ and he finds that

$$2^{n-1} \equiv 1281265953551359064133601216247151836053160074 \pmod{n}. \tag{4.4}$$

The congruence (4.4) immediately tells Bob that n is a composite number, although it does not give him any indication of how to factor n. Why? Recall Fermat's Little Theorem 1.22, which says that if p is prime, then $a^{p-1} \equiv 1 \pmod{p}$ (unless p divides a). Thus if n were prime, then the right-hand side of (4.4) would equal 1; since it does equal 1, Bob concludes that n is not prime.

Before continuing the saga of Bob's quest for large primes, we note that it is often convenient to use the following version of Fermat's Little Theorem that does not put any restrictions on a.

Theorem 4.15 (Fermat's Little Theorem, Version 2)**.** *Let p be a prime number. Then*

$$a^p \equiv a \pmod{p} \qquad \textit{for every integer } a. \tag{4.5}$$

Proof. If $p \nmid a$, then the first version of Fermat's Little Theorem (Theorem 1.22) implies that $a^{p-1} \equiv 1 \pmod{p}$. Multiplying both sides by a gives (4.5). On the other hand, if $p|a$, then both sides of (4.5) are 0 modulo p. □

Returning to Bob's quest, we see him undaunted as he randomly chooses another large number,

$$n = 296795298595169276282041874013832900431516513 1. \tag{4.6}$$

After checking for divisibility by small primes, Bob computes $2^n \bmod n$ and finds that

$$2^n \equiv 2 \pmod{n}. \tag{4.7}$$

Does (4.7) combined with Fermat's Little Theorem 4.15 prove that n is prime? The answer is NO! Fermat's theorem only works in one direction:

$$\text{If } p \text{ is prime, then } a^p \equiv a \pmod{p}.$$

There is nothing to prevent an equality such as (4.7) being true for composite values of n, and indeed a brief search reveals the example

$$2^{341} \equiv 2 \pmod{341} \qquad \text{with} \quad 341 = 11 \cdot 13.$$

However, in some vague philosophical sense, the fact that $2^n \equiv 2 \pmod{n}$ makes it more likely that n is prime, since if the value of $2^n \bmod n$ had turned out differently, we would have known that n was composite. This leads us to make the following definition.

Definition. Fix an integer n. We say that an integer a is a *witness for n* (more precisely, a witness for the compositeness of n) if

$$a^n \not\equiv a \pmod{n}.$$

As we observed earlier, a single witness for n combined with Fermat's Little Theorem 4.15 is enough to prove beyond a shadow of a doubt that n is composite.[1] Thus one way to assess the likelihood that n is prime is to try a lot of numbers $a_1, a_2, a_3, \ldots$. If any one of them is a witnesses for n, then Bob knows that n is composite; and if none of them is a witness for n, then Bob suspects that n is prime.

Unfortunately, intruding on this idyllic scene are barbaric numbers such as 561. The number 561 is composite, $561 = 3 \cdot 11 \cdot 17$, yet 561 has no witnesses! In other words,

$$a^{561} \equiv 561 \pmod{561} \qquad \text{for every integer } a.$$

Composite numbers having no witnesses are called *Carmichael numbers*, after R.D. Carmichael, who in 1910 published a paper listing 15 such numbers. The fact that 561 is a Carmichael number can be verified by checking each value $a = 0, 1, 2, \ldots, 560$, but see Exercise 4.11 for an easier method and for more examples of Carmichael numbers. Although Carmichael numbers are rather rare, Alford, Granville, and Pomerance proved in 1984 that there are infinitely many of them. So Bob needs something stronger than just Fermat's Little Theorem in order to test whether a number is (probably) prime. What is needed is a better test for compositeness. The following property of prime numbers is used to formulate the Miller-Rabin test, which has the agreeable property that every composite number has a large number of witnesses.

Proposition 4.16. *Let p be an odd prime and write*

$$p - 1 = 2^k q \qquad \textit{with } q \textit{ odd.}$$

Let a be any number not divisible by p. Then one of the following two conditions is true:

(i) *a^q is congruent to* 1 *modulo p.*

(ii) *One of a^q, a^{2q}, a^{4q},..., $a^{2^{k-1}q}$ is congruent to* -1 *modulo p.*

Proof. Fermat's Little Theorem 1.22 tells us that $a^{p-1} \equiv 1 \pmod{p}$. This means that when we look at the list of numbers

$$a^q,\ a^{2q},\ a^{4q}, \ldots,\ a^{2^{k-1}q},\ a^{2^k q},$$

we know that the last number in the list, which equals 2^{p-1}, is congruent to 1 modulo p. Further, each number in the list is the square of the previous number. Therefore one of the following two possibilities must occur.

[1] In the great courthouse of mathematics, witnesses never lie!

```
    Input. Integer $n$ to be tested, integer $a$ as potential witness.
1.  If $n$ is even or $1 < \gcd(a,n) < n$, return Composite.
2.  Write $n-1 = 2^k q$ with $q$ odd.
3.  Set $a = a^q \pmod{n}$.
4.  If $a \equiv 1 \pmod{n}$, return Test Fails.
5.  Loop $i = 0, 1, 2, \ldots, k-1$
     6.  If $a \equiv -1 \pmod{n}$, return Test Fails.
     7.  Set $a = a^2 \bmod n$.
8.  Increment $j$ and loop again at Step 5.
9.  Return Composite.
```

Table 4.2: Miller-Rabin test for composite numbers

(i) The first number in the list is congruent to 1 modulo p.

(ii) Some number in the list is not congruent to 1 modulo p, but when it is squared, it becomes congruent to 1 modulo p. But the only number satisfying both

$$b \not\equiv 1 \pmod{p} \qquad \text{and} \qquad b^2 \equiv 1 \pmod{p}$$

is -1, so one of the numbers in the list is congruent to -1 modulo p.

This completes the proof of Proposition 4.16. □

Definition. Let n be an odd number and let a be an integer with $\gcd(a,n) = 1$. Then a is called a *Miller-Rabin witness for* n if, when we write $n = 2^k q$ with q odd, both of the following conditions are true:

(a) $a^q \not\equiv 1 \pmod{n}$.

(b) $a^{2^i q} \not\equiv -1 \pmod{n}$ for all $i = 0, 1, 2, \ldots, k-1$.

Proposition 4.16 says that if a is a Miller-Rabin witness for n, then n is a composite number. This leads to the Miller-Rabin test for composite numbers described in Table 4.2. More precisely, Bob runs the Miller-Rabin test using a large number of randomly selected values of a. The reason that the Miller-Rabin test is so much better than the test based on Fermat's Little Theorem is the fact that there are no Carmichael-like numbers for the Miller-Rabin test, i.e. every composite number has many Miller-Rabin witnesses. This can be made quite precise.

Proposition 4.17. *Let n be an odd composite number. Then at least 75% of the numbers a between 1 and $n-1$ are Miller-Rabin witnesses for n.*

Proof. The proof is not hard, but we will not give it here. See for example [52, Theorem 10.6]. □

Consider now Bob's quest to identify large prime numbers. He takes his potentially prime number n and he runs the Miller-Rabin test on n for (say) 10 different values of a. If any a value is a Miller-Rabin witness for n, then Bob knows that n is composite. But suppose that none of his a values is a Miller-Rabin witness for n. Proposition 4.17 says that if n were composite, then each time Bob tries a value for a, he has at least a 75% chance of getting a witness. Since Bob found no witnesses in 10 tries, it is reasonable to conclude that the probability of n being composite is at most $(25\%)^{10}$, which is approximately 10^{-6}. And if this is not good enough, Bob can use 100 different values of a, and if none of them proves n to be composite, then the probability that n is actually composite is less than $(25\%)^{100} \approx 10^{-60}$.

Example 4.18. We illustrate the Miller-Rabin test with $a = 2$ and the number $n = 561$, which you may recall is a Carmichael number. We factor

$$n - 1 = 560 = 2^4 \cdot 35$$

and then compute

$$\begin{aligned} 2^{35} &\equiv 263 && \pmod{561} \\ 2^{2\cdot 35} &\equiv 263^2 \equiv 166 && \pmod{561} \\ 2^{4\cdot 35} &\equiv 166^2 \equiv 67 && \pmod{561} \\ 2^{8\cdot 35} &\equiv 67^2 \;\;\equiv 1 && \pmod{561} \end{aligned}$$

The first number $2^{35} \bmod 561$ is neither 1 nor -1, and the other numbers in the list are not equal to -1, so 2 is a Miller-Rabin witness to the fact that 561 is composite.

Example 4.19. We do a second example, taking $n = 172947529$ and

$$n - 1 = 172947528 = 2^3 \cdot 21618441.$$

We apply the Miller-Rabin test with $a = 17$ and find that

$$17^{21618441} \equiv 1 \pmod{172947529}.$$

Thus 17 is not a Miller-Rabin witness for n. Next we try $a = 3$, but unfortunately

$$3^{21618441} \equiv -1 \pmod{172947529},$$

so 3 also fails to be a Miller-Rabin witness. At this point we might suspect that n is prime, but if we try another value, say $a = 23$, we find

$$\begin{aligned} 23^{21618441} &\equiv 40063806 && \pmod{172947529}, \\ 23^{2\cdot 21618441} &\equiv 2257065 && \pmod{172947529}, \\ 23^{4\cdot 21618441} &\equiv 1 && \pmod{172947529}. \end{aligned}$$

Thus 23 is a Miller-Rabin witness and n is actually composite. In fact, n is a Carmichael number, but it's not so easy to factor (by hand).

4.4.1 The distribution of the set of primes

If Bob picks a number at random, what is the likelihood that it is prime? The answer is provided by one of number theory's most famous theorems. In order to state the theorem, we need one definition.

Definition. For any number X, let

$$\pi(X) = (\#\text{ of primes between 2 and } X).$$

For example, $\pi(10) = 4$, since the primes between 2 and 10 are 2, 3, 5, and 7.

Theorem 4.20 (The Prime Number Theorem)**.**

$$\lim_{X\to\infty} \frac{\pi(X)}{X/\ln(X)} = 1.$$

Using little-oh notation, this may be rephrased as

$$\frac{\pi(X)}{X} = \frac{1}{\ln X} + o\left(\frac{1}{\ln(X)}\right).$$

Intuitively, the Prime Number Theorem says that if we look at all of the numbers between 1 and X, the proportion of them that are prime is approximately $1/\ln(X)$. Turning this statement around, the Prime Number Theorem says:

$$\boxed{\begin{array}{l}\text{A randomly chosen number } N \text{ has} \\ \text{probability } 1/\ln(N) \text{ of being prime.}\end{array}} \tag{4.8}$$

Of course, taken at face value, statement (4.8) is utter nonsense. A chosen number either is prime or is not prime, it cannot be partially prime and partially composite. See Exercise 4.15 for an interpretation of (4.8) that is both meaningful and mathematically correct.

Example 4.21. We illustrate statement (4.8) and the Prime Number Theorem by searching for 1024 bit primes, i.e. primes that are approximately 2^{1024}. Statement (4.8) says that the probability that a random number $N \approx 2^{1024}$ is prime is approximately 0.14%. Thus on average, Bob checks about 700 randomly chosen numbers of this size before finding a prime.

If he is clever, Bob can do better. He knows that he doesn't want a number that is even, nor does he want a number that is divisible by 3, nor divisible by 5, etc. Thus rather than choosing numbers completely at random, Bob might restrict attention (say) to numbers that are relatively prime to 2, 3, 5, 7 and 11. To do this,, he first chooses a random number that is relatively prime to $2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 = 2310$, say he chooses 1139. Then he considers only numbers N of the form

$$N = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 \cdot K + 1139 = 2310K + 1139. \tag{4.9}$$

The probability that an N of this form is prime is approximately (see Exercise 4.16)

$$\frac{2}{1} \cdot \frac{3}{2} \cdot \frac{5}{4} \cdot \frac{7}{6} \cdot \frac{11}{10} \cdot \frac{1}{\ln(N)} \approx \frac{4.8}{\ln(N)}.$$

So if Bob chooses a random number N of the form (4.9) with $N \approx 2^{1024}$, then the probability that it is prime is approximately 0.67%. Thus he only needs to check 150 numbers to have a good chance of finding a prime.

We used the Rabin-Miller test with 100 randomly chosen values of a to check the primality of

$$2310K + 1139 \qquad \text{for each} \qquad 2^{1013} \le K \le 2^{1013} + 1000.$$

We found that $2310(2^{1013} + J) + 1139$ is probably prime for the following 12 values of J:

$$J \in \{41, 148, 193, 251, 471, 585, 606, 821, 851, 865, 910, 911\}.$$

This is a bit better than the 7 values predicted by the prime number theorem. The smallest probable prime that we found is $2310 \cdot (2^{1013} + 41) + 1139$, which is equal to the following 308 digit number:

20276714558261473373313940384387925462194955182405899331133959349334105522983

7512127224893854863968851947003448487753250093654475567042186503162873426359974273751 8719

78241831537235413710389881550750303525056818030281312537212445925881220354174468221605146

32796943083444056549712787507063680159820382419821 9369.

Remark 4.22. There are many deep open questions concerning the distribution of prime numbers, of which the most important and famous is certainly the *Riemann Hypothesis*. The usual way to state the Riemann Hypothesis requires some complex analysis. The Riemann zeta function, which is defined by the series

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s} \qquad \text{for } \mathrm{Re}(s) > 1,$$

has an analytic continuation to the entire complex plane with a simple pole at $s = 1$ and no other poles. The Riemann Hypothesis says that in the strip $0 < \mathrm{Re}(s) < 1$, all of the zeros of $\zeta(s)$ lie on the line $\mathrm{Re}(s) = \frac{1}{2}$.

At first glance, this somewhat bizarre statement appears to have little relation to prime numbers. However, it is not hard to show that $\zeta(s)$ is also equal to the product

$$\zeta(s) = \prod_{p \text{ prime}} \left(1 - \frac{1}{p^s}\right)^{-1},$$

so $\zeta(s)$ incorporates information about the set of prime numbers.

There are many statements about prime numbers that are equivalent to the Riemann Hypothesis. For example, recall that the Prime Number Theorem 4.20 says that $\pi(X)$ is approximately equal to $X/\ln(X)$ for large values of X. The Riemann Hypothesis is equivalent to the following more accurate statement:

$$\pi(X) = \int_2^X \frac{dt}{\ln t} + O\left(\sqrt{X} \cdot \ln(X)\right). \tag{4.10}$$

This statement implies the Prime Number Theorem, since the integral is approximately equal to $X/\ln(X)$, see Exercise 4.17.

4.4.2 Primality proofs versus probabilistic tests

The Miller-Rabin test is a powerful and practical method for finding large numbers that are "probably prime." Indeed, Proposition 4.17 says that every composite number has many Miller-Rabin witnesses, so 50 or 100 repetitions of the Miller-Rabin test provides solid evidence that n is prime. However, there is a difference between evidence for a statement and a rigorous proof that the statement is correct. Suppose that Bob is not satisfied with mere evidence. He wants to be completely certain that his chosen number n is a prime.

In principle, nothing could be simpler. Bob checks to see if n is divisible by any of the numbers $1, 2, 3, 4, \dots$ up to $\sqrt{n}$. If none of these numbers divides n, then Bob knows, with complete certainty, that n is prime. Unfortunately, if n is large, say $n \approx 2^{1000}$, then the sun will have burnt out before Bob finishes his task. Notice that the running time of this naive algorithm is $O(\sqrt{n})$, so it is an exponential-time algorithm.

It would be nice if we could use the Rabin-Miller test to efficiently and conclusively prove that a number n is prime. More precisely, we would like a polynomial-time algorithm that proves primality. If a generalized version of the Riemann Hypothesis, which we discussed in Remark 4.22, is true, then the following proposition says that this can be done.

Proposition 4.23. *Assume that the generalized Riemann Hypothesis* (*see Remark 4.22*) *is true. Then every composite number n has a Miller-Rabin witness for its compositeness satisfying*

$$a \le 2(\ln n)^2.$$

Proof. See [35] for a proof that every composite number n has a witness satisfying $a = O\big((\ln n)^2\big)$, and [2] for the more precise estimate $a \le 2(\ln n)^2$. □

Thus if the generalized Riemann Hypothesis is true, then we can prove that n is prime by applying the Miller-Rabin test for every a satisfying $1 \le a \le 2(\ln n)^2$. If some a proves that n is composite, then n is composite, and otherwise Proposition 4.23 tells us that n is prime.

Unfortunately, the proof of Proposition 4.23 assumes that the Riemann Hypothesis is true, and no one has yet been able to prove the Riemann Hypothesis, despite almost two centuries of work on the problem. After the creation of public key cryptography, and especially after the publication of the RSA cryptosystem in 1978, it became of great interest to find a polynomial-time primality test that did not depend on some unproven hypothesis. Many years of research culminated in 2002, when M. Agrawal, N.

Kayal, and N. Saxena [1] found such an algorithm. Subsequent improvements to their algorithm have given the following result.

Theorem 4.24 (AKS Primality Test)**.** *For every $\epsilon > 0$, there is an algorithm that conclusively determines whether a given number n is prime or composite in no more than $O\big((\ln n)^{6+\epsilon}\big)$ steps.*

Proof. The original algorithm was published in [1]. Further analysis and refinements may be found in [23]. The monograph [**?**] contains a nice description of primality testing, including the AKS test. □

Remark 4.25. The result described in Theorem 4.24 represents a triumph of modern algorithmic number theory. The significance for practical cryptography is less clear, since the AKS algorithm is much slower than the Miller-Rabin test, and in practice most people are willing to accept a number as prime if it passes the Miller-Rabin test for (say) 50 to 100 randomly chosen values of a.

4.5 The $p-1$ factorization method

We saw in Section 4.4 that it is relatively easy to check whether or not a large number is composite or (probably) prime. This is good, since the RSA cryptosystem needs large primes in order to operate.

In the opposite direction, the security of RSA relies on the apparent difficulty of factoring large numbers. The study of factorization dates back at least to ancient Greece, but it was only with the advent of computers that people started to develop algorithms capable of the factoring extremely large numbers. In order to make RSA as efficient as possible, we want to use a modulus $N = pq$ that is as small as possible. On the other hand, if an opponent can factor N, then our encrypted messages are not secure. It is thus vital to understand how hard it is to factor large numbers, and in particular, to understand the various algorithms that are currently used for factorization.

In the next few sections we discuss, with varying degrees of detail, some of the known methods for factoring large integers. A further method using elliptic curves is described in Section 5.6. Those readers interested in pursuing this subject further might consult [11, 13, 42, 62] and the references cited in those works.

We begin with an algorithm called *Pollard's $p-1$ method.* Although not useful for all numbers, there are certain types of numbers for which it is quite efficient. Pollard's method demonstrates that there are insecure RSA moduli which, at first glance, appear to be secure. This alone warrants studying Pollard's method. In addition, the $p-1$ method provides the inspiration for Lenstra's elliptic curve factorization method, which we study later in Section 5.6.

We are presented with a number $N = pq$ and our task is to determine the prime factors p and q of N. Suppose that by luck or hard work or some other method, we manage to find an integer L with the property that

$$p-1 \text{ divides } L \qquad \text{and} \qquad q-1 \text{ does not divide } L.$$

This means that there are integers i, j, and k with $k \neq 0$ satisfying

$$L = i(p-1) \qquad \text{and} \qquad L = j(q-1)+k.$$

Consider what happens if we take a randomly chosen integer a and compute a^L. Fermat's Little Theorem 1.22 tells us that[2]

$$\begin{aligned} a^L &= a^{i(p-1)} = (a^{p-1})^i \equiv 1^i \equiv 1 \pmod{p}, \\ a^L &= a^{j(q-1)+k} = a^k(a^{q-1})^j \equiv a^k \cdot 1^j \equiv a^k \pmod{q}. \end{aligned}$$

The exponent k is not equal to 0, so it is quite unlikley that a^k will be congruent to 1 modulo q. Thus with very high probability, i.e. for most choices of a, we find that

$$p \text{ divides } a^L - 1 \qquad \text{and} \qquad q \text{ does not divide } a^L - 1.$$

But this is wonderful, since it means that we can recover p via the simple gcd computation

$$p = \gcd(a^L - 1, N).$$

This is all well and good, but where, you may ask, can we find an exponent L that is divisible by $p-1$ and not by $q-1$? Pollard's observation is that if $p-1$ happens to be a product of a lot of small primes, then we can find a multiple of $p-1$ by taking the product of the first few integers.

[2] We have assumed that $p \nmid a$ and $q \nmid a$, since if p and q are very large, this will almost certainly be the case. Further, if by some chance $p|a$ and $q \nmid a$, then we can recover p as $p = \gcd(a, N)$.

In other words, if $p-1$ is a product of a lot of small primes, then it will divide $n!$ for some not-too-large value of n. So here is the idea. For each value $n = 2, 3, 4, \ldots$ we choose a value of a and compute

$$\gcd(a^{n!} - 1, N).$$

(In practice, we might simply take $a = 2$.) If the gcd is equal to 1, then we go on to the next value of n. If the gcd ever equals N, then we've been quite unlucky, but a different a value will probably work. And if we get a number strictly between 1 and N, then we have a nontrivial factor of N and we're done.

Remark 4.26. There are two important remarks to make before we put Pollard's idea into practice. The first concerns the quantity $a^{n!} - 1$. Even for $a = 2$ and quite moderate values of n, say $n = 100$, it is not feasible to compute $a^{n!} - 1$ exactly. Indeed, the number $2^{100!}$ has more than 10^{157} digits, which is larger than the number of elementary particles in the known universe! Luckily, there is no need to compute it exactly. We are only interested in the greatest common divisor of $a^{n!} - 1$ and N, so it suffices to compute

$$a^{n!} - 1 \pmod{N}$$

and then take the gcd with N. Thus we never need to work with numbers larger than N.

Second, we do not even need to compute the exponent $n!$ exactly. Instead, assuming that we have already computed $a^{n!} \bmod N$ in the previous step, we can compute the next value as

$$a^{(n+1)!} \equiv \left(a^{n!}\right)^{n+1} \pmod{N}.$$

This leads to the algorithm described in Table 4.3.

Remark 4.27. How long does it take to compute the value of $a^{n!} \bmod N$? The Fast Exponentiation Algorithm described in Section 1.3.2 gives a method for computing $a^k \bmod N$ in $O(\log(k))$ steps. Stirling's formula says that $n!$ is approximately equal to $(n/e)^n$ when n is large,[3] so we can compute $a^{n!} \bmod N$ in approximately $n \cdot (\ln(n) - 1)$ steps. Thus it is feasible to compute $a^{n!} \bmod N$ for quite large values of n.

[3] Stirling's formula says more precisely that $\ln(n!) = n\ln(n) - n + \frac{1}{2}\ln(2\pi n) + O(1/n)$.

```
     **Input**. Integer $N$ to be factored.
  1. Set $a = 2$ (or some other convenient value).
  2. Loop $j = 2, 3, 4, \dots$ up to a specified bound.
     3. Set $a = a^j \bmod N$.
     4. Compute $d = \gcd(a-1, N)^\dagger$.
     5. If $1 < d < N$ then **success**, return $d$.
  6. Increment $j$ and loop again at Step **2**.
† For added efficiency, choose an appropriate $k$ and
  only compute the gcd in Step **4** every $k^{\text{th}}$ iteration.
```

Table 4.3: Pollard's $p-1$ factorization algorithm

Example 4.28. We use Pollard's $p-1$ method to factor $N = 13927189$. Starting with $\gcd(2^{9!}-1, N)$ and taking successively larger factorials in the exponent, we find that

$$\begin{aligned}
2^{9!}-1 &\equiv 13867883 \pmod{13927189} & \gcd(2^{9!}-1, 13927189) &= 1\\
2^{10!}-1 &\equiv 5129508 \pmod{13927189} & \gcd(2^{10!}-1, 13927189) &= 1\\
2^{11!}-1 &\equiv 4405233 \pmod{13927189} & \gcd(2^{11!}-1, 13927189) &= 1\\
2^{12!}-1 &\equiv 6680550 \pmod{13927189} & \gcd(2^{12!}-1, 13927189) &= 1\\
2^{13!}-1 &\equiv 6161077 \pmod{13927189} & \gcd(2^{13!}-1, 13927189) &= 1\\
2^{14!}-1 &\equiv 879290 \pmod{13927189} & \gcd(2^{14!}-1, 13927189) &= 3823
\end{aligned}$$

The final line gives us a nontrivial factor $p = 3823$ of N, and the other factor is $q = N/p = 13927189/3823 = 3643$. The reason that an exponent of 14! worked in this instance is because $p-1$ factors into a product of small primes,

$$p - 1 = 3822 = 2 \cdot 3 \cdot 7^2 \cdot 13.$$

The other factor satisfies $q - 1 = 3642 = 2 \cdot 3 \cdot 607$, which is not a product of small primes.

Example 4.29. We present one further example using larger numbers. Let

$N = 168441398857$. Then

$$\begin{aligned}
2^{50!} - 1 &\equiv 114787431143 \pmod{N} & \gcd(2^{50!}-1, N) &= 1\\
2^{51!} - 1 &\equiv 36475745067 \pmod{N} & \gcd(2^{51!}-1, N) &= 1\\
2^{52!} - 1 &\equiv 67210629098 \pmod{N} & \gcd(2^{52!}-1, N) &= 1\\
2^{53!} - 1 &\equiv 8182353513 \pmod{N} & \gcd(2^{53!}-1, N) &= 350437
\end{aligned}$$

So using $2^{53!} - 1$ yields the factor $p = 350437$ of N, and the other factor is 480661. We were lucky, of course, that $p-1$ is a product of small factors,

$$p - 1 = 350436 = 2^2 \cdot 3 \cdot 19 \cdot 29 \cdot 53.$$

Remark 4.30. In practice, there is a refinement that greatly improves the speed of Pollard's $p-1$ and other similar factorization methods. Rather than requiring N to have a prime factor p with the property that $p-1$ is entirely a product of small primes, we instead allow $p-1$ to be a product of small primes multiplied by a single much larger prime. For details of this refinement and other implementation improvements, see for example [11, Section 8.8].

Remark 4.31. In practice, it is easy for Bob and Alice to avoid the dangers of Pollard's $p-1$ method when creating RSA keys. They simply check that for their chosen secret primes p and q, both $p-1$ and $q-1$ do not factor entirely into smll primes. From a cryptographic viewpoint, the importance of Pollard's method lies in the following lesson. At first glance, one would not expect that the factorization of $p-1$ and $q-1$ should have anything to do with the difficulty of factoring pq. So even if we build a cryptosystem based on a seemingly hard problem such as integer factorization, we must be wary of special cases of the problem that, for subtle and nonobvious reasons, are easier to solve than the general case. We have already seen an example of this in the Pohlig-Hellman algorithm for the discrete logarithm problem (Section 2.6) and we will see it again later when we discuss elliptic curves and the elliptic curve discrete logarithm problem (Section 5.9).

Remark 4.32. We have not yet discussed the likelihood that Pollard's method succeeds. Suppose that p and q are randomly chosen primes of about the same size. Pollard's method works if at least one of $p-1$ or $q-1$ factors entirely into a product of small prime powers. Clearly $p-1$ is even, so we can pull off a factor of 2, but after that, the quantity $\frac{1}{2}(p-1)$ should behave more-or-less like a random number of size approximately $p/2$. This leads to the following question:

> What is the probability that a randomly chosen integer of size approximately n divides $B!$?

Notice in particular that if n divides $B!$, then every prime ℓ dividing n must satisfy $\ell \le B$. A number whose prime factors are all less than or equal to B is called a *B-smooth number*. It is thus natural to ask:

> What is the probability that a randomly chosen integer of size approximately n is a B-smooth number?

The efficiency (or lack thereof) of all modern methods of integer factorization is largely determined by the answer to this question. We study smooth numbers, which play a prominent role in the theory of sieves, in Section 4.7.

4.6 Factorization via difference of squares

The most powerful factorization methods known today rely on one of the simplest identities in all of mathematics,

$$X^2 - Y^2 = (X+Y)(X-Y). \tag{4.11}$$

This beautiful formula says that a difference of squares is equal to a product. The potential applicability to factorization is immediate. In order to factor a number N, we look for an integer b so that the quantity $N + b^2$ is a perfect square, say equal to a^2. Then $N + b^2 = a^2$, so

$$N = a^2 - b^2 = (a+b)(a-b),$$

and we have effected a factorization of N.

Example 4.33. We factor $N = 25217$ by looking for an integer b making

$N + b^2$ a perfect square.

$$\begin{aligned}
25217 + 1^2 &= 25218 && \text{not a square}\\
25217 + 2^2 &= 25221 && \text{not a square}\\
25217 + 3^2 &= 25226 && \text{not a square}\\
25217 + 4^2 &= 25233 && \text{not a square}\\
25217 + 5^2 &= 25242 && \text{not a square}\\
25217 + 6^2 &= 25253 && \text{not a square}\\
25217 + 7^2 &= 25266 && \text{not a square}\\
25217 + 8^2 &= 25281 = 159^2 && \text{** square **}
\end{aligned}$$

Thus

$$25217 = 159^2 - 8^2 = (159 + 8)(159 - 8) = 167 \cdot 151.$$

If N is large, then it is unlikely that a randomly chosen value of b makes $N + b^2$ into a perfect square. We need to find a clever way to select a value of b. An important observation is that we don't necessarily need to write N itself as a difference of two squares. It often suffices to write some multiple kN of N as a difference of two squares, since if

$$kN = a^2 - b^2 = (a + b)(a - b),$$

then there is a reasonable chance that the factors of N are scattered on the righthand side, i.e. N has a nontrivial factor in common with each of $a + b$ and $a - b$. It is then a simple matter to recover those factors by computing $\gcd(N, a + b)$ and $\gcd(N, a - b)$. We illustrate with an example.

Example 4.34. We factor $N = 203299$. If we make a list of $N + b^2$ for $b = 1, 2, 3, \ldots$, say up to $b = 100$, we do not find any square values. So next we try listing the values of $3m + b^2$ and we find

$$\begin{aligned}
3 \cdot 203299 + 1^2 &= 609898 && \text{not a square}\\
3 \cdot 203299 + 2^2 &= 609901 && \text{not a square}\\
3 \cdot 203299 + 3^2 &= 609906 && \text{not a square}\\
3 \cdot 203299 + 4^2 &= 609913 && \text{not a square}\\
3 \cdot 203299 + 5^2 &= 609922 && \text{not a square}\\
3 \cdot 203299 + 6^2 &= 609933 && \text{not a square}\\
3 \cdot 203299 + 7^2 &= 609946 && \text{not a square}\\
3 \cdot 203299 + 8^2 &= 609961 = 781^2 && \text{** square **}
\end{aligned}$$

Thus

$$3 \cdot 203299 = 781^2 - 8^2 = (781 + 8)(781 - 8) = 789 \cdot 773.$$

Finally we compute

$$\gcd(203299, 789) = 263 \quad \text{and} \quad \gcd(203299, 773) = 773,$$

which gives nontrivial factors of N. The numbers 263 and 773 are prime, so the full factorization of N is $203299 = 263 \cdot 789$.

Remark 4.35. In Example 4.34, we made a list of values of $3N + b^2$. Why didn't we try $2N + b^2$ first? The answer is that if N is odd, then $2N + b^2$ can never be a square, so it would have been a waste of time to try it. The reason that $2N + b^2$ can never be a square is as follows (cf. Exercise 1.17). We compute modulo 4,

$$2N + b^2 \equiv 2 + b^2 \equiv \begin{cases} 2 + 0 \equiv 2 & \text{if } b \text{ is even,} \\ 2 + 1 \equiv 3 & \text{if } b \text{ is odd.} \end{cases}$$

Thus $2N + b^2$ is congruent to either 2 or 3 modulo 4. But squares are congruent to either 0 or 1 modulo 4. Hence if N is odd, then $2N + b^2$ is never a square.

The multiples of N are the numbers that are congruent to 0 modulo N, so rather than searching for a difference of squares $a^2 - b^2$ that is a multiple of N, we may instead search for distinct numbers a and b satisfying

$$a^2 \equiv b^2 \pmod{N}. \tag{4.12}$$

This is exactly the same problem, of course, but the use of modular arithmetic helps to clarify our task.

In practice it is not feasible to directly search for integers a and b satisfying (4.12). Instead we use the three-step process described in Table 4.4. This procedure, in one form or another, underlies most modern methods of factorization.

Example 4.36. We factor $N = 914387$ using the procedure described in Table 4.4 We first search for integers a with the property that $a^2 \bmod N$ is a product of small primes. For this example, we ask that each $a^2 \bmod N$ be a product of primes in the set $\{2, 3, 5, 7, 11\}$. Ignoring the question of how to

1. **Relation Building**: Find many integers $a_1, a_2, a_3, \ldots, a_r$ with the property that the quantity $c_i \equiv a_i^2 \pmod{N}$ factors as a product of small primes.
2. **Elimination**: Take a product $c_{i_1} c_{i_2} \cdots c_{i_s}$ of some of the c_i's so that every prime appearing in the product appears to an even power. Then $c_{i_1} c_{i_2} \cdots c_{i_s} = b^2$ is a perfect square.
3. **GCD Computation**: Let $a = a_{i_1} a_{i_2} \cdots a_{i_s}$ and compute $d = \gcd(N, a - b)$. Since
$$a^2 = (a_{i_1} a_{i_2} \cdots a_{i_s})^2 \equiv a_{i_1}^2 a_{i_2}^2 \cdots a_{i_s}^2 \equiv c_{i_1} c_{i_2} \cdots c_{i_s} \equiv b^2 \pmod{N},$$
there is a reasonable chance that d is a nontrivial factor of N.

Table 4.4: A three step factorization procedure

find such a, we observe that

$$\begin{aligned}
1869^2 &\equiv 750000 \pmod{914387} &\quad\text{and}\quad& 750000 = 2^4 \cdot 3 \cdot 5^6\\
1909^2 &\equiv 901120 \pmod{914387} &\quad\text{and}\quad& 901120 = 2^{14} \cdot 5 \cdot 11\\
3387^2 &\equiv 499125 \pmod{914387} &\quad\text{and}\quad& 499125 = 3 \cdot 5^3 \cdot 11^3
\end{aligned}$$

None of the numbers on the right is a square, but if we multiply then together, then we do get a square. Thus

$$\begin{aligned}
1869^2 \cdot 1909^2 \cdot 3387^2 &\equiv 750000 \cdot 901120 \cdot 499125 \pmod{914387}\\
&\equiv (2^4 \cdot 3 \cdot 5^6)(2^{14} \cdot 5 \cdot 11)(3 \cdot 5^3 \cdot 11^3) \pmod{914387}\\
&= (2^9 \cdot 3 \cdot 5^5 \cdot 11^2)^2\\
&= 580800000^2\\
&\equiv 164255^2 \pmod{914387}
\end{aligned}$$

We further note that $1869 \cdot 1909 \cdot 3387 \equiv 9835 \pmod{914387}$, so we compute

$$\gcd(914387, 9835 - 164255) = \gcd(914387, 154420) = 1103.$$

Eureka! We have factored $914387 = 1103 \cdot 829$.

Example 4.37. We do a second example to illustrate a potential pitfall in this method. We will factor $N = 636683$. After some searching, we find

$$\begin{aligned} 1387^2 &\equiv 13720 \pmod{636683} \qquad \text{and} \qquad 13720 = 2^3 \cdot 5 \cdot 7^3 \\ 2774^2 &\equiv 54880 \pmod{636683} \qquad \text{and} \qquad 54880 = 2^5 \cdot 5 \cdot 7^3 \end{aligned}$$

Multiplying these two values gives a square,

$$1387^2 \cdot 2774^2 \equiv 13720 \cdot 54880 = (2^4 \cdot 5 \cdot 7^3)^2 = 27440^2.$$

Unfortunately, when we compute the gcd, we find that

$$\gcd(636683, 1387 \cdot 2774 - 27440) = \gcd(636683, 3820098) = 636683.$$

Thus after all our work, we have made no progress! However, all is not lost, we can gather more values of a and try to find a different relation. Extending the above list, we discover that

$$3359^2 \equiv 459270 \pmod{636683} \qquad \text{and} \qquad 459270 = 2 \cdot 3^8 \cdot 5 \cdot 7$$

Multiplying 1387^2 and 3359^2 gives

$$1387^2 \cdot 3359^2 \equiv 13720 \cdot 459270 = (2^2 \cdot 3^4 \cdot 5 \cdot 7^2)^2 = 79380^2,$$

and now when we compute the gcd, we find

$$\gcd(636683, 1387 \cdot 3359 - 79380) = \gcd(636683, 4579553) = 787.$$

This gives the factorization $N = 787 \cdot 809$.

Remark 4.38. How many tries is it likely to take before we find a factor of N? The most difficult case is when $N = pq$ is a product of two primes. Suppose that we can find more-or-less random values of a and b satisfying $a^2 \equiv b^2 \pmod{N}$. What are our chances of finding a nontrivial factor of N when we compute $\gcd(a - b, N)$? We know that

$$(a - b)(a + b) = a^2 - b^2 = kN = kpq \qquad \text{for some value of } k.$$

The prime p must divide at least one of $a - b$ or $a + b$, and it has approximately equal probability of dividing each. Similarly for q. We win if $a - b$ is divisible by exactly one of p and q, which happens approximately 50% of the time. Hence if we can actually generate random a's and b's satisfying $a^2 \equiv b^2 \pmod{N}$, then it won't take us long to find a factor of N.

The factorization procedure described in Table 4.4 consists of three steps:

1.	Relation Building
2.	Elimination
3.	GCD Computation

There is really nothing to say about Step 3, since we already know how to efficiently compute $\gcd(N, a-b)$ in $O(\ln N)$ steps. On the other hand, there is so much to say about Relation Building that we postpone our discussion until Section 4.7. Finally, what of Step 2, the elimination step?

We suppose that each of the numbers $a_1, \ldots, a_r$ found in Step 1 has the property that $c_i \equiv a_i \pmod{m}$ factors into a product of small primes, say each c_i is a product of primes chosen from the set of the first t primes $\{p_1, p_2, p_3, \ldots, p_t\}$. This means that there are exponents e_{ij} so that

$$\begin{aligned} c_1 &= p_1^{e_{11}} p_2^{e_{12}} p_3^{e_{13}} \cdots p_t^{e_{1t}} \\ c_2 &= p_1^{e_{21}} p_2^{e_{22}} p_3^{e_{23}} \cdots p_t^{e_{2t}} \\ \vdots \quad & \qquad \vdots \\ c_r &= p_1^{e_{r1}} p_2^{e_{r2}} p_3^{e_{r3}} \cdots p_t^{e_{rt}} \end{aligned}$$

Our goal is to take a product of some of the c_i's in order to make each prime on the righthand side appear to an even power. Another way to phrase this problem is that of finding $u_1, u_2, \ldots, u_r \in \{0, 1\}$ so that

$$c_1^{u_1} \cdot c_2^{u_2} \cdots c_r^{u_r} \quad \text{is a perferct square.}$$

In other words, we take $u_i = 1$ if we want to include c_i in the product and we take $u_i = 0$ if we do not want to include c_i in the product.

Writing out the product in terms of the prime factorizations of $c_1, \ldots, c_r$ gives the rather messy expression

$$\begin{aligned} &c_1^{u_1} \cdot c_2^{u_2} \cdots c_r^{u_r} \\ &\quad = (p_1^{e_{11}} p_2^{e_{12}} p_3^{e_{13}} \cdots p_t^{e_{1t}})^{u_1} \cdot (p_1^{e_{21}} p_2^{e_{22}} p_3^{e_{23}} \cdots p_t^{e_{2t}})^{u_2} \cdots (p_1^{e_{r1}} p_2^{e_{r2}} p_3^{e_{r3}} \cdots p_t^{e_{rt}})^{u_r} \\ &\quad = p_1^{e_{11}u_1 + e_{21}u_2 + \cdots + e_{r1}u_r} \cdot p_2^{e_{12}u_1 + e_{22}u_2 + \cdots + e_{r2}u_r} \cdots p_t^{e_{1t}u_1 + e_{2t}u_2 + \cdots + e_{rt}u_r}. \qquad (4.13) \end{aligned}$$

You may find this clearer written using summation and product notation,

$$\prod_{i=1}^{r} c_i^{u_i} = \prod_{j=1}^{t} p_j^{\sum_{i=1}^{r} e_{ij} u_i}. \qquad (4.14)$$

In any case, our goal is to choose $u_1, \ldots, u_r$ so that all of the exponents in (4.13) (or equivalently in (4.14)) are even.

To recapitulate, we are given integers

$$e_{11}, e_{12}, \ldots, e_{1t}, e_{21}, e_{22}, \ldots, e_{2t}, \ldots, e_{r1}, e_{r2}, \ldots, e_{rt}$$

and we are searching for integers $u_1, u_2, \ldots, u_r$ so that

$$\begin{aligned} e_{11}u_1 + e_{21}u_2 + \cdots + e_{r1}u_r &\equiv 0 \pmod{2} \\ e_{12}u_1 + e_{22}u_2 + \cdots + e_{r2}u_r &\equiv 0 \pmod{2} \\ \vdots \qquad\qquad & \quad \vdots \\ e_{1t}u_1 + e_{2t}u_2 + \cdots + e_{rt}u_r &\equiv 0 \pmod{2} \end{aligned} \tag{4.15}$$

You have undoubtedly recognized that the system of congruences (4.15) is simply a system of linear equations over the finite field $\mathbb{F}_2$. Hence standard techniques from linear algebra, such as Gaussian elimination, can be used to solve these equations. In fact, doing linear algebra in the field $\mathbb{F}_2$ is much easier than doing linear algebra in the field $\mathbb{R}$, since there is no need to worry about roundoff errors.

Example 4.39. We illustrate the linear algebra elimination step by factoring the number

$$N = 9788111.$$

We look for numbers a with the property that $a^2 \bmod N$ is 50-smooth, i.e. numbers a such that $a^2 \bmod N$ is equal to a product of primes in the set

$$\{2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47\}.$$

The top part of Table 4.5 lists the 20 numbers $a_1, a_2, \ldots, a_{20}$ between 3129 and 4700 having this property,[4] together with the factorization of each

$$c_i \equiv a_i^2 \pmod{N}.$$

The bottom part of Table 4.5 turns the condition that a product $c_1^{u_1} c_2^{u_2} \cdots c_{20}^{u_20}$ is a square into a system of linear equation for $(u_1, u_2, \ldots, u_{20})$ as described by (4.15). For notational convenience, we have written the system of linear equations in Table 4.5 in matrix form.

