Universitext

Robert G. Underwood

Cryptography for Secure Encryption

Springer

Universitext

Universitext is a series of textbooks that presents material from a wide variety of mathematical disciplines at master's level and beyond. The books, often well class-tested by their author, may have an informal, personal even experimental approach to their subject matter. Some of the most successful and established books in the series have evolved through several editions, always following the evolution of teaching curricula, into very polished texts.

Thus as research topics trickle down into graduate-level teaching, first textbooks written for new, cutting-edge courses may make their way into *Universitext*.

Robert G. Underwood

Cryptography for Secure Encryption

Springer

Robert G. Underwood
Auburn University at Montgomery
Montgomery, AL, USA

ISSN 0172-5939 ISSN 2191-6675 (electronic)
Universitext
ISBN 978-3-030-97901-0 ISBN 978-3-030-97902-7 (eBook)
https://doi.org/10.1007/978-3-030-97902-7

Mathematics Subject Classification: 94A60, 68P25, 20K01, 11T71, 14G50

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

Cryptography for Secure Encryption is a book about how concepts in mathematics are used to develop the basic components of cryptography. At times, the mathematical foundations are as important as the cryptography. The text assumes basic knowledge of calculus and linear algebra.

The book can be used for a course in cryptography in a master's degree program in mathematics, computer science, or cyber security. It can also be used in a senior-level undergraduate course in cryptography for mathematics or computer science majors.

In the first part of the book (Chapters 2–7), we introduce concepts from probability theory, information theory, complexity theory, modern algebra, and number theory that will be used later in our study of cryptography.

In the second part of the book (Chapters 8–14), we develop the basic notions of cryptography.

We outline the contents of the second part of the book in detail. In Chapter 8, we present the major symmetric key cryptosystems, including the simple substitution cryptosystem, the affine cipher, the Hill 2×2 cipher, the Vigenère cipher, the Vernam cipher, and the stream cipher. We discuss Feistel-type block ciphers such as DES and AES. Methods of cryptanalysis such as frequency analysis and the Kasiski method are also given.

In Chapter 9, we introduce public key cryptography, including the two most important public key cryptosystems: RSA and ElGamal. We discuss standard attacks on these cryptosystems.

In Chapter 10, we discuss digital signature schemes and hash functions.

In Chapter 11, we consider the construction of bit generators for use in stream ciphers. We give a practical definition of "randomness" of a sequence of bits (the next-bit test) and show that under the discrete logarithm assumption, the Blum-Micali bit generator is pseudorandom. Moreover, under the quadratic residue assumption, we prove that the Blum-Blum-Shub bit generator is pseudorandom.

In Chapter 12, we address the problem of distribution of keys. We introduce the Diffie-Hellman key exchange protocol (DHKEP) and discuss standard attacks on the DHKEP including the man-in-the-middle attack, baby-step/giant-step, and the index calculus. Index calculus is an attack tailored to the choice of group $G = U(\mathbb{Z}_p)$, p prime, in the DHKEP.

In Chapter 13, we introduce elliptic curves and the elliptic curve group. An elliptic curve over a field K is the set of points satisfying an equation of the form

$$y^2 = x^3 + ax + b,$$

where the curve is smooth, that is, the cubic has non-zero elliptic discriminant. The set of points on an elliptic curve, together with the point at infinity, is endowed with a binary operation (point addition) to yield the elliptic curve group $E(K)$.

A cyclic subgroup of $E(K)$ is used in the Diffie-Hellman key exchange protocol in place of $U(\mathbb{Z}_p)$ to define the elliptic curve key exchange protocol (ECKEP). The ECKEP is more secure than the ordinary Diffie-Hellman protocol since the index calculus attack cannot be applied to an elliptic curve group.

In Chapter 14, we consider the case where the curve $y^2 = x^3 + ax + b$ is not smooth. We explore the connection between the group of points on such curves (the non-singular points $E_{ns}(K)$) and another group of points $G_c(K)$, which generalizes the circle group. This is an active area of research which may provide insight into the nature of point addition in the smooth case.

Each chapter contains a set of exercises of various degrees of difficulty that help summarize and review the main ideas of the chapter.

I have found that cryptographic algorithms and protocols provide for good programming problems. At various places in the text, I have included `GAP` code. `GAP` is a powerful computer algebra system, distributed freely, and available at

`gap-system.org`

Course Outlines

Cryptography for Secure Encryption contains more material than can be covered in a one-semester course. Instructors can choose topics that reflect their requirements and interests, and the level of preparation of their students.

For a one-semester course in cryptography for undergraduates, a suggested course outline is:

Chapter 1: Sections 1.1, 1.2, 1.3.
Chapter 2: Sections 2.1, 2.2, 2.3, 2.4.
Chapter 3: Sections 3.1, 3.2, 3.4.
Chapter 4: Sections 4.1, 4.2, 4.3, 4.4.
Chapter 5: Sections 5.1, 5.2, 5.3.
Chapter 6: Sections 6.1, 6.2, 6.3, 6.4.
Chapter 7: Section 7.4.
Chapter 8: Sections 8.1, 8.2, 8.3, 8.4, 8.5.
Chapter 9: Sections 9.1, 9.2, 9.3, 9.4, 9.6.

For a one-semester graduate level course, a suggested course outline is:

Chapters 1–4 as above, but assigned as reading or light coverage in class.
Chapter 5: Sections 5.1, 5.2, 5.3.
Chapter 6: Sections 6.1, 6