[4] Why do we start with $a = 3129$? The answer is that unless a^2 is larger than N, then there is no cancellation in $a^2 \bmod N$, so we cannot hope to gain any information. The value 3129 comes from the fact that $\sqrt{N} = \sqrt{9788111} \approx 3128.6$.

$$\begin{array}{llcl}
3129^2 \equiv 2530 & \pmod{9788111} & \text{and} & 2530 = 2 \cdot 5 \cdot 11 \cdot 23 \\
3130^2 \equiv 8789 & \pmod{9788111} & \text{and} & 8789 = 11 \cdot 17 \cdot 47 \\
3131^2 \equiv 15050 & \pmod{9788111} & \text{and} & 15050 = 2 \cdot 5^2 \cdot 7 \cdot 43 \\
3166^2 \equiv 235445 & \pmod{9788111} & \text{and} & 235445 = 5 \cdot 7^2 \cdot 31^2 \\
3174^2 \equiv 286165 & \pmod{9788111} & \text{and} & 286165 = 5 \cdot 11^3 \cdot 43 \\
3215^2 \equiv 548114 & \pmod{9788111} & \text{and} & 548114 = 2 \cdot 7^3 \cdot 17 \cdot 47 \\
3313^2 \equiv 1187858 & \pmod{9788111} & \text{and} & 1187858 = 2 \cdot 7^2 \cdot 17 \cdot 23 \cdot 31 \\
3449^2 \equiv 2107490 & \pmod{9788111} & \text{and} & 2107490 = 2 \cdot 5 \cdot 7^2 \cdot 11 \cdot 17 \cdot 23 \\
3481^2 \equiv 2329250 & \pmod{9788111} & \text{and} & 2329250 = 2 \cdot 5^3 \cdot 7 \cdot 11^3 \\
3561^2 \equiv 2892610 & \pmod{9788111} & \text{and} & 2892610 = 2 \cdot 5 \cdot 7 \cdot 31^2 \cdot 43 \\
4394^2 \equiv 9519125 & \pmod{9788111} & \text{and} & 9519125 = 5^3 \cdot 7 \cdot 11 \cdot 23 \cdot 43 \\
4425^2 \equiv 4403 & \pmod{9788111} & \text{and} & 4403 = 7 \cdot 17 \cdot 37 \\
4426^2 \equiv 13254 & \pmod{9788111} & \text{and} & 13254 = 2 \cdot 3 \cdot 47^2 \\
4432^2 \equiv 66402 & \pmod{9788111} & \text{and} & 66402 = 2 \cdot 3^2 \cdot 7 \cdot 17 \cdot 31 \\
4442^2 \equiv 155142 & \pmod{9788111} & \text{and} & 155142 = 2 \cdot 3^3 \cdot 13^2 \cdot 17 \\
4468^2 \equiv 386802 & \pmod{9788111} & \text{and} & 386802 = 2 \cdot 3^3 \cdot 13 \cdot 19 \cdot 29 \\
4551^2 \equiv 1135379 & \pmod{9788111} & \text{and} & 1135379 = 7^2 \cdot 17 \cdot 29 \cdot 47 \\
4595^2 \equiv 1537803 & \pmod{9788111} & \text{and} & 1537803 = 3^2 \cdot 17 \cdot 19 \cdot 23^2 \\
4651^2 \equiv 2055579 & \pmod{9788111} & \text{and} & 2055579 = 3 \cdot 23 \cdot 31^3 \\
4684^2 \equiv 2363634 & \pmod{9788111} & \text{and} & 2363634 = 2 \cdot 3^3 \cdot 7 \cdot 13^2 \cdot 37
\end{array}$$

Relation Gathering Step

$$\begin{pmatrix}
1&0&1&0&0&1&1&1&1&1&0&0&1&1&1&1&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&1&0&0&1&1\\
1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&1&0&0&1&1&1&1&0&1&0&0&0&0&0&1\\
1&1&0&0&1&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&1&0&0&0&1&1&1&0&0&0&1&0&1&1&0&1&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0\\
1&0&0&0&0&0&1&1&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0
\end{pmatrix}
\begin{pmatrix} u_1\\u_2\\u_3\\u_4\\u_5\\u_6\\u_7\\u_8\\u_9\\u_{10}\\u_{11}\\u_{12}\\u_{13}\\u_{14}\\u_{15}\\u_{16}\\u_{17}\\u_{18}\\u_{19}\\u_{20} \end{pmatrix}
\equiv
\begin{pmatrix} 0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0 \end{pmatrix}
\pmod{2}$$

Linear Algebra Elimination Step

Table 4.5: Factorization of $N = 9788111$

The next step is to solve the system of linear equations in Table 4.5. This can be done by standard Gaussian elimination, always keeping in mind that all computations are done modulo 2. The set of solutions turns out to be an $\mathbb{F}_2$-vector space of dimension 8. A basis for the set of solutions is given by the following 8 vectors, where we have written the vectors horizontally, rather than vertically, in order to save space.

$$\begin{aligned}
\mathbf{v}_1 &= (0,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0)\\
\mathbf{v}_2 &= (0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0)\\
\mathbf{v}_3 &= (0,0,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0)\\
\mathbf{v}_4 &= (1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0)\\
\mathbf{v}_5 &= (1,0,1,0,1,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0)\\
\mathbf{v}_6 &= (1,0,0,0,0,0,0,1,0,0,0,0,1,0,1,0,0,0,0,0)\\
\mathbf{v}_7 &= (1,0,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,1,0)\\
\mathbf{v}_8 &= (1,0,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,0,0,1)
\end{aligned}$$

Each of the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_8$ gives a congruence $a^2 \equiv b^2 \pmod{N}$ that has the potential to provide a factorization of N. For example, $\mathbf{v}_1$ says that if we multiply the 3^{rd}, 5^{th}, and 9^{th} numbers in the list at the top of Table 4.5, we will get a square, and indeed we find that

$$\begin{aligned}
&3131^2 \cdot 3174^2 \cdot 3481^2\\
&\qquad \equiv (2 \cdot 5^2 \cdot 7 \cdot 43)(5 \cdot 11^3 \cdot 43)(2 \cdot 5^3 \cdot 7 \cdot 11^3) \pmod{9788111}\\
&\qquad = (2 \cdot 5^3 \cdot 7 \cdot 11^3 \cdot 43)^2\\
&\qquad = 100157750^2.
\end{aligned}$$

Next we compute

$$\gcd(9788111, 3131 \cdot 3174 \cdot 3481 - 100157750) = 9788111,$$

which gives back the original number N. This is unfortunate, but all is not lost, since we have seven more independent solutions to our system of linear equations. Trying each of them in turn, the results are listed in Table 4.6.

Seven of the eight solutions to the system of linear equations yield no useful information about N, the resulting gcd being either 1 or N. However, one solution, listed in the penultimate box of Table 4.6, leads to a nontrivial factorization of N. Thus 2741 is a factor of N, and hence $N = 9788111 = 2741 \cdot 3571$. Since both 2741 and 3571 are prime, this gives the complete factorization of N.

$\mathbf{v}_1 = (0,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0)$ $3131^2 \cdot 3174^2 \cdot 3481^2 \equiv (2 \cdot 5^3 \cdot 7 \cdot 11^3 \cdot 43)^2$ $= 100157750^2$ $\gcd(9788111, 3131 \cdot 3174 \cdot 3481 - 100157750) = 9788111$
$\mathbf{v}_2 = (0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$ $3130^2 \cdot 3131^2 \cdot 3166^2 \cdot 3174^2 \cdot 3215^2 \equiv (2 \cdot 5^2 \cdot 7^3 \cdot 11^2 \cdot 17 \cdot 31 \cdot 43 \cdot 47)^2$ $= 2210173785050^2$ $\gcd(9788111, 3130 \cdot 3131 \cdot 3166 \cdot 3174 \cdot 3215 - 2210173785050) = 1$
$\mathbf{v}_3 = (0,0,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0)$ $3131^2 \cdot 3166^2 \cdot 3561^2 \equiv (2 \cdot 5^2 \cdot 7^2 \cdot 31^2 \cdot 43)^2$ $= 101241350^2$ $\gcd(9788111, 3131 \cdot 3166 \cdot 3561 - 101241350) = 9788111$
$\mathbf{v}_4 = (1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0)$ $3129^2 \cdot 3131^2 \cdot 4394^2 \equiv (2 \cdot 5^3 \cdot 7 \cdot 11 \cdot 23 \cdot 43)^2$ $= 19038250^2$ $\gcd(9788111, 3129 \cdot 3131 \cdot 4394 - 19038250) = 9788111$
$\mathbf{v}_5 = (1,0,1,0,1,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0)$ $3129^2 \cdot 3131^2 \cdot 3174^2 \cdot 3313^2 \cdot 4432^2 \equiv (2^2 \cdot 3 \cdot 5^2 \cdot 7^2 \cdot 11^2 \cdot 17 \cdot 23 \cdot 31 \cdot 43)^2$ $= 927063776100^2$ $\gcd(9788111, 3129 \cdot 3131 \cdot 3174 \cdot 3313 \cdot 4432 - 927063776100) = 1$
$\mathbf{v}_6 = (1,0,0,0,0,0,0,1,0,0,0,0,1,0,1,0,0,0,0,0)$ $3129^2 \cdot 3449^2 \cdot 4426^2 \cdot 4442^2 \equiv (2^2 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 23 \cdot 47)^2$ $= 3311167860^2$ $\gcd(9788111, 3129 \cdot 3449 \cdot 4426 \cdot 4442 - 3311167860) = 1$
$\mathbf{v}_7 = (1,0,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,1,0)$ $3129^2 \cdot 3313^2 \cdot 3449^2 \cdot 4426^2 \cdot 4651^2 \equiv (2^2 \cdot 3 \cdot 5 \cdot 7^2 \cdot 11 \cdot 17 \cdot 23^2 \cdot 31^2 \cdot 47)^2$ $= 13136082114540^2$ $\gcd(9788111, 3129 \cdot 3313 \cdot 3449 \cdot 4426 \cdot 4651 - 13136082114540) = \mathbf{2741}$
$\mathbf{v}_8 = (1,0,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,0,0,1)$ $3129^2 \cdot 3449^2 \cdot 4425^2 \cdot 4426^2 \cdot 4684^2 \equiv (2^2 \cdot 3^2 \cdot 5 \cdot 7^2 \cdot 11 \cdot 13 \cdot 17 \cdot 23 \cdot 37 \cdot 47)^2$ $= 857592475740^2$ $\gcd(9788111, 3129 \cdot 3449 \cdot 4425 \cdot 4426 \cdot 4684 - 857592475740) = 1$

Table 4.6: Factorization of $N = 9788111$ — Computation of gcd's

Remark 4.40. In order to factor a large number N, it may be necessary to use a set containing hundreds of thousands, or even millions, of primes. Then the system (4.15) contains millions of linear equations, and even working in the field $\mathbb{F}_2$, it can be very difficult to solve general systems of this size. However, it turns out that the systems of linear equations used in factorization are quite *sparse*, which means that most of their coefficients are zero. (This is plausible because if a number A is a product of primes smaller than B, then one expects A to be a product of approximately $\log(A)/\log(B)$ primes.) There are special techniques for solving sparse systems of linear equations that are much more efficient than ordinary Gaussian elimination, see for example [12, 28].

4.7 Smooth numbers, sieves and building relations for factorization

4.7.1 Smooth numbers

The relation building step in our three step factorization procedure (Table 4.4) requires us to find many integers with the property that $a^2 \bmod m$ factors as a product of small primes. As noted at the end of Section 4.5, these highly factorizable numbers are given a name.

Definition. An integer M is called *B-smooth* if all of its prime factors are less then or equal to B. We denote the set of B-smooth numbers by

$$\mathbb{Z}(B) = \{M : \text{every prime } p \text{ dividing } M \text{ satisfies } p \le B\}.$$

Example 4.41. Here are the first few 5-smooth numbers and the first few numbers that are not 5-smooth:

$$\begin{aligned} \text{5-smooth:}\quad & 2,3,4,5,6,8,9,10,12,15,16,18,20,24,25,27,30,32,36,\ldots \\ \text{Not 5-smooth:}\quad & 7,11,13,14,17,19,21,22,23,26,28,29,31,33,34,35,37,\ldots \end{aligned}$$

Definition. The function $\psi(X, B)$ counts B-smooth numbers,

$$\psi(X, B) = \text{Number of } B\text{-smooth integers between 1 and } X.$$

For example,

$$\psi(25, 5) = 15,$$

since the 5-smooth numbers between 1 and 25 are the 15 numbers

$$2, 3, 4, 5, 6, 8, 9, 10, 12, 15, 16, 18, 20, 24, 25.$$

Theorem 4.42 (Canfield, Erdös, Pomerance)**.** *Let $L(X)$ be the function*

$$L(X) = e^{\sqrt{(\log X)(\log\log X)}}.$$

Then for any fixed value of c with $0 < c < 1$,

$$\psi\bigl(X, L(X)^c\bigr) \approx X \cdot L(X)^{-1/2c} \qquad \textit{as } X \to \infty.$$

Proof. The upper bound in this important result was proven by de Bruijn, and the full result is due to Canfield, Erdös, Pomerance [9]. They actually prove something a bit stronger than we have stated. They show that for a fixed $\epsilon > 0$, if X and Y go to infinity while satisfying

$$(\log X)^\epsilon < \log Y < (\log X)^{1-\epsilon},$$

and if we let

$$u = \frac{\log X}{\log Y},$$

then

$$\psi(X, Y) = X \cdot u^{-u(1+o(1))},$$

where $o(1)$ denotes a quantity that goes to 0 as X and Y go to infinity. Taking $Y = L(X)^c$ gives the result that we have stated, see Exercise 4.27. □

Remark 4.43. The function $L(X) = e^{\sqrt{(\log X)(\log\log X)}}$ and other similar functions appear prominently in the theory of factorization due to their close relationship to the distribution of smooth numbers. It is thus important to understand how fast $L(X)$ grows as a function of X. A function $f(X)$ is said to grow *exponentially* if there are positive constants α and β so that

$$X^\alpha \ll f(X) \ll X^\beta$$

and it is said to grow *polynomially* if there is are positive constants α and β so that

$$(\log X)^\alpha \ll f(X) \ll (\log X)^\beta.$$

The function $L(X)$ falls between these two categories, so it is called a *subexponential function*. In other words, for every positive constant α, no matter how large, and for every positive constant β, no matter how small, it is true that

$$(\log X)^\alpha \ll L(X) \ll X^\beta. \tag{4.16}$$

Thus $L(X)$ grows slower than exponentially, but faster than polynomially. We leave it to you to prove (4.16) in Exercise 4.25

Suppose that we attempt to factor N by searching for values of $a^2 \pmod{N}$ that are B-smooth. In order to perform the linear equation elimination step, we need (at least) as many B-smooth numbers as there are primes less than B. In other words, we need at least $\pi(B)$ such numbers. We can use the Prime Number Theorem and the formula for $\psi(X,B)$ given in Theorem 4.42 to choose a good value for B and to estimate how many B-smooth numbers we need in order to factor N.

Proposition 4.44. *Let $L(X) = e^{\sqrt{(\log X)(\log\log X)}}$ be as in Theorem 4.42 and let N be a number to be factored. Set $B = L(N)^{1/\sqrt{2}}$. Then we expect to check approximately $L(N)^{\sqrt{2}}$ random numbers of the form $a^2 \pmod{N}$ in order to find $\pi(B)$ of them that are B-smooth.*

Proof. The probability that a random number a^2 modulo N is B-smooth is $\psi(N,B)/N$. We want to find approximately $\pi(B)$ of these B-smooth numbers. Thus we need to check approximately

$$\frac{\pi(B)}{\psi(N,B)/N} \quad \text{numbers} \tag{4.17}$$

to find enough relations to factor N. We want to choose B so as to minimize this function.

Theorem 4.42 says that $\psi(N, L(N)^c)/N \approx L(N)^{-1/2c}$, so we set $B = L(N)^c$ and search for the value of c that minimizes (4.17). The Prime Number Theorem 4.20 tells us that $\pi(B) \approx B/\log(B)$, so (4.17) is equal to

$$\frac{\pi(L(N)^c)}{\psi(N, L(N)^c)/N} \approx \frac{L(N)^c}{c\log L(N)} \cdot \frac{1}{L(N)^{-1/2c}} = L(N)^{c+\frac{1}{2c}} \cdot \frac{1}{c\log L(N)}.$$

The factor $L(N)^{c+1/2c}$ is the largest piece, so we want to choose the value of c that minimizes the quantity $c + \frac{1}{2c}$. This is an elementary calculus problem. We find that it is minimized when $c = \frac{1}{\sqrt{2}}$ and that the minimum value is $\sqrt{2}$. Thus we should choose $B \approx L(N)^{1/\sqrt{2}}$ and we need to check approximately $L(N)^{\sqrt{2}}$ values in order to find enough relations to factor N. □

N	$\log L(N)$	$L(N)$
2^{100}	17.141	$2^{24.73}$
2^{250}	29.888	$2^{43.12}$
2^{500}	45.020	$2^{64.95}$
2^{1000}	67.335	$2^{97.14}$
2^{2000}	100.145	$2^{144.48}$

Table 4.7: The growth of $L(N) = e^{\sqrt{(\log N)(\log\log N)}}$

Remark 4.45. Proposition 4.44 says that we need to check approximately $L(N)^{\sqrt{2}}$ randomly chosen numbers modulo N in order to find enough smooth numbers to factor N. There is an easy way to decrease the search time. Rather than choosing random numbers, we instead select numbers a that are only a little bit larger than $\sqrt{N}$. Then $a^2 \pmod{N}$ is $O(\sqrt{N}\,)$, so it is more likely to be B-smooth, which descreases the estimate in Proposition 4.44 from $L(N)^{\sqrt{2}}$ to $L(N)$. See Exercise 4.28 for details.

On the other hand, in estimating the work needed to factor N, we have completely ignored the work required to check if a given number is B-smooth. For example, if we check for B-smoothness using trial division, i.e. dividing by each prime less than B, then it takes approximately $\pi(B)$ trial divisions to check for B-smoothness. Taking that work into account in the proof of Proposition 4.44, we find that it takes approximately $L(N)^{\sqrt{2}}$ trial divisions to find enough smooth numbers to factor N, even using values of $a \approx \sqrt{N}$ as in the first part of this remark.

The Quadratic Sieve, which we describe in Section 4.7.2, avoids trial divisions and reduces the running time back down to $L(N)$. It is instructive to see how $L(N)$ grows as N increases. We list a few values in Table 4.7, and in Exercise 4.24 we ask you to estimate how long it takes to perform $L(N)$ operations on a moderately fast computer. For a number of years it was thought that no factorization algorithm could take fewer than a power of $L(N)$ steps, but the invention of the Number Field Sieve (Section 4.7.3) showed this to be incorrect. The Number Field Sieve achieves a running time faster than a power of $L(N)$ by moving beyond the realm of the ordinary integers.

4.7.2 The quadratic sieve

In this section we address the final fundamental problem whose solution is required in order to factor large numbers via the difference of squares method described in Section 4.6:

How can we efficiently find many numbers $a > \sqrt{N}$ so that $a^2 \pmod{N}$ is B-smooth?

If we take B very large, then there are lots of B-smooth numbers, which is good. However, we need to find approximately $\pi(B)$ such numbers, so taking B very large is bad. On the other hand, if we choose B to be small, we need only a few B-smooth numbers, but then B-smooth numbers will be rare.

Definition. The set of primes less than B (or sometimes the set of prime powers less than B) is called the *factor base.*

Example 4.46 (Continued Fraction Factorization Algorithm). An early approach to finding B-smooth squares modulo N was to look for fractions $\frac{a}{b}$ that are as close as possible to $\sqrt{kN}$ for $k = 1, 2, 3, \ldots$. Then $a^2 \approx b^2kN$, so $a^2 \pmod{N}$ is reasonably small and thus more likely to be B-smooth. The theory of *continued fractions* gives an algorithm for finding such $\frac{a}{b}$. See [11, §10.1] for details.

An alternative approach that turns out to be much faster in practice is to allow slightly larger values of a and to use an efficient cancelation process called a sieve to simultaneously create a large number of B-smooth $a^2 \pmod{N}$ values. We briefly describe Pomerance's *Quadratic Sieve*, which is still the fastest known method for factoring large numbers $N = pq$ up to about 2^{350}. For numbers considerably larger than this, say larger than 2^{450}, the more complicated *Number Field Sieve* holds the world record for factorization. In the remainder of this section we describe the simplest version of the quadratic sieve as an illustration of modern factorization methods. For a description of the history of sieve methods and an overview of how they work, see Pomerance's delightful article "A Tale of Two Sieves" [41].

We start with the simpler problem of rapidly finding many B-smooth numbers less than some bound X. To do this we adapt the *Sieve of Eratosthenes*, which is an ancient Greek method for making lists of prime numbers. Eratosthenes' idea for finding primes is as follows. Start by circling the first

	$\boxed{2}$	$\boxed{3}$	$\cancel{4}$	$\boxed{5}$	$\cancel{\cancel{6}}$	$\boxed{7}$	$\cancel{8}$	$\cancel{9}$	$\cancel{\cancel{10}}$	11	$\cancel{\cancel{12}}$	13	$\cancel{\cancel{14}}$	$\cancel{\cancel{15}}$	$\cancel{16}$	17	$\cancel{\cancel{18}}$	19	$\cancel{\cancel{20}}$
$\cancel{\cancel{21}}$	$\cancel{22}$	23	$\cancel{\cancel{24}}$	$\cancel{25}$	$\cancel{26}$	$\cancel{27}$	$\cancel{\cancel{28}}$	29	$\cancel{\cancel{\cancel{30}}}$	31	$\cancel{32}$	$\cancel{33}$	$\cancel{34}$	$\cancel{\cancel{35}}$	$\cancel{\cancel{36}}$	37	$\cancel{38}$	$\cancel{39}$	$\cancel{\cancel{40}}$
41	$\cancel{\cancel{\cancel{42}}}$	43	$\cancel{44}$	$\cancel{\cancel{45}}$	$\cancel{46}$	47	$\cancel{\cancel{48}}$	$\cancel{49}$	$\cancel{\cancel{50}}$	$\cancel{51}$	$\cancel{52}$	53	$\cancel{\cancel{54}}$	$\cancel{55}$	$\cancel{\cancel{56}}$	$\cancel{57}$	$\cancel{58}$	59	$\cancel{\cancel{\cancel{60}}}$
61	$\cancel{62}$	$\cancel{\cancel{63}}$	$\cancel{64}$	$\cancel{65}$	$\cancel{\cancel{66}}$	67	$\cancel{68}$	$\cancel{69}$	$\cancel{\cancel{\cancel{70}}}$	71	$\cancel{\cancel{72}}$	73	$\cancel{74}$	$\cancel{\cancel{75}}$	$\cancel{76}$	$\cancel{77}$	$\cancel{\cancel{78}}$	79	$\cancel{\cancel{80}}$
$\cancel{81}$	$\cancel{82}$	83	$\cancel{\cancel{\cancel{84}}}$	$\cancel{85}$	$\cancel{86}$	$\cancel{87}$	$\cancel{88}$	89	$\cancel{\cancel{\cancel{90}}}$	$\cancel{91}$	$\cancel{92}$	$\cancel{93}$	$\cancel{94}$	$\cancel{95}$	$\cancel{\cancel{96}}$	97	$\cancel{\cancel{98}}$	$\cancel{99}$	

Figure 4.2: The sieve of Eratosthenes

prime 2 and crossing off every larger multiple of 2. Then circle the next prime 3 and cross off every larger multiple of 3. The smallest uncircled number is 5, so circle 5 and cross off all larger multiples of 5, and so on. At the end, the circled numbers are the primes.

This sieving process is illustrated in Figure 4.2, where we have sieved all primes less than 10. The remaining uncrossed numbers in the list consist of all primes smaller than 100. Notice that some numbers are crossed off several times. For example, 6 and 12 and 18 are crossed off twice, once because they are multiples of 2 and once because they are multiples of 3. Similarly, numbers such as 30 and 42 are crossed off three times. Suppose that instead of crossing numbers off, we instead divided. That is, we begin by dividing every even number by 2, then we divide every multiple of 3 by 3, then we divide every multiple of 5 by 5, and so on. If we do this for all primes less than B, which numbers end up being divided all the way down to 1? The answer is that these are the numbers that are a product of distinct primes in B, in particular, they are B-smooth! So we end up with a list of many B-smooth numbers.

Unfortunately, we miss some B-smooth numbers, but it is easy to remedy this problem by sieving with prime powers. Thus after sieving by 3, rather than proceeding to 5, we first sieve by 4. To do this, we cancel an additional factor of 2 from every multiple of 4. (Notice that we've already canceled 2 from these numbers, since they are even, so we can only cancel one additional factor of 2.) If we do this, then at the end, the B-smooth numbers less than X are precisely the numbers that have been reduced to 1. One can show that the total number of divisions required is approximately $X \ln\ln(B)$. The double logarithm function, $\ln\ln(B)$, grows extremely slowly, so the average number of divisions required to check each individual number for smoothness is approximately constant.

However, our goal is not to make a list of numbers from 1 to X that are B-smooth. What we need is a list of numbers of the form $a^2 \pmod{N}$ that are B-smooth. Let $F(T)$ be the polynomial

$$F(T) = T^2 - N.$$

We search for such numbers by starting with a value of a that is slightly larger than $\sqrt{N}$ and looking at the list of numbers

$$F(a), F(a+1), F(a+2), \ldots, F(b). \tag{4.18}$$

The idea is to find the B-smooth numbers in this list by sieving away the primes smaller than B and seeing which numbers in the list get sieved all the way down to 1.

Let p be a prime in our factor base. Which of the numbers in the list (4.18) are divisible by p? In other words, which numbers t between a and b satisfy

$$t^2 \equiv N \pmod{p}. \tag{4.19}$$

If the congruence (4.19) has no solutions, then we discard the prime p, since it can divide none of the numbers in the list 4.18. Otherwise the congruence (4.19) has two solutions (see Exercise 1.25 on page 50), which we denote by

$$t = \alpha_p \qquad \text{and} \qquad t = \beta_p.$$

(If $p = 2$, there is only one solution α_p.) It follows that each of the numbers

$$F(\alpha_p), F(\alpha_p + p), F(\alpha_p + 2p), F(\alpha_p + 3p), \ldots$$

and each of the numbers

$$F(\beta_p), F(\beta_p + p), F(\beta_p + 2p), F(\beta_p + 3p), \ldots$$

is divisible by p. Thus we can sieve away a factor of p from every p^{th} entry in the list (4.18) starting with the smallest a value satisfying $a \equiv \alpha_p \pmod{p}$, and similarly we can sieve away a factor of p from every p^{th} entry in the list (4.18) starting with the smallest a value satisfying $a \equiv \beta_p \pmod{p}$.

Example 4.47. We illustrate the quadratic sieve applied to the composite number $N = 221$. The smallest number whose square is larger than N is $a = \lfloor 221 \rfloor + 1 = 15$. We set

$$F(T) = T^2 - 221$$

and sieve the numbers from $F(15) = 4$ up to $F(30) = 679$ using successively the prime powers from 2 to 7. The initial list of numbers $T^2 - N$ is[5]

4 35 68 103 140 179 220 263 308 355 404 455 508 563 620 679

We first sieve by $p = 2$, which means that we cancel 2 from every second entry in the list. This gives

4	35	68	103	140	179	220	263	308	355	404	455	508	563	620	679
$\downarrow_2$		$\downarrow_2$		$\downarrow_2$		$\downarrow_2$		$\downarrow_2$		$\downarrow_2$		$\downarrow_2$		$\downarrow_2$	
2	35	34	103	70	179	110	263	154	355	202	455	254	563	310	679

Next we sieve by $p = 3$. However, it turns out that the congruence

$$t^2 \equiv 221 \equiv 2 \pmod{3}$$

has no solutions, so none of the entries in our list are divisible by 3.

We move on to the prime power 2^2. Every odd number is a solution of the congruence

$$t^2 \equiv 221 \equiv 1 \pmod{4},$$

which means that we can sieve another factor of 2 from every second entry in our list. We put a small 4 next to the sieving arrows to indicate that in this step we are sieving by 4, although we only cancel a factor of 2 from each entry.

2	35	34	103	70	179	110	263	154	355	202	455	254	563	310	679
$\downarrow_4$		$\downarrow_4$		$\downarrow_4$		$\downarrow_4$		$\downarrow_4$		$\downarrow_4$		$\downarrow_4$		$\downarrow_4$	
1	35	17	103	35	179	55	263	77	355	101	455	127	563	155	679

Next we move on to $p = 5$. The congruence

$$t^2 \equiv 221 \equiv 1 \pmod{5}$$

has two solutions $\alpha_5 = 1$ and $\beta_5 = 4$ modulo 5. So starting with $F(16)$, since 16 is the first t value that is congruent to 1 modulo 5, we find that

[5] In practice when N is large, the t values used in the quadratic sieve are close enough to $\sqrt{N}$ that the value of $t^2 - N$ is between 1 and N. For our small numerical example, this is not the case, so it would be more efficient to reduce our values of t^2 modulo N, rather than merely subtracting N from t^2. However, since our aim is illumination, not efficiency, we will pretend that there is no advantage to subtracting additional multiples of N from $t^2 - N$.

every 5th entry in our list is divisible by 5. Sieving out those factors of 5 gives

1	35	17	103	35	179	55	263	77	355	101	455	127	563	155	679
	$\downarrow_5$					$\downarrow_5$					$\downarrow_5$				
1	7	17	103	35	179	11	263	77	355	101	91	127	563	155	679

Similarly, every 5th entry starting with $F(19)$ is divisible by 5, so we sieve out those factors

1	7	17	103	35	179	11	263	77	355	101	91	127	563	155	679
				$\downarrow_5$					$\downarrow_5$					$\downarrow_5$	
1	7	17	103	7	179	11	263	77	71	101	91	127	563	31	679

To conclude our example, we sieve the prime $p = 7$. The congruence

$$t^2 \equiv 221 \equiv 4 \pmod{7}$$

has the two solutions $\alpha_7 = 2$ and $\beta_7 = 5$. We thus sieve 7 away from every 7th entry starting with $F(16)$ and also every 7th entry starting with $F(19)$. This yields

1	7	17	103	7	179	11	263	77	71	101	91	127	563	31	679
	$\downarrow_7$							$\downarrow_7$							$\downarrow_7$
1	1	17	103	7	179	11	263	11	71	101	91	127	563	31	679
				$\downarrow_7$							$\downarrow_7$				
1	1	17	103	1	179	11	263	11	71	101	13	127	563	31	679

Notice that the original entries

$$F(15) = 4, \qquad F(16) = 35, \qquad \text{and} \qquad F(19) = 140$$

have been sieved all the way down to 1. This tells us that

$$F(15) = 15^2 - 221, \qquad F(16) = 16^2 - 221, \qquad \text{and} \qquad F(19) = 19^2 - 221$$

are each a product of small primes, so we have discovered several squares modulo 221 that are products of small primes,

$$\begin{aligned} 15^2 &\equiv 2^2 \pmod{221}, \\ 16^2 &\equiv 5 \cdot 7 \pmod{221}, \\ 19^2 &\equiv 2^2 \cdot 5 \cdot 7 \pmod{221}. \end{aligned} \tag{4.20}$$

We can use the congruences (4.20) to obtain various relations between squares. For example,[6]

$$(16 \cdot 19)^2 \equiv (2 \cdot 5 \cdot 7)^2 \pmod{221}.$$

Computing

$$\gcd(16 \cdot 19 - 2 \cdot 5 \cdot 7, 221) = \gcd(234, 221) = 13$$

gives a nontrivial factor of 221.

We have successfully factored $N = 221$, but to illustrate the sieving process further, we continue sieving up to $B = 11$. The next prime power to sieve is 3^2. However, the fact that $t^2 \equiv 221 \pmod{3}$ has no solutions means that $t^2 \equiv 221 \pmod{9}$ also has no solutions, so we move on to the prime $p = 11$.

The congruence $t^2 \equiv 221 \equiv 1 \pmod{11}$ has the solutions $\alpha_{11} = 1$ and $\beta_{11} = 10$, which allows us to sieve a factor of 11 from $F(23)$ and from $F(21)$. We recapitulate the entire sieving process in Figure 4.3, where the top row gives values of t and the subsequent rows sieve the values of $F(t) = t^2 - 221$ using prime powers up to 11. Notice that two more entries, $F(21)$ and $F(23)$, have been sieved down to 1, which gives us two additional relations

$$F(21) \equiv 21^2 \equiv 2^2 \cdot 5 \cdot 11 \pmod{221} \quad \text{and} \quad F(23) \equiv 23^2 \equiv 2^2 \cdot 7 \cdot 11 \pmod{221}.$$

We can combine these relations with the earlier relations (4.20) to obtain new square equalities, for example

$$(19 \cdot 21 \cdot 23)^2 \equiv (2^3 \cdot 5 \cdot 7 \cdot 11)^2 \pmod{221},$$

which gives another way to factor

$$\gcd(19 \cdot 21 \cdot 23 - 2^3 \cdot 5 \cdot 7, 221) = \gcd(6097, 221) = 13.$$

Remark 4.48. If p is an odd prime, then the congruence $t^2 \equiv N \pmod{p}$ has either 0 or 2 solutions modulo p. Further, the congruences $t^2 \equiv N \pmod{p^i}$ modulo higher powers of p all have the same number of solutions. This

[6] You may have noticed that it is even easier to use the fact that 15^2 is itself congruent to a square modulo 221, yielding $\gcd(15 - 2, 221) = 13$. In practice, the true power of the quadratic sieve only appears when it is applied to numbers much too large to describe fully in a textbook example.

15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
4	35	68	103	140	179	220	263	308	355	404	455	508	563	620	679
$\downarrow_2$		$\downarrow_2$		$\downarrow_2$		$\downarrow_2$		$\downarrow_2$		$\downarrow_2$		$\downarrow_2$		$\downarrow_2$	
2	35	34	103	70	179	110	263	154	355	202	455	254	563	310	679
$\downarrow_4$		$\downarrow_4$		$\downarrow_4$		$\downarrow_4$		$\downarrow_4$		$\downarrow_4$		$\downarrow_4$		$\downarrow_4$	
1	35	17	103	35	179	55	263	77	355	101	455	127	563	155	679
	$\downarrow_5$					$\downarrow_5$					$\downarrow_5$				
1	7	17	103	35	179	11	263	77	355	101	91	127	563	155	679
				$\downarrow_5$					$\downarrow_5$					$\downarrow_5$	
1	7	17	103	7	179	11	263	77	71	101	91	127	563	31	679
	$\downarrow_7$							$\downarrow_7$							$\downarrow_7$
1	1	17	103	7	179	11	263	11	71	101	91	127	563	31	679
				$\downarrow_7$							$\downarrow_7$				
1	1	17	103	1	179	11	263	11	71	101	13	127	563	31	679
								$\downarrow_{11}$							
1	1	17	103	1	179	11	263	1	71	101	13	127	563	31	679
						$\downarrow_{11}$									
1	1	17	103	1	179	1	263	1	71	101	13	127	563	31	679

Figure 4.3: Sieving $N = 221$ using prime powers up to $B = 11$

makes sieving odd prime powers relatively straightforward. Sieving with powers of 2 is a bit trickier, since the number of solutions may be different modulo 2, modulo 4, and modulo higher powers of 2. Further, there may be more than two solutions. For example, $t^2 \equiv N \pmod{8}$ has 4 different solutions modulo 8 if $N \equiv 1 \pmod{8}$. So although sieving powers of 2 is not intrinsically difficult, it must be dealt with as a special case.

Remark 4.49. There are many implementation ideas that can be used to greatly increase the practical speed of the quadratic sieve, i.e. although the running time remains a constant multiple of $L(N)$, the multiple is significantly reduced. Probably the most important idea is the use of approximate logarithms to replace division by subtraction. A time consuming part of the sieve is the necessity to divide every p^{th} entry by p, since if the numbers are large, division by p is moderately complicated. Of course, computers perform division quite rapidly, but the sieving process requires approximately $L(N)$ divisions, so anything that decreases this time will have an immediate effect.

The basic idea is to replace the list of values $F(a), F(a+1), F(a+2), \dots$ by a list of integer values that are approximately equal to

$$\log\bigl(F(a)\bigr),\ \log\bigl(F(a)+1\bigr),\ \log\bigl(F(a)+2\bigr),\ \log\bigl(F(a)+3\bigr),\dots.$$

Then in order to sieve p from $F(t)$, we subtract an approximate value of $\log(p)$ from $\log\bigl(F(t)\bigr)$. If we had used exact values for the logarithms, then at the end of the sieving process, the entries that are reduced to 0 are the values of $F(t)$ that are B-smooth. However, since the logarithm values are only approximate, we instead look for entries that have been reduced to a small number. Then we use division on only those few entries to find which ones are actually B-smooth.

A second idea to speed the process is to use the polynomial $F(t) = t^2 - N$ only until t reaches a certain size and then to replace it by a new polynomial. For details of these two implementation ideas and many others, see for example [11, §10.4], [13], or [42] and the references that they list.

4.7.3 The number field sieve

4.8 The index calculus method for computing discrete logarithms in $\mathbb{F}_p$

4.9 Quadratic residues and quadratic reciprocity [M]

Let p be a prime number. Here is a simple mathematical question:

> How can Bob tell whether or not a given number a is equal to a square modulo p?

$1^2 \equiv 1$	$2^2 \equiv 4$	$3^2 \equiv 9$	$4^2 \equiv 16$	$5^2 \equiv 25$	$6^2 \equiv 36$	$7^2 \equiv 49$	$8^2 \equiv 64$	$9^2 \equiv 81$
$10^2 \equiv 100$	$11^2 \equiv 121$	$12^2 \equiv 144$	$13^2 \equiv 169$	$14^2 \equiv 196$	$15^2 \equiv 225$	$16^2 \equiv 256$	$17^2 \equiv 289$	$18^2 \equiv 324$
$19^2 \equiv 361$	$20^2 \equiv 400$	$21^2 \equiv 441$	$22^2 \equiv 484$	$23^2 \equiv 529$	$24^2 \equiv 576$	$25^2 \equiv 625$	$26^2 \equiv 676$	$27^2 \equiv 729$
$28^2 \equiv 784$	$29^2 \equiv 841$	$30^2 \equiv 900$	$31^2 \equiv 961$	$32^2 \equiv 1024$	$33^2 \equiv 1089$	$34^2 \equiv 1156$	$35^2 \equiv 2$	$36^2 \equiv 73$
$37^2 \equiv 146$	$38^2 \equiv 221$	$39^2 \equiv 298$	$40^2 \equiv 377$	$41^2 \equiv 458$	$42^2 \equiv 541$	$43^2 \equiv 626$	$44^2 \equiv 713$	$45^2 \equiv 802$
$46^2 \equiv 893$	$47^2 \equiv 986$	$48^2 \equiv 1081$	$49^2 \equiv 1178$	$50^2 \equiv 54$	$51^2 \equiv 155$	$52^2 \equiv 258$	$53^2 \equiv 363$	$54^2 \equiv 470$
$55^2 \equiv 579$	$56^2 \equiv 690$	$57^2 \equiv 803$	$58^2 \equiv 918$	$59^2 \equiv 1035$	$60^2 \equiv 1154$	$61^2 \equiv 52$	$62^2 \equiv 175$	$63^2 \equiv 300$
$64^2 \equiv 427$	$65^2 \equiv 556$	$66^2 \equiv 687$	$67^2 \equiv 820$	$68^2 \equiv 955$	$69^2 \equiv 1092$	$70^2 \equiv 8$	$71^2 \equiv 149$	$72^2 \equiv 292$
$73^2 \equiv 437$	$74^2 \equiv 584$	$75^2 \equiv 733$	$76^2 \equiv 884$	$77^2 \equiv 1037$	$78^2 \equiv 1192$	$79^2 \equiv 126$	$80^2 \equiv 285$	$81^2 \equiv 446$
$82^2 \equiv 609$	$83^2 \equiv 774$	$84^2 \equiv 941$	$85^2 \equiv 1110$	$86^2 \equiv 58$	$87^2 \equiv 231$	$88^2 \equiv 406$	$89^2 \equiv 583$	$90^2 \equiv 762$
$91^2 \equiv 943$	$92^2 \equiv 1126$	$93^2 \equiv 88$	$94^2 \equiv 275$	$95^2 \equiv 464$	$96^2 \equiv 655$	....		

Table 4.8: A table of squares modulo 1223

For example, suppose that Alice asks Bob if 181 is a square modulo 1223? Bob might start making a table of squares modulo 1223 as illustrated in Table 4.8, but he gets tired after computing up to 96^2 mod 1223. Alice continues the computations and eventually finds that $437^2 \equiv 181 \pmod{1223}$. Thus the answer to Alice's question is that 181 is indeed a square modulo 1223. And Alice is also able to deduce that the number 385 is not a square modulo 1223, because she continued the table all the way up to 1222^2 mod 1223 and checked that 385 does not appear in her table.[7]

Definition. Let p be a prime number and let a be a number with $p \nmid a$. We say that a is a *quadratic residue modulo* p if a is a square modulo p, i.e. if there is a number c so that $c^2 \equiv a \pmod{p}$. If a is not a square modulo p, then a is called a *quadratic nonresidue modulo* p.

Example 4.50. The two numbers 968 and 1203 are both quadratic residues modulo 1223, since

$$968 \equiv 453 \pmod{1223} \qquad \text{and} \qquad 1203 \equiv 375 \pmod{1223}.$$

On the other hand, the numbers 209 and 888 are quadratic nonresidues modulo 1223, since the congruences

$$c^2 \equiv 209 \pmod{1223} \qquad \text{and} \qquad c^2 \equiv 888 \pmod{1223}$$

have no solution.

[7] Alice could have saved half her time by only computing up to 611^2 mod 1223, since a^2 and $(p-a)^2$ have the same values modulo p.

It should be clear that if we multiply two quadratic residues together, we get another quadratic residue. But what if we multiply a residue by a nonresidue, or if we multiply two nonresidues.

Proposition 4.51. *Let p be a prime number.*

(a) *The product of two quadratic residues modulo p is again a quadratic residue modulo p. residue modulo p.*

(b) *The product of a quadratic residue and a quadratic nonresidue modulo p is a quadratic nonresidue modulo p.*

(c) *The product of two quadratic nonresidues modulo p is a quadratic*

Proof. It is easy to prove (a) and (b) directly from the definition of quadratic residue, but we use a different approach that gives all three parts simultaneously. Let g be a primitive root modulo p as described in Theorem 1.25, which recall means that the powers $1, g, g^2, \ldots, g^{p-2}$ are all distinct modulo p.

Which powers of g are quadratic residues modulo p? Certainly if m is even, then g^m is a square, since $g^m = (g^{m/2})^2$. On the other hand, let m be odd, say $m = 2k+1$, and suppose that g^m is a quadratic residue, say $g^m \equiv c^2 \pmod{p}$. We compute

$$1 \equiv c^{p-1} \equiv (c^2)^{\frac{p-1}{2}} \equiv (g^m)^{\frac{p-1}{2}} \equiv (g^{2k+1})^{\frac{p-1}{2}} \equiv g^{k(p-1)} \cdot g^{\frac{p-1}{2}} \equiv g^{\frac{p-1}{2}} \pmod{p}.$$

This contradicts the fact that g is a primitive root, so we conclude that odd powers of g are nonresidues.

To summarize, we have proven that

$$g^m \text{ is a } \begin{cases} \text{Quadratic Residue} & \text{if } m \text{ is even,} \\ \text{Nonresidue} & \text{if } m \text{ is odd.} \end{cases}$$

It is now a simple matter to prove the theorem. We write a and b as powers of g, multiply a and b by adding the exponents, and read off the result. For example, if a and b are nonresidues, then $a = g^{2i+1}$ and $b = g^{2j+1}$, so $ab = g^{2i+2j+2}$, so ab is a quadratic residue. This proves (c). The proofs of (a) and (b) are similar. □

If we write QR to denote a quadratic residue and NR to denote a nonresidue, then Proposition 4.51 may be succintly summarized by the three equations

$$\mathsf{QR} \cdot \mathsf{QR} = \mathsf{QR}, \qquad \mathsf{QR} \cdot \mathsf{NR} = \mathsf{NR}, \qquad \mathsf{NR} \cdot \mathsf{NR} = \mathsf{QR}.$$

Do these equations look familiar? They resemble the rules for multiplying 1 and -1. This leads to the following definition.

Definition. Let p be a prime. The *Legendre symbol* of a is the quantity $\left(\frac{a}{p}\right)$ defined by the rules

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{if } a \text{ is a quadratic residue modulo } p, \\ -1 & \text{if } a \text{ is a quadratic nonresidue modulo } p, \\ 0 & \text{if } p|a. \end{cases}$$

With this definition, Proposition 4.51 becomes the simple multiplication rule[8]

$$\left(\frac{a}{p}\right)\left(\frac{b}{p}\right) = \left(\frac{ab}{p}\right). \tag{4.21}$$

We also make the obvious, but useful, observation that

$$\text{If} \qquad a \equiv b \pmod{p} \qquad \text{then} \qquad \left(\frac{a}{p}\right) = \left(\frac{b}{p}\right). \tag{4.22}$$

Thus in computing $\left(\frac{a}{p}\right)$, we may reduce a modulo p into the interval from 0 to $p-1$.

Returning to our original question, the following beautiful and powerful theorem provides a method for checking if a given number a is a quadratic residue modulo p.

Theorem 4.52 (Quadratic Reciprocity)**.** *Let p and q be odd primes.*

(a) $$\left(\frac{-1}{p}\right) = \begin{cases} 1 & \textit{if } p \equiv 1 \ (\textit{mod } 4), \\ -1 & \textit{if } p \equiv 3 \ (\textit{mod } 4). \end{cases}$$

(b) $$\left(\frac{2}{p}\right) = \begin{cases} 1 & \textit{if } p \equiv 1 \textit{ or } 7 \ (\textit{mod } 8), \\ -1 & \textit{if } p \equiv 3 \textit{ or } 5 \ (\textit{mod } 8). \end{cases}$$

(c) $$\left(\frac{p}{q}\right) = \begin{cases} \left(\frac{q}{p}\right) & \textit{if } p \equiv 1 \ (\textit{mod } 4) \textit{ or } q \equiv 1 \ (\textit{mod } 4), \\ -\left(\frac{q}{p}\right) & \textit{if } p \equiv 3 \ (\textit{mod } 4) \textit{ and } q \equiv 3 \ (\textit{mod } 4). \end{cases}$$

[8] Proposition 4.51 only deals with the case that $p \nmid a$ and $p \nmid b$. But if p divides a or b, then p also divides ab, so both sides of (4.21) are zero.

Proof. We do not give a proof of Quadratic Reciprocity, but you may find a proof in any introductory number theory textbook, such as [14, 21, 22, 39, 44]. □

The name "Quadratic Reciprocity" comes from property (c), which tells us how $\left(\frac{p}{q}\right)$ is related to its "reciprocal" $\left(\frac{q}{p}\right)$. It is worthwhile spending some time contemplating this result, since despite the simplicity of its statement, it is saying something quite unexpected and profound. The value of $\left(\frac{p}{q}\right)$ tells us whether or not p is a square modulo q. Similarly, $\left(\frac{q}{p}\right)$ tells us whether or not q is a square modulo p. There is no *a priori* reason to suspect that these questions should have anything to do with one another. Quadratic reciprocity tells us that they are intimately related, and indeed, related by a very simple rule.

We indicated earlier that Quadratic Reciprocity can be used to determine if a is a square modulo p. The way to apply Quadratic Reciprocity is to use (c) to repeatedly flip the Legendre symbol, where each time that we flip, we're allowed to reduce the top number modulo the bottom number. This leads to a rapid reduction in the size of the numbers, as illustrated by the following example.

Example 4.53. We determine if -15750 is a quadratic residue modulo 37907 by using Quadratic Reciprocity to compute the Legendre symbol $\left(\frac{-15750}{37907}\right)$.

$$\begin{aligned}
\left(\frac{-15750}{37907}\right) &= \left(\frac{-1}{37907}\right)\left(\frac{15750}{37907}\right) && \text{Multiplication rule (4.21)}\\
&= -\left(\frac{15750}{37907}\right) && \text{Quadratic Reciprocity 4.52(a)}\\
&= -\left(\frac{2\cdot 3^2\cdot 5^3\cdot 7}{37907}\right) && \text{Factor 15750}\\
&= -\left(\frac{2}{37907}\right)\left(\frac{3}{37907}\right)^2\left(\frac{5}{37907}\right)^3\left(\frac{7}{37907}\right)\\
& && \text{Multiplication rule (4.21)}\\
&= -\left(\frac{2}{37907}\right)\left(\frac{5}{37907}\right)\left(\frac{7}{37907}\right) && \text{since } (-1)^2 = 1\\
&= \left(\frac{5}{37907}\right)\left(\frac{7}{37907}\right) && \text{Quadratic Reciprocity 4.52(b)}
\end{aligned}$$

$$\begin{aligned}
&= \left(\frac{37907}{5}\right) \times -\left(\frac{37907}{7}\right) && \text{Quadratic Reciprocity 4.52(c)}\\
&= -\left(\frac{2}{5}\right)\left(\frac{2}{7}\right) && \text{since } 37907 \equiv 2 \pmod 5\\
& && \quad \text{and } 37907 \equiv 2 \pmod 7\\
&= -(-1) \times 1 && \text{Quadratic Reciprocity 4.52(b)}\\
&= 1.
\end{aligned}$$

Thus $\left(\frac{-15750}{37907}\right) = 1$, so we conclude that -15750 is a square modulo 37907, although our computation using Legendre symbols does not tell us how to solve $c^2 \equiv -15750 \pmod{37907}$. It turns out that $c = 10982$ is a solution.

Example 4.53 shows how Quadratic Reciprocity can be used to evaluate the Legendre symol. However, you may have noticed that in the middle of our calculation, we needed to factor the number 15750. We were lucky that 15750 is easy to factor, but suppose that we were faced with a more difficult factorization problem. For example, suppose we want to determine if $p = 228530738017$ is a square modulo $q = 9365449244297$. It turns out that both p and q are prime.[9] Hence we can use Quadratic Reciprocity to compute

$$\begin{aligned}
\left(\frac{228530738017}{9365449244297}\right) &= \left(\frac{9365449244297}{228530738017}\right) && \text{since } 228530738017 \equiv 1 \pmod 4,\\
&= \left(\frac{224219723617}{228530738017}\right) && \text{reducing } 9365449244297\\
& && \text{modulo } 228530738017.
\end{aligned}$$

Unfortunately, the number 224219723617 is not prime, so we cannot apply Quadratic Reciprocity directly, and even more unfortunately, it is not an easy number to factor. So it appears that we are stuck and that Quadratic Reciprocity is only useful for numbers that we can factor.

Luckly, there is a fancier version of Quadratic Reciprocity that allows us to avoid this difficulty. We need one definition.

Definition. Let a and b be integers with b odd and positive, and suppose that the factorization of b into primes is

$$b = p_1^{e_1} p_2^{e_2} p_3^{e_3} \cdots p_t^{e_t}.$$

[9] If you don't believe p and q are prime, use the the Miller-Rabin test (Table 4.2) to check.

The *Jacobi symbol* $\left(\frac{a}{b}\right)$ is defined by the formula

$$\left(\frac{a}{b}\right) = \left(\frac{a}{p_1}\right)^{e_1}\left(\frac{a}{p_2}\right)^{e_2}\left(\frac{a}{p_3}\right)^{e_3}\cdots\left(\frac{a}{p_t}\right)^{e_t}.$$

Notice that if b is itself prime, then $\left(\frac{a}{b}\right)$ is the usual Legendre symbol, so the Jacobi symbol is a generalization of the Legendre symbol.

Example 4.54. Here is a simple example of a Jacobi symbol, computed directly from the definition.

$$\left(\frac{123}{323}\right) = \left(\frac{123}{17\cdot 19}\right) = \left(\frac{123}{17}\right)\left(\frac{123}{19}\right) = \left(\frac{4}{17}\right)\left(\frac{9}{19}\right) = 1.$$

Here is a more complicated example.

$$\begin{aligned}\left(\frac{171337608}{536134436237}\right) &= \left(\frac{171337608}{29^3\cdot 59\cdot 67^2\cdot 83}\right)\\ &= \left(\frac{171337608}{29}\right)^3\left(\frac{171337608}{59}\right)\left(\frac{171337608}{67}\right)^2\left(\frac{171337608}{83}\right)\\ &= \left(\frac{171337608}{29}\right)\left(\frac{171337608}{59}\right)\left(\frac{171337608}{83}\right)\\ &= \left(\frac{11}{29}\right)\left(\frac{15}{59}\right)\left(\frac{44}{83}\right) = (-1)\cdot 1\cdot 1 = -1\end{aligned}$$

From the definition, it appears that in order to compute the Jacobi symbol $\left(\frac{a}{b}\right)$, we need to know how to factor b. However, it turns out that the Jacobi symbol has most of the same properties as does the Legendre symbol, and this allows us to compute $\left(\frac{a}{b}\right)$ extremely rapidly without doing any factorization at all. We start with the basic multiplication and reduction properties.

Proposition 4.55. *Let a, a_1, a_2, b, b_1, b_2 be integers with b, b_1 and b_2 positive and odd.*

(a)

$$\left(\frac{a_1a_2}{b}\right) = \left(\frac{a_1}{b}\right)\left(\frac{a_2}{b}\right) \qquad \textit{and} \qquad \left(\frac{a}{b_1b_2}\right) = \left(\frac{a}{b_1}\right)\left(\frac{a}{b_2}\right).$$

(b)

$$\textit{If} \qquad a_1 \equiv a_2 \ (\textit{mod } b) \qquad \textit{then} \qquad \left(\frac{a_1}{b}\right) = \left(\frac{a_2}{b}\right).$$

Proof. Both parts of Proposition 4.55 follows fairly directly from the corresponding properties of the Legendre symbol and the definition of the Jacobi symbol. We leave the details as an exercise. □

Now we come to the amazing fact that the Jacobi symbol satisfies exactly the same reciprocity law as the Legendre symbol.

Theorem 4.56 (Quadratic Reciprocity—Version II)**.** *Let a and b be integers that are* odd *and* positive*.*

(a) $$\left(\frac{-1}{b}\right) = \begin{cases} 1 & \textit{if } b \equiv 1 \ (\textit{mod } 4), \\ -1 & \textit{if } b \equiv 3 \ (\textit{mod } 4). \end{cases}$$

(b) $$\left(\frac{2}{b}\right) = \begin{cases} 1 & \textit{if } b \equiv 1 \textit{ or } 7 \ (\textit{mod } 8), \\ -1 & \textit{if } b \equiv 3 \textit{ or } 5 \ (\textit{mod } 8). \end{cases}$$

(c) $$\left(\frac{a}{b}\right) = \begin{cases} \left(\dfrac{b}{a}\right) & \textit{if } a \equiv 1 \ (\textit{mod } 4) \textit{ or } b \equiv 1 \ (\textit{mod } 4), \\ -\left(\dfrac{b}{a}\right) & \textit{if } a \equiv 3 \ (\textit{mod } 4) \textit{ and } b \equiv 3 \ (\textit{mod } 4). \end{cases}$$

Proof. It is not hard to use the orginal result (Theorem 4.52) to prove this general version of Quadratic Reciprocity, see for example [22, Proposition 5.2.2]. □

Example 4.57. When we tried to use the original version of Quadratic Reciprocity (Theorem 4.52) to compute $\left(\frac{228530738017}{9365449244297}\right)$, we ran into the problem of needing to factor the number 224219723617. Using the new and improved version of Quadratic Reciprocity (Theorem 4.56), we can perform the computation without doing any factoring.

$$\begin{aligned} \left(\frac{228530738017}{9365449244297}\right) &= \left(\frac{9365449244297}{228530738017}\right) = \left(\frac{224219723617}{228530738017}\right) = \left(\frac{228530738017}{224219723617}\right) \\ &= \left(\frac{4311014400}{224219723617}\right) = \left(\frac{2^{10}\cdot 4209975}{224219723617}\right) = \left(\frac{224219723617}{4209975}\right) = \left(\frac{665092}{4209975}\right) \\ &= \left(\frac{2^{2}\cdot 166273}{4209975}\right) = \left(\frac{4209975}{166273}\right) = \left(\frac{53150}{166273}\right) = \left(\frac{2\cdot 26575}{166273}\right) = \left(\frac{26575}{166273}\right) \\ &= \left(\frac{166273}{26575}\right) = \left(\frac{6823}{26575}\right) = -\left(\frac{26575}{6823}\right) = -\left(\frac{6106}{6823}\right) = -\left(\frac{2\cdot 3053}{6823}\right) \end{aligned}$$

$$
\begin{aligned}
&= -\left(\frac{3053}{6823}\right) = -\left(\frac{6823}{3053}\right) = -\left(\frac{717}{3053}\right) = -\left(\frac{3053}{717}\right) = -\left(\frac{185}{717}\right) = -\left(\frac{717}{185}\right)\\
&= -\left(\frac{162}{185}\right) = -\left(\frac{2\cdot 81}{185}\right) = -\left(\frac{81}{185}\right) = -\left(\frac{185}{81}\right) = -\left(\frac{23}{81}\right) = -\left(\frac{81}{23}\right)\\
&= -\left(\frac{12}{23}\right) = -\left(\frac{2^2\cdot 3}{23}\right) = \left(\frac{23}{3}\right) = \left(\frac{2}{3}\right) = -1
\end{aligned}
$$

Remark 4.58. Suppose that $\left(\frac{a}{b}\right) = 1$. Does that tell us that a is a square modulo b? It does if b is prime, since that's how we defined the Legendre symbol, but what if b is composite. For example, suppose that $b = pq$ is a product of two primes. Then by definition,

$$\left(\frac{a}{b}\right) = \left(\frac{a}{pq}\right) = \left(\frac{a}{p}\right)\left(\frac{a}{q}\right).$$

If $\left(\frac{a}{b}\right) = 1$, then there are two possibilities. Either $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = 1$, in which case a is a square modulo p and also a square modulo q, so it is a square modulo pq.[10] Or else $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = -1$, in which case a is certainly not a square modulo pq. Thus although it is easy to compute the value of $\left(\frac{a}{pq}\right)$, that value does not tell you whether or not a is a square modulo pq. This dichotomy can be exploited for cryptographic purposes, as we explain in the next section.

4.10 Probabilistic encryption and the Goldwasser-Micali public key cryptosystem

Suppose that Bob wants a public key cryptosystem in which Alice can encrypt and send him one bit at a time. How could this possibly be secure? All that Eve has to do is to encrypt the two possible plaintexts $m = 0$ and $m = 1$, and then she compares the encryptions with Alice's ciphertext. More generally, if the set of possible plaintexts is small, Eve can encrypt each plaintext using Bob's public key until she finds the one that is Alice's.

Probabilistic encyption was invented by Goldwasser and Micali as a way around this problem. The idea is that Alice chooses both a plaintext m and

[10] Suppose that $c_1^2 \equiv a \pmod{p}$ and $c_2^2 \equiv a \pmod{q}$. Then the Chinese Remainder Theorem 2.13 says that we can find an integer c satisfying $c \equiv c_1 \pmod{p}$ and $c \equiv c_2 \pmod{q}$, and then $c^2 \equiv a \pmod{pq}$.

a random string of data r, and then she uses Bob's public key to encrypt the pair (m, r). Ideally, as r varies over all of its possible values, the ciphertext for (m, r) will vary "randomly" over the possible ciphertexts. More precisely, for any fixed m and varying r, the distribution of values of the quantity

$$e(m, r) = \text{the ciphertext for plaintext } m \text{ and random string } r$$

should be essentially indistinguishable. Note that it is not necessary that Bob be able to recover the full pair (m, r) when he performs the decryption. He only needs to recover the plaintext m.

The abstract idea of probabilistic encryption is clear, but how might one create a probabilistic encryption scheme in practice? Goldwasser and Micali describe one such scheme which, although impractical since it only encrypts one bit at a time, has the advantage of being quite simple to describe and analyze. The idea is based on the difficulty of the following problem.

> Let p and q be (secret) prime numbers and let $N = pq$ be given. For a given integer a, determine whether or not a is a square modulo N, i.e. determine whether there exists an integer u satisfying $u^2 \equiv a \pmod{N}$.

Note that Bob, who knows how to factor $N = pq$, is able to solve this problem very easily, since

$$a \text{ is a square modulo } pq \qquad \text{if and only if} \qquad \left(\frac{a}{p}\right) = 1 \quad \text{and} \quad \left(\frac{a}{q}\right) = 1.$$

Eve, on the other hand, has a harder time, since she only knows the value of N. Eve can compute $\left(\frac{a}{N}\right)$, but as we noted earlier (Remark 4.58), this does not tell her if a is a square modulo N. Goldwasser and Micali exploit this fact[11] to create the probabilistic public key cryptosystem described in Table 4.9.

It is easy to check that the Goldwasser-Micali cryptosystem works as

[11] Goldwasser and Micali were not the first to use the problem of squares modulo pq for cryptography. Indeed, an early public key cryptosystem due to Rabin that is provably secure against chosen plaintext attacks (assuming the hardness of factorization) relies on this problem.

Bob	**Alice**
Key Creation	
Choose secret primes p and q. Choose a with $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = -1$. Publish $N = pq$ and a.	
Encryption	
	Choose plaintext $m \in \{0, 1\}$. Choose random r with $1 < r < N$. Use Bob's public key (N, a) to compute $c = \begin{cases} r^2 \bmod N & \text{if } m = 0, \\ ar^2 \bmod N & \text{if } m = 1. \end{cases}$ Send ciphtertext c to Bob.
Decryption	
Compute $\left(\frac{c}{p}\right)$. Decrypt to $m = \begin{cases} 0 & \text{if } \left(\frac{c}{p}\right) = 1, \\ 1 & \text{if } \left(\frac{c}{p}\right) = -1. \end{cases}$	

Table 4.9: Goldwasser-Micali probabilistic public key cryptosystem

advertised, since

$$\left(\frac{c}{p}\right) = \begin{cases} \left(\frac{r^2}{p}\right) = \left(\frac{r}{p}\right)^2 = 1 & \text{if } m = 0, \\ \left(\frac{ar^2}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{r}{p}\right)^2 = \left(\frac{a}{p}\right) = -1 & \text{if } m = 1. \end{cases}$$

Further, since Alice is choosing r randomly, the set of values that Eve sees when Alice encrypts $m = 0$ consists of all possible squares modulo N, and the set of values that Eve sees when Alice encrypts $m = 1$ consists of all possible numbers c satisfying $\left(\frac{c}{N}\right) = 1$ that are not squares modulo N.

What information does Eve obtain if she computes the Jacobi symbol $\left(\frac{c}{N}\right)$, which she can do since N is a public quantity? If $m = 0$, then $c \equiv r^2 \pmod{N}$, so clearly $\left(\frac{c}{N}\right) = \left(\frac{r^2}{N}\right) = \left(\frac{r}{N}\right)^2 = 1$. On the other hand, if $m = 1$, then $c \equiv ar^2 \pmod{N}$, so

$$\left(\frac{c}{N}\right) = \left(\frac{ar^2}{N}\right) = \left(\frac{a}{N}\right) = \left(\frac{a}{pq}\right) = \left(\frac{a}{p}\right)\left(\frac{a}{q}\right) = (-1)\cdot(-1) = 1,$$

since Bob chose a to satisfy $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = -1$. Thus $\left(\frac{c}{n}\right)$ is equal to 1, regardless of the value of N, so Eve gains no useful information.

Example 4.59. Bob creates a Goldwasser-Micali public key by choosing

$$p = 2309, \qquad q = 5651, \qquad N = pq = 13048159, \qquad a = 6283665.$$

Note that a has the property that $\left(\frac{a}{p}\right) = \left(\frac{a}{q}\right) = -1$. He publishes the pair (N, a) and keeps the values of the primes p and q secret.

Alice begins by sending Bob the plaintext bit $m = 0$. To do this, she chooses $r = 1642087$ randomly between 1 and 13048158, computes

$$c \equiv r^2 \equiv 1642087^2 \equiv 8513742 \pmod{13048159},$$

and sends the ciphertext $c = 8513742$ to Bob. Bob decrypts the ciphertext $c = 8513742$ by computing $\left(\frac{8513742}{2309}\right) = 1$, which gives the plaintext bit $m = 0$.

Next Alice decides to send Bob the plaintext bit $m = 1$. She chooses a random value $r = 11200984$ and computes

$$c \equiv ar^2 \equiv 6283665 \cdot 11200984^2 \equiv 2401627 \pmod{13048159}.$$

Bob decrypts $c = 2401627$ by computing $\left(\frac{2401627}{2309}\right) = -1$, which tells him that the plaintext bit $m = 1$.

Finally, Alice wants to send Bob another plaintext bit $m = 1$. She chooses the random value $r = 11442423$ and computes

$$c \equiv ar^2 \equiv 6283665 \cdot 11442423^2 \equiv 4099266 \pmod{13048159}.$$

Notice that the ciphertext for this encryption of $m = 1$ is completely unrelated to the previous encryption of $m = 1$. Bob decrypts $c = 4099266$ by computing $\left(\frac{4099266}{2309}\right) = -1$ to conclude that the plaintext bit is $m = 1$.

Remark 4.60. The Goldwasser-Micali public key cryptosystem is not practical, because each bit of plaintext is encrypted with a number modulo N. To be secure, it is necessary that Eve be unable to factor the number $N = pq$, so in practice N will be (at least) a 1024 bit number. Thus if Alice wants to send k bits of plaintext to Bob, her ciphertext will be $1024k$ bits long, in which case we say that the Goldwasswer-Micali public key cryptosystem has *message expansion* ratio of 1024. In general, the Goldwasswer-Micali public key cryptosystem has message expansion by a factor $\log_2(N)$.

There are other probabilistic public key cryptosystems whose message expansion is much smaller. Indeed, we have already seen one, since the ephemeral key k used by the ElGamal public key cryptosystem (Section 2.4) makes it probabilistic. ElGamal has a message expansion ratio of 2, as explained in Remark 2.10. Other natural probabilistic cryptosystems include the Rabin cryptosystem and the NTRU cryptosystem (Section 7.6). It is also possible to take a deterministic cryptosystem and turn it into a probabilistic system, at the cost of increasing its message expansion ratio, see Exercise 4.35.

Exercises

Section 4.1. Euler's theorem and roots modulo pq

4.1. Solve the following congruences.

(a) $x^{19} \equiv 36 \pmod{97}$.
(b) $x^{137} \equiv 428 \pmod{541}$.
(c) $x^{73} \equiv 614 \pmod{1159}$.
(d) $x^{751} \equiv 677 \pmod{8023}$.
(e) $x^{38993} \equiv 328047 \pmod{401227}$. (*Hint.* $401227 = 607 \cdot 661$.)

4.2. Let p and q be distinct primes and let e and d be integers satisfying

$$de \equiv 1 \pmod{(p-1)(q-1)}.$$

Suppose further that c is an integer with $\gcd(c, pq) > 1$. Prove that

$$x \equiv c^d \pmod{pq} \quad \text{is a solution to the congruence} \quad x^e \equiv c \pmod{pq},$$

thereby completing the proof of Proposition 4.4.

4.3. Recall that *Euler's phi function* $\phi(m)$ is the function defined by

$$\phi(N) = \#\{0 \le k < N : \gcd(k, N) = 1\}.$$

In other words, $\phi(N)$ is the number of integers between 0 and $N-1$ that are relatively prime to N, or equivalently, the number of elements in $\mathbb{Z}/N\mathbb{Z}$ that have inverses modulo N (see Section 1.3).

(a) Compute the values of $\phi(6)$, $\phi(9)$, $\phi(15)$, and $\phi(17)$.

(b) If p is prime, what is the value of $\phi(p)$?

(c) Prove *Euler's formula*

$$a^{\phi(N)} \equiv 1 \pmod{N} \quad \text{for all integers } a \text{ satisfying } \gcd(a, N) = 1.$$

(*Hint.* Mimic the proof of Fermat's Little Theorem 1.22, but instead of looking at all of the multiples of a as was done in (**??**), just take the multiples ka of a for values of k satisfying $\gcd(k, N) = 1$.)

4.4. Euler's phi function has many beautiful properties.

(a) If p and q are distinct prime, how is $\phi(pq)$ related to $\phi(p)$ and $\phi(q)$?

(b) If p is prime, what is the value of $\phi(p^2)$? How about $\phi(p^j)$? Prove that your formula for $\phi(p^j)$ is correct. (*Hint.* Among the numbers between 0 and $p^j - 1$, remove the ones that have a factor of p? The ones that are left are relatively prime to p.)

(c) Let M and N be integers satisfying $\gcd(M, N) = 1$. Prove the multiplication formula

$$\phi(MN) = \phi(M)\phi(N).$$

(d) Use your results from (b) and (c) to prove the following formula. Let $p_1, p_2, \ldots, p_r$ be the distinct primes that divide N. Then

$$\phi(N) = N \prod_{i=1}^{r} \left(1 - \frac{1}{p_i}\right).$$

(e) Use the formula in (d) to compute the following values of $\phi(N)$.

(i) $\phi(1728)$. (ii) $\phi(1575)$. (iii) $\phi(889056)$ (*Hint.* $889056 = 2^5 \cdot 3^4 \cdot 7^3$).

4.5. Let N, c, and e be positive integers satisfying the conditions $\gcd(N, c) = 1$ and $\gcd\bigl(e, \phi(N)\bigr) = 1$.

(a) Explain how to solve the congruence

$$x^e \equiv c \pmod{N},$$

assuming that you know the value of $\phi(N)$. (*Hint.* Use the formula in Exercise 4.3(c).)

(b) Solve the following congruences. (The formula in Exercise 4.4(d) may be helpful for computing the value of $\phi(N)$.)

(i) $x^{577} \equiv 60 \pmod{1463}$.
(ii) $x^{959} \equiv 1583 \pmod{1625}$.
(iii) $x^{133957} \equiv 224689 \pmod{2134440}$.

Section 4.2. The RSA public key cryptosystem

4.6. Alice's publishes her RSA public key: modulus $N = 2038667$ and exponent $e = 103$.

(a) Bob wants to send Alice the message $m = 892383$. What ciphertext does Bob send to Alice?

(b) Alice knows that her modulus factors into a product of two primes, one of which is $p = 1301$. Find a decryption exponent d for Alice.

(c) Alice receives the ciphertext $c = 317730$ from Bob. Decrypt the message.

4.7. Bob's RSA public key has modulus $N = 12191$ and exponent $e = 37$. Alice sends to Bob the ciphertext $c = 587$. Unfortunately, Bob has chosen too small a modulus. Help Eve by factoring N and decrypting Alice's message. (*Hint.* N has a factor smaller than 100.)

4.8. For each of the given values of $N = pq$ and $(p-1)(q-1)$, use the method described in Remark 4.9 to determine p and q.

(a) $N = pq = 352717$ and $(p-1)(q-1) = 351520$.
(b) $N = pq = 77083921$ and $(p-1)(q-1) = 77066212$.
(c) $N = pq = 109404161$ and $(p-1)(q-1) = 109380612$.
(d) $N = pq = 172205490419$ and $(p-1)(q-1) = 172204660344$.

4.9. A *decryption exponent* for an RSA public key (N, e) is an integer d with the property that $a^{de} \equiv a \pmod{N}$ for all integers a that are relatively prime to N.

(a) Suppose that Eve has a magical box that creates decryption exponents for (N, e) for a fixed modulus N and for a lot of different encryption exponents e. Explain how Eve can use her magical box to try to factor N.

(b) Let $N = 38749709$. Eve's magic box tells her that the encryption exponent $e = 10988423$ has decryption exponent $d = 16784693$ and that the encryption exponent $e = 25910155$ has decryption exponent $d = 11514115$. Use this information to factor N.

(c) Let $N = 225022969$. Eve's magic box tells her that the encryption exponent $e = 70583995$ has decryption exponent $d = 4911157$, that the encryption exponent $e = 173111957$ has decryption exponent $d = 7346999$, and that the encryption exponent $e = 180311381$ has decryption exponent $d = 29597249$. Use this information to factor N.

(d) Let $N = 1291233941$. Eve's magic box tells her the following three encryption/decryption pairs for N:

$$(1103927639, 76923209), (1022313977, 106791263), (387632407, 7764043).$$

Use this information to factor N.

Section 4.3. Implementation and security issues

4.10. Formulate a Man-in-the-Middle Attack, similar to the attack described in Example 4.12 on page 189, for the following public key cryptosystems.

(a) The ElGamal public key cryptosystem (Table 2.3 on page 69).

(b) The RSA public key cryptosystem (Table 4.1 on page 186).

Section 4.4. Primality testing

4.11. We stated that the number 561 is a Carmichael number, but we never checked that $a^{561} \equiv a \pmod{561}$ for every value of a.

(a) The number 561 factors as $3 \cdot 11 \cdot 17$. First use Fermat's Little Theorem to prove that

$$a^{561} \equiv a \pmod{3}, \quad a^{561} \equiv a \pmod{11}, \quad \text{and} \quad a^{561} \equiv a \pmod{17}$$

for every value of a. Then explain why these three congruences imply that $a^{561} \equiv a \pmod{561}$ for every value of a.

(b) Mimic the idea used in (a) to prove that each of the following numbers is a Carmichael number. (To assist you, we have factored each number into primes.)

(i) $1729 = 7 \cdot 13 \cdot 19$

(ii) $10585 = 5 \cdot 29 \cdot 73$

(iii) $75361 = 11 \cdot 13 \cdot 17 \cdot 31$
(iv) $1024651 = 19 \cdot 199 \cdot 271$

(c) Prove that a Carmichael number must be odd.
(d) Prove that a Carmichael number must be a product of *distinct* primes.
(e) Look up Korselt's criterion in a book or online, write a brief description of how it works, and use it to show that $29341 = 13 \cdot 37 \cdot 61$ and $172947529 = 307 \cdot 613 \cdot 919$ are Carmichael numbers.

4.12. Use the Miller-Rabin test on each of the following numbers. In each case, either provide a Miller-Rabin witness for the compositeness of n, or conclude that n is probably prime by providing 10 numbers that are not Miller-Rabin witnesses for n.
(a) $n = 1105$. (Yes, 5 divides n, but this is just a warm-up exercise!)
(b) $n = 294409$.
(c) $n = 294439$.
(d) $n = 118901509$.
(e) $n = 118901521$.
(f) $n = 118901527$.
(g) $n = 118915387$.

4.13. The function $\pi(X)$ gives the number of primes between 2 and X.
(a) Compute the following values of $\pi(X)$:

(i) $\pi(20)$. (ii) $\pi(30)$. (iii) $\pi(100)$.

(b) Write a program to compute $\pi(X)$ and use it to compute $\pi(X)$ and the ratio $\pi(X)/(X/\ln(X))$ for $X = 100$, $X = 1000$, $X = 10000$, and $X = 100000$. Does your data make the Prime Number Theorem plausible?

4.14. Let

$$\pi_1(X) = (\#\text{ of primes } p \text{ between 2 and } X \text{ satisfying } p \equiv 1 \pmod{4}), \quad \text{and}$$
$$\pi_3(X) = (\#\text{ of primes } p \text{ between 2 and } X \text{ satisfying } p \equiv 3 \pmod{4}).$$

Thus every prime except for 2 gets counted by either $\pi_1(X)$ or by $\pi_3(X)$.
(a) Compute the values of $\pi_1(X)$ and $\pi_3(X)$ for each of the following values of X. (i) $X = 10$. (ii) $X = 25$. (iii) $X = 100$.
(b) Write a program to compute $\pi_1(X)$ and $\pi_3(X)$ and use it to compute their values and the ratio $\pi_3(X)/\pi_1(X)$ for $X = 100$, $X = 1000$, $X = 10000$, and $X = 100000$.

(c) Based on your data from (b), make a conjecture about the relative sizes of $\pi_1(X)$ and $\pi_3(X)$ (which one do you think is larger) and a conjecture about the limit of the ratio $\pi_3(X)/\pi_1(X)$ as $X \to \infty$.

4.15. We noted in the text that the statement (4.8) in Section 4.4 does not make make sense, since any particular number that you choose either will be prime or will not be prime. There are no numbers that are 35% prime and 65% composite! In this exercise you will prove a result that makes sense of (4.8). You may use the Prime Number Theorem 4.20 for this problem.

(a) For each (large) number N, suppose that Bob chooses a random number n in the interval $\frac{1}{2}N \le n \le \frac{3}{2}N$. If he repeats this process many times, prove that approximately $1/\ln(N)$ of his numbers will be prime? In mathematical terms, let

$$P(N) = \left(\begin{array}{l}\text{Probability that an integer } n \text{ in the inter-}\\ \text{val } \frac{1}{2}N \le n \le \frac{3}{2}N \text{ is a prime number}\end{array}\right)$$

and prove that

$$\lim_{N\to\infty} \frac{P(N)}{1/\ln(N)} = 1.$$

(b) More generally, fix two numbers c_1 and c_2 satisfying $c_1 < c_2$. For each N, Bob chooses random numbers n in the interval $c_1N \le n \le c_2N$. Keeping c_1 and c_2 fixed, let

$$P(c_1, c_2; N) = \left(\begin{array}{l}\text{Probability that an integer } n \text{ in the inter-}\\ \text{val } c_1N \le n \le c_2N \text{ is a prime number}\end{array}\right)$$

In the following formula, fill in the box with a simple function of N so that the statement is true:

$$\lim_{N\to\infty} \frac{P(c_1, c_2; N)}{\boxed{}} = 1.$$

4.16. Continuing with the previous exercise, explain how to make mathematical sense of the following statements.

(a) A randomly chosen *odd* number N has probability $2/\ln(N)$ of being prime. (What is the probability that a randomly chosen even number is prime?)

(b) A randomly chosen number N satisfying $N \equiv 1 \pmod 3$ has probability $3/(2\ln(N))$ of being prime.

(c) A randomly chosen number N satisfying $N \equiv 1 \pmod 6$ has probability $3/\ln(N)$ of being prime.

(d) Let $m = p_1 p_2 \cdots p_r$ be a product of distinct primes and let k be a number satisfying $\gcd(k, m) = 1$. What number should go into the box to make the following statement correct? Why?

> A randomly chosen number N satisfying $N \equiv k \pmod{m}$ has probability $\boxed{}\,/\ln(N)$ of being prime. (4.23)

(e) Same question, but for arbitrary m, not just for m that are products of distinct primes.

4.17. The *logarithmic integral function* $\mathrm{Li}(X)$ is defined to be

$$\mathrm{Li}(X) = \int_2^X \frac{dt}{\ln t}.$$

(a) Prove that

$$\mathrm{Li}(X) = \frac{X}{\ln X} + \int_2^X \frac{dt}{(\ln t)^2} + O(1).$$

(*Hint.* Integration by parts.)

(b) Compute the limit

$$\lim_{X\to\infty} \frac{\mathrm{Li}(X)}{X/\ln X}.$$

(*Hint.* Break the integral into two pieces, $2 \le t \le \sqrt{X}$ and $\sqrt{X} \le t \le X$, and estimate each piece separately.)

(c) Use (b) to show that formula (4.10) on page 198 implies the Prime Number Theorem 4.20.

Section 4.5. The $p - 1$ factorization method

4.18. A prime of the form $2^n - 1$ is called a Mersenne prime.

(a) Factor each of the numbers $2^n - 1$ for $n = 2, 3, \ldots, 10$. Which ones are Mersenne primes?

(b) Find the first seven Mersenne primes. (You may need a computer.)

(c) If n is even and $n > 2$, prove that $2^n - 1$ is not prime.

(d) If $3|n$ and $n > 3$, prove that $2^n - 1$ is not prime.

(e) More generally, prove that if n is a composite number, then $2^n - 1$ is not prime. Thus Mersenne primes all have the form $2^p - 1$ with p a prime number.

(f) What is the largest known Mersenne prime? Are there any larger primes known? (You can find out at the "Great Internet Mersenne Prime Search" web site `www.mersenne.org/prime.htm`.)

(g) Write a one page essay on Mersenne primes, starting with the discoveries of Father Mersenne and ending with GIMPS.

Section 4.6. Factorization via difference of squares

4.19. For each of the following numbers N, compute the values of

$$N+1^2, \quad N+2^2, \quad N+3^2, \quad N+4^2, \ldots$$

as we did in Example 4.33 until you find a value $N+b^2$ that is a perfect square a^2. Then use the values of a and b to factor N.

(a) $N = 53357$

(b) $N = 34571$

(c) $N = 25777$

(d) $N = 64213$

4.20. For each of the listed values of N, k, and b_{init}, factor N by making a list of values of $k \cdot N + b^2$, starting at $b = b_{\text{init}}$ and incrementing b until $k \cdot N + b^2$ is a perfect square. Then take greatest common divisors as we did in Example 4.34.

(a)	$N = 143041$	$k = 247$	$b_{\text{init}} = 1$
(b)	$N = 1226987$	$k = 3$	$b_{\text{init}} = 36$
(c)	$N = 2510839$	$k = 21$	$b_{\text{init}} = 90$

4.21. For each part, use the data provided to find values of a and b satisfying $a^2 \equiv b^2 \pmod{N}$ and then compute $\gcd(N, a-b)$ in order to find a nontrivial factor of N, as we did in Examples 4.36 and 4.37.

(a) $N = 61063$

$$\begin{array}{rlcr} 1882^2 \equiv 270 & \pmod{61063} & \text{and} & 270 = 2 \cdot 3^3 \cdot 5 \\ 1898^2 \equiv 60750 & \pmod{61063} & \text{and} & 60750 = 2 \cdot 3^5 \cdot 5^3 \end{array}$$

(b) $N = 52907$

$$\begin{array}{rlcr} 399^2 \equiv 480 & \pmod{52907} & \text{and} & 480 = 2^5 \cdot 3 \cdot 5 \\ 763^2 \equiv 192 & \pmod{52907} & \text{and} & 192 = 2^6 \cdot 3 \\ 773^2 \equiv 15552 & \pmod{52907} & \text{and} & 15552 = 2^6 \cdot 3^5 \\ 976^2 \equiv 250 & \pmod{52907} & \text{and} & 250 = 2 \cdot 5^3 \end{array}$$

(c) $N = 198103$

$$\begin{array}{llll}
1189^2 \equiv 27000 & \pmod{198103} & \text{and} & 27000 = 2^3 \cdot 3^3 \cdot 5^3 \\
1605^2 \equiv 686 & \pmod{198103} & \text{and} & 686 = 2 \cdot 7^3 \\
2378^2 \equiv 108000 & \pmod{198103} & \text{and} & 108000 = 2^5 \cdot 3^3 \cdot 5^3 \\
2815^2 \equiv 105 & \pmod{198103} & \text{and} & 105 = 3 \cdot 5 \cdot 7
\end{array}$$

(d) $N = 2525891$

$$\begin{array}{llll}
1591^2 \equiv 5390 & \pmod{2525891} & \text{and} & 5390 = 2 \cdot 5 \cdot 7^2 \cdot 11 \\
3182^2 \equiv 21560 & \pmod{2525891} & \text{and} & 21560 = 2^3 \cdot 5 \cdot 7^2 \cdot 11 \\
4773^2 \equiv 48510 & \pmod{2525891} & \text{and} & 48510 = 2 \cdot 3^2 \cdot 5 \cdot 7^2 \cdot 11 \\
5275^2 \equiv 40824 & \pmod{2525891} & \text{and} & 40824 = 2^3 \cdot 3^6 \cdot 7 \\
5401^2 \equiv 1386000 & \pmod{2525891} & \text{and} & 1386000 = 2^4 \cdot 3^2 \cdot 5^3 \cdot 7 \cdot 11
\end{array}$$

Section 4.7. Smooth numbers, sieves and building relations for factorization

4.22. Compute the following values of $\psi(X, B)$, the number of B-smooth numbers between 1 and X (see page 215).

(a) $\psi(25, 3)$ (b) $\psi(35, 5)$ (c) $\psi(50, 7)$ (d) $\psi(100, 5)$ (e) $\psi(100, 7)$

4.23. An integer M is called *B-power-smooth* if every prime power p^e dividing M satisfies $p^e \le B$. For example, $180 = 2^2 \cdot 3^2 \cdot 5$ is 10-power-smooth, since the largest prime power dividing 180 is 9, which is smaller than 10.

(a) Suppose that M is B-power-smooth. Prove that M is also B-smooth.

(b) Suppose that M is B-smooth. Is it always true that M is also B-power-smooth? Either prove that it is true or give an example for which it is not true.

(c) The following is a list of 20 randomly chosen numbers between 1 and 1000, sorted from smallest to largest. Which of these numbers are 10-power-smooth? Which of them are 10-smooth?

$$\{84, 141, 171, 208, 224, 318, 325, 366, 378, 390, 420, 440,$$
$$504, 530, 707, 726, 758, 765, 792, 817\}$$

(d) Prove that M is B-power-smooth if and only if M divides $\mathrm{LCM}[1, 2, \ldots, B]$.

4.24. Let $L(N) = e^{\sqrt{(\log N)(\log\log N)}}$. Suppose that a computer does one billion operations per second.

(a) How many seconds does it take to perform $L(2^{100})$ operations?
(b) How many hours does it take to perform $L(2^{250})$ operations?
(c) How many days does it take to perform $L(2^{350})$ operations?
(d) How many years does it take to perform $L(2^{500})$ operations?
(e) How many years does it take to perform $L(2^{750})$ operations?
(f) How many years does it take to perform $L(2^{1000})$ operations?
(g) How many years does it take to perform $L(2^{2000})$ operations?

(For simplicity, assume that there are 365.25 days in a year.)

4.25. Prove that the function $L(N) = e^{\sqrt{(\log N)(\log\log N)}}$ is subexponential. That is, prove the following two statements.

(a) For every positive constant α, no matter how large, $(\log N)^\alpha \ll L(N)$.
(b) For every positive constant β, no matter how small, $L(N) \ll N^\beta$.

4.26. More generally, for any fixed positive constants a and b, define the function

$$F_{a,b}(N) = e^{(\log N)^{1/a}(\log\log N)^{1/b}}.$$

Prove the following properties of $F_{a,b}(N)$.

(a) If $a > 1$, prove that $F_{a,b}(N)$ is subexponential.
(b) If $a = 1$, prove that $F_{a,b}(N)$ is exponential. What happens if $a < 1$?

4.27. This exercise asks you to verify an assertion in the proof of Theorem 4.42. Let $L(X)$ be the usual function $L(X) = e^{\sqrt{(\log X)(\log\log X)}}$.

(a) Prove that there is a value of $\epsilon > 0$ so that

$$(\log X)^\epsilon < \log L(X) < (\log X)^{1-\epsilon} \qquad \text{for all } X > 10.$$

(b) Let $c > 0$, let $Y = L(X)^c$, and let $u = (\log X)/(\log Y)$. Prove that

$$u^{-u} = L(X)^{-\frac{1}{2c}(1+o(1))}.$$

4.28. Proposition 4.44 assumes that we choose random numbers a modulo N, compute $a^2 \pmod{N}$, and check if the result is B-smooth. We can achieve better results if we take a values of the form

$$a = \left\lfloor \sqrt{N} \right\rfloor + k \qquad \text{for } 1 \le k \le K.$$

(For simplicity, you may treat K as a fixed integer, independent of N. More rigorously, it is necessary to take K equal to a power of $L(N)$, which has a small effect on the final answer.)

(a) Prove that $a^2 - N \leq 2K\sqrt{N} + K^2$, so in particular, $a^2 \pmod{N}$ is smaller than a multiple of $\sqrt{N}$.

(b) Prove that $L(\sqrt{N}\,) \approx L(N)^{1/\sqrt{2}}$. More generally, prove that, $L(N^{1/r}) \approx L(N)^{1/\sqrt{r}}$.

(c) Reprove Proposition 4.44 using this better choice of values for a. Set $B = L(N)^c$ and find the optimal value of c. How many relations are needed to factor N?

4.29. Illustrate the quadartic sieve as was done in Figure 4.3 on page 225 by sieving prime powers up to B on the values of $F(T) = T^2 - N$ in the indicated range.

(a) Sieve $N = 493$ using prime powers up to $B = 11$ on values from $F(23)$ to $F(38)$. Use the relation(s) that you find to factor N.

(b) Extend the computations in (a) by using prime powers up to $B = 16$ and sieving values from $F(23)$ to $F(50)$. What additional value(s) are sieved down to 1 and what additional relation(s) do they yield?

Section 4.9. Quadratic residues and quadratic reciprocity

4.30. Let p be an odd prime and let a be an integer with $p \nmid a$.

(a) Prove that $a^{(p-1)/2}$ is congruent to either 1 or -1 modulo p. (*Hint.* Use Fermat's Little Theorem).

(b) Prove that $a^{(p-1)/2}$ is congruent to 1 modulo p if and only if a is a quadratic residue modulo p. (*Hint.* Let g be a primitive root for p and look at the discrete logarithm of a with respect to g.)

(c) Prove that $a^{(p-1)/2} \equiv \left(\frac{a}{p}\right) \pmod{p}$. (This even holds if $p|a$.)

(d) Use (c) to prove Theorem 4.52(a), that is, prove that

$$\left(\frac{-1}{p}\right) = \begin{cases} 1 & \text{if } p \equiv 1 \pmod{4}, \\ -1 & \text{if } p \equiv 3 \pmod{4}. \end{cases}$$

4.31. Let p be a prime satisfying $p \equiv 3 \pmod{4}$.

(a) Let a be a quadratic residue modulo p. Prove that the number

$$b \equiv a^{\frac{p+1}{4}} \pmod{p}$$

has the property that $b^2 \equiv a \pmod{p}$. (*Hint.* Write $\frac{p+1}{2}$ as $1 + \frac{p-1}{2}$ and use Exercise 4.30.) This gives an easy way to take square roots modulo p for primes that are congruent to 3 modulo p.

(b) Use (a) to compute the following square roots modulo p. Be sure to check your answers.

(i) Solve $b^2 \equiv 116 \pmod{587}$.

(ii) Solve $b^2 \equiv 3217 \pmod{8627}$.

(iii) Solve $b^2 \equiv 9109 \pmod{10663}$.

4.32. Recall that for any $a \in \mathbb{F}_p^*$, the discrete logarithm of a (with respect to a primitive root g) is a number $\log_g(a)$ satisfying

$$g^{\log_g(a)} \equiv a \pmod{p}.$$

Prove that

$$\left(\frac{a}{p}\right) = (-1)^{\log_g(a)} \qquad \text{for all } a \in \mathbb{F}_p^*.$$

4.33. Let p be a prime. We say that a is a *cubic residue modulo p* if $p \nmid a$ and there is an integer c satisfying $a \equiv c^3 \pmod{p}$.

(a) Let a and b be cubic residues modulo p. Prove that ab is a cubic residue modulo p.

(b) Give an example to show that it is possible for none of a, b or ab to be cubic residues modulo p.

(c) Let g be a primitive root modulo p. Prove that a is a primitive root modulo p if and only if $3 \mid \log_g(a)$, where recall that $\log_g(a)$ is the discrete logarithm of a.

Section 4.10. Probabilistic encryption and and the Goldwasser-Micali public key cryptosystem

4.34. Perform the following encryptions and decryptions using the Goldwasser-Micali public key cryptosystem (Table 4.9).

(a) Bob's public key is $N = 1842338473$ and $a = 1532411781$. Alice encrypts three bits and sends Bob the ciphertext blocks

$$1794677960, \quad 525734818, \quad \text{and} \quad 420526487.$$

Decrypt Alice's message using the factorization

$$N = pq = 32411 \cdot 56843.$$

(b) Bob's public key is $N = 3149$ and $a = 2013$. Alice encrypts three bits and sends Bob the ciphertext blocks 2322, 719, and 202. Unfortunately, Bob used primes that are much too small. Factor N and decrypt Alice's message.

(c) Bob's public key is $N = 781044643$ and $a = 568980706$. Encrypt the three bits 1, 1, 0 using, respectively, the three random values $r = 705130839$, $r = 631364468$, and $r = 67651321$.

4.35. Suppose that the plaintext space $\mathcal{M}$ of a certain cryptosystem is the set of bit strings of length $2b$. Let e_k and d_k be the encryption and decryption functions associated to a key $k \in \mathcal{K}$. This exercise describes one method of turning the original cryptosystem into a probabilistic cryptosystem. Most current practical cryptosystems use more complicated variants of this idea.

Alice sends Bob an encrypted message by performing the following steps:

1. Alice chooses a b-bit message m' to be encrypted.
2. Alice chooses a string r consisting of b random bits.
3. Alice sets $m = r \| (r \oplus m')$, where $\|$ denotes concatenation[12] and $\oplus$ denotes exclusive or (see Section 1.6.4). Notice that m has length $2b$ bits.
4. Alice computes $c = e_k(m)$ and sends the ciphtertext c to Bob

(a) Explain how Bob decrypts Alice's message and recovers the plaintext m'. We asssume, of course, that Bob knows the decryption function d_k.

(b) If the plaintexts and the ciphertexts of the original cryptosystem have the same length, what is the message expansion ratio of the new probabilistic cryptosystem?

(c) More generally, if the original cryptosystem has a message expansion ratio of μ, what is the message expansion ratio of the new probabilistic cryptosystem?

[12] The *concatenation* of two bit strings is formed by placing the first string before the second string. For example, 1101‖1001 is the bit string 11011001.

Chapter 5

Elliptic Curves and Cryptology

The subject of elliptic curves encompasses a vast amount of mathematics.[1] Our aim in this section is to summarize just enough of the basic theory for cryptographic applications. For additional reading, there are a number of survey articles and books devoted to elliptic curve cryptography [3, 27, 32, 56], and many others that describe the number theoretic aspects of the theory of elliptic curves, including [10, 25, 29, 30, 53, 54, 57].

5.1 Elliptic curves [M]

An elliptic curve is the set of solutions to an equation of the form

$$Y^2 = X^3 + AX + B.$$

For example, the elliptic curves

$$E_1 : Y^2 = X^3 - 3X + 3 \qquad \text{and} \qquad E_2 : Y^2 = X^3 - 6X + 5$$

are illustrated in Figure 5.1.

The amazing feature of elliptic curves is that there is a natural way to take two points on an elliptic curve and "add" them to produce a third point. The most natural way to describe this "addition law" is using geometry.

Let P and Q be two points on an elliptic curve E, as illustrated in Figure 5.2. We start by drawing the line L going through P and Q. This line L

[1] Indeed, even before elliptic curves burst into cryptographic prominence, a well-known mathematician [29] noted that "it is possible to write endlessly on elliptic curves!"

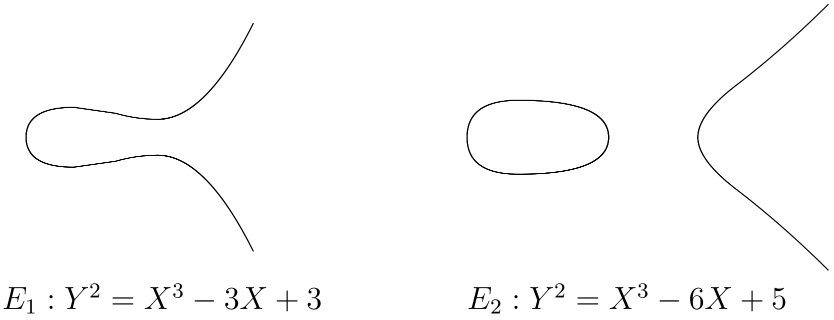

Figure 5.1: Two examples of elliptic curves

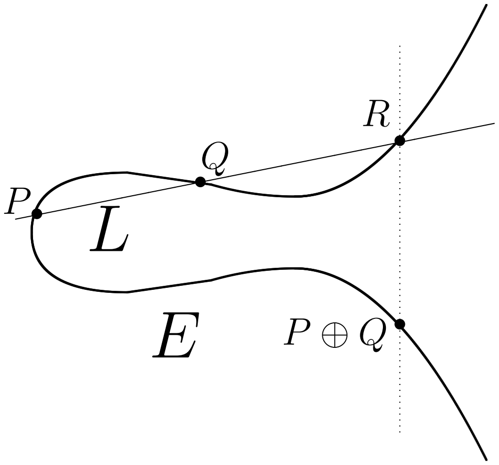

Figure 5.2: The addition law on an elliptic curve

intersects E at three points, namely the points P and Q together with one other point R. We take that point R and reflect it across the x-axis (i.e. we multiply its Y-coordinate by -1) to get a new point R'. The point R' is called the "sum of P and Q," although as you can see, this process is nothing like ordinary addition, so for now we denote this operation using the symbol $\oplus$. Thus we write

$$P \oplus Q = R'.$$

Example 5.1. Let E be the elliptic curve

$$Y^2 = X^3 - 15X + 18. \tag{5.1}$$

The points $P = (7, 16)$ and $Q = (1, 2)$ are on the curve E. The line L connecting them is given by the equation[2]

$$L : Y = \frac{7}{3}X - \frac{1}{3}. \tag{5.2}$$

In order to find the points where E and L intersect, we substitute (5.2) into (5.1) and solve for X. Thus

$$\begin{aligned}\left(\frac{7}{3}X - \frac{1}{3}\right)^2 &= X^3 - 15X + 18 \\ \frac{49}{9}X^2 - \frac{14}{9}X + \frac{1}{9} &= X^3 - 15X + 18 \\ 0 &= X^3 - \frac{49}{9}X^2 + \frac{121}{9}X + \frac{161}{9}.\end{aligned}$$

We need to find the roots of this cubic polynomial. In general, finding the roots of a cubic is difficult, but in this case, we already know two of the roots, namely $X = 7$ and $X = 1$, since we know that P and Q are in the intersection $E \cap L$. It is then easy to find the other factor,

$$X^3 - \frac{49}{9}X^2 + \frac{121}{9}X + \frac{161}{9} = (X - 7) \cdot (X - 1) \cdot (9X + 23),$$

so the third point of intersection of L and E has X-coordinate equal to $\frac{22}{9}$. Then we can find the Y-coordinate by subsituting $X = -\frac{23}{9}$ into the equation 5.2 for L. This gives $R = \left(\frac{22}{9}, -\frac{170}{27}\right)$. Finally, we reflect across the X-axis to obtain

$$P \oplus Q = \left(\frac{22}{9}, \frac{170}{27}\right).$$

There are a few subtleties to elliptic curve addition that need to be addressed. First, what happens if we want to add a point P to itself? Imagine what happens to the line L connecting P and Q if the point Q slides along the curve and gets closer and closer to P. In the limit, as Q approaches P, the line L becomes the tangent line to E at P. Thus in order to add P to itself, we simply take L to be the tangent line to E and P, as illustrated in Figure 5.3. Then L intersects E at P and at one other point R, so we can proceed as before. In some sense, L still intersects E at three points, but P counts for two of them.

[2] Recall that the equation of the line through two points (x_1, y_1) and (x_2, y_2) is given by the point-slope formula $Y - y_1 = \lambda \cdot (X - x_1)$, where the slope λ is equal to $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$.

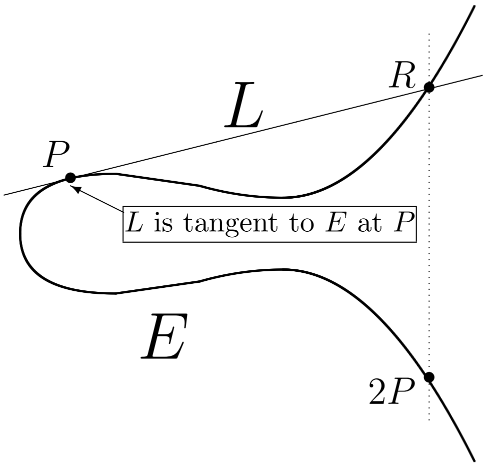

Figure 5.3: Adding a point P to itself

Example 5.2. Continuing with the curve E and point P from Example 5.1, we compute $P \oplus P$. The slope of E at P is computed by implicitly differentiating the equation 5.1 for E:

$$2Y\frac{dY}{dX} = 3X^2 - 15, \qquad \text{so} \qquad \frac{dY}{dX} = \frac{3X^2-15}{2Y}.$$

Substituting the coordinates of $P = (7, 16)$ gives slope $\lambda = \frac{33}{8}$, so the tangent line to E at P is given by the equation

$$L : Y = \frac{33}{8}X - \frac{103}{8}. \tag{5.3}$$

Now substitute (5.3) into the equation (5.1) for E, simplify, and factor,

$$\begin{aligned} \left(\frac{33}{8}X - \frac{103}{8}\right)^2 &= X^3 - 15X + 18 \\ X^3 - \frac{1089}{64}X^2 + \frac{2919}{32}X - \frac{9457}{64} &= 0 \\ (X-7)^2 \cdot (64X - 193) &= 0. \end{aligned}$$

Notice that the X-coordinate of P appears as a double root of the cubic polynomial, so it was easy for us to factor the cubic. Finally, we substitute

$X = \frac{193}{64}$ into the equation (5.3) for L to get $Y = -\frac{223}{512}$ and then switch the sign on Y to get

$$P \oplus P = \left(\frac{193}{64}, \frac{223}{512} \right).$$

A second potential problem with our "addition law" arises if we try to add a point $P = (a, b)$ to its reflection $P' = (a, -b)$. The line L through P and P' is the vertical line $x = a$, and this line intersects E in only the two points P and P'. (See Figure 5.4.) There is no third point of intersection, so it appears that we are stuck! The solution is to create an extra point $\mathcal{O}$ that lives "at infinity." More precisely, the point $\mathcal{O}$ does not live in the XY-plane, but it lies on every vertical line. We then declare by fiat that

$$P \oplus P' = \mathcal{O}.$$

But now we need to figure out how to add $\mathcal{O}$ to an ordinary point $P = (a, b)$ on E. The line L connecting P to $\mathcal{O}$ is the vertical line through P (since $\mathcal{O}$ lies on vertical lines), and that vertical line intersects E at the points P, $\mathcal{O}$, and $P' = (a, -b)$. To add P to $\mathcal{O}$, we reflect P' across the X-axis, which gets us back to P. In other words, $P \oplus \mathcal{O} = P$, so $\mathcal{O}$ acts sort of like zero for addition.

Example 5.3. Continuing with the curve E from Example 5.1, notice that the point $T = (3, 0)$ is on the curve E and that the tangent line to E at T is the vertical line $X = 3$. Thus if we add T to itself, we get $T \oplus T = \mathcal{O}$.

Definition. An *elliptic curve* E is the set of solutions to an equation of the form

$$E : Y^2 = X^3 + AX + B,$$

together with an extra point $\mathcal{O}$, where the constants A and B must satisfy $4A^3 + 27B^2 \neq 0$.

The *addition law on* E is defined as follows. Let P and Q be two points on E. Let L be the line connecting P and Q, or the tangent line to E at P if $P = Q$. Then the intersection of E and L consists of three points, counted with appropriate multiplicities and with $\mathcal{O}$ lying on every vertical line, namely P, Q, and R. Writing $R = (a, b)$, the sum of P and Q is defined to be the reflection $R' = (a, -b)$ of R across the X-axis. This sum is denoted by $P \oplus Q$, or simply by $P + Q$.

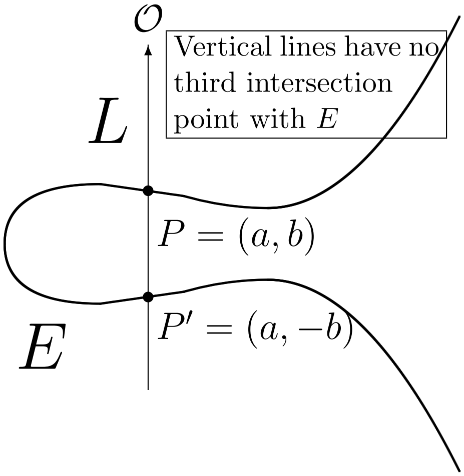

Figure 5.4: The vertical line L through $P = (a,b)$ and $P' = (a,-b)$

Further, if $P = (a,b)$, we denote the reflected point by $\ominus P = (a,-b)$, or simply by $-P$; and we define $P \ominus Q$ (or $P - Q$) to be $P \oplus (\ominus Q)$. Similarly, repeated addition is denoted as multiplication of a point by an integer,

$$nP = \underbrace{P + P + P + \cdots + P}_{n\ copies}.$$

Remark 5.4. What is this extra condition $4A^3 + 27B^2 \neq 0$? The quantity $\Delta_E = 4A^3 + 27B^2$ is called the *discriminant of E.* The condition $\Delta_E \neq 0$ is equivalent to the condition that the cubic polynomial $X^3 + AX + B$ has no repeated roots. In other words, if we factor $X^3 + AX + B$ (using complex numbers if necessary) as

$$X^3 + AX + B = (X - e_1)(X - e_2)(X - e_2),$$

then one can prove (Exercise 5.3) that

$$4A^3 + 27B^2 \neq 0 \qquad \text{if and only if} \qquad e_1, e_2, e_3 \text{ are distinct.}$$

Curves with $\Delta_E = 0$ have singular points (Exercise 5.4). The addition law does not work well on these curves. That is why we include the requirement that $\Delta_E \neq 0$ in our definition of an elliptic curve.

Theorem 5.5. *Let E be an elliptic curve. Then the addition law on E has the following properties:*

(a) $P + \mathcal{O} = \mathcal{O} + P = P$ *for all* $P \in E$. [Identity]

(b) $P + (-P) = \mathcal{O}$ *for all* $P \in E$. [Inverse]

(c) $(P + Q) + R = P + (Q + R)$ *for all* $P, Q, R \in E$. [Associative]

(d) $P + Q = Q + P$ *for all* $P, Q \in E$. [Commutative]

In other words, the addition law makes the points of E into an abelian group.

Proof. As we explained earlier, the Identity Law (a) and Inverse Law (b) are true because $\mathcal{O}$ lies on all vertical lines. The Commutative Law (d) is easy to verify, since the line that goes through P and Q is the same as the line that goes through Q and P, the order of the points does not matter.

The remaining piece of Theorem 5.5 is the Associative Law (c). One might not think that this would be hard to prove, but if you draw a picture and start to put in all of the lines needed to verify (c), you will see that it is quite complicated. There are several different ways to prove the Associative Law, but none of them is easy. After we develop explicit formulas for the addition law on E (see Theorem 5.6), you can use those formulas to check the Associative Law by a direct (but painful) calculation. More perspicacious, but less elementary, proofs may be found in [30, 54, 57] and other books on elliptic curves. □

The next step is to find explicit formulas that allow us to easily add and subtract points on an elliptic curve. The derivation of these formulas uses elementary analytic geometry, a little bit of differential calculus to find a tangent line, and a certain amount of algebraic manipulation. We state the results in the form of an algorithm, and then briefly indicate the proof.

Theorem 5.6 (Elliptic Curve Addition Algorithm)**.** *Let*

$$E : Y^2 = X^3 + AX + B$$

be an elliptic curve and let P_1 and P_2 be points on E.

(a) *If $P_1 = \mathcal{O}$, then $P_1 + P_2 = P_2$.*

(b) *Otherwise if $P_2 = \mathcal{O}$, then $P_1 + P_2 = P_1$.*

(c) *Otherwise write $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$.*

(d) *If $x_1 = x_2$ and $y_1 = -y_2$, then $P_1 + P_2 = \mathcal{O}$.*

(e) *Otherwise set* λ *equal to*

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \textit{if } P_1 \neq P_2, \\ \dfrac{3x_1^2 + A}{2y_1} & \textit{if } P_1 = P_2, \end{cases}$$

and let

$$x_3 = \lambda^2 - x_1 - x_2 \qquad \textit{and} \qquad y_3 = \lambda(x_1 - x_3) - y_1.$$

Then $P_1 + P_2 = (x_3, y_3)$.

Proof. Parts (a) and (b) are clear, and (d) is the case that the line through P_1 and P_2 is vertical, so $P_1 + P_2 = \mathcal{O}$. (Note that if $y_1 = y_2 = 0$, then the tangent line is vertical, so that case works, too.) For (e), we note that if $P_1 \neq P_2$, then λ is the slope of the line through P_1 and P_2, and if $P_1 = P_2$, then λ is the slope of the tangent line. Then in both cases this line L is given by the equation $Y = \lambda X + \nu$ with $\nu = y_1 - \lambda x_1$. Substituting the equation of L into the equation of E gives

$$(\lambda X + \nu)^2 = X^3 + AX + B,$$

so

$$X^3 - \lambda^2 X^2 + (A - 2\lambda\nu)X + (B - \nu^2) = 0.$$

We know that this cubic has x_1 and x_2 as two of its roots. If we call the third root x_3, then it factors as

$$X^3 - \lambda^2 X^2 + (A - 2\lambda\nu)X + (B - \nu^2) = (X - x_1)(X - x_2)(X - x_3).$$

Now multiply out the righthand side and look at the coefficient of X^2 on each side. That coefficient of X^2 on the righthand side is $-x_1 - x_2 - x_3$. This must equal $-\lambda^2$, which is the coefficient of X^2 on the lefthand side. This allows us to solve for $x_3 = \lambda^2 - x_1 - x_2$, and then Y-coordinate of the third intersection point of E and L is given by $\lambda x_3 + \nu$. Finally, we must reflect across the X-axis, which means replacing the Y-coordinate with its negative, to get $P_1 + P_2$. □

5.2 Elliptic curves over finite fields [M]

In the previous section we developed the theory of elliptic curves geometrically. For example, the sum of two distinct points P and Q on an elliptic curve E is defined by drawing the line L connecting P to Q and then finding the third point where L and E intersect, as illustrated in Figure 5.2. In order to apply the theory of elliptic curves to cryptography, we need to look at elliptic curves whose points have coordinates in a finite field $\mathbb{F}_p$. This is easy to do, we simply define an *elliptic curve over* $\mathbb{F}_p$ to be an equation of the form

$$E : Y^2 = X^3 + AX + B \qquad \text{with } A, B \in \mathbb{F}_p \text{ satisfying } 4A^3 + 27B^2 \neq 0,$$

and then we look at the points on E with coordinates in $\mathbb{F}_p$, which we denote by

$$E(\mathbb{F}_p) = \bigl\{(x,y) : x, y \in \mathbb{F}_p \text{ satisfy } y^2 = x^3 + Ax + B\bigr\} \cup \{\mathcal{O}\}.$$

Remark 5.7. For reasons that are explained later, we also require that $p \geq 3$. Elliptic curves over $\mathbb{F}_2$ are actually quite important in cryptography, but they are somewhat more complicated, so we delay our discussion of them until Section 5.8.

Example 5.8. Consider the elliptic curve

$$E : Y^2 = X^3 + 3X + 8 \quad \text{over the field } \mathbb{F}_{13}.$$

We can find the points of $E(\mathbb{F}_{13})$ by substituting in all possible values $X = 0, 1, 2, \ldots, 12$ and checking for which X values the quantity $X^3 + 3X + 8$ is a square modulo 13. For example, putting $X = 0$ gives 8, and 8 is not a square modulo 13. Next we try $X = 1$, which gives $1 + 3 + 8 = 12$. It turns out that 12 is a square modulo 13, in fact it has two square roots,

$$5^2 \equiv 12 \pmod{13} \qquad \text{and} \qquad 8^2 \equiv 12 \pmod{13}.$$

This gives the two points $(1,5)$ and $(1,8)$ in $E(\mathbb{F}_{13})$. Continuing in this fashion, we end up with a complete list of points,

$$E(\mathbb{F}_{13}) = \{\mathcal{O}, (1,5), (1,8), (2,3), (2,10), (9,6), (9,7), (12,2), (12,11)\}.$$

Thus $E(\mathbb{F}_{13})$ consists of nine points.

Suppose now that P and Q are two points in $E(\mathbb{F}_p)$ and that we want to "add" the points P and Q. One possibility is to develop a theory of geometry using the field $\mathbb{F}_p$ instead of $\mathbb{R}$. Then we could mimic our earlier constructions to define $P+Q$. This can be done and it leads to a fascinating field of mathematics called algebraic geometry. However, in the interests of brevity of exposition, we instead use the explicit formulas given in Theorem 5.6 to add points in $E(\mathbb{F}_p)$. But we note that if one wants to gain a deeper understanding of the theory of elliptic curves, it is necessary to use some of the machinery and some of the formalism of algebraic geometry.

Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ be points in $E(\mathbb{F}_p)$. We define the sum $P_1 + P_2$ to be the point (x_3, y_3) obtained by applying the Elliptic Curve Addition Algorithm (Theorem 5.6). Notice that in this algorithm, the only operations used are addition, subtraction, multiplication, and division of the coefficients of E and the coordinates of P and Q. Since those coefficients and coordinates are in the field $\mathbb{F}_p$, we end up with a point (x_3, y_3) whose coordinates are in $\mathbb{F}_p$. Of course, it is not completely clear that (x_3, y_3) is a point in $E(\mathbb{F}_p)$.

Theorem 5.9. *Let E be an elliptic curve over $\mathbb{F}_p$ and let P and Q be points in $E(\mathbb{F}_p)$.*

(a) *The Elliptic Curve Addition Algorithm (Theorem 5.6) applied to P and Q yields a point in $E(\mathbb{F}_p)$. We denote this point by $P + Q$.*

(b) *This addition law on $E(\mathbb{F}_p)$ satisfies all of the properties listed in Theorem 5.5. In other words, this addition law makes $E(\mathbb{F}_p)$ into a finite group.*

Proof. The formulas in Theorem 5.6(e) are derived by substituting the equation of a line into the equation for E and solving for X, so the resulting point is automatically a point on E, i.e. it is a solution to the equation defining E. This shows why (a) is true, although when $P = Q$, a small additional argument is needed to indicate why the resulting cubic polynomial has a double root. For (b), the Identity Law follows from the Addition Algorithm Steps (a) and (b), the Inverse Law is clear from the Addition Algorithm Step (d), and the Commutative Law is easy since a brief examination of the Addition Algorithm shows that switching the two points leads to the same result. Unfortunately, the Associative Law is not so clear. It is possible to verify the Associative Law directly using the Addition Algorithm formulas, although there are many special cases to consider. The alternative is to develop more of the general theory of elliptic curves, as is done in the references cited in the proof of Theorem 5.5. □

Example 5.10. We continue with the elliptic curve

$$E : Y^2 = X^3 + 3X + 8 \qquad \text{over } \mathbb{F}_{13}$$

from Example 5.8 and we use the Addition Algorithm (Theorem 5.6) to add the points $P = (9, 7)$ and $Q = (1, 8)$ in $E(\mathbb{F}_{13})$. Step (e) of that algorithm tells us to first compute

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} = \frac{8 - 7}{1 - 9} = \frac{1}{-8} = \frac{1}{5} = 8,$$

where remember that all computations are being performed in the field $\mathbb{F}_{13}$, so $-8 = 5$ and $\frac{1}{5} = 8$. Next we compute

$$\nu = y_1 - \lambda x_1 = 7 - 8 \cdot 9 = -65 = 0.$$

Finally, the addition algorithm tells us to compute

$$\begin{aligned} x_3 &= \lambda^2 - x_1 - x_2 = 64 - 9 - 1 = 54 = 2 \quad \text{and} \\ y_3 &= -(\lambda x_3 + \nu) = -8 \cdot 2 = -16 = 10. \end{aligned}$$

This completes the computation of

$$P + Q = (1, 8) + (9, 7) = (2, 10) \qquad \text{in } E(\mathbb{F}_{13}).$$

Similarly, we can use the Addition Algorithm to add $P = (9, 7)$ to itself. We have (remember that all calculations are in $\mathbb{F}_{13}$)

$$\lambda = \frac{3x_1^2 + A}{2y_1} = \frac{3 \cdot 9^2 + 3}{2 \cdot 7} = \frac{246}{14} = 1 \quad \text{and} \quad \nu = y_1 - \lambda x_1 = 7 - 1 \cdot 9 = 11.$$

Then

$$x_3 = \lambda^2 - x_1 - x_2 = 1 - 9 - 9 = 9 \quad \text{and} \quad y_3 = -(\lambda x_3 + \nu) = -1 \cdot 9 - 11 = 6,$$

so $P + P = (9, 7) + (9, 7) = (9, 6)$ in $E(\mathbb{F}_{13})$. In a similar fashion, we can compute the sum of every pair of points in $E(\mathbb{F}_{13})$. The results are listed in Table 5.1.

	$\mathcal{O}$	$(1,5)$	$(1,8)$	$(2,3)$	$(2,10)$	$(9,6)$	$(9,7)$	$(12,2)$	$(12,11)$
$\mathcal{O}$	$\mathcal{O}$	$(1,5)$	$(1,8)$	$(2,3)$	$(2,10)$	$(9,6)$	$(9,7)$	$(12,2)$	$(12,11)$
$(1,5)$	$(1,5)$	$(2,10)$	$\mathcal{O}$	$(1,8)$	$(9,7)$	$(2,3)$	$(12,2)$	$(12,11)$	$(9,6)$
$(1,8)$	$(1,8)$	$\mathcal{O}$	$(2,3)$	$(9,6)$	$(1,5)$	$(12,11)$	$(2,10)$	$(9,7)$	$(12,2)$
$(2,3)$	$(2,3)$	$(1,8)$	$(9,6)$	$(12,11)$	$\mathcal{O}$	$(12,2)$	$(1,5)$	$(2,10)$	$(9,7)$
$(2,10)$	$(2,10)$	$(9,7)$	$(1,5)$	$\mathcal{O}$	$(12,2)$	$(1,8)$	$(12,11)$	$(9,6)$	$(2,3)$
$(9,6)$	$(9,6)$	$(2,3)$	$(12,11)$	$(12,2)$	$(1,8)$	$(9,7)$	$\mathcal{O}$	$(1,5)$	$(2,10)$
$(9,7)$	$(9,7)$	$(12,2)$	$(2,10)$	$(1,5)$	$(12,11)$	$\mathcal{O}$	$(9,6)$	$(2,3)$	$(1,8)$
$(12,2)$	$(12,2)$	$(12,11)$	$(9,7)$	$(2,10)$	$(9,6)$	$(1,5)$	$(2,3)$	$(1,8)$	$\mathcal{O}$
$(12,11)$	$(12,11)$	$(9,6)$	$(12,2)$	$(9,7)$	$(2,3)$	$(2,10)$	$(1,8)$	$\mathcal{O}$	$(1,5)$

Table 5.1: Addition table for $E : Y^2 = X^3 + 3X + 8$ over $\mathbb{F}_{13}$

It is clear that the set of points $E(\mathbb{F}_p)$ is a finite set, since there are only finitely many possibilities for the X and Y-coordinates. More precisely, there are p possibilities for X, and then for each X, the equation

$$Y^2 = X^3 + AX + B$$

shows that there are at most two possibilities for Y. (See Exercise 1.25.) Adding in the extra point $\mathcal{O}$, this shows that $\#E(\mathbb{F}_p)$ is at most $2p+1$ points. However, this estimate is considerably too large.

When we plug in a value for X, there are three possibilities for the value of the quantity

$$X^3 + AX + B.$$

First, it may be a quadratic residue modulo p, in which case it has two square roots and we get two points in $E(\mathbb{F}_p)$. This happens about 50% of the time. Second, it may be a nonresidue modulo p, in which case we discard X. This also happens about 50% of the time. Third, it might equal 0, in which case we get one point in $E(\mathbb{F}_p)$, but this case happens very rarely.[3] Thus we might expect the number of points in $E(\mathbb{F}_p)$ to be approximately

$$\#E(\mathbb{F}_p) \approx 50\% \cdot 2 \cdot p + 1 = p + 1.$$

A famous theorem of Hasse, later vastly generalized by Weil and Deligne, says that this is true up to random fluctuations.

Theorem 5.11 (Hasse)**.** *Let E be an elliptic curve over $\mathbb{F}_p$. Then*

$$\#E(\mathbb{F}_p) = p + 1 - t_p \qquad \text{with } t_p \text{ satisfying } |t_p| \le 2\sqrt{p}.$$

[3] The congruence $X^3 + AX + B \equiv 0 \pmod{p}$ has at most three solutions.

Definition. The quantity

$$t_p = p + 1 - \#E(\mathbb{F}_p)$$

appearing in Theorem 5.11 is called the *trace of Frobenius* for $E/\mathbb{F}_p$. We will not explain the somewhat technical reasons for this name, other than to say that t_p appears as the trace of a certain 2-by-2 matrix that acts as a linear transformation on a certain two-dimensional vector space associated to $E/\mathbb{F}_p$.

Example 5.12. Let E be given by the equation

$$E : Y^2 = X^3 + 4X + 6.$$

We can think of E as an elliptic curve over $\mathbb{F}_p$ for different finite fields $\mathbb{F}_p$ and count the number of points in $E(\mathbb{F}_p)$. Tabele 5.2 lists the results for the first few primes, together with the value of t_p and, for comparison purposes, the value of $2\sqrt{p}$.

p	$\#E(F_p)$	t_p	$2\sqrt{p}$
3	4	0	3.46
5	8	-2	4.47
7	11	-3	5.29
11	16	-4	6.63
13	14	0	7.21
17	15	3	8.25

Table 5.2: Number of points and trace of Frobenius for $E : Y^2 = X^3 + 4X + 6$

Remark 5.13. Hasse's Theorem 5.11 gives a bound for $\#E(\mathbb{F}_p)$, but it does not provide a method for calculating this quantity. In principle, one can substitute in each value for X and check the value of $X^3 + AX + B$ against a table of squares modulo p, but this takes time $O(p)$, so is very inefficient. Schoof [47] found an algorithm to compute $\#E(\mathbb{F}_p)$ in time $O\big((\log p)^6\big)$, i.e. he found a polynomial-time algorithm. Schoof's algorithm was improved and made practical by Elkies and Atkin, so it is now known as the *SEA Algorithm*. We will not describe SEA, which uses advanced techniques from the theory of elliptic curves, but see [48]. Also see Remark 5.24 in Section 5.8 for another counting algorithm due to Satoh that is designed for a different type of finite field.

5.3 The elliptic curve discrete logarithm problem (ECDLP) [M]

In Chapter 2 we talked about the Discrete Logarithm Problem (DLP) modulo a prime number p. In the DLP for $\mathbb{F}_p^*$, Alice publishes two numbers g and h, and her secret is the exponent x that makes

$$h \equiv g^x \pmod{p}.$$

Can Alice do something similar with an elliptic curve E over $\mathbb{F}_p$? If Alice views g and h as being elements of the group $\mathbb{F}_p^*$, then the discrete logarithm problem requires Alice's adversary Eve to find an x so that

$$h \equiv \underbrace{g \cdot g \cdot g \cdots g}_{x \text{ multiplications}} \pmod{p}.$$

In other words, Eve needs to determine how many times g must be multiplied by itself in order to get to h.

With this formulation, it is clear that Alice can do the same thing with an elliptic curve E over a finite field $E(\mathbb{F}_p)$. She chooses and publishes two points P and Q in $E(\mathbb{F}_p)$, and her secret is an integer n that makes

$$Q = \underbrace{P + P + P + \cdots + P}_{n \text{ additions on } E} = nP.$$

Then Eve needs to find out how many times P must be added to itself in order to get Q. Keep in mind that although the "addition law" on an elliptic curve is conventionally written with a plus sign, addition on E is actually a very complicated operation, so this elliptic analog of the discrete logarithm problem may be quite difficult to solve.

Definition. Let E be an elliptic curve over the finite field $\mathbb{F}_p$ and let P and Q be points in $E(\mathbb{F}_p)$. The *Elliptic Curve Discrete Logarithm Problem* (ECDLP) is the problem of finding an integer n so that $Q = nP$. By analogy with the discrete logarithm problem for $\mathbb{F}_p^*$, we denote this integer n by

$$n = \log_P(Q)$$

and we call n the *elliptic discrete logarithm of Q with respect to P*.

Remark 5.14. Our definition of $\log_P(Q)$ is not quite precise. The first difficulty is that there may be points $P, Q \in E(\mathbb{F}_p)$ for which Q is not a multiple of P. In this case, $\log_P(Q)$ is not defined. However, for cryptographic purposes, Alice starts out with a public point P and a private integer n and she computes and publishes the value of $Q = nP$. So in practical applications, $\log_P(Q)$ exists and its value is Alice's secret.

The second difficulty is that if there is one value of n satisfying $Q = nP$, then there are many such values. Thus if n_0 is the smallest such positive integer and if we let s be the smallest integer such that $sP = \mathcal{O}$, then the solutions to $Q = nP$ are the integers $n = n_0 + is$ with $i \in \mathbb{Z}$. (See Exercise 5.9.) Thus $\log_P(Q)$ may either be defined as an integer modulo s, or for concreteness we can set it equal n_0. The advantage of the former approach is that the elliptic discrete logarithm then defines a group homomorphism

$$\log_P : E(\mathbb{F}_p) \longrightarrow \mathbb{Z}/s\mathbb{Z},$$

which means that it satisfies

$$\log_P(Q_1 + Q_2) = \log_P(Q_1) + \log_P(Q_2) \qquad \text{for all } Q_1, Q_2 \in E(\mathbb{F}_p).$$

Notice the analogy with the ordinary logarithm $\log(\alpha\beta) = \log(\alpha) + \log(\beta)$ and the discrete logarithm for $\mathbb{F}_p$ (cf. Remark 2.2).

Example 5.15. Consider the elliptic curve

$$E : Y^2 = X^3 + 8X + 7 \quad \text{over } \mathbb{F}_{73}.$$

The points $P = (32, 53)$ and $Q = (39, 17)$ are both in $E(\mathbb{F}_{73})$, and it is easy to verify (by hand if you're patient and with a computer if not) that

$$Q = 11P, \qquad \text{so} \qquad \log_P(Q) = 11.$$

Similarly, $R = (35, 47) \in E(\mathbb{F}_{73})$ and $S = (58, 4) \in E(\mathbb{F}_{73})$, and after some computation we find that they satisfy $R = 37P$ and $S = 28P$, so

$$\log_P(R) = 37 \qquad \text{and} \qquad \log_P(S) = 28.$$

Finally, we mention that $\#E(\mathbb{F}_{73}) = 82$, but that P satisfies $41P = \mathcal{O}$, so the multiples of P only sweep out half of the points in $E(\mathbb{F}_{73})$. For example, $(20, 65)$ is in $E(\mathbb{F}_{73})$, but it does not equal a multiple of P.

5.3.1 The double-and-add algorithm

It appears to be quite difficult to recover the value of n from the two points P and $Q = nP$ in $E(\mathbb{F}_p)$, i.e. it is difficult to solve the ECDLP. We will say more about the difficulty of the ECDLP in later sections. However, in order to use the function

$$\mathbb{Z} \longrightarrow E(\mathbb{F}_p), \qquad n \longmapsto (P, nP),$$

for cryptography, we need to efficiently compute nP from the known values n and P. If n is large, we certainly do not want to compute nP by computing $P, 2P, 3P, 4P, \ldots$.

The most efficient way to compute nP is very similar to the method that we described in Section 1.3.2 for computing powers $a^n \pmod N$, which we need for Diffie-Hellman Key Exchange (Section 2.3) and for the ElGamal and RSA Public Key Cryptosystems (Sections 2.4 and 4.2). However, since the operation on an elliptic curve is written as addition instead of as multiplication, we call it "double-and-add" instead of "square-and-multiply."

The underlying idea is the same as before. We first write n in binary form as

$$n = n_0 + n_1 \cdot 2 + n_2 \cdot 4 + n_3 \cdot 8 + \cdots + n_r \cdot 2^r \qquad \text{with } n_0, n_1, \ldots, n_r \in \{0, 1\}.$$

(We also assume that $n_r = 1$.) Next we compute the 2-power multiples of P in r duplication steps,

$$Q_0 = P, \quad Q_1 = 2Q_0, \quad Q_2 = 2Q_1, \ldots, \quad Q_r = 2Q_{r-1}.$$

Notice that $Q_i = 2^i P$. Finally, we compute nP in at most r addition steps,

$$nP = n_0 Q_0 + n_1 Q_1 + n_2 Q_2 + \cdots + n_r Q_r.$$

The total time to compute nP is at most $2r$ point operations in $E(\mathbb{F}_p)$. Notice that $n \geq 2^r$, so it takes no more than $2 \log_2(n)$ point operations to compute nP. This makes it feasible to compute nP even for very large values of n. We have summarized the Double-and-Add Algorithm in Table 5.3.

Example 5.16. We use the Double-and-Add Algorithm as described in Table 5.3 to compute nP in $E(\mathbb{F}_p)$ for

$$n = 947, \qquad E : Y^2 = X^3 + 14X + 19, \qquad p = 3623, \qquad P = (6, 730).$$

```
    Input. Point $P \in E(\mathbb{F}_p)$ and integer $n \ge 1$.
1.  Set $Q = P$ and $R = \mathcal{O}$.
2.  Loop while $n > 0$.
    3.  If $n \equiv 1 \pmod 2$, set $R = R + Q$.
    4.  Set $Q = 2Q$ and $n = \lfloor n/2 \rfloor$.
    5.  If $n > 0$, continue with loop at Step 2.
6.  Return the point $R$, which equals $nP$.
```

Table 5.3: The double-and-add algorithm for elliptic curves

Step i	n	$Q = 2^i P$	R
0	947	$(6, 730)$	$\mathcal{O}$
1	473	$(2521, 3601)$	$(6, 730)$
2	236	$(2277, 502)$	$(2149, 196)$
3	118	$(3375, 535)$	$(2149, 196)$
4	59	$(1610, 1851)$	$(2149, 196)$
5	29	$(1753, 2436)$	$(2838, 2175)$
6	14	$(2005, 1764)$	$(600, 2449)$
7	7	$(2425, 1791)$	$(600, 2449)$
8	3	$(3529, 2158)$	$(3247, 2849)$
9	1	$(2742, 3254)$	$(932, 1204)$
10	0	$(1814, 3480)$	$(3492, 60)$

Figure 5.5: Computing $947 \cdot (6, 730)$ on $Y^2 = X^3 + 14X + 19$ modulo 3623

The binary expansion of n is

$$n = 947 = 1 + 2 + 2^4 + 2^5 + 2^7 + 2^8 + 2^9.$$

The step-by-step calculation, which requires 9 doublings and 6 additions, is given in Figure 5.5. The final result is $947P = (3492, 60)$.

5.3.2 How hard is the ECDLP?

The collision algorithms described in Section 3.4 are easily adapted to any group, for example to the group of point $E(\mathbb{F}_p)$ on an elliptic curve. In order to solve $Q = nP$, Eve chooses random integers $j_1, \ldots, j_n$ and $k_1, \ldots, k_n$

between 1 and $p-1$ and makes two lists of points:

$$\begin{array}{ll} \text{List \#1.} & j_1P,\ j_2P,\ j_3P,\ldots,\ j_nP \\ \text{List \#2.} & k_1P+Q,\ k_2P+Q,\ k_3P+Q,\ldots,\ k_nP+Q \end{array}$$

As soon as she finds a match (collision) between the two lists, she is done, since as if she finds $j_uP = k_vP + Q$, then $Q = (j_u - k_v)P$ provides the solution. As we saw in Section 3.4, if n is somewhat larger than $\sqrt{p}$, say $n \approx 3\sqrt{p}$, then there is a very good chance that there will be a collision.

This naive collision algorithm requires quite a lot of storage for the two lists. However, it is not hard to adapt Pollard's ρ method from Section 3.5 to devise a storage-free collision algorithm with a similar running time. (See Exercise 5.11.) In any case, there are certainly algorithms that solve the ECDLP for $E(\mathbb{F}_p)$ in approximately $\sqrt{p}$ steps.

We have seen that there are much faster ways to solve the discrete logarithm problem for $\mathbb{F}_p^*$, including especially the Index Calculus (Section 4.8) whose running time is subexponential, i.e. whose running time is smaller than a multiple of p^ϵ for every $\epsilon > 0$. The principal reason that elliptic curves are used in cryptography is the fact that there are no index calculus algorithms known for the ECDLP, and indeed, there are no general algorithms known that solve the ECDLP in fewer than $O(\sqrt{p}\,)$ step. In other words, despite the highly structured nature of the group $E(\mathbb{F}_p)$, the fastest known algorithms to solve ECDLP are no better than the generic algorithm that works equally well to solve the discrete logarithm problem on any group. This fact is sufficiently important that it bears repeating.

The fastest known algorithm to solve ECDLP in $E(\mathbb{F}_p)$ takes approximately $\sqrt{p}$ steps.

Thus the ECDLP appears to be much more difficult than the DLP. Recall, however, there are some primes p for which the DLP in $\mathbb{F}_p^*$ is comparatively easy. For example, if $p-1$ is a product of small primes, we saw that the Pohlig-Hellman Algorithm 2.23 gives a quick solution to the DLP in $\mathbb{F}_p^*$. In a similar way, there are some elliptic curves and some primes for which the ECDLP in $E(\mathbb{F}_p)$ is comparatively easy. We discuss these special cases, which must be avoided when devising secure cryptosystems, in Section 5.9.

5.4 Elliptic curve cryptography

It is finally time to apply elliptic curves to cryptography. We start with the easiest application, Diffie-Hellman key exchange, which involves little more than replacing the discrete logarithm problem for the finite field $\mathbb{F}_p$ with the discrete logarithm problem for an elliptic curve $E(\mathbb{F}_p)$. We then describe an elliptic analog of the ElGamal public key cryptosystem.

5.4.1 Elliptic Diffie-Hellman key exchange

Alice and Bob agree to use a particular elliptic curve $E(\mathbb{F}_p)$ and a particular point $P \in E(\mathbb{F}_p)$. Alice chooses a secret integer n_A and Bob chooses a secret integer n_B. They compute the associated multiples

$$\overbrace{Q_A = n_A P}^{\text{Alice computes this}} \qquad \text{and} \qquad \overbrace{Q_B = n_B P}^{\text{Bob computes this}}$$

and exchange the values of Q_A and Q_B. Alice then uses her secret multiplier to compute $n_A Q_B$ and Bob similarly computes $n_B Q_A$. They have now shared the secret value

$$n_A Q_B = (n_A n_B)P = n_B Q_A.$$

Table 5.4 summarizes elliptic Diffie-Hellman key exchange.

Example 5.17. Alice and Bob decide to use elliptic Diffie-Hellman with the following prime, curve, and point:

$$p = 3851, \qquad E : Y^2 = X^3 + 324X + 1287, \qquad P = (920, 303) \in E(\mathbb{F}_{3851}).$$

Alice and Bob choose respective secret values $n_A = 1194$ and $n_B = 1759$, and then

$$\begin{array}{ll} \text{Alice computes} & Q_A = 1194P = (2067, 2178) \in E(\mathbb{F}_{3851}), \\ \text{Bob computes} & Q_B = 1759P = (3684, 3125) \in E(\mathbb{F}_{3851}). \end{array}$$

Alice sends Q_A to Bob and Bob sends Q_B to Alice. Finally,

$$\begin{array}{ll} \text{Alice computes} & n_A Q_B = 1194(3684, 3125) = (3347, 1242) \in E(\mathbb{F}_{3851}), \\ \text{Bob computes} & n_B Q_A = 1759(2067, 2178) = (3347, 1242) \in E(\mathbb{F}_{3851}). \end{array}$$

Bob and Alice have exchanged the secret point $(3347, 1242)$. As explained in Remark 5.18, they should discard the y-coordinate and treat only the value $x = 3347$ as a secret shared value.

Public Parameter Creation	
A trusted party chooses and publishes a (large) prime p, an elliptic curve E over $\mathbb{F}_p$, and a point P in $E(\mathbb{F}_p)$.	
Private Computations	
Alice	**Bob**
Choose a secret integer n_A. Compute the point $Q_A = n_A P$.	Choose a secret integer n_B. Compute the point $Q_B = n_B P$.
Public Exchange of Values	
Alice sends Q_A to Bob	$\longrightarrow$ Q_A
Q_B $\longleftarrow$	Bob sends Q_B to Alice
Further Private Computations	
Alice	**Bob**
Compute the point $n_A Q_B$.	Compute the point $n_B Q_A$.
The shared secret value is $n_A Q_B = n_A(n_B P) = n_B(n_A P) = n_B Q_A$.	

Table 5.4: Diffie-Hellman key exchange using elliptic curves

One way for Eve to discover Alice and Bob's secret is to solve the ECDLP

$$nP = Q_A,$$

since if Eve can solve this problem, then she knows n_A and can use it to compute $n_A Q_B$. Of course, there might some other way for Eve to compute their secret without actually solving the ECDLP. The precise problem that Eve needs to solve is the elliptic analog of the Diffie-Hellman problem described on page 65.

Definition. : Let $E(\mathbb{F}_p)$ be an elliptic curve over a finite field and let $P \in E(\mathbb{F}_p)$. The *Elliptic Diffie-Hellman Problem* (EDHP) is the problem of computing the value of $n_1 n_2 P$ from the known values of $n_1 P$ and $n_2 P$.

Remark 5.18. Ellliptic Diffie-Hellman key exchange requires Alice and Bob to exchange the value of points on an elliptic curves. A point Q in $E(\mathbb{F}_p)$ consists of two coordinates $Q = (x_Q, y_Q)$, where x_Q and y_Q are elements of the finite field $\mathbb{F}_p$, so it appears that Alice must send Bob two numbers in $\mathbb{F}_p$. However, those two numbers modulo p do not contain as much information as one might expect, since they are related by the formula

$$y_Q^2 = x_Q^3 + Ax_Q + B \qquad \text{in } \mathbb{F}_p.$$

Note that Eve knows A and B, so if she can guess the correct value of x_Q, then there are only two possible values for y_Q, and in practice it is not too hard for her to actually compute the two values of y_Q.

There is thus little reason for Alice to send both coordinates of Q_A to Bob, since the y-coordinate contains so little additional information. Instead, she sends Bob only the x-coordinate of Q_A. Bob then computes and uses one of the two possible y-coordinates. If he happens to choose the "correct" y, then he is using Q_A, and if he chooses the "incorrect" y (which is the negative of the correct y), then he is using $-Q_A$. In any case, Bob ends up computing one of

$$\pm n_B Q_A = \pm(n_A n_B)P.$$

Similarly, Alice ends up computing one of $\pm(n_A n_B)P$. Then Alice and Bob use the x-coordinate as their shared secret value, since that x-coordinate is the same regardless of which y they use.

Example 5.19. Alice and Bob decide to exchange another secret value using the same public parameters as in Example 5.17:

$$p = 3851, \qquad E : Y^2 = X^3 + 324X + 1287, \qquad P = (920, 303) \in E(\mathbb{F}_{3851}).$$

However, this time they want to send fewer bits to one another. Alice and Bob respectively choose new secret values $n_A = 2489$ and $n_B = 2286$, and as before,

$$\begin{array}{ll} \text{Alice computes} & Q_A = n_A P = 2489(920, 303) = (593, 719) \in E(\mathbb{F}_{3851}), \\ \text{Bob computes} & Q_B = n_B P = 2286(920, 303) = (3681, 612) \in E(\mathbb{F}_{3851}). \end{array}$$

However, rather than sending both coordinates, Alice sends only $x_A = 593$ to Bob and Bob sends only $x_B = 3681$ to Alice.

Alice substitutes $x_B = 3681$ into the equation for E and finds that

$$y_B^2 = x_B^3 + 324x_B + 1287 = 3681^3 + 324 \cdot 3681 + 1287 = 997.$$

(Remember that all calculations are performed in $\mathbb{F}_{3851}$.) Alice needs to compute a square root of 997 modulo 3851. This is not hard to do, especially since for primes satisfying $p \equiv 3 \pmod 4$, Proposition 2.15 tells her that $b^{(p+1)/4}$ is a square root of b modulo p. So Alice sets

$$y_B = 997^{(3851+1)/4} = 997^{963} \equiv 612 \pmod{3851}.$$

It happens that she gets the same point $Q_B = (x_B, y_B) = (3681, 612)$ that Bob used and she computes $n_A Q_B = 2489(3681, 612) = (509, 1108)$.

Similarly, Bob substitutes $x_A = 593$ into the equation for E and takes a square root,

$$\begin{aligned} y_A^2 &= x_A^3 + 324x_A + 1287 = 593^3 + 324 \cdot 593 + 1287 = 927 \\ y_A &= 927^{(3851+1)/4} = 927^{963} \equiv 3132 \pmod{3851}. \end{aligned}$$

Bob then uses the point $Q'_A = (593, 3132)$, which is not actually the point that Alice used, to compute $n_B Q'_A = 2286(593, 3132) = (509, 2743)$. Bob and Alice end up with points that are negatives of one another, but that is all right, since their shared secret value is the x-coordinate $x = 593$, which is the same for both points.

5.4.2 Elliptic ElGamal public key cryptosystem

It is easy to create a direct analog of the ElGamal public key cryptosystem described in Section 2.4. Briefly, Alice and Bob agree to use a particular prime p, elliptic curve E, and point $P \in E(\mathbb{F}_p)$. Alice chooses a secret multiplier n_A and publishes the point $Q_A = n_A P$ as her public key. Bob's plaintext is a point $M \in E(\mathbb{F}_p)$. He chooses an integer k to be his ephemeral key and computes

$$C_1 = kP \qquad \text{and} \qquad C_2 = M + kQ_A.$$

He sends the two points (C_1, C_2) to Alice, who computes

$$C_2 - n_A C_1 = (M + kQ_A) - n_A(kP) = M + k(n_A P) - n_A(kP) = M$$

to recover the plaintext. The elliptic ElGamal public key cryptosystem is summarized in Table 5.5.

In principle, the elliptic ElGamal cryptosystem works fine, but there are some practical difficulties.

1. There is no obvious way to attach plaintext messages to points in $E(\mathbb{F}_p)$.
2. The elliptic ElGamal cryptosystem has has 4-to-1 message expansion, as compared to the 2-to-1 expansion ratio of ElGamal using $\mathbb{F}_p$ (cf. Remark 2.10).

Public Parameter Creation	
A trusted party chooses and publishes a (large) prime p, an elliptic curve E over $\mathbb{F}_p$, and a point P in $E(\mathbb{F}_p)$.	
Alice	**Bob**
Key Creation	
Choose a private key n_A. Compute $Q_A = n_A P$ in $E(\mathbb{F}_p)$. Publish the public key Q_A.	
Encryption	
	Choose plaintext $M \in E(\mathbb{F}_p)$. Choose random ephemeral key k. Use Alice's public key Q_A to compute $C_1 = kP \in E(\mathbb{F}_p)$. and $C_2 = M + kQ_A \in E(\mathbb{F}_p)$. Send ciphtertext (C_1, C_2) to Alice.
Decryption	
Compute $C_2 - n_A C_1 \in E(\mathbb{F}_p)$. This quantity is equal to M.	

Table 5.5: Elliptic ElGamal key creation, encryption, and decryption

The reason that elliptic ElGamal has a 4-to-1 message expansion lies in the fact that the plaintext M is a single point in $E(\mathbb{F}_p)$, so there are only (approximately) p different plaintexts from Hasse's theorem 5.11. However, the ciphertext (C_1, C_2) consists of four numbers modulo p, since each point in $E(\mathbb{F}_p)$ has two coordinates.

Various methods have been proposed to solve these problems. The difficulty with associating plaintext to points can be circumvented by choosing M randomly and using it as a mask for the actual plaintext. One such method, which also decreases message expansion, is described in Exercise 5.15.

Another natural way to improve message expansion is to send only the x-coordinates of C_1 and C_2, as was suggested for Diffie-Hellman key exchange in Remark 5.18. Unfortunately, since Alice must compute the difference $C_2 - n_A C_1$, she needs the correct values of both the x and y coordinates of C_1 and C_2. (Note that the points $C_2 - n_A C_1$ and $C_2 + n_A C_1$ are quite different!) However, the x-coordinate of a point determines the y coordinate up to change of sign, so Bob can send one extra bit, for example

$$\text{Extra bit} = \begin{cases} 0 & \text{if } 0 \le y < \frac{1}{2}p, \\ 1 & \text{if } \frac{1}{2}p < y < p \end{cases}$$

(See Exercise 5.14.) In this way, Bob only needs to send the x-coordinates of C_1 and C_2 plus two extra bits.

5.5 The evolution of public key cryptography [H]

5.6 The elliptic curve factorization algorithm

Pollard's $p-1$ factorization method, which we discussed in Section 4.5, finds factors of $N = pq$ by searching for a power a^L with the property that

$$a^L \equiv 1 \pmod{p} \qquad \text{and} \qquad a^L \not\equiv 1 \pmod{q}.$$

Fermat's Little Theorem tells us that this is likely to work if $p-1$ divides L and $q-1$ does not divide L. So what we did was to take $L = n!$ for some moderate value of n. Then we hoped that $p-1$ (or $q-1$, but not both) is a product of small primes, hence divides $n!$. Thus Pollard's method works well for some numbers, but not for others. The determining factor is whether $p-1$ or $q-1$ is a product of small primes.

What is it about the quantity $p-1$ that makes it so important for Pollard's method? The answer is Fermat's Little Theorem, or in more intrinsic terms, it is because there are $p-1$ elements in $\mathbb{F}_p^*$, so every element α of $\mathbb{F}_p^*$ satisfies $\alpha^{p-1} = 1$. Now consider that last statement as it relates to the theme of this chapter, which is that the points and addition law for an elliptic curve $E(\mathbb{F}_p)$ are very much analogous to the elements and multiplication law for $\mathbb{F}_p^*$. Hendrik Lenstra [31] made this analogy precise by devising a factorization algorithm that uses the group law on an elliptic curve E in place of multiplication in $\mathbb{Z}/N\mathbb{Z}$.

In order to describe Lenstra's algorithm, we need to work with an elliptic curve modulo N, where the integer N is not prime, so the ring $\mathbb{Z}/N\mathbb{Z}$ is not a field. However, suppose that we start with an equation

$$E : Y^2 = X^3 + AX + B$$

and suppose that $P = (a, b)$ is a point on E modulo N, by which we mean that

$$b^2 \equiv a^3 + A \cdot a + B \pmod{N}.$$

Then we can apply the Elliptic Curve Addition Algorithm (Theorem 5.6) to compute $2P, 3P, 4P, \ldots$, since the only operations required by that algorithm are addition, subtraction, multiplication and division.

Example 5.20. Let $N = 187$ and consider the elliptic curve

$$E : Y^2 = X^3 + 3X + 7$$

modulo 187 and the point $P = (38, 112)$ that is on E modulo 187. In order to compute $2P \bmod 187$, we follow the Elliptic Curve Addition Algorithm

(Theorem 5.6) and compute

$$
\begin{aligned}
\frac{1}{2y(P)} &= \frac{1}{224} \equiv 91 \pmod{187} \\
\lambda = \frac{3x(P)^2 + A}{2y(P)} &= \frac{4335}{224} \equiv 34 \cdot 91 \equiv 102 \pmod{187} \\
x(2P) = \lambda^2 - 2x(P) &= 10328 \equiv 43 \pmod{187} \\
y(2P) = \lambda\bigl(x(P) - x(2P)\bigr) - y(P) &= 102(38 - 43) - 112 \equiv 126 \pmod{187}
\end{aligned}
$$

Thus $2P = (43, 126)$ on the curve E modulo 187.

Notice that during this calculation, we needed to find the reciprocal of 224 modulo 187, or equivalently, we needed to solve the congruence

$$224d \equiv 1 \text{ (mod 187)}.$$

This is easily accomplished using the Extended Euclidean Algorithm (Theorem 1.11, see also Remark 1.14 and Exercise 1.11), since it turns out that $\gcd(224, 187) = 1$.

In a similar fashion we compute $3P = 2P + P$. In this case, we are adding distinct points, so the formula for λ is different, but the computation is similar.

$$
\begin{aligned}
\frac{1}{x(2P) - x(P)} &= \frac{1}{5} \equiv 75 \pmod{187} \\
\lambda = \frac{y(2P) - y(P)}{x(2P) - x(P)} &= \frac{14}{5} \equiv 14 \cdot 75 \equiv 115 \pmod{187} \\
x(3P) = \lambda^2 - x(2P) - x(P) &= 13144 \equiv 54 \pmod{187} \\
y(3P) = \lambda\bigl(x(P) - x(3P)\bigr) - y(P) &= 115(38 - 54) - 112 \equiv 105 \pmod{187}
\end{aligned}
$$

Thus $3P = (54, 105)$ on the curve E modulo 187. Again we needed to compute a reciprocal, in this case, the reciprocal of 5 modulo 187. We leave it to you to continue the calculations. For example, it is instructive to check that $P + 3P$ and $2P + 2P$ give the same answer, namely $4P = (93, 64)$.

Example 5.21. Continuing with Example 5.20, we attempt to compute $5P$ for the point $P = (38, 112)$ on the elliptic curve

$$E : Y^2 = X^3 + 3X + 7 \qquad \text{modulo } 187.$$

We already computed $2P = (43, 126)$ and $3P = (54, 105)$. The first step in computing $5P = 3P + 2P$ is to compute the reciprocal of

$$x(3P) - x(2P) = 54 - 43 = 11 \quad \text{modulo } 187.$$

However, when we apply the Extended Euclidean Algorithm to 11 and 187, we find that $\gcd(11, 187) = 11$, so 11 does not have a reciprocal modulo 187. It seems that we have hit a dead end!

However, we have actually struck it rich, because notice that since the quantity $\gcd(11, 187)$ is greater than 1, it gives us a divisor of 187. So our failure to compute $5P$ also tells us that 11 divides 187, which allows us to factor 187 as $187 = 11 \cdot 17$. This idea underlies Lenstra's elliptic curve factorization algorithm.

Let us examine more closely why we were not able to compute $5P$ modulo 187. If we instead look at the elliptic curve E modulo 11, then a quick computation shows that the point

$$P = (38, 112) \equiv (5, 2) \equiv 11 \quad \text{satisfies } 5P = \mathcal{O} \text{ in } E(\mathbb{F}_{11}).$$

This means that when we attempt to compute $5P$ modulo 11, we end up with the extra point $\mathcal{O}$, so at some stage of the calculation we have tried to divide by zero. But here "zero" means in $\mathbb{F}_{11}$, so we actually end up trying to find the reciprocal modulo 11 of some integer that is divisible by 11.

Following the lead from Examples 5.20 and 5.21, we replace multiplication modulo N in Pollard's factorization method with addition modulo N on an elliptic curve. We start with an elliptic curve E and a point P on E modulo N and we compute

$$2! \cdot P,\ 3! \cdot P,\ 4! \cdot P, 5! \cdot P, \ldots \quad (\text{mod } N).$$

Notice it that once we have computed $Q = (n-1)! \cdot P$, it is easy to compute $n! \cdot P$, since it equals nQ. At each stage, there are three things that may happen. First, we may be able to compute $n! \cdot P$. Second, during the computation we may need to find the reciprocal of a number d that is a multiple of N, which would not be helpful, but luckly this is quite unlikely. Third, we may need to find the reciprocal of a number d that satisfies $1 < \gcd(d, N) < N$, in which case the computation of $n! \cdot P$ fails, but $\gcd(d, N)$ is a nontrivial factor of N, so we are happy.

This completes the description of Lenstra's elliptic curve factorization algorithm, other than the minor problem of finding an initial point P on an

```
   Input. Integer $N$ to be factored.
1. Choose random values $A$, $a$ and $b$ modulo $N$.
2. Set $P = (a, b)$ and $B \equiv b^2 - a^3 - A \cdot a \pmod{N}$.
   Let $E$ be the elliptic curve $E : Y^2 = X^3 + AX + B$.
3. Loop $j = 2, 3, 4, \ldots$ up to a specified bound.
    4. Compute $Q \equiv jP \pmod{N}$ and set $P = Q$.
    5. If computation in (4) fails, then have found a $d > 1$ with $d|N$.
    6. If $d < N$ then success, return $d$.
    7. If $d = N$, go to Step 1 and choose a new curve and point.
8. Increment $j$ and loop again at Step 2.
```

Table 5.6: Lenstra's elliptic curve factorization algorithm

elliptic curve E modulo N. The obvious method is to fix an equation for the curve E, plug in values of X, and check if the quantity $X^3 + AX + B$ is a square modulo N. Unfortunately, this is difficult to do unless we know how to factor N. The solution to this dilemma is to first choose the point $P = (a, b)$ at random, second choose a random value for A, and third set

$$B \equiv b^2 - a^3 - A \cdot a \pmod{N}.$$

Then the point P is automatically on the curve $E : Y^2 = X^3 + AX + B$ modulo N. Lenstra's algorithm is summarized in Table 5.6.

Example 5.22. We illustrate Lenstra's algorithm by factoring $N = 6887$. We begin by randomly selecting a point $P = (1512, 3166)$ and a number $A = 14$ and computing

$$B \equiv 3166^2 - 1512^3 - 14 \cdot 1512 \equiv 19 \pmod{6887}.$$

We let E be the elliptic curve

$$E : Y^2 = X^3 + 14X + 19,$$

and then by construction the point P is automatically on E modulo 6887. Next we start computing multiples of P modulo 6887. First we find that $2P \equiv (3466, 2996) \pmod{6887}$. Next we compute

$$3! \cdot P = 3 \cdot (2P) = 3 \cdot (3466, 2996) \equiv (3067, 396) \pmod{6887}.$$

n	$n! \cdot P \bmod 6887$
1	$P = (1512, 3166)$
2	$2! \cdot P = (3466, 2996)$
3	$3! \cdot P = (3067, 396)$
4	$4! \cdot P = (6507, 2654)$
5	$5! \cdot P = (2783, 6278)$
6	$6! \cdot P = (6141, 5581)$

Table 5.7: Multiples of $P = (1512, 3166)$ on $Y^2 = X^3 + 14X + 19 \mod 6887$

And so on. The values up to $6! \cdot P$ are listed in Table 5.7. These values are not, in and of themselves, interesting. It is only when we try, and fail, to compute $7! \cdot P$ that something interesting happens.

From Table 5.7 we read off the value of $Q = 6! \cdot P = (6141, 5581)$, and we want to compute $7Q$. First we compute

$$\begin{aligned} 2Q &\equiv (5380, 174) && \pmod{6887} \quad \text{and} \\ 4Q &\equiv 2 \cdot 2Q \equiv (203, 2038) && \pmod{6887}. \end{aligned}$$

Then we compute $7Q$ as

$$\begin{aligned} Q &\equiv (Q + 2Q) + 4Q && \pmod{6887} \\ &\equiv \big((6141, 5581) + (5380, 174)\big) + (203, 2038) && \pmod{6887} \\ &\equiv (984, 589) + (203, 2038) && \pmod{6887}. \end{aligned}$$

When we attempt to perform the final step, we need to compute the reciprocal of $203 - 984$ modulo 6887, but we find that

$$\gcd(203 - 984, 6887) = \gcd(-781, 6887) = 71.$$

Thus we have discovered a nontrivial divisor of 6887, namely 71, which gives the factorization $6887 = 71 \cdot 97$.

It turns out that in $E(\mathbb{F}_{71})$ the point P satisfies $63P \equiv \mathcal{O} \pmod{71}$, while in $E(\mathbb{F}_{97})$ the point P satisfies $107P \equiv \mathcal{O} \pmod{97}$. The reason that we succeeded in factoring 6887 using $7! \cdot P$, but not with a smaller multiple of P, is precisely because $7!$ is the smallest factorial that is divisible by 63.

Remark 5.23. In Section 4.7 we discussed the speed of sieve factorization methods and saw that the average running time of the quadratic sieve to factor a composite number N is approximately

$$O\left((e^{\sqrt{(\log N)(\log\log N)}}\right) \text{ steps.} \tag{5.4}$$

Notice that the running time depends on the size of the integer N.

On the other hand, the most naive possible factorization method, namely trying each possible divisor $2, 3, 4, 5, \ldots$, has a running time that depends on the smallest prime factor of N. More precisely, this trial division algorithm takes exactly p steps, where p is the smallest prime factor of N. If it happens that $N = pq$ with p and q approximately the same size, then the running time is approximately $\sqrt{N}$, which is much slower than sieve methods; but if N happens to have a very small prime factor, trial division may be useful.

It is an interesting and useful property of the Elliptic Curve Factorization Algorithm that its expected running time depends on the smallest prime factor of N, rather than on N itself. More precisely, if p is the smallest factor of N, then the Elliptic Curve Factorization Algorithm have average running time approximately

$$O\left((e^{\sqrt{2(\log p)(\log\log p)}}\right) \text{ steps.} \tag{5.5}$$

If $N = pq$ is a product of two primes with $p \approx q$, the running times in (5.4) and (5.5) are approximately equal, and then the fact that a sieve step is much faster than an elliptic curve step makes sieve methods faster in practice. However, the elliptic curve method is quite useful for finding moderately large factors of extremely large numbers, due to its running time depending on the smallest prime factor.

5.7 Finite fields with p^k elements [M]

5.8 Elliptic fields over $\mathbb{F}_2$ and over $\mathbb{F}_2^k$

Remark 5.24. The SEA algorithm and its variants [47, 48] are designed to efficiently count the number of points in $E(\mathbb{F}_q)$ for any fields with a large number of elements. Satoh [46] devised an alternative method that is often faster than SEA when $q = p^e$ for a small prime p and (moderately) large exponent e. Satoh's original paper dealt only with the case $p \geq 3$, but subsequent work [19, 58] covers also the cryptographically important case of $p = 2$.

5.9 Survey of other methods to solve ECDLP [M]

Exercises

Section 5.1. Elliptic curves [M]

5.1. Let E be the elliptic curve $E : Y^2 = X^3 - 2X + 4$ and let $P = (0, 2)$ and $Q = (3, -5)$. (You should check that P and Q are on the curve E.)
(a) Compute $P \oplus Q$.
(b) Compute $P \oplus P$ and $Q \oplus Q$.
(c) Compute $P \oplus P \oplus P$ and $Q \oplus Q \oplus Q$.

5.2. Check that the points $P = (-1, 4)$ and $Q = (2, 5)$ are points on the elliptic curve $E : Y^2 = X^3 + 17$.
(a) Compute the points $P + Q$ and $P - Q$.
(b) Compute the points $2P$ and $2Q$.

5.3. Suppose that the cubic polynomial $X^3 + AX + B$ factors as

$$X^3 + AX + B = (X - e_1)(X - e_2)(X - e_2).$$

Prove that $4A^3 + 27B^2 = 0$ if and only if two (or more) of e_1, e_2, and e_3 are the same. (*Hint.* Multiply out the righthand side and compare coefficients to relate A and B to e_1, e_2, and e_3.)

5.4. Sketch each of the following curves, as was done in Figure 5.1 on page 252.

(a) $E : Y^2 = X^3 - 7X + 3$
(b) $E : Y^2 = X^3 - 7X + 9$
(c) $E : Y^2 = X^3 - 7X - 12$
(d) $E : Y^2 = X^3 - 3X + 2$.
(e) $E : Y^2 = X^3$.

Notice that the curves in (d) and (e) have $\Delta_E = 0$, so they are not elliptic curves. How do their pictures differ from the pictures in (a), (b) and (c)? Each of the curves (d) and (e) has one point that is somewhat unusual. These unusual points are called *singular points*.

Section 5.2. Elliptic curves over finite fields [**M**]

5.5. For each of the following elliptic curves E and finite fields $\mathbb{F}_p$, make a list of the set of points $E(\mathbb{F}_p)$.
(a) $E : Y^2 = X^3 + 3X + 2$ over $\mathbb{F}_7$.
(b) $E : Y^2 = X^3 + 2X + 7$ over $\mathbb{F}_{11}$.
(c) $E : Y^2 = X^3 + 4X + 5$ over $\mathbb{F}_{11}$.
(d) $E : Y^2 = X^3 + 9X + 5$ over $\mathbb{F}_{11}$.
(e) $E : Y^2 = X^3 + 9X + 5$ over $\mathbb{F}_{13}$.

5.6. Make an addition table for E over $\mathbb{F}_p$, as we did in Table 5.1.
(a) $E : Y^2 = X^3 + \ X + 2$ over $\mathbb{F}_5$
(b) $E : Y^2 = X^3 + 2X + 3$ over $\mathbb{F}_7$.
(c) $E : Y^2 = X^3 + 2X + 5$ over $\mathbb{F}_{11}$.

You may want to write a computer program for (c), since $E(\mathbb{F}_{11})$ has a lot of points!

5.7. Let E be the elliptic curve

$$E : y^2 = x^3 + x + 1.$$

Compute the number of points in the group $E(\mathbb{F}_p)$ for each of the following primes:
(a) $p = 3$. (b) $p = 5$. (c) $p = 7$. (d) $p = 11$.

In each case, also compute the trace of Frobenius

$$t_p = p + 1 - \#E(\mathbb{F}_p)$$

and verify that $|t_p|$ is smaller than $2\sqrt{p}$.

Section 5.3. The elliptic curve discrete logarithm problem [**M**]

5.8. Let E be the elliptic curve

$$E : y^2 = x^3 + x + 1$$

and let $P = (4, 2)$ and $Q = (0, 1)$ be points on E modulo 5. Solve the elliptic curve discrete logarithm problem for P and Q, that is, find a positive integer n so that $Q = nP$.

5.9. Let E be an elliptic curve over $\mathbb{F}_p$ and let P and Q be points in $E(\mathbb{F}_p)$. Assume that Q is a multiple of P and let $n_0 > 0$ be the smallest solution to $Q = nP$. Also let $s > 0$ be the smallest solution to $sP = \mathcal{O}$. Prove that every solution to $Q = nP$ looks like $n_0 + is$ for some $i \in \mathbb{Z}$. (*Hint.* Write n as $n = is + r$ for some $0 \le r < s$ and determine the value of r.)

5.10. Use the Double-and-Add Algorithm (Table 5.3) to compute nP in $E(\mathbb{F}_p)$ for each of the following curves and points, as we did in Figure 5.5.

(a)	$E : Y^2 = X^3 + 23X + 13$	$p = 83$	$P = (24, 14)$	$n = 19$
(b)	$E : Y^2 = X^3 + 143X + 367$	$p = 613$	$P = (195, 9)$	$n = 23$
(c)	$E : Y^2 = X^3 + 1828X + 1675$	$p = 1999$	$P = (1756, 348)$	$n = 11$
(d)	$E : Y^2 = X^3 + 1541X + 1335$	$p = 3221$	$P = (2898, 439)$	$n = 3211$

5.11. In Section 3.5 we gave an abstract description of Pollard's ρ method, and in Section 3.5.2 we gave an explicit version to solve the discrete logarithm problem in $\mathbb{F}_p$. Adapt this material to create a Pollard ρ algorithm to solve the ECDLP.

Section 5.4. Elliptic curve cryptography

5.12. Alice and Bob agree to use elliptic Diffie-Hellman key exchange with the prime, elliptic curve, and point

$$p = 2671, \qquad E : Y^2 = X^3 + 171X + 853, \qquad P = (1980, 431) \in E(\mathbb{F}_{2671}).$$

(a) Alice sends Bob the point $Q_A = (2110, 543)$. Bob decides to use the secret multiplier $n_B = 1943$. What point should Bob send to Alice?

(b) What is their secret shared value?

(c) How difficult is it for Eve to figure out Alice's secret multiplier n_A? If you know how to program, use a computer to find n_A.

(d) Alice and Bob decide to exchange a new piece of secret information using the same prime, curve, and point. This time Alice sends Bob only the x-coordinate $x_A = 2$ of her point Q_A. Bob decides to use the secret multiplier $n_B = 875$. What single number modulo p should Bob send to Alice, and what is their secret shared value?

5.13. Exercise 2.8 on page 83 describes a secret sharing scheme based on the discrete logarithm problem for $\mathbb{F}_p$. Describe a version of this secret sharing scheme that uses the elliptic curve discrete logarithm problem. (You may assume that Alice and Bob know the order of the point P in the group $E(\mathbb{F}_p)$, i.e. they know the smallest integer $n \geq 1$ with the property that $nP = \mathcal{O}$.)

5.14. A potential drawback to using an elliptic curve $E(\mathbb{F}_p)$ for cryptography is the fact that it takes two coordinates to specify a point in $E(\mathbb{F}_p)$, but the second coordinate actually conveys very little additional information.

(a) Suppose that Bob wants to send Alice the value of a point $R \in E(\mathbb{F}_p)$. Explain why it suffices for Bob to send Alice the x-coordinate of $R = (x_R, y_R)$ together with the single bit

$$\beta_R = \begin{cases} 0 & \text{if } 0 \leq y_R < \frac{1}{2}p, \\ 1 & \text{if } \frac{1}{2}p < y_R < p. \end{cases}$$

(You may assume that Alice is able to efficiently compute square roots modulo p.)

(b) Alice and Bob decide to use the prime $p = 1123$ and the elliptic curve $E : Y^2 = X^3 + 54X + 87$. Bob sends to Alice the x-coordinate $x = 278$ and the bit $\beta = 0$. What point is Bob trying to convey to Alice? What about if instead Bob had sent $\beta = 1$?

5.15. The Menezes-Vanstone variant of the elliptic ElGamal public key cryptosystem improves message expansion while avoiding the difficulty of directly attaching plaintexts to points in $E(\mathbb{F}_p)$. The MV-ElGamal cryptosystem is described in Table 5.8 on page 286.

(a) The last line of Table 5.8 claims that $m_1' = m_1$ and $m_2' = m_2$. Prove that this is true, so the decryption process does work.

(b) What is the message expansion of MV-ElGamal?

(c) Alice and Bob agree to use

$$p = 1201, \qquad E : Y^2 = X^3 + 19X + 17, \qquad P = (278, 285) \in E(\mathbb{F}_p).$$

for MV-ElGamal. Alice's secret value is $n_A = 595$. What is her public key? Bob sends Alice the encrypted message $((1147, 640), 279, 1189)$. What is the plaintext?

5.16. This exercise continues the discussion of the MV-ElGamal cryptosystem described in Table 5.8 on page 286.

(a) Eve knows the elliptic curve E and the ciphertext values c_1 and c_2. Show how Eve can use this knowledge to write down a polynomial equation (modulo p) that relates the two pieces m_1 and m_2 of the plaintext. In particular, if Eve can figure out one piece of the plaintext, then she can recover the other piece by finding the roots of a certain polynomial modulo p.

(b) Alice and Bob exchange a message using MV-ElGamal with the prime, elliptic curve, and point in Exercise 5.15(c). Eve intercepts the ciphertext $((269, 339), 814, 1050)$ and through other sources, she discovers that the first part of the plaintext is $m_1 = 1050$. Use your algorithm in (a) to recover the second part of the plaintext.

Section 5.5. The evolution of public key cryptography [**H**]

Section 5.6. The elliptic curve factorization algorithm

5.17. Use the Elliptic Curve Factorization Algorithm to factor each of the numbers N using the given elliptic curve E and point P.

(a)	$N = 589,$	$E : Y^2 = X^3 + 4X + 9,$	$P = (2, 5).$
(b)	$N = 26167,$	$E : Y^2 = X^3 + 4X + 128,$	$P = (2, 12).$
(c)	$N = 1386493,$	$E : Y^2 = X^3 + 3X - 3,$	$P = (1, 1).$
(d)	$N = 28102844557,$	$E : Y^2 = X^3 + 18X - 453,$	$P = (7, 4).$

Section 5.7. Finite fields with p^k elements [**M**]

Section 5.8. Elliptic fields over $\mathbb{F}_2$ and over $\mathbb{F}_2^k$

Section 5.9. Survey of other methods for ECDLP [**M**]

<table>
<tr><th colspan="2">Public Parameter Creation</th></tr>
<tr><td colspan="2">A trusted party chooses and publishes a (large) prime p,
an elliptic curve E over $\mathbb{F}_p$, and a point P in $E(\mathbb{F}_p)$.</td></tr>
<tr><th>Alice</th><th>Bob</th></tr>
<tr><th colspan="2">Key Creation</th></tr>
<tr><td>Choose a secret multiplier n_A.
Compute $Q_A = n_A P$.
Publish the public key Q_A.</td><td></td></tr>
<tr><th colspan="2">Encryption</th></tr>
<tr><td></td><td>Choose plaintext values m_1 and m_2 modulo p.
Choose a random number k.
Compute $R = kP$.
Compute $S = kQ_A$ and write it as $S = (x_S, y_S)$.
Set $c_1 \equiv x_S m_1 \pmod{p}$ and $c_2 \equiv y_S m_2 \pmod{p}$.
Send ciphtertext (R, c_1, c_2) to Alice.</td></tr>
<tr><th colspan="2">Decryption</th></tr>
<tr><td>Compute $T = n_A R$ and write it as $T = (x_T, y_T)$.
Set $m_1' \equiv x_T^{-1} c_1 \pmod{p}$ and $m_2' \equiv y_T^{-1} c_2 \pmod{p}$.
Then $m_1' = m_1$ and $m_2' = m_2$.</td><td></td></tr>
</table>

Table 5.8: Menezes-Vanstone variant of ElGamal (Exercises 5.15, 5.16)

Chapter 6

Lattices and Cryptology

Until now the only examples we have seen of public key cryptosystems have been based (at least indirectly) on the hard problems of factorization and finding discrete logarithms. Another interesting and potentially useful class of cryptosystems can be based on problems arising from the study of lattices. In this chapter we will begin the study of lattices and the hard problems that arise in this context.

6.1 Subset sum problems and knapsacks [M]

Although the general problems of factoring integers or finding discrete logarithms are acknowledged to be hard, they do not, as far as anyone knows, belong to the class of NP-complete problems. This is a class of problems that have been studied for quite a long time. No one has yet found an algorithm that will solve any NP-complete problem in polynomial time, and if an algorithm existed for one, the entire class would be solvable in polynomial time.

The notion of having this sort of degree of certainty that your underlying hard problem is really hard is naturally very attractive when building a cryptosystem. Consequently, since Diffie and Hellman first introduced the concept of public key cryptography there has been a great deal of interest in the problem of building an efficient public key cryptosystem based on an NP-complete (or NP-hard) problem. (A problem is called NP-hard if its solution would imply the solution of all NP-complete problems, but a solution to an NP-complete problem would not necessarily give a solution to the NP-hard problem.)

The first such attempts were made by Merkle and Hellman in the late 70's [?], using the subset sum problem. This NP-complete problem is stated as follows:

Suppose one is given a list of positive integers $\{m_1, m_2, \ldots, m_n\}$. An unknown subset of the list is selected and summed to give an integer S. Given S, recover the subset that summed to S, or find another subset with the same property.

Here's another way of writing this problem. A list of positive integers $\mathbf{m} = \{m_1, m_2, \ldots, m_n\}$ is public knowledge. Bernard chooses a secret binary vector $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$, where each x_i can take on the value 1 or 0. Bernard computes $S = \sum_{i=1}^{n} x_i m_i$ and passes S to Arthur. Can Arthur recover the original vector $\mathbf{x}$ in an efficient way?

Presumably a general person with no extra knowledge would have a hard time extracting $\mathbf{x}$, as this would appear to be a random example of a subset sum problem. Suppose, though, that Arthur possessed some secret knowledge or trapdoor information about $\mathbf{m}$ that enabled him to solve the problem easily. Then this scenario could be used as a public key cryptosystem, with $\mathbf{x}$ as the secret plain text, $\mathbf{m}$ as the public key and S as the encrypted message.

The question is: what sort of sneaky trick could Arthur use to ensure that he could solve the subset sum problem, but nobody else could? There is a general form of the subset sum problem that is extremely easy to solve. Suppose that one takes a sequence of positive integers $\mathbf{r} = \{r_1, r_2, \ldots, r_n\}$ with the property that each $r_{i+1} \geq 2r_i$ for each $1 \leq i \leq n-1$. Such a sequence is called *super increasing*. If one is given an integer S, with $S = \mathbf{x} \cdot \mathbf{r}$ for an unknown binary vector $\mathbf{x}$ it is a very easy matter to recover $\mathbf{x}$ from S.

For example, suppose $\mathbf{r} = \{3, 11, 24, 50, 115\}$ and $S = 142$. Then 115 must be part of the subset, as otherwise the entries couldn't sum that high. Subtracting 115 from 142 leaves 27. The next entry, 50, can not belong to the subset as it is too large, and the following entry, 24, must be part of it, as otherwise the sum could not reach 27. Subtracting 24 from 27 yields 3. Thus 3 is the only remaining possibility, as 11 is too large. The corresponding binary vector is $\mathbf{x} = \{1, 0, 1, 0, 1\}$.

The basic idea that Merkle and Hellman proposed was this: Arthur begins with a secret super increasing sequence $\mathbf{r}$. He then chooses two large secret integers A, B, with $B > 2r_n$ and $(A, B) = 1$. Arthur multiplies the entries of $\mathbf{r}$ by A and reduces modulo B, obtaining a new sequence $\mathbf{m}$, with each $m_i \equiv Ar_i$

(mod B). This new sequence $\mathbf{m}$ is his public list. Encryption then works as follows. Bernard chooses a secret binary vector $\mathbf{x}$ and computes the encrypted message $S = \mathbf{x} \cdot \mathbf{m}$, which sends by an insecure channel to Arthur. To decrypt S, Arthur multiplies by $A^{-1} \pmod{B}$, obtaining $S' \equiv \mathbf{x} \cdot \mathbf{r} \pmod{B}$. Choosing S' in the range $0 \leq S' \leq B - 1$ Arthur obtains an exact inequality $S' = \mathbf{x} \cdot \mathbf{r} \pmod{B}$ as any subset of the integers r_i must sum to an integer smaller than B. As the $\mathbf{r}$ is super increasing, he now easily recovers $\mathbf{x}$.

A cryptosystem of this type became known as a knapsack system. The general idea was: start with s a secret super increasing sequence, disguise it by some collection of secret linear operations, then reveal the transformed list as the public key. The original Merkle and Hellman system suggested applying a secret permutation to the entries of $A\mathbf{r} \pmod{B}$ as an additional layer of security. Later versions were proposed by a number of people, involving multiple multiplications and reductions with respect to various moduli. For an excellent survey see the article by Odlyzko [].

The first question one must ask about this system is: what minimal properties must $\mathbf{r}, A, B$ have to obtain a given level of security. Some very easy attacks are possible if r_1 is too small, and so one generally takes $2^n < r_1$. Because of the super increasing nature of the sequence one then has

$$r_n \approx S \approx 2^{2n}. \tag{6.1}$$

The space of all binary vectors $\mathbf{x}$ has size 2^n, and thus an exhaustive search for a solution would require effort on the order of 2^n. In fact, a meet in the middle attack is possible which roughly square roots the time in exchange for a heavy use of memory. Thus the security of a knapsack system with a list of length n is $O(2^{n/2})$ from a meet in the middle attack. To obtain security on the order of 2^{80} one then requires at a minimum that $n > 160$.

We should also consider the corresponding key lengths and message lengths. A message consists of n bits of information. The public key is a list of n integers, each approximately $2n$ bits long. Thus the length of the public key will be about $2n^2$ bits. Because of this $n = 160$ corresponds to a public key size of about 51200 bits. This does not contrast well with RSA or Diffie-Hellman, where for security on the order of 2^{80} the public key size is about 1000 bits.

However the temptation to use a knapsack system rather than RSA or Diffie-Hellman was very great. There was a mild disadvantage in the size of the public key, but in the original example only one modular multiplication was required to decrypt a message. This was far more efficient than the computationally intensive modular exponentiations in RSA and Diffie-Hellman.

Unfortunately, although a meet in the middle attack is still the best known attack on the general subset sum problem, there proved to be other, far more effective, attacks on knapsacks with trapdoors. At first some very specific attacks were announced by Shamir, Odlyzko, Lagarias and others. Eventually, however, after the publication of the famous LLL paper [] in 1985 it became clear that there was a fundamental weakness inherent in any knapsack based system. Roughly speaking, these attacks showed that whenever n is less than around 300 a secret message $\mathbf{x}$ can be recovered from an encrypted message S in a disconcertingly short time by means of a process called lattice reduction. Thus in order to have even a hope of being secure, a knapsack would have to have $n > 300$, and a corresponding public key length that was greater than 180000 bits. This was sufficiently impractical that knapsacks were abandoned for some years.

The basic idea of the lattice reduction attack is as follows. First, one transforms the knapsack problem into one involving vectors. Given a public list $\mathbf{m}$ and encrypted message S, construct the matrix

$$\begin{pmatrix} 1 & 0 & 0 & \dots & 0 & m_1 \\ 0 & 1 & 0 & \dots & 0 & m_2 \\ & & & \vdots & & \\ 0 & 0 & 0 & \dots 1 \dots & 0 & m_i \\ & & & \vdots & & \\ 0 & 0 & 0 & \dots & 1 & m_n \\ 0 & 0 & 0 & \dots & 0 & S \end{pmatrix}. \tag{6.2}$$

The relevant vectors are the rows of the $n+1$ by $n+1$ matrix (6.2). Thus we consider the vectors $\mathbf{v}_1 = (1, 0, 0, \dots, 0, m_1)$, $\mathbf{v}_2 = (0, 1, 0, \dots, 0, m_2)$, $\dots$, $\mathbf{v}_n = (0, 0, 0, \dots, 1, m_n)$ and $\mathbf{v}_{n+1} = (0, 0, 0, \dots, 0, S)$.

The collection of all linear combinations of the $\mathbf{v}_i$ with integer coefficients is called a lattice L (to be defined formally in the next section). The determinant of this lattice is the same as the determinant of the matrix and equals S. The statement that the sum of some subset of the m_i equals S translates into the statement that there exists a vector $\mathbf{t} \in L$, with

$$\mathbf{t} = \sum_{i=1}^{n} x_i \mathbf{v}_i - \mathbf{v}_{n+1} = (x_1, x_2, \dots, x_n, 0),$$

where each x_i is chosen from the set $\{0, 1\}$.

As the x_i are binary, $\|\mathbf{t}\| \leq \sqrt{n}$. In fact, as roughly half of the x_i will be 0, it is more likely that $\|\mathbf{t}\| \approx \sqrt{n/2}$. On the other hand, the size of each $\|\mathbf{v}_i\|$ is roughly 2^{2n}. The key observation is that it seems rather improbable that a linear combination of vectors that are quite so large should have a norm that is quite so small. Thus the solution vector $\mathbf{t}$ can be viewed as an unexpectedly small vector in the lattice L. With high probability $\mathbf{t}$ will be unique, and if techniques exist for efficiently finding the shortest (non-zero) vector in a lattice L then $\mathbf{t}$ can potentially be located by this approach. The process of finding short vectors in a lattice is known as lattice reduction, and the LLL paper of [] was the first major advance in this field.

Later in this chapter we will precisely quantify what we mean by "shorter than expected". In this case we will show that in some sense the expected smallest vector in L will have size roughly (for large n)

$$S^{1/(n+1)}\sqrt{\frac{n+1}{2\pi e}} \approx 4\sqrt{\frac{n}{2\pi e}} \approx 0.97\sqrt{n}.$$

As this is larger than $\sqrt{n/2}$, the target vector $\mathbf{t}$ will in fact, with very high probability, turn out to be the smallest vector in L.

6.2 Another motivation for the study of lattices [M]

In this section we'll present another possible public key cryptosystem. It will turn out to have an unexpected connection with lattices of dimension 2, and hence a fatal vulnerability. However it is of some interest. For one thing it illustrates a tendency of lattices to be useful in cryptanalysis even though the problem involved may appear to have nothing to do with lattices. And for another thing, it provides a simple, low-dimensional example of NTRU, (yet another lattice based cryptosystem) which will be described in the next chapter.

In the following, Arthur will set up a public-private key pair. He begins by fixing a large integer q. He then chooses smaller integers f, g satisfying $(f, q) = 1$ and

$$\sqrt{q/4} < f, g < \sqrt{q/2}.$$

Arthur computes $h \equiv f^{-1}g \pmod{q}$ and posts the pair h, q as his public key. Arthur does not reveal f, his private key. Although f and g are small, f^{-1}

(mod q) will, with high probability, not be small, and h will also be on the order of q and appear random modulo q.

Bernard would now like to send a secret message $m < \sqrt{q/4}$ to Arthur. To do this he chooses an integer r at random, $r < \sqrt{q/2}$, and computes $e \equiv rh + m \pmod{q}$. Bernard sends the encrypted message e to Arthur.

To decrypt e, Arthur first computes $fe \pmod{q}$ and chooses $a \equiv fe \pmod{q}$ with $0 \leq a < q$. Arthur then computes $f^{-1}a \pmod{g}$. But $m \equiv f^{-1}a \pmod{g}$, and as $m < g$, Arthur has recreated m.

For example, take $q = 122430513841, f = 231231, g = 195698$. Then $f^{-1} \equiv 49194372303 \pmod{q}$ and

$$h \equiv f^{-1}g \pmod{q} \equiv 39245579300 \pmod{q}.$$

Thus Arthur's public key is the pair $39245579300, 122430513841$. Bernard would like to encrypt the message $m = 123456$. He chooses $r = 101010$. His encrypted message is thus

$$e \equiv rh + m \pmod{q} \equiv 18357558717 \pmod{q}.$$

To decrypt e, Arthur first computes

$$a \equiv fe \pmod{q} \equiv 48314309316 \pmod{q}.$$

He then computes,

$$f^{-1}a \pmod{g} \equiv 193495a \pmod{g} \equiv 123456,$$

which miraculously equals m.

Why does this work? When Arthur computes $a \equiv fe \pmod{q}$ he is actually computing

$$a \equiv f(rh + m) \equiv frf^{-1}g + fm \equiv rg + fm \pmod{q}.$$

By the size restrictions on f, g, r, m, $0 \leq rg + fm < q$ and thus $a \equiv rg + fm \pmod{q}$ and $0 \leq a < q$ implies that $a = rg + fm$ as an integer, not simply modulo q. Reducing $rg + fm$ modulo g now gives $fm \pmod{g}$ and multiplying by the inverse of f modulo g recovers $m \pmod{g}$. But $m < \sqrt{q/4} < g$, so recovering m modulo g recovers m over the integers.

How would one attack this system? First of all, there is the question of recovering the private key from the public key. To search for the private key f an attacker would multiply h by a candidate $F < \sqrt{q/2}$ and compute $G \equiv Fh$ modulo q. If $G < \sqrt{q/2}$ then the chances are very good that $f = F$ and $g = G$. This is because as F varies, the corresponding G will be distributed randomly modulo q. The chance that a random G will be this small compared to q is $1/\sqrt{1/(2q)}$, which will be quite small if q is large. Interestingly, even if $(f, g) \neq (F, G)$, F will still function perfectly well as a private key.

A brute force attack on the public key would then be expected to require $O(\sqrt{q})$ operations. Similarly, brute force attacks can be launched on an encrypted message. The attacker would simply try a random $r < \sqrt{q/2}$ and check to see if $e - rh \pmod{q}$ were small. Thus it appears that to obtain security on the order of 2^{80} one should choose $q \approx 2^{160}$. However, unfortunately for Arthur, there is another, completely different attack on this system. In the example given above an attacker, given the public key h, q, could simply compute the continued fraction expansion of h/q. In the example given above, the continued fraction expansion has coefficients

$$\{0, 3, 8, 2, 1, 3, 3, 8, 1, 4, 2, 1, 1, 2, 8, 1, 5, 3, 1, 4, 2, 1, 3, 1, 3, 3\}.$$

Expanding the intermediate continued fraction

$$\{0, 3, 8, 2, 1, 3, 3, 8, 1, 4, 2, 1, 1\}$$

one obtains the fraction

$$\frac{74122}{231231}.$$

Thus f surprisingly appears in the denominator of one of the continued fraction approximations to h/q. “What in the world is going on here?”, one might well ask. Later in this chapter we will see the this connection between this method and lattices. We will also explain why it will always return the secret f, given the public h, q, in the short (i.e polynomial) time that it takes to compute a continued fraction.

There is another way of thinking about attacks on this system that is more clearly connected to lattices. Consider the 2 by 2 matrix formed with the public key:

$$\begin{pmatrix} 1 & h \\ 0 & q \end{pmatrix}.$$

The linear combinations of these rows generate a two dimensional lattice L, with determinant q. As $h \equiv f^{-1}g \pmod{q}$ it follows that $fh = g + kq$ for some integer k. The linear combination

$$f(1,h) - k(0,q) = (f,g)$$

will thus be contained in L. This target vector has norm

$$\|(f,g)\| < \sqrt{\frac{q}{2} + \frac{q}{2}} = \sqrt{q}.$$

On the other hand, we will see later in this chapter that as the determinant equals q, the expected length of the shortest vector in L is $\approx 0.564\sqrt{q}$. (See (6.3).) The target vector has length somewhat larger than this, but it is still small enough that (f,g) will still, with high probability, be the smallest vector in L. An efficient method of searching for small vectors in a two dimensional lattice, for example the method of Gauss, introduced later in this chapter, will certainly locate (f,g).

6.3 Integer lattices and the shortest vector problem [M]

Fix an integer $n \geq 1$ and let $\mathbf{v}_1.\mathbf{v}_2,\ldots,\mathbf{v}_n$ be n independent vectors in $\mathbb{R}^n$. Recall that the collection of all linear combinations of the $\mathbf{v}_i$ with coefficients in $\mathbb{R}$ is an n-dimensional subspace of $\mathbb{R}^n$, the span of $\mathbf{v}_1.\mathbf{v}_2,\ldots,\mathbf{v}_n$. If one instead restricts the coefficients to be integers the result is called an n-dimensional lattice sitting inside $\mathbb{R}^n$. The study of lattices has a long and colorful history and there have been many applications to number theory, computer algebra, discrete mathematics, combinatorics, and cryptography.

Fundamental notions associated to lattices are:

1. A lattice L of dimension n is the $\mathbb{Z}$-linear span of n independent vectors:

$$L = \{a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \cdots + a_n\mathbf{v}_n \;:\; a_1,\ldots,a_n \in \mathbb{Z}\}.$$

2. A **fundamental domain** for the lattice L is

$$\mathcal{F}(\mathbf{v}_1,\ldots,\mathbf{v}_n) = \{t_1\mathbf{v}_1 + t_2\mathbf{v}_2 + \cdots + t_n\mathbf{v}_n \;:\; 0 \leq t_i < 1\}.$$

3. The **determinant** of L is the determinant of a matrix formed with the n vectors $\mathbf{v}_1,\mathbf{v}_2,\ldots,\mathbf{v}_n$, with the property that

$$|\det(L)| = \operatorname{Vol}\bigl(\mathcal{F}(\mathbf{v}_1,\ldots,\mathbf{v}_n)\bigr).$$

This matrix is the n by n matrix formed by considering the n basis vectors as rows (or alternatively columns). For example, if $m = n = 3$ and $\mathbf{v}_1 = (1, 0, 1), \mathbf{v}_2 = (1, 2, 0), \mathbf{v}_3 = (1, -1, 2)$ then the lattice L with basis $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ has determinant $\det(L) = 1$ because the determinant of the matrix

$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 2 & 0 \\ 1 & -1 & 2 \end{pmatrix}$$

equals 1.

A lattice can have many bases but the volume of the fundamental domain is independent of the choice of basis. Changing a basis of a lattice is the same as multiplying the matrix of (old) basis vectors by a matrix P of determinant plus or minus 1, which corresponds to doing elementary row or column operations using integer coefficients. We sometimes refer to a matrix as a "lattice", meaning that the rows of the matrix are the basis vectors. Finally, usually we'll be working with lattices whose basis vectors also consist of integers, but this is not a required in the definition of an integer lattice.

6.3.1 Dimension 2 examples

To give a feeling of concreteness to this usually high dimensional subject, we will begin with a basic discussion of lattices in the case $n = 2$. Fix two vectors

$$\mathbf{a} = (a_1, a_2), \quad \mathbf{b} = (b_1, b_2)$$

with $a_1, a_2, b_1, b_2 \in \mathbb{Z}$. Then

$$L = \{m\mathbf{a} + n\mathbf{b} | m, n \in \mathbb{Z}\}$$

is the lattice generated by $\mathbf{a}, \mathbf{b}$. This can also be thought of as the collection of all linear combinations of the rows of

$$\begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix}$$

or as

$$\{(m, n) \begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix} | m, n \in \mathbb{Z}\}$$

If $\mathbf{a}, \mathbf{b}$ are independent then the dimension of L is $n = 2$ and the determinant of L is defined by

$$d = \det \begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix} = a_1 b_2 - a_2 b_1.$$

When $\mathbf{a}, \mathbf{b}$ are independent $\{\mathbf{a}, \mathbf{b}\}$ is known as a *basis* for L. Clearly L can have many bases, and in fact $\{\mathbf{a}', \mathbf{b}'\}$ will be a basis for L if and only if there exist $m_1, m_2, m_3, m_4 \in \mathbb{Z}$ with $m_1 m_4 - m_2 m_3 = \pm 1$ such that $\mathbf{a}' = m_1 \mathbf{a} + m_2 \mathbf{b}$ and $\mathbf{b}' = m_3 \mathbf{a} + m_4 \mathbf{b}$.

As it happens, some bases are better than other bases. The "length" or Euclidean norm of a vector is given by

$$\|\mathbf{a}\| = \sqrt{a_1^2 + a_2^2}$$

and we will see below that a basis chosen so that $\|\mathbf{a}\|, \|\mathbf{b}\|$ are as short as possible will have certain advantages.

We may now state (in this context) two of the fundamental problems in the study of lattices. Suppose we are given a lattice L, described by a basis $\{\mathbf{a}, \mathbf{b}\}$. Then:

1. Find the shortest non-zero vector in L, i.e find $0 \neq \mathbf{v} \in L$ such that $\|\mathbf{v}\|$ is minimized.

2. Given a vector $\mathbf{w}$ which is in $\mathbb{Z}^2$ but not L, find the vector $\mathbf{v} \in L$ closest to $\mathbf{w}$, i.e. find $\mathbf{v} \in L$ such that $\|\mathbf{v} - \mathbf{w}\|$ is minimized.

One might very reasonably ask why these are considered to be interesting questions. The usual mathematical justifications for interest are that the problems should be hard and the solutions, if they can be found, should have useful consequences. Of course useful is another dangerous word. But as the focus of this book is cryptography, we'll let useful mean: "has interesting applications to cryptogaraphy". We still can't handle the word "interesting", but never mind.

The problems are indeed hard (at least for moderately large n), as will be elaborated on later. For the moment we will take the special case $n = 2$ and use it to illustrate some of the basic ideas we will be exploring in this chapter.

Recall we are given a lattice L, described by a basis $\{\mathbf{a}, \mathbf{b}\}$ and having determinant d. First let's ask what the likely size is of the shortest (non-zero) vector of L. A well known theorem of Minkowski states that if a body in space is convex, symmetric about the origin and has volume greater than or equal to $2^n|\det L|$ then it must contain a non-zero point of L. In our case $n = 2$. If we take as our convex symmetric body a circle of radius r about the origin, Minkowski's theorem translates to the statement that this circle is large enough if $\pi r^2 \geq 2^2|\det L|$. Thus there must exist a point $0 \neq \mathbf{a} \in L$ such that

$$\|\mathbf{a}\| \leq 2\sqrt{\frac{|\det L|}{\pi}}.$$

It's interesting to ask how this compares with the answer one might expect from a probabilistic argument. If one takes a circle of increasing radius r and counts the number of lattice points inside, this will be given by the volume of the circle divided by $|\det L|$ plus an error term on the order of r (with a constant depending on L). Thus the main term will be $\pi r^2/|\det L|$. Supposing for the moment that the main term gives the correct answer we can ask for the radius r at which this main term equals 1. This occurs at

$$r = \sqrt{\frac{|\det L|}{\pi}} \approx 0.564\sqrt{|\det L|}, \tag{6.3}$$

which is off only by a factor of 2 from the maximum it can be before guaranteeing the existence of a lattice point.

We will use analogs of this argument in higher dimensions and refer to it as the "gaussian heuristic". While not something that can be used as a basis for proving theorems, it is nevertheless quite useful for understanding the behavior of "average" lattices in high dimensions.

Moving on to the second problem, suppose we are now presented with a vector $\mathbf{w} = (w_1, w_2)$ and are asked the question: Is $\mathbf{w} \in L$? This can be answered by solving the following equation:

$$(x_1, x_2)\begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix} = (w_1, w_2)$$

One obtains

$$(x_1, x_2) = (w_1, w_2)\begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix}^{-1} = (w_1, w_2)\begin{pmatrix} b_2 & -a_2 \\ -b_1 & a_1 \end{pmatrix} d^{-1}.$$

The x_1, x_2 are the coefficients of the unique rational linear combination sending $\{\mathbf{a}, \mathbf{b}\}$ to $x_1\mathbf{a} + x_2\mathbf{b} = \mathbf{w}$. The point $\mathbf{w}$ is contained in L if and only if $(x_1, x_2) \in \mathbb{Z}^2$.

Now let's change the question. Suppose **w** is not contained in L. Can we find the point **v** in L that is closest to, or at least reasonably close to, w? The obvious first thing to try is: take (x_1, x_2) obtained above and round each coordinate to the nearest integer, i.e set $y_1 = \lfloor x_1 \rceil, y_2 = \lfloor x_2 \rceil$. Thus $x_1 = y_1 + \epsilon_1, x_2 = y_2 + \epsilon_2$, where $|\epsilon_1|, |\epsilon_2| \leq 1/2$. Then let $\mathbf{v} = y_1\mathbf{a} + y_2\mathbf{b}$ be our prospective closest lattice point.

To see whether this has a right to work, set $w_1 = v_1 + e_1, w_2 = v_2 + e_2$, with $(v_1, v_2) \in L$ and $(e_1, e_2) \in \mathbb{Z}^2$, with $\|(e_1, e_2)\|$ smaller than the shortest vector in L. We are hoping that

$$(y_1, y_2)\begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix} = (v_1, v_2),$$

or in other words, that

$$(x_1, x_2)\begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix} - (y_1, y_2)\begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix} = (\epsilon_1, \epsilon_2)\begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix} = (e_1, e_2).$$

Solving for (ϵ_1, ϵ_2) one obtains

$$(\epsilon_1, \epsilon_2) = (\frac{e_1b_2 - e_2b_1}{d}, \frac{e_2a_1 - e_1a_2}{d}).$$

By the Cauchy-Schwarz inequality

$$\frac{|e_1b_2 - e_2b_1|}{|d|} = \frac{|(e_1, e_2) \cdot (b_2, -b_1)|}{|d|} \leq \frac{\|(e_1, e_2)\|\|(b_1, b_2)\|}{|d|}$$

and similarly for (a_1, a_2). Our construction will work if the (ϵ_1, ϵ_2) obtained this way does in fact satisfy $|\epsilon_1|, |\epsilon_2| \leq 1/2$. Thus we require that $\|(e_1, e_2)\|\|(b_1, b_2)\| < |d|/2$ and $\|(e_1, e_2)\|\|(a_1, a_2)\| < |d|/2$.

Clearly, if $\|(a_1, a_2)\|, \|(b_1, b_2)\| \approx |d|$, then there is not much hope of success unless $\|(e_1, e_2)\|$ is extremely small. We know from the above arguments that the best we can expect in general is that the smaller of the basis vectors, say $\|\mathbf{a}\|$, is $\approx \sqrt{|d|}$. How small can $\|\mathbf{b}\|$ be? We have an inequality stating that

$$|d| \leq \|\mathbf{a}\|\|\mathbf{b}\|,$$

or more generally that for any n, $|d|$ is less than or equal to the product of the lengths of the basis vectors. Thus if $\|\mathbf{a}\|$ is considerably below $\sqrt{|d|}$, $\|\mathbf{b}\|$ must be considerably above. Clearly the most balanced situation comes about when $\|\mathbf{a}\| \approx \|\mathbf{b}\| \approx \alpha\sqrt{|d|}$ for some $\alpha > 1$. We should then correspondingly choose $\|(e_1, e_2)\| \approx \beta\sqrt{|d|}$, where $\alpha\beta < 1/2$.

With this background we can illustrate a simple (and insecure) two dimensional version of the GGH cryptosystem. We will need a private small basis and a public large basis for a lattice L. Our security will rest on the hope (incorrect in dimension 2) that it is a hard problem to construct a short basis from a large basis.

The GGH system works as follows:

1. To create a public-private key pair: Construct a private short basis $\mathbf{a}, \mathbf{b}$ for a lattice L, and multiply by a matrix of determinant 1 with large coefficients to obtain a public basis $\mathbf{A}, \mathbf{B}$.

2. To encrypt: The plain text message is a pair of integers (m_1, m_2). Choose at random a pair of small integers (ϵ_1, ϵ_2) and construct the encrypted message (e_1, e_2) by computing

$$(e_1, e_2) = (m_1, m_2)\begin{pmatrix} A_1 & A_2 \\ B_1 & B_2 \end{pmatrix} + (\epsilon_1, \epsilon_2).$$

3. To decrypt: Compute

$$(x_1, x_2) = (e_1, e_2)\begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix}^{-1}$$

and round to the nearest integer, setting $(y_1, y_2) = \lfloor (x_1, x_2) \rceil$. Then let

$$(e_1', e_2') = (y_1, y_2)\begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix}$$

and set

$$(m_1', m_2') = (e_1', e_2')\begin{pmatrix} A_1 & A_2 \\ B_1 & B_2 \end{pmatrix}^{-1}.$$

If the basis $\mathbf{a}, \mathbf{b}$ and the perturbation vector (ϵ_1, ϵ_2) are short enough we should have recovered the message with $(m_1, m_2) = (m_1', m_2')$.

A small number example will help to illustrate the basic operations, and some potential pitfalls. For a private basis take four integers of about the same size at random: $\mathbf{a} = (1324, 2376)$, $\mathbf{b} = (4928, 3751)$. The determinant of L is then $d = -6742604$. In relation to the determinant, $\|\mathbf{a}\| \approx 1.05\sqrt{|d|}$ and $\|\mathbf{b}\| \approx 2.39\sqrt{|d|}$. Referring to (6.3) we see that these are 1.86 and 4.23 (respectively) times the length of the expected shortest vector in L. These are reasonably short, so we have reason to hope that decryption will behave as advertised.

Now choose a large matrix of determinant 1:

$$\begin{pmatrix} 10723 & 10631 \\ 1049 & 1040 \end{pmatrix}.$$

and compute

$$\begin{pmatrix} 10723 & 10631 \\ 1049 & 1040 \end{pmatrix}\begin{pmatrix} 1324 & 2376 \\ 4928 & 3751 \end{pmatrix} = \begin{pmatrix} 66586820 & 65354729 \\ 6513996 & 6393464 \end{pmatrix}.$$

This gives us our public basis: $\mathbf{A} = (66586820, 65354729)$, $\mathbf{B} = (6513996, 6393464)$. In relation to the determinant, $\|\mathbf{A}\| \approx 35931\sqrt{|d|}$ and $\|\mathbf{B}\| \approx 3515\sqrt{|d|}$. These are not at all short.

Our object is to encrypt and then successfully decrypt the message $(512, 379)$. To encrypt it we first choose a random small perturbation vector, say $(\epsilon_1, \epsilon_2) = (2, -3)$, and then compute

$$(512, 379)\begin{pmatrix} 66586820 & 65354729 \\ 6513996 & 6393464 \end{pmatrix} + (2, -3) = (36561256326, 35884744101).$$

Thus the encryption of $(m_1, m_2) = (512, 379)$ is the vector

$$(e_1, e_2) = (36561256326, 35884744101).$$

To decrypt we first multiply by the inverse of the private basis, obtaining

$$(36561256326, 35884744101)\begin{pmatrix} 1324 & 2376 \\ 4928 & 3751 \end{pmatrix}^{-1} \approx (5887746.997, 5837232.001).$$

Rounding this to the nearest integer gives $(5887747, 5837232)$ and multiplying on the right by the private basis gives us our conjectured closest lattice point:

$$(5887747, 5837232)\begin{pmatrix} 1324 & 2376 \\ 4928 & 3751 \end{pmatrix} = (36561256324, 35884744104).$$

This is indeed close to the original encrypted message, and multiplying by the inverse of the public matrix we obtain an integral solution:

$$\begin{aligned}(m_1', m_2') &= (36561256324, 35884744104)\begin{pmatrix} 66586820 & 65354729 \\ 6513996 & 6393464 \end{pmatrix}^{-1} \\ &= (512, 379)\end{aligned}$$

which is indeed our recovered message.

What if we tried this decryption process using the public basis? We would obtain

$$\begin{gathered}(36561256326, 35884744101)\begin{pmatrix} 66586820 & 65354729 \\ 6513996 & 6393464 \end{pmatrix}^{-1} \\ \approx (507.205277367616, 428.012209229549).\end{gathered}$$

Rounding this to the nearest integer gives $(507, 428)$ and multiplying on the right by the public basis gives us our conjectured closest lattice point:

$$(507, 428)\begin{pmatrix} 66586820 & 65354729 \\ 6513996 & 6393464 \end{pmatrix} = (36547508028, 35871250195).$$

This is not very close to the original point at all, and multiplying by the public basis yields (of course) $(m_1', m_2') = (507, 428) \neq (512, 379),$, the original message.

One can then ask: how big can we allow (ϵ_1, ϵ_2) to be while still retaining our ability to decrypt using the private basis? Some experimentation shows that if $|\epsilon_1|, |\epsilon_2| < 380$ we can decrypt successfully most of the time, while if $|\epsilon_1|, |\epsilon_2|$ grow a bit beyond this we will fail most of the time. Note that the maximum perturbation is $\|(380, 380)\| \approx 0.21\sqrt{|d|}$, a smallish multiple of $\sqrt{|d|}$, as expected.

Interestingly, even our private basis is not optimal, in the sense that there exist even shorter bases. For example, the simple linear combination $\mathbf{a}, \mathbf{b} - \mathbf{a}$ will have better decryption properties. In fact a still better basis exists: $(-1324, -2376), (2280, -1001)$, short enough to allow decryption for $|\epsilon_1|, |\epsilon_2| < 900$.

The system just described is not secure because even if the numbers involved were chosen to be large enough to make a brute force search for the message or a short basis impractical, a method exists, dating back to Gauss, for efficiently locating short bases for lattices of dimension 2. This, and a powerful generalization known as LLL will be discussed in **??**.

6.3.2 General lattices

We will now discuss the general statement of the fundamental lattice problems mentioned in the previous section. Suppose we are given a lattice L of dimension n. We assume L sits inside $\mathbb{R}^m$ for some $m \geq n$.

1. **Shortest Vector Problem (SVP)**: Find the shortest non-zero vector in L, i.e find $0 \neq \mathbf{v} \in L$ such that $\|\mathbf{v}\|$ is minimized.

2. **Closest Vector Problem (CVP)**:Given a vector $\mathbf{w}$ which is not in L, find the vector $\mathbf{v} \in L$ closest to $\mathbf{w}$, i.e. find $\mathbf{v} \in L$ such that $\|\mathbf{v} - \mathbf{w}\|$ is minimized.

Both of these problems appear to be profound and very difficult as the dimension n becomes large. In full generality, CVP is known to be NP-hard and SVP is NP-hard under a certain “randomized reduction” hypothesis. In practice a CVP can often be reduced to a SVP and is thought of as being “a little bit harder” than SVP. As a cautionary note, full generality refers to something which rarely if ever occurs in practice. In a real world scenario it is likely that a particular class or family of lattices will be used, and there is always the possibility that something special about that class will make SVP and SVP easier to solve.

Secondary problems, that are also highly significant, arise from SVP and CVP. For example, one could look for a basis $\mathbf{v}_1, \ldots, \mathbf{v}_n$ of L consisting of all “short” vectors (e.g., minimize $\max \|\mathbf{v}_i\|$). This is known as the Short Basis Problem or SBP. Alternatively, one might search for a nonzero vector $\mathbf{v} \in L$ satisfying

$$\|\mathbf{v}\| \leq \phi(n)\|\mathbf{v}_{\text{shortest}}\|$$

for some slowly growing function of $n = \dim(L)$. For example, for a fixed constant κ one could try to find $\mathbf{v} \in L$ satisfying

$$\|\mathbf{v}\| \leq \kappa\sqrt{n}\|\mathbf{v}_{\text{shortest}}\|$$

and similarly for CVP. These generalizations are known as approximate shortest and closest vector problems, or ASVP,ACVP.

How big, in fact, is the shortest vector in terms of the determinant and the dimension of L? A theorem of Hermite from the 19^{th} century says that there is a constant γ_n so that in every lattice L of dimension n, the shortest vector satisfies

$$\|\mathbf{v}_{\text{shortest}}\| \leq \gamma_n |\det(L)|^{1/n},$$

where γ_n is a constant that depends only on the dimension n. Hermite gave the value $\gamma_n = (4/3)^{(n-1)/4}$. The smallest possible value one take for γ_n is called *Hermite's constant.* Its exact value is known only for $1 \le n \le 8$. For large n it satisfies

$$\sqrt{n/2\pi e} \le\approx \gamma_n \le\approx \sqrt{n/\pi e}.$$

A tool for understanding the approximate size of the shortest non-zero vector of a lattice L is given by Minkowski's Theorem, mentioned earlier.

Theorem 6.1. *Let L be a lattice of dimension n. If $S \subset \mathbb{R}^n$ is a symmetric convex set of volume* $\mathrm{Vol}(S) > 2^n|\det(L)|$*, then S contains a nonzero lattice vector.*

To see how this can be used, let S be a sphere of radius r centered at 0. Then S is symmetric and convex and has volume given by

$$\mathrm{Vol}(S) = \frac{\pi^{n/2} r^n}{\Gamma(1 + n/2)}. \tag{6.4}$$

Here $\Gamma(s)$ is the classical gamma function, defined by

$$\Gamma(s) = \int_0^\infty t^s e^{-s} \frac{dt}{t}.$$

The integral expression for $\Gamma(s)$ is absolutely convergent for $s > 0$, and by integrating by parts can be shown to satisfy $\Gamma(s+1) = s\Gamma(s)$. As $\Gamma(1) = 1$, one immediately sees that for integral N, $\Gamma(1 + N) = N!$, so $\Gamma(s)$ is actually the interpolation of the factorial function to all real (and even complex) numbers. For large R, not necessarily integral, Stirling's formula gives the asymptotic value of $\Gamma(1 + R)$ (meaning, in the limit as R tends to ∞):

$$\Gamma(1 + R) \sim \frac{R^R}{e^R}. \tag{6.5}$$

By Minkowski's Theorem, if $\mathrm{Vol}(S) > 2^n|\det(L)|$ then S must contain a non-zero point of L. We ask then: how large must r be to ensure that

$$\frac{\pi^{n/2} r^n}{\Gamma(1 + n/2)} > 2^n|\det(L)|?$$

Applying Stirling's formula (6.5) this translates to the approximate statement

$$\frac{\pi^{n/2} r^n e^{n/2}}{(n/2)^{n/2}} >\approx 2^n|\det(L)|,$$

which implies

$$r >\approx \sqrt{\frac{2n}{\pi e}}(|\det(L)|)^{1/n}.$$

Thus the shortest non-zero vector must certainly satisfy

$$\|\mathbf{v}_{\text{shortest}}\| <\approx \sqrt{\frac{2n}{\pi e}}(|\det(L)|)^{1/n}.$$

Although exact bounds for the size of the shortest vector of a lattice are unknown for large n, one can make probabilistic arguments using the Gaussian heuristic. One variant of the Gaussian heuristic states that for a fixed lattice L and a sphere of radius r centered at 0, as r tends to infinity the ratio of the volume of the sphere divided by $|\det L|$ will approach the number of points of L inside the sphere. In two dimensions, if L is simply $\mathbb{Z}^2$ the question of how precisely the area of a circle approximates the number of integer points inside the circle is a classical problem in number theory. In higher dimensions the problem becomes far more difficult . This is because as n increases the error created by lattice points near the surface of the sphere can be quite large. This becomes particularly problematic for small values of r. Still, one can ask the question: For what value of r does the ratio

$$\frac{\text{Vol}(S)}{|\det L|}$$

approach 1. This gives us in some sense an expected value for r, the smallest radius at which the expected number of points of L with length less than r equals 1. Performing this computation we find that this value is

$$r = \sqrt{\frac{n}{\pi e}}(|\det(L)|)^{1/n}.$$

For this reason we make the following definition:

Definition. If L is a lattice of dimension n we define the Gaussian expected shortest length to be

$$\sigma(L) = \sqrt{\frac{n}{\pi e}}(|\det(L)|)^{1/n}.$$

We will find this value $\sigma(L)$ to be useful in quantifying the difficulty of locating short vectors in lattices. It can be thought of as the probable length of the shortest vector of a “random” lattice. We will discover that if the actual shortest vector of a lattice L is significantly shorter than $\sigma(L)$ then LLL and related algorithms have an easier time locating the shortest vector.

6.4 The LLL lattice reduction algorithm [M]

6.4.1 The case of dimension 2

In this section we will explain some of the techniques that exist today for finding short bases for lattices. The simplest of these techniques was developed by Gauss and applies to lattices of dimension 2. This is best illustrated with an example. Consider the public basis for the 2 dimensional GGH example of **??**:

$$\{\mathbf{A}, \mathbf{B}\} = \{(66586820, 65354729), (6513996, 6393464)\}.$$

We'll begin by ordering them according to size. As $\|\mathbf{A}\|^2 \approx 8.70505\,10^{15}$ and $\|\mathbf{B}\|^2 \approx 8.33085\,10^{13}$, we'll set $\mathbf{b}_1 = \mathbf{B}, \mathbf{b}_2 = \mathbf{A}$.

Now we would like to obtain a basis with shorter vectors, and so we are allowed to add or subtract from one basis vector only integer multiples of another.

To subtract off from $\mathbf{b}_2$ an appropriate integral multiple of $\mathbf{b}_1$, we choose this integral multiple $m_{2,1}$ so that $\|\mathbf{b}_2 - m_{2,1}\mathbf{b}_1\|$ is minimized. To see how $m_{2,1}$ should be chosen, write $r(t) = \|\mathbf{b}_2 - t\mathbf{b}_1\|^2 = \|\mathbf{b}_2\|^2 - 2t\mathbf{b}_1 \cdot \mathbf{b}_2 + t^2\|\mathbf{b}_1\|^2$. This has a minimum when $r'(t) = -2\mathbf{b}_1 \cdot \mathbf{b}_2 + 2t\|\mathbf{b}_1\|^2 = 0$, i.e when $t = \mathbf{b}_1 \cdot \mathbf{b}_2/\|\mathbf{b}_1\|^2$. With this choice of t, if we set $\mathbf{b}_2^* = \mathbf{b}_2 - t\mathbf{b}_1$ then it is easily checked that $\mathbf{b}_2^* \cdot \mathbf{b}_1 = 0$. In fact $\mathbf{b}_2^*$ is simply the projection of $\mathbf{b}_2$ on to the orthogonal complement of $\mathbf{b}_1$. The best choice we can make for $m_{2,1}$ is thus $m_{2,1} = \lfloor \mathbf{b}_1 \cdot \mathbf{b}_2/\|\mathbf{b}_1\|^2 \rceil$, the largest integer in $\mathbf{b}_1 \cdot \mathbf{b}_2/\|\mathbf{b}_1\|^2$.

In the case at hand a computation gives $t = 10.2221$ so $m_{2,1} = 10$ and we obtain

$$\mathbf{b}_2' = \mathbf{b}_2 - m_{2,1}\mathbf{b}_1 = (1446860, 1420089).$$

We now replace $\mathbf{b}_2$ by $\mathbf{b}_2'$ and obtain a new basis

$$\{\mathbf{b}_1, \mathbf{b}_2\} = \{(6513996, 6393464), (1446860, 1420089)\}.$$

The second vector has smaller norm than the first so we re-order this to obtain

$$\{\mathbf{b}_1, \mathbf{b}_2\} = \{(1446860, 1420089), (6513996, 6393464)\}.$$

Our norms are now $\|\mathbf{b}_1\|^2 \approx 4.11006\,10^{12}$ and $\|\mathbf{b}_2\|^2 \approx 8.33085\,10^{13}$, so we've made progress. Incidentally, if we had not re-ordered the original $\{\mathbf{A}, \mathbf{B}\}$ we would have obtained $t = 0.09783$ with a consequent $m_{2,1} = 0$, so no progress would have been made.

We now repeat this process, setting $t = \mathbf{b}_2 \cdot \mathbf{b}_1 / \|\mathbf{b}_1\|^2 \approx 4.50216$, so $m_{2,1} = \lfloor t \rceil = 5$. Then

$$\mathbf{b}_2' = \mathbf{b}_2 - m_{2,1}\mathbf{b}_1 = (-720304, -706981).$$

We then replace $\mathbf{b}_2$ by $\mathbf{b}_2'$ and swap $\mathbf{b}_1, \mathbf{b}_2$, obtaining

$$\{\mathbf{b}_1, \mathbf{b}_2\} = \{(-720304, -706981), (1446860, 1420089)\}.$$

This in turn leads to

$$\{\mathbf{b}_1, \mathbf{b}_2\} = \{(6252, 6127), (-720304, -706981)\},$$

followed by $\{(-1324, -2376), (6252, 6127)\}$, followed by

$$\{\mathbf{b}_1, \mathbf{b}_2\} = \{(2280, -1001), (-1324, -2376)\}.$$

At this point the procedure terminates, as $\|\mathbf{b}_1\| < \|\mathbf{b}_2\|$ and $t = -0.103275$. Notice that this new basis is exactly the basis mentioned at the end of Section **??**.

We're left with an obvious question; Is $\mathbf{b}_1$ now the shortest vector in the lattice? In fact it is, as the following argument shows.

The vectors $\{\mathbf{b}_1, \mathbf{b}_2\}$ satisfy $\|\mathbf{b}_1\| < \|\mathbf{b}_2\|$ and

$$\frac{|\mathbf{b}_2 \cdot \mathbf{b}_1|}{\|\mathbf{b}_1\|^2} < \frac{1}{2}.$$

Any other vector $\mathbf{b}$ in the lattice generated by $\{\mathbf{b}_1, \mathbf{b}_2\}$ has the form $\mathbf{b} = \alpha\mathbf{b}_1 + \beta\mathbf{b}_2$ for some integers α, β. Clearly if $\alpha = 0$ or $\beta = 0$ with $(\alpha, \beta) \neq (0, 0)$ then $\|\mathbf{b}\| \geq \|\mathbf{b}_1\|$ or $\|\mathbf{b}\| \geq \|\mathbf{b}_2\|$. If $\alpha \neq 0$ and $\beta \neq 0$ then

$$\begin{aligned}\|\mathbf{b}\|^2 &= \alpha^2\|\mathbf{b}_1\|^2 + \beta^2\|\mathbf{b}_2\|^2 + (\alpha + \beta)(\mathbf{b}_1 \cdot \mathbf{b}_2) \\ &\geq \alpha^2\|\mathbf{b}_1\|^2 + \beta^2\|\mathbf{b}_2\|^2 - (1/2)|\alpha|\|\mathbf{b}_1\|^2 - (1/2)|\beta|\|\mathbf{b}_1\|^2 \\ &\geq ((\alpha - 1)^2 + (3/2)|\alpha| - 1)\|\mathbf{b}_1\|^2 + ((\beta - 1)^2 + (3/2)|\beta| - 1)\|\mathbf{b}_1\|^2 \\ &\geq \|\mathbf{b}_1\|^2.\end{aligned}$$

There is still one question left hanging: why does this procedure ever terminate? We will address that in greater generality next.

6.4.2 The general case and LLL

The problem of finding the shortest vector seemed to be quite straight forward in the previous section, when the dimension equalled 2. When the dimension increases the problem becomes considerably subtler, and it was not until 1982 that the next significant advance occurred. This was the very profound paper of **??** and the object of this section is to give a description of the LLL algorithm and some of its generalizations.

Suppose we are given a basis $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ for a lattice L. Our object is to transform this basis into "better" basis. But what do we mean by a better basis? We would like this better basis to be as short as possible, beginning with the shortest vector we can find and increasing gently until we reach the end. We would also like it to be as orthogonal as possible, i.e with the dot products of distinct basis vectors with each other as close to zero as possible. Hadamard's Inequality states that

$$|\det L| = \mathrm{Vol}(L) \leq \|\mathbf{v}_1\| \, \|\mathbf{v}_2\| \cdots \|\mathbf{v}_n\|.$$

and the closer that the basis is to being orthogonal, the closer the inequality will be to being an equality.

To help create an improved basis we will construct an auxiliary Gram - Schmidt basis as follows. Let $\mathbf{v}_1^* = \mathbf{v}_1$ and for $i \geq 2$ let

$$\mathbf{v}_i^* = \mathbf{v}_i - \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{v}_j^*,$$

where

$$\mu_{i,j} = \frac{\mathbf{v}_i \cdot \mathbf{v}_j^*}{\|\mathbf{v}_j^*\|^2} \quad \text{with} \quad 1 \leq j \leq i-1.$$

The collection of vectors $\{\mathbf{v}_1^*, \mathbf{v}_2^*, \ldots, \mathbf{v}_n^*\}$ is now an orthogonal (but not orthonormal!) basis for the subspace spanned by $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$. Because of this

$$|\det L| = \prod \|\mathbf{v}_i^*\|.$$

The intuition behind this construction is:

$$\mathbf{v}_i^* = \text{Projection of } \mathbf{v}_i \text{ onto } \mathrm{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_{i-1})^\perp,$$

where by $\mathrm{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_{i-1})^\perp$ we are referring to the orthogonal complement of the subspace spanned by $\{\mathbf{v}_1, \ldots, \mathbf{v}_{i-1}\}$.

This is a variant of the usual Gram-Schmidt procedure, but we have not divided by the lengths of the resulting vectors. It is important to remember that the Gram-Schmidt collection $\{\mathbf{v}_1^*, \mathbf{v}_2^*, \ldots, \mathbf{v}_n^*\}$ is not a basis for the original integer lattice, since our elementary operations were performed with non-integer quantities. But we shall use the properties of this Gram-Schmidt collection associated to L to define and produce what is called a *reduced basis.* There are two important properties that we can read off from the construction of the Gram-Schmidt vectors. First, computing the value of the $\mu_{i,j}$ will tell us if we've subtracted from $\mathbf{b}_i$ the as much of $\mathbf{b}_j$ as possible. Second, even though the collection of vectors $\{\mathbf{v}_i^*\}$ does not span the same lattice, we can still find the volume of the fundamental parallelopiped (the $\det L$) by taking the product $\prod \|\mathbf{v}_i^*\|$.

Definition. Let $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ be a basis for a lattice L. The basis is LLL reduced if

1. The size condition:

$$\mu_{i,j} = \frac{|\mathbf{v}_i \cdot \mathbf{v}_j^*|}{\|\mathbf{v}_j^*\|^2} \leq \frac{1}{2} \qquad \text{for all } 1 \leq j < i \leq n$$

holds, and if

2. The Lovász condition:

$$\|\mathbf{v}_i^*\|^2 \geq \left(\frac{3}{4} - \mu_{i,i-1}^2\right) \|\mathbf{v}_{i-1}^*\|^2 \qquad \text{for all } 1 < i \leq n$$

holds.

The Lovász condition is equivalent to

$$\|\mathbf{v}_i^* + \mu_{i,i-1}\mathbf{v}_{i-1}^*\|^2 \geq \frac{3}{4}\|\mathbf{v}_{i-1}^*\|^2.$$

and is also equivalent to:

Projection of $\mathbf{v}_i$ onto $\mathrm{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_{i-2})^\perp$

$$\geq \frac{3}{4} \cdot \text{Projection of } \mathbf{v}_{i-1} \text{ onto } \mathrm{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_{i-2})^\perp.$$

The authors prove in **??** that

Theorem 6.2. *Given a basis for a lattice L of dimension n, an LLL reduced basis $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ can be found in $O(n^6)$ operations. This basis will satisfy*

1.
$$|\det L| \le \prod_{i=1}^{n} \|\mathbf{v}_i\| \le 2^{n(n-1)/4} |\det L|, \text{ and}$$

2.
$$\|\mathbf{v}_j\| \le 2^{(i-1)/2} \|\mathbf{v}_i^*\| \qquad \text{for all } 1 \le j \le i \le n$$

In particular
$$\|\mathbf{v}_1\| \le 2^{(n-1)/4} |\det L|^{1/n}.$$

The authors prove the theorem by introducing the LLL algorithm and proving that it will terminate after $O(n^6)$ operations yielding an LLL reduced basis which must in turn satisfy the conditions of the theorem.

In the following we describe the LLL algorithm and explain why it must terminate (in polynomial time). First, a couple of remarks on the definition. Given a basis $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$, we satisfy condition (1) of the definition by, roughly speaking, subtracting from $\mathbf{b}_k$ as many integer multiples of the previous $k-1$ vectors as possible to get a shorter vector. We won't do this all at once, but in stages, and we'll see that the size reduction condition depends on the ordering of the vectors. After size reduction, we'll check to see if the Lovász condition is satisfied, which insures that a (nearly) optimal ordering of the vectors has been arranged. If not, we reorder the vectors. Here are the steps of the algorithm.

1. Assume the vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{k-1}$ are LLL reduced. (Initially, then $k = 2$, and the index is set to 1.) The first step is to reduce the size of $\mathbf{v}_k$ without affecting the size reduction of the previous $k-1$ vectors. We work from the top down. Let l be the largest integer less than k such that $|\mu_{k,j}| \le 1/2$ for all $l < j < k$. (This could be $k-1$.) Subtract off q multiples of $\mathbf{v}_l$ from $\mathbf{v}_k$ where $q = \lfloor \mu_{k,l} \rceil$. Note that no change has been made any $\mu_{k,j}$ for $j > l$, since $\mathbf{v}_j^*$ is orthogonal to these $\mathbf{v}_l$. Once $\mathbf{v}_k$ is replaced by $\mathbf{v}_k - q\mathbf{v}_l$, the new $\mu_{k,l}$ equals the old $\mu_{k,l} - q$, which is now less than or equal to $1/2$. The process continues until $\mathbf{v}_k$ has been size reduced with respect to all the previous $k-1$ vectors.

2. In Step 2, we check the Lovász condition for $\mathbf{v}_k$. If this is satisfied, we move to the next vector, and increment our index by 1. If not, we swap $\mathbf{v}_k$

and $\mathbf{v}_{k-1}$. But now the first $k-1$ vectors may no longer be *LLL* reduced, and we must decrement our index by 1. Return to Step 1.

There are two important questions that arise. Why are we swapping vectors in step 2? And why does the process, which involves incrementing and decrementing indices, ever stop?

The intuition behind the swap step is this. If the Lovász condition fails for $\mathbf{b}_k$, then

$$\text{Projection of } \mathbf{v}_i \text{ onto } \operatorname{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_{i-2})^{\perp} \\ < \frac{3}{4} \cdot \text{Projection of } \mathbf{v}_{i-1} \text{ onto } \operatorname{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_{i-2})^{\perp}.$$

But the object is to produce a list of vectors, with the shortest one in first position and the lenths in increasing order. That is, the sequence whose kth element is the product of the lengths $\prod_{i-1}^{k} \|\mathbf{v}_i\|^2$ should be an increasing sequence. If L_k denotes the partial lattice spanned by the first k vectors, then the product above is an upper bound for $(\det L_k)^2$. (As the algorithm progresses, L_k changes since the order of the vectors changes.) LLL attempts to minimize this product by producing an ordering of the vectors which comes close to minimizing the product $\prod_{i=1}^{k} \|\mathbf{v}_i^*\|^2$, which equals $(\det L_k)^2$. If the number 3/4 were replaced by the number 1 in this inequality, the algorithm would do precisely this: swapping $\mathbf{v}_k$ for $\mathbf{v}_{k-1}$ whenever this reduces $\det L_{k-1}$. When we use the constant 3/4, or for that matter any constant less than 1, we may miss an opportunity to reduce the size of this determinant by passing up a swap. For instance, in the very first step, we'll only swap if the length of $\mathbf{v}_2$ is less than 3/4 of the length of $\mathbf{v}_1$. But if the objective is to place the shortest vector in first position, we should swap whenever $\|\mathbf{v}_2\| < \|\mathbf{v}_1\|$. In this sense, it would be ideal to use the constant 1 in condition (2), but then one can not prove that LLL terminates in polynomial time. In fact, the number 3/4 is a chosen fairly arbitrarily - the only properties it needs to have is that it is less than one and larger than 1/4. In practice, the constant is frequently chosen to be very near 1. One immediate effect of swapping at stage i is that the new constant $\mu_{i,i-1}$ usually becomes larger and so it is now possible to size reduce the i^{th} vector more (relative to the previous one) if $\mathbf{v}_i$ comes before $\mathbf{v}_{i-1}$. So, swapping results in more size reductions among the basis vectors, making them more orthogonal.

We now must convince ourselves that the algorithm terminates, using our Gram-Schmidt vectors to compute determinants. Set

$$d_i = \det(\mathbf{v}_j \cdot \mathbf{v}_k)_{1 \le j,k \le i},$$

the determinant squared of the (partial) lattice generated by the vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_i$. Another expression for d_i is

$$d_i = \prod \|\mathbf{v}_i^*\|^2.$$

Thus $d_0 = 1$, and $d_n = (\det L)^2$. Let

$$D = \prod_{i=1}^{n-1} d_i.$$

The algorithm changes the value of D only in Step 2, the swap step. Indeed, one can argue that the object of LLL is to minimize these partial determinants. In a single swap, the value of D is reduced by a factor which, could be less, but is certainly not more than 3/4. But D is bounded from below, because each d_i is bounded from below by a constant which depends only i, the dimension of the partial lattice, and on the length of the smallest basis vector of L. Since D has a lower bound, we cannot decrement the value of D by 3/4 (or less) forever.

The key aspect of the LLL algorithm is that it runs in polynomial time (although we have omitted the analysis of running time here) and it delivers an output vector which is not necessarily the smallest vector of L, but which has length no worse than $2^{(n-1)/4}$ times the length of the smallest vector. In practice, for small n LLL will often find the actual smallest vector, but this does not continue to be the case as n increases.

Various improvements have been made to LLL which trade lengthening the time the algorithm takes to terminate for an improved output. For example, the "deep insertion method replaces a swap of $\mathbf{v}_k$ and $\mathbf{v}_{k-1}$ with an insertion of $\mathbf{v}_k$ between two $\mathbf{v}_{i-1}$ and $\mathbf{v}_i$. The algorithm is no longer polynomial, but in practice may give better results.

Another improvement is the "block reduction method". This involves taking a block of vectors

$$\mathbf{v}_i, \mathbf{v}_{i+1}, \ldots, \mathbf{v}_{i+\beta-1}$$

and replacing it with vectors minimizing $\mathbf{v}_i^*$. The algorithm is exponential in the blocksize β, but yields better results as β increases. For large β, the running time becomes

$$O(\beta^{c\beta} n^{c'})$$

and one is guaranteed an output

$$\|\mathbf{v}_1\| \le (\beta/\pi e)^{\frac{n-1}{\beta-1}} \min\bigl\{\|\mathbf{v}\| \,:\, \mathbf{v} \in L, \quad \mathbf{v} \neq 0\bigr\}.$$

6.5 Applications of LLL to cryptanalysis

The LLL algorithm has many applications to cryptanalysis. These range from the original knapsack public key cryptosystems to more recent variants such as GGH and NTRU. However the tools that LLL and its generalizations provide for finding small or close vectors in lattices have multitudes of other applications. Exercise 1.35, given at the end of Chapter 1 provides a particularly appealing way of demonstrating this. Recall the setup:

Alice and Bob create a symmetric cipher as follows. Their private key k is a large integer and their messages (plaintexts) are d-digit integers

$$\mathcal{M} = \{m \in \mathbb{Z} : 0 \le m < 10^d\}.$$

In order to encrypt a message, Alice computes $\sqrt{k}$ to d decimal places, throws away the part to the left of the decimal point, and keeps the remaining d digits. Let α be this d-digit number. (For example, if $k = 23$ and $d = 6$, then $\sqrt{23} = 9.32737905\ldots$ and $\alpha = 327379$.)

Alice encrypts a message m as

$$c \equiv m + \alpha \pmod{10^d}.$$

Since Bob knows k, he can also find α, and then he decrypts c by computing $m \equiv c - \alpha \pmod{10^d}$. The number α used for encryption and decryption was given by the formula

$$\alpha = \left\lfloor 10^d \left(\sqrt{k} - \lfloor\sqrt{k}\,\rfloor\right)\right\rfloor,$$

where $\lfloor t \rfloor$ is the greatest integer that is less than or equal to t.

The following question was asked: If Eve steals a plaintext/ciphertext pair (m, c), then she clearly can recover the number α, since $\alpha \equiv c - m \pmod{10^d}$. If 10^d is large compared to k, can she also recover the number k? This might be useful, for example, if Alice and Bob use the second d digits of $\sqrt{k}$ to encrypt their next message. We are now in a position to give a reasonably complete answer to this question.

Write $\beta = \lfloor \sqrt{k} \, \rfloor$ so

$$\sqrt{k} = \beta + \alpha 10^{-d} + \epsilon,$$

where $0 < \epsilon < 10^{-d}$. Here α and d are known, but k, β are unknown. Note that $0 \le \alpha < 10^d$. Multiplying both sides by 10^d and squaring, we obtain

$$10^{2d} k = 10^{2d} \beta^2 + 2\alpha\beta 10^d + \alpha^2 + e,$$

where the error e is

$$e = 2(10^d \beta + \alpha) 10^d \epsilon + 10^{2d} \epsilon^2 = O(10^d \sqrt{k}).$$

The question we now face is: how can we set up a lattice with the property that the shortest vector in the lattice, if we could locate it, would reveal k? One possibility is to consider the lattice L generated by the rows of the following matrix:

$$\begin{pmatrix} 1 & 0 & 10^{2d} \\ 0 & 1 & -2\alpha 10^d \\ 0 & 0 & -\alpha^2 \end{pmatrix}.$$

The key point here is that $10^{2d}, -2\alpha 10^d, -\alpha^2$ are all of roughly the same size, and yet a linear combination of them turns out to equal e, which is considerably smaller. This is what we need to take advantage of. In particular, $k - \beta^2$ times the first row plus β times the second row plus the third row yields $(k - \beta^2, \beta, e)$. Recovering this vector would reveal k, and if this were the shortest vector in L we would have a good chance of locating it. Unfortunately, it turns out that this is not in fact the shortest vector in L. However, there is something we can do to salvage the situation. We actually had more flexibility than we allowed ourselves, and we could have introduced an arbitrary parameter $A > 0$ and considered the lattice L generated by

$$\begin{pmatrix} A & 0 & 10^{2d} \\ 0 & A & -2\alpha 10^d \\ 0 & 0 & -\alpha^2 \end{pmatrix}.$$

Then the same linear combination of rows yields the target vector

$$\mathbf{t} = (A(k - \beta^2), A\beta, e).$$

As $k - \beta^2 = O(\beta) = O(\sqrt{k})$ it follows that if $A > 10^d$ then $\|t\| = O(A\sqrt{k})$.

The determinant of L has absolute value $A^2\alpha^2 = O(A^2 10^{2d}$ and thus by the Gaussian heuristic in dimension 3 (or by Minkowski's bound) , the expected shortest length σ of a vector in L is on the order of $(A\alpha)^{2/3}$. Thus the ratio r of $\|\mathbf{t}\|$ over σ is on the order of

$$r = \frac{A\sqrt{k}}{(A\alpha)^{2/3}}.$$

For the target to be as small as possible compared to the expected smallest we want $r < 1$. However we also want $A > 10^d$ (why?) and combining these constraints yields $A = 10^d$ and $10^d > k^{3/2}$. Thus as long as d is sufficiently larger than $(3/2)\log_{10} k$ to overcome the implied constants in the order symbol, the target $\mathbf{t}$ should in fact be the smallest vector in L.

Let's demonstrate this with a small example. Suppose the the first 21 digits after the decimal point of the square root of a 12 digit integer k is $\alpha = 977499789696409173668$. If we set $d = 21$ and $A = 10^{21}$ and feed the rows of the above matrix into the LLL algorithm we get out a basis of three vectors, the shortest of which is

$$(4371511000000000000000000000, 2236067000000000000000000000, e),$$

with $e = 3270366745550727053415425776$. This reveals that $k - \beta^2 = 4371511$ and $\beta = 2236067$, from which it follows that $k = 5 \cdot 10^{12}$. If we had used only 20 digits, this would not have worked, so in fact we had to go up to 21 from the rough estimate of 18 that the above argument gave.

So, to summarize lattice reduction strategy, the idea is to construct a lattice where the information you want to find is contained in a very short vector. If it's short enough it will, with high probability, be the shortest vector in the lattice. If this is the case, then LLL or another lattice reduction algorithm has a chance of locating it.

Section 6.1. Subset sum problems and knapsacks [**M**]

6.1. Suppose you are given a public list for a knapsack cryptosystem:

$$\{5186, 2779, 5955, 2307, 6599, 6771, 6296, 7306, 4115, 7039\}.$$

You are also handed an encrypted message: $S = 26560$. Your problem: You have the secret multiplier $A = 4392$, and the secret modulus B = 8387. Find the private list and decrypt the message.

Section 6.2. Another motivation for the study of lattices [**M**]

6.2. You have been spying on Arthur for some time and overhear him receive an encrypted message 83493429501. You know that his public key is $h = 24201896593$ and $q = 148059109201$. Luckily, Arthur isn't aware of how easily broken his neat public key scheme is. Use the method of continued fractions do recover the private key f and the message m.

Section 6.3. Integer lattices and the shortest vector problem [**M**]

6.3. Let L be the lattice generated by $\{(1,3,-2),(2,1,0),(-1,2,5)\}$. Find the volume of the fundamental domain for L.

6.4. Let L be a lattice, let $\mathcal{B}$ be a basis for L, and let $\mathcal{F}$ be the associated fundamental domain for L. Let $\mathbf{w} \in \mathbb{R}^n$ be a vector.

(a) Prove that there is a vector $\mathbf{v} \in L$ in the lattice and a vector $\mathbf{u} \in \mathcal{F}$ in the given fundamental domain so that

$$\mathbf{w} = \mathbf{v} + \mathbf{u}. \tag{6.6}$$

(b) Prove that for a given vector $\mathbf{w} \in \mathbb{R}^n$, there is only one choice for $\mathbf{v} \in L$ and only one choice for $\mathbf{u} \in \mathcal{F}$ that makes (6.6) true.

(c) Explain why (a) and (b) imply that $\mathbb{R}^n$ is equal to the disjoint union of the translates of $\mathcal{F}$ by the vectors in L. In other words, for any $v \in L$, let $\mathcal{F} + \mathbf{v}$ denote the set

$$\mathcal{F} + \mathbf{v} = \{\mathbf{v} + \mathbf{u} : \mathbf{u} \in \mathcal{F}\}.$$

Explain why (a) says that for different $\mathbf{v} \in L$, the sets $\mathcal{F} + \mathbf{v}$ have no points in common, and why (b) says that the union of all of the set $\mathcal{F} + \mathbf{v}$ for all $\mathbf{v} \in L$ is equal to $\mathbb{R}^n$.

6.5. Let A and B be matrices in $\mathrm{GL}_n(\mathbb{Z})$.

(a) Prove that $AB \in \mathrm{GL}_n(\mathbb{Z})$.

(b) Prove that $A^{-1} \in \mathrm{GL}_n(\mathbb{Z})$.

(c) Prove that the n-by-n identity matrix is in $\mathrm{GL}_n(\mathbb{Z})$.

(d) Prove that $\mathrm{GL}_n(\mathbb{Z})$ is a group. (*Hint.* You have already done most of the work in proving (a), (b) and (c). For the associative law, either reprove it directly or use the fact that you know that it is true for matrices with real coefficients.)

(e) Is $\mathrm{GL}_n(\mathbb{Z})$ a commutative group?

6.6. Let L be the lattice given by the basis

$$\mathcal{B} = \{(3,1,-2),\ (1,-3,5),\ (4,2,1)\}$$

Which of the following sets of vectors are also bases for L? If so, express the new basis in terms of the basis $\mathcal{B}$, i.e. find the change of basis matrix.

(a) $\mathcal{B}_1 = \{(5,13,-13),\ (0,-4,2),\ (-7,-13,18)\}$.

(b) $\mathcal{B}_2 = \{(4,-2,3),\ (6,6,-6),\ (-2,-4,7)\}$.

6.7. Which of the following matrices are in $\mathrm{GL}_n(\mathbb{Z})$? For those matrices in $\mathrm{GL}_n(\mathbb{Z})$, find the inverse.

(a) $A_1 = \begin{pmatrix} 3 & 1 \\ 2 & 2 \end{pmatrix}$ (b) $A_2 = \begin{pmatrix} 3 & -2 \\ 2 & -1 \end{pmatrix}$

(c) $A_3 = \begin{pmatrix} 3 & 2 & 2 \\ 2 & 1 & 2 \\ -1 & 3 & 1 \end{pmatrix}$ (d) $A_4 = \begin{pmatrix} -3 & -1 & 2 \\ 1 & -3 & -1 \\ 3 & 0 & -2 \end{pmatrix}$

6.8. Alice and Bob agree to use the GGH public key cryptosystem with $n = 2$. Alice's public key is the basis $\mathcal{B}_{\mathsf{pub}} = \big\{(-800,-643),(-1752,-1408)\big\}$ for the lattice L.

(a) Bob decides to send Alice the message $m = (-4,7)$ using the ephemeral key $r = (1,1)$. What is the ciphertext that Bob sends to Alice?

(b) Alice receives the ciphertext $c = (5752,4624)$. Her private (short) basis for L is $\mathcal{B}_{\mathsf{priv}} = \big\{(-8,7),(8,10)\big\}$. Decrypt and find the plaintext.

(c) Eve intercepts the ciphertext $c = (5752,4624)$ and she tries to decrypt it using the public basis $\mathcal{B}_{\mathsf{pub}}$. What "plaintext" does she get? Is it correct?

6.9. Construct a 2 dimensional GGH example where the public basis is on the order of 10 digits per vector. Determine the maximum size of the perturbations that allows consistent decryption.

6.10. If L is the lattice generated by the vectors $\mathbf{v}_1 = (20, 14, 8)$, $\mathbf{v}_2 = (13, 0, 12)$, and $\mathbf{v}_3 = (0, 18, 9)$, what does the (Gaussian) expected length of the shortest vector?

6.11. An n-dimensional lattice L can also be generated by n independent vectors sitting inside of $\mathbb{R}^m$, where $m > n$. In this case the square of the volume of the fundamental domain can be computed by constructing an n by m matrix, multiplying it on the right by the transpose of this matrix and taking the determinant of the resulting n by n matrix. For example, if $n = 3, m = 4$ and $\mathbf{v}_1 = (1, 0, 1, -1), \mathbf{v}_2 = (1, 2, 0, 4), \mathbf{v}_3 = (1, -1, 2, 1)$ then we compute the determinant of

$$\begin{pmatrix} 1 & 0 & 1 & -1 \\ 1 & 2 & 0 & 4 \\ 1 & -1 & 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 \\ 0 & 2 & -1 \\ 1 & 0 & 2 \\ -1 & 4 & 1 \end{pmatrix} = \begin{pmatrix} 3 & -3 & 2 \\ -3 & 21 & 3 \\ 2 & 3 & 7 \end{pmatrix}$$

which equals 231. Thus the determinant of the lattice with basis

$$\{(1, 0, 1, -1), (1, 2, 0, 4), (1, -1, 2, 1)\}$$

is $\pm\sqrt{231}$. Explain why this works.

Section 6.4. The LLL lattice reduction algorithm [M]

6.12. Verify that if $t = \mathbf{b}_1 \cdot \mathbf{b}_2 / \|\mathbf{b}_1\|^2$ and $\mathbf{b}_2^* = \mathbf{b}_2 - t\mathbf{b}_1$ then $\mathbf{b}_2^* \cdot \mathbf{b}_1 = 0$ and $\mathbf{b}_2^*$ is the projection of $\mathbf{b}_2$ on to the orthogonal complement of $\mathbf{b}_1$.

6.13. Suppose L is the lattice with basis vectors $\mathbf{v}_1 = (161, 120)$ and $\mathbf{v}_2 = (104, 77)$.

(a) Is $(0, 1)$ in the lattice?

(b) Find the LLL reduced basis.

(c) Use the reduced basis to find the closest lattice vector to $(-4.5, 11)$.

6.14. Use the method of Gauss to break the GGH example you constructed in the previous section.

6.15. Use the LLL algorithm to reduce the basis given by $\mathbf{v}_1 = (20, 16, 3)$, $\mathbf{v}_2 = (15, 0, 10)$, $\mathbf{v}_3 = (0, 18, 9)$. How does the length of the shortest vector found by *LLL* compare to the expected shortest length predicted by the Gaussian heuristic?

Section 6.5. Applications of LLL to cryptanalysis [**M**]

6.16. The first 15 digits after the decimal place of the square root of a 10 digit integer are 418400286617716. Find the 10 digit integer.

Chapter 7

Polynomial rings, quotient rings, and convolutions

The collection of all polynomials with coefficients taken from $\mathbb{Z}$ in a variable x is usually denoted $\mathbb{Z}[x]$. Thus we write

$$\mathbb{Z}[x] = \{a_0 + a_1x + a_2x^2 + \dots a_nx^n | n \geq 0, n, a_0, a_1, \dots, a_n \in \mathbb{Z}\}$$

For example 0, -203, $2-3x+2x^3-3x^4+2x^8$ and $-72+14x+23x^2-7x^3+47x^{203}-14x^{2001}$ are polynomials in $\mathbb{Z}[x]$. The usual sum, difference or product of polynomials remains a polynomial. Polynomials in $\mathbb{Z}[x]$ form a group under addition and obey the distributive law. Because of this $\mathbb{Z}[x]$ is an example of a commutative ring. In this chapter we will describe how polynomials can be used to create some interesting cryptographic constructions. These constructions will fit naturally into the more general subject of lattice based cryptography, which will be covered in the next chapter.

7.1 Polynomial quotient rings with integer coefficients [M]

A new and surprisingly useful ring can be obtained by choosing an $N \in \mathbb{Z}, N \geq 1$ and adding the rule that $x^N = 1$. This has the effect of reducing exponents of polynomials modulo N. Thus, for example, if $N = 5$ we would have

$$\begin{aligned} &3 + 5x - 2x^2 + x^3 + 7x^4 - 8x^5 + 12x^7 + 13x^{11} \\ &= 3 + 5x - 2x^2 + x^3 + 7x^4 - 8 + 12x^2 + 13x \\ &= -5 + 18x + 10x^2 + x^3 + 7x^4. \end{aligned}$$

In formal terms, we have defined the commutative quotient ring

$$R = \mathbb{Z}[x]/(x^N - 1).$$

In this ring polynomials $\mathbf{r}(x)$ are replaced by cosets $\bar{\mathbf{r}}(x)$, with $\bar{\mathbf{r}}(x) = \bar{\mathbf{s}}(x)$ if and only if $\mathbf{r}(x) - \mathbf{s}(x)$ is divisible by $x^N - 1$. In effect, this means that the cosets $\bar{\mathbf{r}}(x)$ and $\bar{\mathbf{s}}(x)$ are equal if an only if $\mathbf{r}(x)$ and $\mathbf{s}(x)$ are equal after each exponent i has been reduced modulo N to the range $0 \le i \le N - 1$. Thus the polynomial cosets $\bar{\mathbf{r}} \in R$ are in one to one correspondence with the set of all polynomials of degree strictly less than N. For simplicity we will henceforth drop the bar from the notation.

As we mentioned above, R is a commutative ring under the operations of $+, *$. Thus polynomials of R form a group under addition. Also multiplication satisfies

$$\mathbf{f} * \mathbf{g} = \mathbf{g} * \mathbf{f}$$

and

$$\mathbf{f} * (\mathbf{g}_1 + \mathbf{g}_2) = \mathbf{f} * \mathbf{g}_1 + \mathbf{f} * \mathbf{g}_2$$

for all $\mathbf{f}, \mathbf{g}, \mathbf{g}_1, \mathbf{g}_2 \in R$.

For example, if $\mathbf{a} = 1 - 2x + 4x^3 - x^4, \mathbf{b} = 3 + 4x - 2x^2 + 5x^3 + 2x^4, c = x^2 - 3x^3$ with $N = 5$ then

$$\mathbf{a} * \mathbf{b} = 3 - 2\,x - 10\,x^2 + 21\,x^3 + 5\,x^4 - 16\,x^5 + 22\,x^6 + 3\,x^7 - 2\,x^8$$

which equals

$$-13 + 20\,x - 7\,x^2 + 19\,x^3 + 5\,x^4 = \mathbf{b} * \mathbf{a}.$$

Similarly

$$\mathbf{a} * \mathbf{c} = 4 - 13\,x + 4\,x^2 - 5\,x^3 + 6\,x^4.$$

On the other hand

$$\mathbf{a} * (\mathbf{b} + \mathbf{c}) = 3 - 2\,x - 9\,x^2 + 16\,x^3 + 11\,x^4 - 12\,x^5 + 9\,x^6 + 6\,x^7 - 2\,x^8$$

which equals

$$-9 + 7\,x - 3\,x^2 + 14\,x^3 + 11\,x^4 = \mathbf{a} * \mathbf{b} + \mathbf{a} * \mathbf{c}.$$

Thus we have verified in this case that $\mathbf{a} * (\mathbf{b} + \mathbf{c}) = \mathbf{a} * \mathbf{b} + \mathbf{a} * \mathbf{c}$.

The polynomials of R can be identified with N-dimensional vectors of coefficients. Thus $\mathbf{r}(x) = r_0 + r_1 x + r_2 x^2 + \cdots + r_{N-1} x^{N-1}$ corresponds to the vector $(r_0, r_1, r_2, \ldots, r_{N-1}) \in \mathbb{Z}^N$. Addition of two polynomials $\mathbf{r}$ and $\mathbf{s}$ corresponds to pointwise addition of the corresponding vectors, but multiplication of polynomials corresponds to a more intricate vector operation.

Consider two polynomial cosets $\mathbf{r}, \mathbf{s} \in R$. Letting $\mathbf{r}(x) = r_0 + r_1 x + r_2 x^2 + \cdots + r_{N-1} x^{N-1}$, $\mathbf{s}(x) = s_0 + s_1 x + s_2 x^2 + \cdots + s_{N-1} x^{N-1}$ the product has the form

$$\mathbf{r}(x)\mathbf{s}(x) = \sum_{k=0}^{2N-2} \left(\sum_{i+j=k} r_i s_j \right) x^k$$

and after applying the equivalence relation this reduces to

$$\mathbf{r}(x) * \mathbf{s}(x) = \sum_{k=0}^{N-1} c_k x^k,$$

where

$$c_k = \sum_{i+j \equiv k \pmod{N}} r_i s_j. \tag{7.1}$$

Thus, in terms of vectors we have the convolution relation

$$(r_0, r_1, r_2, \ldots, r_{N-1}) * (s_0, s_1, s_2, \ldots, s_{N-1}) = (c_0, c_1, c_2, \ldots, c_{N-1}),$$

where the c_k are defined as in (7.1). We will use $*$ to denote this convolution multiplication both in R and for the corresponding vectors.

7.2 Norms of polynomials [M]

Because of the identification between the ring of polynomials $\mathbf{a} = a_0 + a_1 x + \cdots + a_{N-1} x^{N-1} \in R$ and the ring of vectors $(a_0, a_1, \ldots, a_{N-1})$ under addition and convolution multiplication it is natural to think of introducing a norm on R. However there are a number of possible ways to do this. The most straightforward approach is to use the usual Euclidean norm:

Definition. Let $\mathbf{a} = a_0 + a_1 x + \cdots + a_{N-1}x^{N-1} \in R$. Then

$$\|\mathbf{a}\| = \sqrt{a_0^2 + a_1^2 + \cdots + a_{N-1}^2}. \tag{7.2}$$

This is also known as the L^2 norm. It satisfies the three requirements for a norm, namely $\|\mathbf{a}\| \geq 0$ for all $\mathbf{a}$, $\|\mathbf{a}\| = 0$ if and only if $\mathbf{a} = 0$ and the triangle inequality:

$$\|\mathbf{a} + \mathbf{b}\| \leq \|\mathbf{a}\| + \|\mathbf{b}\|$$

is satisfied.

An alternative is the L^∞ norm defined by

$$\|\mathbf{a}\|_\infty = \max\{|a_0|, |a_1|, \ldots, |a_{N-1}|\}. \tag{7.3}$$

This also satisfies the three requirements for a norm.

We will find it most advantageous to use a variation on the usual Euclidean norm, which we'll refer to as the "centered norm". To define this, let

$$\bar{a} = N^{-1} \sum_{i=0}^{N-1} a_i$$

denote the mean value of the coefficients of $\mathbf{a}$. We will refer to

$$(a_0 - \bar{a}, a_1 - \bar{a}, + \cdots + a_{N-1} - \bar{a})$$

as the centered vector corresponding to $\mathbf{a}$. Then the centered norm is

$$\|\mathbf{a}\|_{\text{cent}} = \sqrt{(a_0 - \bar{a})^2 + (a_1 - \bar{a})^2 + \cdots + (a_{N-1} - \bar{a})^2}. \tag{7.4}$$

This is "almost " a norm, in the sense that $\|\mathbf{a}\|_{\text{cent}} \geq 0$ for all $\mathbf{a}$, and the triangle inequality is satisfied. However, the requirement $\|\mathbf{a}\|_{\text{cent}} = 0$ if and only if $\mathbf{a} = 0$ is not satisfied. In fact, any vector of the form $(c, c, \ldots, c) = c(1, 1, \ldots, 1)$, where c is any constant has centered norm equal to 0 and $\|\mathbf{a}\|_{\text{cent}} = 0$ implies that $\mathbf{a} = (c, c, \ldots, c)$ for some c. The difference between the centered norm and the usual norm has the potential for being a source of great confusion among students, and we will therefore take some time to discuss the reasons for taking this path and the right way to think about centered norms.

First notice, that if the mean value of the coefficients of $\mathbf{a}$ is 0 then the centered norm is the same as the usual norm. Thus, for example, if $N = 7$

$$\|(1, 0, -1, -2, 0, 0, 2)\| = \|(1, 0, -1, -2, 0, 0, 2)\|_{\text{cent}} = \sqrt{10}.$$

If the mean value of the coefficients is non-zero but close to 0, the usual norm and the centered norm will still often be good approximations of each other. For example,

$$\begin{aligned}&\|(1,0,-1,-1,1,0,1)\|\\&\qquad=\sqrt{1^2+0^2+(-1)^2+(-1)^2+1^2+0^2+1^2}=\sqrt{5}\approx 2.236,\end{aligned}$$

while

$$\begin{aligned}&\|(1,0,-1,-1,1,0,1)\|_{\text{cent}}=\\&\sqrt{(1-\frac{1}{7})^2+(0-\frac{1}{7})^2+(-1-\frac{1}{7})^2+(-1-\frac{1}{7})^2+(1-\frac{1}{7})^2+(0-\frac{1}{7})^2+(1-\frac{1}{7})^2}\\&=\sqrt{5-\frac{1}{7}}\approx 2.204\end{aligned}$$

This suggests the easiest way to get adjusted to the idea of using the centered norm. Simply imagine that all vectors we use have coefficients with mean value 0. This will essentially be true in most cases and the centered norm allows us to formally act as if it is true. Also, the use of the centered norm makes formulas such as part (a) of Proposition 7.1 cleaner and will ultimately prove to be the most effective norm to use when applying the LLL algorithm.

There is another reason why the centered norm makes intuitive sense. If we compute the convolution product $(1,1,\ldots,1)*(a_0,a_1,\ldots,a_{N-1})$ we obtain

$$\begin{aligned}(1,1,\ldots,1)*(a_0,a_1,\ldots,a_{N-1})&=\left(\sum_{i=0}^{N-1}a_i,\sum_{i=0}^{N-1}a_i,\ldots,\sum_{i=0}^{N-1}a_i\right)\\&=\left(\sum_{i=0}^{N-1}a_i\right)(1,1,\ldots,1).\end{aligned}$$

Thus if we think of the vector $(c,c,\ldots,c)$ as being “zero” for all constants c, then it has the same property that 0 has under convolution multiplication, namely that “zero” times “anything” equals “zero”. Similarly, if we interpret vectors as polynomials in R then $c+cx+cx^2+\cdots+cx^{N-1}$ has a zero centered norm for any constant c. This is not too unreasonable, as

$$(x-1)(1+x+x^2+\cdots+x^{N-1})=1-x^N.$$

In our quotient ring R we divide out by $1 - x^N$, essentially declaring by fiat that that the polynomial $1 - x^N$ is equivalent to zero. Because the polynomial $1 + x + x^2 + \cdots + x^{N-1}$ is a zero divisor it makes sense that its norm, appropriately defined, should also be zero.

7.3 Norms of products and sums

Now that we have a norm to apply to R it is reasonable to ask how this norm interacts with addition and with convolution products. In fact it interacts in a very natural and beautiful way that is tied up with some of the odder aspects of the geometry of N-dimensional spheres. We will label the two statements describing this behavior as a proposition, but they are really heuristics. This is because the statements can be made precise and proved in some very specific cases. However, experimental evidence indicates that they actually hold in vast generality. They also provide a very useful conceptual framework for analyzing N-dimensional constructs and we have not yet encountered a reasonable situation where they fail to hold. From this point on we will drop the "cent" subscript on the norm and $\|\mathbf{r}\|$ will refer to the centered norm of $\mathbf{r}$.

Proposition 7.1. *Let $\mathbf{a}, \mathbf{b} \in R$ be interpreted as vectors. Suppose that the coefficients of the centered vectors corresponding to $\mathbf{a}$ and $\mathbf{b}$ are reasonably uniformly and symmetrically distributed about 0. Then*

(a)

$$\|\mathbf{a} * \mathbf{b}\| \approx \|\mathbf{a}\| \|\mathbf{b}\|.$$

(b)

$$\|\mathbf{a} + \mathbf{b}\|^2 \approx \|\mathbf{a}\|^2 + \|\mathbf{b}\|^2.$$

We refer to these two properties as quasi-multiplicativity and quasi-additivity. The first question, of course, is what do we mean by reasonably uniformly and symmetrically distributed about 0? And the answer, unfortunately, is that it's hard to say. However, it's quite easy to get a feeling for why this heuristic should hold in a very general setting.

Let $\mathbf{c} = \mathbf{a} * \mathbf{b}$. Then by (7.1)

$$\begin{aligned}\|\mathbf{c}\|^2 &= \sum_{k \pmod N} c_k^2 = \sum_{k \pmod N} \left(\sum_{i+j \equiv k \pmod N} a_i b_j \right)^2 \\ &= \sum_{k \pmod N} \sum_{i+j\equiv k, l+m \equiv k \pmod N} a_i a_l b_j b_m \\ &= \sum_{i+j \equiv l+m \pmod N} a_i a_l b_j b_m.\end{aligned}$$

Here the last sum is over all 4-tuples $(i, j, l, m) \pmod N$ such that $i + j \equiv l + m \pmod N$.

Let us think of the a_i, b_j as $2N$ independent random variables, each with expected value 0, i.e $E(a_i) = E(b_j) = 0$. Then the expected value of each contribution to the sum above will be 0 unless $i = l$ and $j = m$. Thus

$$E(\|\mathbf{c}\|^2) = E\left(\sum_{i,j \pmod N} a_i^2 b_j^2 \right) = E(\|\mathbf{a}\|^2 \|\mathbf{b}\|^2) = E(\|\mathbf{a}\|^2) E(\|\mathbf{b}\|^2).$$

Similarly,

$$\|\mathbf{a} + \mathbf{b}\|^2 = \sum_{k \pmod N} (a_k + b_k)^2 = \sum_{k \pmod N} (a_k^2 + 2a_k b_k + b_k^2)$$

so

$$\begin{aligned}E(\|\mathbf{a} + \mathbf{b}\|^2) &= \sum_{k \pmod N} E((a_k + b_k)^2) = \sum_{k \pmod N} (E(a_k^2) + 2E(a_k b_k) + E(b_k^2)) \\ &= E(\|\mathbf{a}\|^2) + E(\|\mathbf{b}\|^2).\end{aligned}$$

The properties of quasi-multiplicativity and quasi-additivity thus reduce to a tendency of sums arising in $\mathbf{a} * \mathbf{b}$ and $\mathbf{a} + \mathbf{b}$ to remain close to their mean value. In fact, in many specific instances it can be shown that the probability of such sums departing from their mean value decreases exponentially as the distance from the mean increases. For example, consider the following scenario. Fix $\alpha > 0$ and take the a_i, b_j coefficients to be random variables that take the value 1 with probability α, the value -1 with probability α and the value 0 with probability $1 - 2\alpha$. Then $E(\|\mathbf{a}\|^2) = E(\|\mathbf{b}\|^2) = 2\alpha N$ and so

$E(\|\mathbf{a}\|^2)E(\|\mathbf{b}\|^2) = 4\alpha^2N^2$. For any $\epsilon > 0$ let $P(\epsilon, *)$ denote the probability that

$$1 - \epsilon < \|a * b\|^2/(\|a\|^2\|b\|^2) < 1 + \epsilon$$

does *not* hold. Applying what is called a Khintchine type inequality one can show that there exists a constant $c > 0$ such that

$$P(\epsilon, *) < ce^{-4\alpha^3\epsilon^2N}. \tag{7.5}$$

In fact, if $P(\epsilon, +)$ denotes the probability that

$$1 - \epsilon < \|a + b\|^2/(\|a\|^2 + \|b\|^2) < 1 + \epsilon$$

then (7.5) also holds with $P(\epsilon, *)$ replaced by $P(\epsilon, +)$.

There is a very simple class of examples that approximates this scenario and most of our constructs will use this class as a source of building blocks.

Definition. For fixed d_1, d_2, with $0 \leq d_1, d_2 \leq N/2$ we define $S(d_1, d_2)$ to be the collection of all $f \in R$ such that precisely d_1 coefficients of f equal 1, d_2 coefficients equal -1 and the remainder of the coefficients equal 0. In this definition we suppress the N from the notation, just as we do with R, as it will remain constant through most discussions.

For any d_1, d_2, if $\mathbf{a}, \mathbf{b}$ are selected randomly from $S(d_1, d_2)$ then Proposition 7.1 holds. For example, take $d_1 = d_2 = [N/3]$. This approximates the random variable setting described above with $\alpha = 1/3$. If we let $\lambda = \|f * g\|//(\|f\|\|g\|)$ then if $\lambda = 1$ the quasi-multiplicative property holds strongly, and as λ departs from 1 the quasi-multiplicativity weakens. In Table 7.1 we show the distribution of λ in this case as N changes, running 10 million examples for each N.

Similarly, if we let $\lambda = \|f + g\|/\sqrt{\|f\|^2 + \|g\|^2}$ then the distance of λ from 1 measures the departure from quasi-additivity. The distribution of λ in this case for varying N, 10 million examples froor each N is given in Table 7.2

Notice that as N increases the tendency toward quasi-multiplicativity and quasi-additivity increases. This is in accordance with the theoretical estimate (7.5).

Picking points at random from $S(d_1, d_2)$ amounts to picking random vertices from a subset of all the vertices of rectangle in N dimensional space. At the far opposite end of the spectrum, one could pick points at random from the surface of a sphere. This is actually far harder to achieve in a real

$\|f * g\|//(\|f\|\|g\|)$	N=100	N=200	N=300	N=400	N=500
$0.60 \leq \lambda < 0.70$	0.00012	0	0	0	0
$0.70 \leq \lambda < 0.80$	0.09219	0.00081	0	0	0
$0.80 \leq \lambda < 0.90$	6.13065	1.64909	0.47569	0.13983	0.04172
$0.90 \leq \lambda < 0.92$	4.80055	2.86506	1.50575	0.73377	0.35834
$0.92 \leq \lambda < 0.94$	6.97291	5.86250	4.31625	3.01148	2.02820
$0.94 \leq \lambda < 0.96$	9.19447	9.90668	9.37605	8.36982	7.28266
$0.96 \leq \lambda < 0.98$	10.81305	13.89269	15.38263	16.17615	16.49717
$0.98 \leq \lambda < 1.00$	11.84338	16.06206	19.29374	21.90226	24.13480
$1.00 \leq \lambda < 1.02$	11.39148	15.62508	18.71422	21.25882	23.39351
$1.02 \leq \lambda < 1.04$	10.42015	12.95543	14.35643	15.13189	15.51815
$1.04 \leq \lambda < 1.06$	8.61696	9.23614	8.88981	8.15212	7.28724
$1.06 \leq \lambda < 1.08$	6.56270	5.75761	4.54691	3.44915	2.55305
$1.08 \leq \lambda < 1.10$	4.72300	3.19218	1.98561	1.19529	0.70104
$1.10 \leq \lambda < 1.20$	7.79039	2.94506	1.17247	0.47864	0.20405
$1.20 \leq \lambda < 1.30$	0.60012	0.04847	0.00555	0.00078	0.00007
$1.30 \leq \lambda < 1.40$	0.04329	0.00109	0.00008	0	0
$1.40 \leq \lambda < 1.50$	0.00403	0.00005	0.00001	0	0
$1.50 \leq \lambda < 1.60$	0.00043	0	0	0	0
$1.60 \leq \lambda < 1.70$	0.00012	0	0	0	0
$1.70 \leq \lambda < 1.80$	0.00001	0	0	0	0

Table 7.1: Percent of $\lambda = \|f * g\|//(\|f\|\|g\|)$ values in given interval. Trinary f, g, with $d = [N/3]$.

$\|f+g\|/\sqrt{\|f\|^2+\|g\|^2}$	N=100	N=200	N=300	N=400	N=500
$0.70 \leq \lambda < 0.80$	0.01567	0.00001	0	0	0
$0.80 \leq \lambda < 0.90$	2.94158	0.30722	0.04093	0.00629	0.00072
$0.90 \leq \lambda < 0.92$	2.72483	1.10175	0.36972	0.10617	0.03162
$0.92 \leq \lambda < 0.94$	7.29552	3.47068	1.68,797	0.97448	0.44174
$0.94 \leq \lambda < 0.96$	7.45514	8.24474	6.91786	5.09843	3.25223
$0.96 \leq \lambda < 0.98$	14.92908	14.75866	16.85245	15.34214	14.46492
$0.98 \leq \lambda < 1.00$	17.60753	24.24458	25.85888	29.95905	33.12513
$1.00 \leq \lambda < 1.02$	11.65335	20.00447	25.13378	26.97141	30.45653
$1.02 \leq \lambda < 1.04$	14.93284	14.76248	15.44146	16.21904	15.00521
$1.04 \leq \lambda < 1.06$	10.38154	9.22058	5.99744	4.59399	2.96253
$1.06 \leq \lambda < 1.08$	4.37478	2.81857	1.51691	0.68846	0.25048
$1.08 \leq \lambda < 1.10$	3.61501	0.90787	0.16775	0.03890	0.00875
$1.10 \leq \lambda < 1.20$	2.07294	0.15839	0.01485	0.00164	0.00014
$1.20 \leq \lambda < 1.30$	0.00019	0	0	0	0

Table 7.2: Percent of $\lambda = \|f+g\|/\sqrt{\|f\|^2+\|g\|^2}$ values in given interval. Trinary f, g, with $d = [N/3]$.

life scenario, in the sense that it's computationally more difficult to simulate this sort of random choice. However, one can prove by different methods that for random choices of $\|\mathbf{a}\|, \|\mathbf{b}\|$ made with respect to this distribution the heuristic of Proposition 7.1 also holds. The relevant point here is the well known fact that if two points are chosen at random on the surface of an N-dimensional sphere they will, with high probability, be close to perpendicular to each other for large N. Because Proposition 7.1 is provable in several widely differing settings, and because it agrees with experimental evidence in more general settings, we feel comfortable stating it as a heuristic.

7.4 The width of a product

Another basic question one can ask about products is: What is the probable difference between the largest coefficient and the smallest coefficient. More precisely, for $\mathbf{a} \in R$, with $\mathbf{a} = a_0 + a_1 x + \cdots + a_{N-1} x^{N-1}$, let

$$\text{width}(\mathbf{a}) = \max a_i - \min a_i. \tag{7.6}$$

Thus, for example,

$$\text{width}(1 + 2x - 3x^2 + x^4 - x^5 + x^6) = 2 - (-3) = 5.$$

Although it is relatively simple to write down heuristics governing the norms of sums and products, it is actually quite difficult to do the same for widths. However, it is quite easy to estimate widths from above. For example, consider the trinary case above, and choose $\mathbf{f}, \mathbf{g}$ at random from $S(d, d)$, $d \leq [N/3]$. Then a coefficient of $\mathbf{f} * \mathbf{g}$ will reach it's maximum possible value if the d 1's of $\mathbf{f}$ are multiplied by the d 1 coefficients of $\mathbf{g}$ and the d -1's of $\mathbf{f}$ are multiplied by the d -1 coefficients of g. This will lead to a maximum coefficient of $2d$. Similarly, the smallest possible coefficient occurs when the 1's correspond to the $-1's$, leading to a coefficient value of $-2d$. Thus the maximal width is $4d$. As

$$\|\mathbf{f} * \mathbf{g}\| \approx \|\mathbf{f}\| \|\mathbf{g}\| = 2d,$$

in this case

$$\text{width}(\mathbf{f} * \mathbf{g}) \leq 4d \approx 2\|\mathbf{f} * \mathbf{g}\|. \tag{7.7}$$

On the other hand, the probability that a randomly chosen $\mathbf{f}, \mathbf{g}$ will actually achieve this width is extremely small. The expected value of the square of a coefficient of $\mathbf{f} * \mathbf{g}$ is approximately

$$\frac{\|\mathbf{f} * \mathbf{g}\|^2}{N}$$

and as the actual coefficient values should drop off exponentially as we depart from the mean it seems not unreasonable to expect that with high probability the width of $\mathbf{f} * \mathbf{g}$ would be bounded above by a constant times $\|\mathbf{f} * \mathbf{g}\|/\sqrt{N}$.

7.5 Polynomial quotient rings with $\mathbb{Z}/q\mathbb{Z}$ coefficients

If q is any fixed positive integer we can define another useful object

$$R_q = \mathbb{Z}_q[x]/(x^N - 1).$$

This is the quotient ring where in addition to reducing exponents modulo N, the coefficients are reduced modulo q, i.e taken from $\mathbb{Z}/q\mathbb{Z}$. For example, setting $q = 3$, if we take the same polynomials $\mathbf{a}, \mathbf{b}, \mathbf{c}$ as above but think of them as being in R_3, then $\mathbf{a} = 1 - 2x + 4x^3 - x^4 = 1 + x + x^3 + 2x^4$, $\mathbf{b} = 3 + 4x - 2x^2 + 5x^3 + 2x^4 = x + x^2 + 2x^3 + 2x^4$ and $\mathbf{c} = x^2 - 3x^3 = x^2$.

It will be convenient, in connection with the our definition of centered norm, to reduce modulo q into an interval centered around zero. (If q is even, this "centering" will not be exact.) Let's assume for the moment that q is odd. Then, instead of identifying every integer with an integer in $\{0, 1, \ldots, q-1\}$ we'll use the $q-1$ integers $\{-(q-1)/2, \ldots, 0, \ldots, (q-1)/2\}$. To reduce into this range, do reduction modulo q as usual by finding the remainder after division by q, and then, for the integers larger than $(q-1)/2$, subtract q. Reducing a polynomial into this range means reducing each coefficient.

Example 7.2. Fix N to be 5. Find the reduction of $\mathbf{a} = 10 + 8x^2 + 21x^3 + 13x^4$ into R_7 using the centered range. Each coefficient must be reduced modulo 7, and identified with an integer in $\{-3, -2, -1, 0, 1, 2, 3\}$. Thus, $10 = 3 \pmod 7, 8 = 1 \pmod 7, 21 = 0 \pmod 7, 13 = 6 = -1 \pmod 7$ and the new polynomial is $3 + x^2 - x^4$.

When q is even, our interval range around 0 won't be exactly centered, and we'll make the convention that the integer $q/2$ will be used, and $-q/2$ will not.

It is important to realize that in the context of R_q, Proposition 7.1 no longer holds. This is because when we reduce coefficients modulo q the notion of length becomes meaningless.

An important new phenomenon that occurs in the context of R_q is that certain polynomials become invertible. For example, take $q = 2$ and $N = 5$ and consider the polynomial $1 + x + x^4$. Multiplying it by $1 + x^2 + x^3$ one obtains

$$(1 + x + x^4) * (1 + x^2 + x^3) = 1 + x + x^4 + x^2 + x^3 + x + x^3 + x^4 + x^2 = 1,$$

after reducing exponents modulo 5 and coefficients modulo 2.

Thus $\mathbf{a} = 1 + x + x^4$ is invertible with inverse $\mathbf{a}^{-1} = 1 + x^2 + x^3$.

In general most, but not all, of the elements of R_q will be invertible. In particular we have:

Proposition 7.3. *Let* $\mathbf{a} \in R_q$*. Then there exists* $\mathbf{b} \in R_q$ *such that* $\mathbf{a} * \mathbf{b} = 1$ *if and only if* $(\mathbf{a}(1), q) = 1$ *The polynomials* $\mathbf{a}, 1 + x + x^2 + \cdots + x^{N-1}$ *are relatively prime when reduced modulo* q*. A necessary condition for this is*

$$(\mathbf{a}(1), q) = 1$$

and this will in fact be sufficient if

$$\frac{x^N - 1}{x - 1} = 1 + x + x^2 + \cdots + x^{N-1}$$

does not factor modulo q *and* $\mathbf{a} \neq 1 + x + x^2 + \cdots + x^{N-1}$.

To understand this Proposition, consider what it means for $\mathbf{a}$ to be invertible in R_q. Saying that there exists $\mathbf{b} \in R_q$ such that $\mathbf{a} * \mathbf{b} = 1$ is equivalent to saying that there exists $\mathbf{b} \in R_q$ and $\mathbf{r} \in R_q$ such that

$$\mathbf{ab} + \mathbf{r}(x^N - 1) \equiv 1 \pmod{q}.$$

Clearly if this is true then $\mathbf{a}, x^N - 1$ are relatively prime modulo q. Conversely, if $\mathbf{a}, x^N - 1$ are relatively prime modulo q then the extended Euclidean algorithm for polynomials with coefficients in q provides a method for determining $\mathbf{b}$, given $\mathbf{a}$ and $x^N - 1$. For example, to locate the inverse in the case of $q = 2, N = 5$ above, we start with $1 + x + x^4$ and $x^5 - 1 \equiv 1 + x^5 \pmod 2$ and perform long division, with coefficients in $\mathbb{Z}/2\mathbb{Z}$ obtaining

$$x^5 + 1 = (x)(x^4 + x + 1) + x^2 + x + 1$$

followed by

$$x^4 + x + 1 = (x^2 + x)(x^2 + x + 1) + 1.$$

Working backwards to write the greatest common divisor, 1, as a linear combination of the initial two polynomials we obtain

$$\begin{aligned} 1 &= (x^4 + x + 1) + (x^2 + x)(x^2 + x + 1) \\ &= (x^4 + x + 1) + (x^5 + 1 + x(x^4 + x + 1)) \\ &= (x^4 + x + 1)(x^3 + x^2 + 1) + (x^5 + 1)(x^2 + x), \end{aligned}$$

which is, in fact, the statement that $1 + x + x^4$ and $1 + x^2 + x^3$ are inverses in R_2 when $N = 5$.

To understand the second statement of the Proposition, note that as $x^N - 1 = (x - 1)(1 + x + x^2 + \cdots + x^{N-1})$, if the second factor on the right is irreducible modulo q and not equal to $\mathbf{a}$ then the statement $\mathbf{a}$ is relatively prime to $x^N - 1$ modulo q is equivalent to the statement that $\mathbf{a}$ is relatively prime to $x - 1$ modulo q, which is in turn equivalent to the statement that $\mathbf{a}$ evaluated at 1 is relatively prime to q.

In practice the extended Euclidean algorithm is easiest to apply if q is a prime. If $q = p^\alpha$ for p a prime and $\alpha \geq 2$ then one finds an inverse modulo p and "lifts" it to a solution modulo p^2, followed by a solution modulo p^4, and so on. Similarly, if $q = q_1 q_2$ with q_1, q_2 relatively prime, one uses a solution modulo q_1 and a solution modulo q_2 to build a solution modulo $q_1 q_2$ via the Chinese Remainder Theorem.

7.6 The NTRU public key cryptosystem

In this section we will show how the ideas introduced earlier in this chapter can be used to create a public key cryptosystem which we will call NTRU. Throughout this section we will fix an N and q and let R, R_q have the definitions given in Sections 7.1 and 7.5. Typically we will take N to be a prime greater than 100 and less than 1000, and q to be an integer between $N/2$ and $2N$ that is relatively prime to N.

We will find it convenient to sometimes think of the same polynomial as an element of R and of R_q. When interpreting $\mathbf{r} \in R$ as an element of R_q we will simply reduce the coefficients of $\mathbf{r}$ modulo q. When interpreting $\mathbf{r} \in R_q$ as an element of R we will mean the polynomial whose coefficients are in $\mathbb{Z}$ and selected from the interval $[-(q-1)/2, (q-1)/2]$ (or $(-q/2, q/2]$ if q is even) as described in Section 7.5. When we do this we will say that we are "lifting" $\mathbf{r}$ from R_q to R.

A vital notion in NTRU is the concept of a "short" polynomial or equivalently a "short" vector. For example, recall the sets of polynomials $S(d_1, d_2)$ described in Definition 7.3 These consisted of all $\mathbf{f} \in R$ such that precisely d_1 coefficients of f equal 1, d_2 coefficients equal -1 and the remainder of the coefficients equal 0. Fix $1 \leq d, d' \leq N/3$ and select $\mathbf{f} \in S(d+1, d)$ and $\mathbf{g} \in S(d', d')$ at random. The centered norm of $\mathbf{f}$ may be computed, as defined in (7.4), and we see that

$$\|\mathbf{f}\| \approx \sqrt{2d+1} \approx \sqrt{2d}.$$

More precisely,

$$\|\mathbf{f}\| = \sqrt{(d+1)(1-1/N)^2 + d(1+1/N)^2 + (N-2d-1)(1/N)^2},$$

but as N will generally be greater than 100, for our purposes $\sqrt{2d}$ will be a close enough approximation. Similarly, $\|\mathbf{g}\| = \sqrt{2d'}$, although this time, as the mean of the coefficients is 0, the statement is exact.

This is a convenient point to introduce the order notation.

Definition. We say that $A(y)$ is on the order of $B(y)$ for large y if there exists an absolute constant $c > 0$ such that for sufficiently large y,

$$|A(y)| < c|B(y)|.$$

If this is the case, we write $A(y) = O(B(y))$

Thus for $\mathbf{f} \in S(d+1, d)$ and $\mathbf{g} \in S(d', d')$,

$$\|\mathbf{f}\| = O(\sqrt{N}) \quad \text{and} \quad \|\mathbf{g}\| = O(\sqrt{N})$$

Also, for q, N in the ranges we have specified, $q = O(N)$. In general, we make the follwing definition:

Definition. A polynomial or vector$\|\mathbf{f}\|$ is said to be short if $\|\mathbf{f}\| = O(\sqrt{N})$.

Thus $S(d+1, d)$ and $S(d', d')$ supply us with a large class of short vectors.

Now, let's choose an $\mathbf{f} \in S(d+1, d)$ at random. Although $\mathbf{f} \in R$ we may also regard it as being in R_q. As $\mathbf{f}(1) = d + 1 - d = 1$, if $1 + x + \cdots + x^{N-1}$ does not factor modulo q then, by Proposition 7.3, $\mathbf{f}$ will be invertible in R_q. Assume this is the case and denote by $\mathbf{f}_q$ the inverse of f in R_q. Although the coefficients of $\mathbf{f}$ have a very special form, the coefficients of $\mathbf{f}_q$ will appear to be random and uniformly distributed modulo q. (There is no known "proof" of this fact, but experimental evidence supports it.).

For example, consider a small example, with $N = 17$ and $q = 31$. In this case $1 + x + \cdots + x^{N-1}$ is irreducible modulo q. Set $d = 5$. Choosing a random $\mathbf{f} \in S(6, 5)$ we get

$$\mathbf{f} = -x^{15} + x^{14} - x^{11} - x^{10} - x^8 + x^7 + x^6 + x^4 - x^2 + x + 1.$$

Computing its inverse modulo 31 we obtain

$$\begin{aligned}\mathbf{f}_q =& 14x^{16} + 7x^{15} - 11x^{14} + 3x^{13} - 12x^{12} - 5x^{11} + 13x^{10} + 4x^9 \\ & - 4x^8 + 6x^7 + 10x^6 + 7x^5 - 15x^4 - 11x^3 - 4x^2 - 11x + 10.\end{aligned}$$

Notice that the coefficients of $\mathbf{f}_q$ do indeed appear to be random and uniformly distributed modulo 31.

We now take $d' = d$ and choose $\mathbf{g}$ at random from $S(d, d)$. For example, in the case $N = 17, q = 31$ above we might obtain

$$\mathbf{g} = -x^{16} - x^{14} - x^{13} + x^9 + x^8 - x^7 - x^6 + x^5 + x^4 + x^2.$$

Our next step is to multiply $\mathbf{f}_q$ and $\mathbf{g}$ in R_q, setting $\mathbf{h} = \mathbf{f}_q * \mathbf{g}$. Then by the definition of convolution multiplication in (7.1) each coefficient of $\mathbf{h}$ will be a sum of $2d$ distinct coefficients of $\mathbf{f}_q$ reduced modulo q. Of these d will be multiplied by 1 and d will be multiplied by -1. As the coefficients of $\mathbf{f}_q$ appear to be randomly and uniformly distributed modulo q it is reasonable to expect that the coefficients of $\mathbf{h}$ will also appear to be randomly and uniformly distributed modulo q. For example, in the case we have chosen we obtain

$$\begin{aligned}\mathbf{h} =& - 7x^{16} - 3x^{15} + 14x^{14} - 11x^{13} + 10x^{12} - 2x^{11} + 6x^{10} + 8x^9 \\ & + 5x^8 + 10x^7 + 12x^6 + 11x^5 + 6x^4 - 14x^2 - 10x - 4.\end{aligned}$$

Although the coefficients of $\mathbf{h}$ appear random modulo q, there is an important hidden relationship between $\mathbf{h}$ and $\mathbf{f}$. This is the fact that, for general

N, q, in R_q.

$$\mathbf{f} * \mathbf{h} = \mathbf{f} * (\mathbf{f}_q * \mathbf{g}) = (\mathbf{f} * \mathbf{f}_q) * \mathbf{g} = 1 * \mathbf{g} = \mathbf{g}.$$

Remarkably, this R_q relationship can be translated into a statement that holds in R. To see this, lift $\mathbf{f}, \mathbf{h}$ to elements of R and compute $\mathbf{f} * \mathbf{h}$. Reduce the coefficients of $\mathbf{f} * \mathbf{h}$ into the interval $[-(q-1)/2, (q-1)/2]$ and then lift back to $\mathbb{Z}$. The resulting product is then congruent to $\mathbf{g}$ modulo q and also has all its coefficients lying in the interval $[-(q-1)/2, (q-1)/2]$ As the coefficients of$\mathbf{g}$ are taken from the set $\{1, 0, -1\}$ it follows that this reduced product lifted to $\mathbb{Z}$ must exactly equal $\mathbf{g}$. Thus in the particular example we took above

$$\begin{aligned} \mathbf{f} * \mathbf{h} = &-32x^{16} - x^{14} + 30x^{13} + 31x^{12} + 62x^{11} + 31x^{10} + 32x^9 \\ &- 30x^8 - x^7 - x^6 - 30x^5 + 32x^4 - 31x^3 - 30x^2 - 31 \\ &\equiv -x^{16} - x^{14} - x^{13} + x^9 + x^8 - x^7 - x^6 + x^5 + x^4 + x^2 \quad \mathrm{mod}\ 31 \\ &= \mathbf{g}. \end{aligned}$$

The above discussion can be summarized as follows:

1. Fix $1 \le d \le N/3$ and select $\mathbf{f} \in S(d+1, d)$ and $\mathbf{g} \in S(d, d)$ at random.
2. Assuming that $\mathbf{f}$ is invertible, compute $\mathbf{f}_q = \mathbf{f}^{-1}$ and $\mathbf{h} = \mathbf{f}_q * \mathbf{g}$, both elements of R_q. The coefficients of $\mathbf{f}_q$ and $\mathbf{h}$ should appear random and unifirm modulo q.
3. Compute $\mathbf{f} * \mathbf{h}$ in R_q and lift to a polynomial $\mathbf{a}$ of R with coefficients in the interval $[-(q-1)/2, (q-1)/2]$.
4. It should be true that $\mathbf{a} = \mathbf{g}$.

We are now almost ready to describe the NTRU Hard Problem. Recall that $\mathbf{f}$ is called a short vector if it satisfies $\|\mathbf{f}\| = O(\sqrt{N})$. A randomly chosen vector from R_q will have its coefficients randomly and uniformly distributed between $-q/2$ and $q/2$. One might ask what the norm squared of such a vector might be expected to be. The answer is easily seen to be approximated by the integral

$$\frac{N}{q} \int_{-q/2}^{q/2} x^2 dx = \frac{Nq^2}{12}$$

Thus we would expect an element chosen at random from R_q and lifted back to R to have norm about $q\sqrt{N/12}$. As $q = O(N)$, this implies that a random lifted vector will have norm $O(N^{3/2})$. Notice that this is a factor of N longer than a short vector.

In the scenario above $\mathbf{h}$ is apparently random, with $\|\mathbf{h}\| = O(N^{3/2})$. There exists a special vector $\mathbf{f}$ which is short, and has the property that $\mathbf{f} * \mathbf{h}$ is also short when lifted to R. If one were simply given $\mathbf{h}$, how hard would it be to find $\mathbf{f}$? Thus we have

The NTRU Hard Problem: Given $\mathbf{h} = \mathbf{f}^{-1} * \mathbf{g} \in R_q$, with $\mathbf{f}, \mathbf{g}$ short, recover $\mathbf{f}, \mathbf{g}$. Alternatively, given $\mathbf{h}$, recover any $\mathbf{f}'$ which simultaneously satisfies $\|\mathbf{f}'\| = O(\sqrt{N})$ and $\|\mathbf{f}' * \mathbf{h}\| = O(\sqrt{N})$.

We call this a hard problem for several reasons. For one thing, this specific problem has been studied by the mathematical and cryptographic communities since 1996 without significant advances being made. We will also see shortly that it's solution can be tied to the solution of a certain class of shortest vector problems in high dimensional lattices. It is possible that this specific class of lattice problems can be solved without solving the general class, but to this date there has been no indication that this class is easier than the general class.

7.6.1 A description of NTRU

We now describe how a public key cryptosystem can be based on this hard problem. Fix p to be a small prime, relatively prime to q. The quantities p, q, N, d, d' will be system parameters and public knowledge.

Suppose Alice wishes to create a public, private key pair. She chooses $\mathbf{f} \in S(d+1, d), \mathbf{g} \in S(d', d')$ at random and checks to make sure that $\mathbf{f}$ is invertible in R_q and R_p. If not, she re-chooses $\mathbf{f}$ until it is. She then computes $\mathbf{f}_q$, the inverse of $\mathbf{f}$ in R_q and sets $\mathbf{h} = \mathbf{f}_q * \mathbf{g}$ as above. The polynomial $\mathbf{h}$ becomes her public key, and the polynomial $\mathbf{f}$ becomes her private key. Alice also computes and stores $\mathbf{f}_p$, the inverse of $\mathbf{f}$ in R_p.

If Bob wishes to encrypt and send a message M to Alice that only she can decrypt, he first uses some encoding process to transform it into a polynomial $\mathbf{m}$ with coefficients taken modulo p. For example, if $p = 2$ the coefficients would be integers of the form 0 or 1; If $p = 3$ the coefficients would be integers of the form $-1, 0, 1$, and so on. Longer messages would be broken into a sequence of polynomials $m_1, m_2, \ldots$.

Bob then chooses $\mathbf{r} \in S(d', d')$ at random and constructs the encrypted message

$$\mathbf{e} \equiv p\mathbf{h} * \mathbf{r} + \mathbf{m} \pmod{q}.$$

The coefficients are reduced modulo q and we think of $\mathbf{e}$ as being an element of R_q.

Bob sends $\mathbf{e}$ to Alice, who would now like to decrypt it and recover $\mathbf{m}$. Alice uses her secret knowledge of $\mathbf{f}$ to compute the following element of R_q:

$$\mathbf{a} \equiv \mathbf{f} * \mathbf{e} \equiv \mathbf{f} * (p\mathbf{h} * \mathbf{r}) + \mathbf{f} * \mathbf{m} \equiv p\mathbf{g} * \mathbf{r} + \mathbf{f} * \mathbf{m} \pmod{q}. \tag{7.8}$$

By the additive and quasi-multiplicative properties described in Proposition 7.1 and in Section **??**, when normalized to the proper interval modulo q the coefficients on the right hand side of (7.8) and consequently of $\mathbf{a}$ should not be too large.

In particular, the very largest that a positive coefficient of $\mathbf{g} * \mathbf{a}$ can get is $2d'$, the largest a positive coefficient of $\mathbf{f} * \mathbf{m}$ can get is $(2d+1)p/2$ and thus the largest that a positive coefficient of $\mathbf{a}$ can get is $2d'p + (2d+1)p/2$. The bound for negative coefficients is the same, and so a coefficient a_i of $\mathbf{a}$ must satisfy

$$|a_i| \leq 2d'p + (2d+1)p/2.$$

Because of this, if we require

$$q > 4d'p + (2d+1)p$$

then a lifting of $\mathbf{a}$ to R, with the coefficients in their usual range between $-q/2$ and $q/2$ will lead to an exact recovery of $p\mathbf{g} * \mathbf{r} + \mathbf{f} * \mathbf{m}$ over $\mathbb{Z}$ rather than merely the recovery of something congruent to it modulo q.

Having recovered $p\mathbf{g} * \mathbf{r} + \mathbf{f} * \mathbf{m}$ over $\mathbb{Z}$ Alice now reduces modulo p and recovers $\mathbf{f} * \mathbf{m}$ as an element of R_p. Multiplying by $\mathbf{f}_p$ she recovers

$$m \equiv \mathbf{f}_p * (\mathbf{f} * \mathbf{m}) \pmod{p}$$

which is the decrypted message.

7.6.2 An $N = 11, q = 127$ example of NTRU

Consider the special case $N = 11$ and $q = 127, p = 3$. Choosing a trinary $\mathbf{f}, \mathbf{g}$ at random we have

$$\mathbf{f} = -x^{10} - x^9 + x^8 + x^7 + x^5 - x^2 + x, \quad \mathbf{g} = x^9 + x^7 + x^6 - x^4 - x^3 - x.$$

Computing the inverse of $\mathbf{f}$ modulo q we obtain

$$\mathbf{f}_q = 90x^{10}+14x^9+91x^8+18x^7+119x^6+9x^5+121x^4+25x^3+96x^2+117x+63$$

and computing the inverse modulo q gives

$$\mathbf{f}_p = 2x^{10} + x^7 + x^6 + x^5 + x^4 + x^3 + 2x^2 + 2x + 2.$$

Taking the convolution product modulo q, the public key is

$$\mathbf{h} = \mathbf{f}_q * g = 112x^{10}+92x^9+28x^8+56x^7+114x^6+42x^5+19x^4+123x^3+37x^2+112x+27.$$

Notice that the coefficients of $\mathbf{f}_q$ and $\mathbf{h}$ do in fact appear random modulo q.

Now we choose a trinary message $\mathbf{m}$:

$$\mathbf{m} = x^{10} - x^9 + x^8 - x^5 + x^4 - x^2$$

and pick the ephemeral trinary key $\mathbf{r}$ at random:

$$\mathbf{r} = -x^9 + x^6 + x^5 + x^4 - x^3 - x.$$

Thus the encrypted message is the element of R_q:

$$\mathbf{e} = 3\mathbf{rh}+\mathbf{m} = 44x^9+35x^8+14x^7+76x^6+35x^5+28x^4+124x^3+52x^2+122x+105.$$

To decrypt we multiply by $\mathbf{f}$ and reduce modulo q to obtain

$$\mathbf{a} = 120x^{10}+118x^9+118x^8+119x^7+124x^6+125x^5+10x^4+x^3+11x^2+6x+10.$$

We then subtract q from coefficients larger than $q/2$ to translate into the range $(-q/2.q/2)$. This yields the lifting of $\mathbf{a}$ to $\mathbb{Z}$:

$$-7x^{10} - 9x^9 - 9x^8 - 8x^7 - 3x^6 - 2x^5 + 10x^4 + x^3 + 11x^2 + 6x + 10.$$

This should exactly equal the polynomial in R equal to $3\mathbf{rg}+\mathbf{fm}$, and in fact we can verify that

$$3\mathbf{rg}+\mathbf{fm} = -7x^{10} - 9x^9 - 9x^8 - 8x^7 - 3x^6 - 2x^5 + 10x^4 + x^3 + 11x^2 + 6x + 10.$$

Continuing, we now multiply by $\mathbf{f}_p$ and reduce modulo p, obtaining

$$x^{10} + 2x^9 + x^8 + 2x^5 + x^4 + 2x^2$$

which does in fact equal $\mathbf{m}$ after the coefficients are translated modulo p into the range $[-1, 1]$.

7.6.3 Combinatorial security of NTRU

7.6.4 Lattice security of NTRU

Exercises

7.1. Compute the polynomial convolution product $c = a * b$ using the given value of N.

(a) $a = -1 + 4X + 5X^2, \qquad b = -1 - 3X - 2X^2, \qquad N = 3.$

(b) $a = 2 - X + 3X^3 - 3X^4, \qquad b = 1 - 3X^2 - 3X^3 - X^4, \qquad N = 5.$

(c) $a = X + X^2 + X^3, \qquad b = 1 + X + X^5, \qquad N = 6.$

(d) $a = X + X^2 + X^3 + X^4 + X^6 + X^7 + X^9, \qquad b = X^2 + X^3 + X^6 + X^8, \qquad N = 10.$

7.2. Compute the polynomial convolution product $c = a * b$ modulo q using the given values of q and N.

(a) $a = 1 + X, \qquad b = -5 + 4X + 2X^2, \qquad q = 7, \qquad N = 3.$

(b) $a = 2 + 2X - 2X^2 + X^3 - 2X^4, \qquad b = -1 + 3X - 3X^2 - 3X^3 - 3X^4, \qquad q = 4, \qquad N = 5.$

(c) $a = X + X^3, \qquad b = X + X^2 + X^4 + X^6, \qquad q = 3, \qquad N = 7.$

(d) $a = X^2 + X^5 + X^7 + X^8 + X^9, \qquad b = 1 + X + X^3 + X^4 + X^5 + X^7 + X^8 + X^9, \qquad q = 2, \qquad N = 10.$

7.3. Compute the inverse of $x^4 + x^3 - x^2 - x + 1$ in R_3

7.4. Construct your own NTRU public and private key set, using $N = 5, q = 127, p = 3, d = 2$. Verify that it works by encrypting and decrypting a message. (The parameters are small enough that this can be done by hand, or by using a computer algebra package.)

7.5. Alice and Bob agree to communicate using NTRU with $N = 7$ and $q = 37$. Alice's private key is

$$f = X + X^3 + X^6 \qquad \text{and} \qquad F = 1 + X + X^4 + X^5 + X^6.$$

(You can check that $f * F \equiv 1 \pmod{2}$.) Alice receives the ciphertext $c = 1 + 3X + 3X^2 + 4X^3 + 4X^4 + X^5 + 35X^6$ from Bob. Decipher the message and find the plaintext.

7.6. Alice and Bob plan to communicate using NTRU with $N = 7$ and $q = 29$. Alice's public key is $h = 23 + 23X + 23X^2 + 24X^3 + 23X^4 + 24X^5 + 23X^6$. Bob wants to send Alice the plaintext message $m = 1 + X^5$ and he decides to use the ephemeral key $r = 1 + X + X^3 + X^6$.

(a) What ciphertext does Bob send to Alice?

(b) Alice's secret key is $f = 1 + X + X^2 + X^4 + X^5$ and $F = 1 + X^5 + X^6$. Check your answer in (a) by using f and F to decrypt the message.

7.7. Prove that the centered norm satisfies triangle inequality.

7.8. Show that $\|\mathbf{a}\|_{\text{cent}} = 0$ implies that $\mathbf{a} = (c, c, \ldots, c)$ for some c.

7.9. Choose $\mathbf{f}, \mathbf{g}$ at random from $S(d, d)$, $d = [N/3]$. Suppose that when constructing the coefficients $c_k = \sum_{i+j \equiv k \pmod{N}} f_i g_j$ we think of the f_i, g_j as independent random variables with probability $1/3$ of taking the values $1, 0, -1$. Compute the probability the the maximum width of (7.7) is achieved. (In reality this model is not quite correct, but it provides a reasonable approximation.)

7.10. Let $\mathbf{a} = 12 + 19x^2 + 22x^3 + 5x^4 + 11x^5$ and $\mathbf{b} = 17 + 24x + 30x^2 + 14x^4 + 68x^6$. If $N = 5$ and $q = 8$, express these polynomial as elements of R_q and compute their norms, the convolution product, and the norm of the convolution.

7.11. Let $N = 5$ and $q = 3$. The inverse of one of the two polynomials $1 + x^2 + x^3$ and $1 + x^2 - x^3$ exists in R_3. Find the inverse that exists, and explain why the other doesn't exist.

Chapter 8

Signatures, Hash Functions, and the Science of Cryptology

8.1 Digital signatures

*** SECTION TO BE WRITTEN ***

8.2 RSA digital signatures

*** SECTION TO BE WRITTEN ***

8.3 Discrete logarithm digital signatures and DSA

*** SECTION TO BE WRITTEN ***

8.4 NTRU digital signatures

*** SECTION TO BE WRITTEN ***

8.5 Hash functions

*** SECTION TO BE WRITTEN ***

8.6 Random numbers and pseudorandom number generators

*** SECTION TO BE WRITTEN ***

As noted in Section 1.6 (page 41) *** (check that that section wasn't split into two) ***, ad hoc constructions of pseudorandom number generators tend to be much more efficient than those constructed via classical hard mathematical problems. However, the following simple example of a mathematically based pseudorandom number generator is farily efficient.

*** Jeff wrote this. It could be placed in the quadartic residue section, but maybe best to leave it in the PRNG section as an example. It needs some editting. ***

Let k, the key, be a prime number on the order of 160 bits. For $n = 1, 2, 3, \ldots$ let p_n denote the n^{th} odd prime. So $p_1 = 3, p_2 = 5, p_3 = 7p_4 = 11, \ldots$.... Define the function $F(k, n)$ by $F(k, n) = 1$ if $x^2 \equiv k \pmod{p}_n$ has a solution and $F(k, n) = 0$ otherwise. We will see in Section *** that $F(k, n)$ is essentially the Legendre or quadratic residue symbol (k/p_n). This is a very familiar mathematical object which can be efficiently computed by means of the quadratic reciprocity law, and whose properties are well understood. In particular, it is a consequence of the Generalized Riemann Hypothesis that for large k the values of $F(k, n)$ closely resemble a random distribution. Also, it is generally accepted that k can not be recovered from a list of T values of (k/p_n) until T is on the order of $\sqrt{k}$.

8.7 Identification schemes

*** SECTION TO BE WRITTEN ***

8.8 Zero-knowledge proofs

*** SECTION TO BE WRITTEN ***

Exercises

Section 8.1. Digital signatures

Section 8.2. RSA digital signatures

Section 8.3. Discrete logarithm digital signatures and DSA

Section 8.4. NTRU digital signatures

Section 8.5. Hash functions

Section 8.6. Random numbers and pseudorandom number generators

Section 8.7. Identification schemes

Section 8.8. Zero-knowledge proofs

Chapter 9

Practical Cryptology and the Search for Certainty in an Uncertain World

9.1 Building protocols from cryptographic primitives

*** SECTION TO BE WRITTEN ***

9.2 Quantum cryptography and quantum computing

*** SECTION TO BE WRITTEN ***

9.3 *** more stuff ***

Exercises

Section 9.1. Building protocols from cryptographic primitives

Section 9.2. Quantum cryptography and quantum computing

Section 9.3. *** more stuff ***

Notation

$\lfloor x \rfloor$ The *greatest integer* in x, also sometimes called the *floor* of x. It is the largest integer n satisfying $n \leq x$.

$b|a$ b divides a, which by definition means that there is an integer c so that $a = bc$. We also say that a is divisible by b.

$\equiv$ congruence relation. The congruence $a \equiv b \pmod{m}$ means that $a - b$ is divisible by m.

$\mathbb{Z}/m\mathbb{Z}$ the ring of integers modulo m.

$\mathbb{Z}/p\mathbb{Z}$ the finite field with p elements (p prime).

$\mathbb{F}_p$ an alternative notation for the finite field $\mathbb{Z}/p\mathbb{Z}$.

$\mathbb{F}_{p^k}$ the finite field with p^k elements.

$\mathrm{GF}(p^k)$ an alternative notation for $\mathbb{F}_{p^k}$. The letters "GF" are an abbreviation for "Galois Field."

R^* the group of invertible elements (units) in the ring R. An alternative notation sometimes used in cryptography is $U(R)$.

Bibliography

[1] M. Agrawal, N. Kayal, and N. Saxena. Primes are in p. preprint, August 6, 2002. `http://www.cse.iitk.ac.in/primality.pdf`.

[2] Eric Bach. Explicit bounds for primality testing and related problems. *Math. Comp.*, 55(191):355–380, 1990.

[3] I. F. Blake, G. Seroussi, and N. P. Smart. *Elliptic curves in cryptography*, volume 265 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, Cambridge, 2000.

[4] Johannes Blömer and Alexander May. Low secret exponent RSA revisited. In *Cryptography and lattices (Providence, RI, 2001)*, volume 2146 of *Lecture Notes in Comput. Sci.*, pages 4–19. Springer, Berlin, 2001.

[5] Dan Boneh and Glenn Durfee. Cryptanalysis of RSA with private key d less than $N^{0.292}$. In *Advances in cryptology—EUROCRYPT '99 (Prague)*, volume 1592 of *Lecture Notes in Comput. Sci.*, pages 1–11. Springer, Berlin, 1999.

[6] Dan Boneh and Glenn Durfee. Cryptanalysis of RSA with private key d less than $N^{0.292}$. *IEEE Trans. Inform. Theory*, 46(4):1339–1349, 2000.

[7] Dan Boneh and Ramarathnam Venkatesan. Breaking RSA may not be equivalent to factoring (extended abstract). In *Advances in cryptology—EUROCRYPT '98 (Espoo)*, volume 1403 of *Lecture Notes in Comput. Sci.*, pages 59–71. Springer, Berlin, 1998.

[8] Richard P. Brent. An improved Monte Carlo factorization algorithm. *BIT*, 20(2):176–184, 1980.

[9] E. R. Canfield, Paul Erdős, and Carl Pomerance. On a problem of Oppenheim concerning "factorisatio numerorum". *J. Number Theory*, 17(1):1–28, 1983.

[10] J. W. S. Cassels. *Lectures on elliptic curves*, volume 24 of *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, 1991.

[11] Henri Cohen. *A course in computational algebraic number theory*, volume 138 of *Graduate Texts in Mathematics*. Springer-Verlag, Berlin, 1993.

[12] Don Coppersmith. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Math. Comp.*, 62(205):333–350, 1994.

[13] Richard Crandall and Carl Pomerance. *Prime numbers*. Springer-Verlag, New York, 2001.

[14] H. Davenport. *The higher arithmetic*. Cambridge University Press, Cambridge, 1999.

[15] Whitfield Diffie. The first ten years of public key cryptology. In *Contemporary cryptology*, pages 135–175. IEEE, New York, 1992.

[16] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, IT-22(6):644–654, 1976.

[17] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inform. Theory*, 31(4):469–472, 1985.

[18] James Ellis. The story of non-secret encryption. released by CSEG 1997, 1987 (released 1997). `http://www.cesg.gov.uk/ellisdox.ps`.

[19] Mireille Fouquet, Pierrick Gaudry, and Robert Harley. An extension of Satoh's algorithm and its implementation. *J. Ramanujan Math. Soc.*, 15(4):281–318, 2000.

[20] Geoffrey R. Grimmett and David R. Stirzaker. *Probability and random processes*. Oxford University Press, New York, 3rd edition, 2001.

[21] G. H. Hardy and E. M. Wright. *An introduction to the theory of numbers*. The Clarendon Press Oxford University Press, New York, 1979.

[22] Kenneth Ireland and Michael Rosen. *A classical introduction to modern number theory*, volume 84 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1990.

[23] Lenstra H. W. Jr. and Pomerance C. Primality testing with gaussian periods. preprint, March 2003.

[24] David Kahn. *The Codebreakers: The Story of Secret Writing*. Scribner Book Company, 1996.

[25] Anthony W. Knapp. *Elliptic curves*, volume 40 of *Mathematical Notes*. Princeton University Press, Princeton, NJ, 1992.

[26] Donald Knuth. *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*. Addison-Wesley, Reading, Mass., 2nd edition, 1981.

[27] Neal Koblitz. *Algebraic aspects of cryptography*, volume 3 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 1998.

[28] B. A. LaMacchia and A. M. Odlyzko. Solving large sparse linear systems over finite fields. In *Advances in cryptology—CRYPTO '90 (Santa Barbara, Calif., 1990)*, Lecture Notes in Comput. Sci. Springer, Berlin, 1990.

[29] Serge Lang. *Elliptic curves: Diophantine analysis*, volume 231 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1978.

[30] Serge Lang. *Elliptic functions*, volume 112 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2nd edition, 1987. With an appendix by J. Tate.

[31] H. W. Lenstra, Jr. Factoring integers with elliptic curves. *Ann. of Math. (2)*, 126(3):649–673, 1987.

[32] Alfred Menezes. *Elliptic curve public key cryptosystems.* The Kluwer International Series in Engineering and Computer Science, 234. Kluwer Academic Publishers, Boston, MA, 1993.

[33] Ralph C. Merkle. Secure communications over insecure channels. In *Secure communications and asymmetric cryptosystems*, volume 69 of *AAAS Sel. Sympos. Ser.*, pages 181–196. Westview, Boulder, CO, 1982.

[34] Ralph C. Merkle and Martin E. Hellman. Hiding information and signatures in trapdoor knapsacks. In *Secure communications and asymmetric cryptosystems*, volume 69 of *AAAS Sel. Sympos. Ser.*, pages 197–215. Westview, Boulder, CO, 1982.

[35] Gary L. Miller. Riemann's hypothesis and tests for primality. *J. Comput. System Sci.*, 13(3):300–317, 1976. Working papers presented at the ACM-SIGACT Symposium on the Theory of Computing (Albuquerque, N.M., 1975).

[36] Peter L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Math. Comp.*, 48(177):243–264, 1987.

[37] NBS–AES. Advanced Encryption Standard (AES). FIPS Publication 197, National Bureau of Standards, 2001. `http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`.

[38] NBS–DES. Data Encryption Standard (DES). FIPS Publication 46-3, National Bureau of Standards, 1999. http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf.

[39] Ivan Niven, Herbert S. Zuckerman, and Hugh L. Montgomery. *An introduction to the theory of numbers.* John Wiley & Sons Inc., New York, 1991.

[40] Stephen C. Pohlig and Martin E. Hellman. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. *IEEE Trans. Information Theory*, IT-24(1):106–110, 1978.

[41] Carl Pomerance. A tale of two sieves. *Notices Amer. Math. Soc.*, 43(12):1473–1485, 1996.

[42] Hans Riesel. *Prime numbers and computer methods for factorization*, volume 126 of *Progress in Mathematics.* Birkhäuser Boston Inc., Boston, MA, 1994.

[43] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Comm. ACM*, 21(2):120–126, 1978.

[44] Kenneth H. Rosen. *Elementary number theory and its applications.* Addison-Wesley, Reading, MA, 4th edition, 2000.

[45] Sheldon Ross. *A first course in probability.* Prentice Hall, 6th edition, 2001.

[46] Takakazu Satoh. The canonical lift of an ordinary elliptic curve over a finite field and its point counting. *J. Ramanujan Math. Soc.*, 15(4):247–270, 2000.

[47] René Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Math. Comp.*, 44(170):483–494, 1985.

[48] René Schoof. Counting points on elliptic curves over finite fields. *J. Théor. Nombres Bordeaux*, 7(1):219–254, 1995. Les Dix-huitièmes Journées Arithmétiques (Bordeaux, 1993).

[49] Adi Shamir. A polynomial-time algorithm for breaking the basic Merkle-Hellman cryptosystem. *IEEE Trans. Inform. Theory*, 30(5):699–704, 1984.

[50] C. E. Shannon. A mathematical theory of communication. *Bell System Tech. J.*, 27:379–423, 623–656, 1948.

[51] C. E. Shannon. Communication theory of secrecy systems. *Bell System Tech. J.*, 28:656–715, 1949.

[52] Victor Shoup. *A Computational Introduction to Number Theory and Algebra.* Cambridge University Press, 2005. http://shoup.net/ntb/ntb-b5.pdf.

[53] Joseph H. Silverman. *The arithmetic of elliptic curves*, volume 106 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1986.

[54] Joseph H. Silverman. *Advanced topics in the arithmetic of elliptic curves*, volume 151 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1994.

[55] Joseph H. Silverman. *A Friendly Introduction to Number Theory*. Prentice Hall, Upper Saddle River, NJ, 2nd edition, 2001.

[56] Joseph H. Silverman. Elliptic curves and cryptography. Proceedings of Symposia in Applied Mathematics. American Mathematical Society, Providence, RI, to appear.

[57] Joseph H. Silverman and John Tate. *Rational points on elliptic curves*. Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1992.

[58] Berit Skjernaa. Satoh's algorithm in characteristic 2. *Math. Comp.*, 72(241):477–487 (electronic), 2003.

[59] Edlyn Teske. Speeding up Pollard's rho method for computing discrete logarithms. In *Algorithmic number theory (Portland, OR, 1998)*, volume 1423 of *Lecture Notes in Comput. Sci.*, pages 541–554. Springer, Berlin, 1998.

[60] Edlyn Teske. Square-root algorithms for the discrete logarithm problem (a survey). In *Public-key cryptography and computational number theory (Warsaw, 2000)*, pages 283–301. de Gruyter, Berlin, 2001.

[61] Michael J. Wiener. Cryptanalysis of short RSA secret exponents. *IEEE Trans. Inform. Theory*, 36(3):553–558, 1990.

[62] Song Y. Yan. *Primality testing and integer factorization in public-key cryptography*, volume 11 of *Advances in Information Security*. Kluwer Academic Publishers, Boston, MA, 2004.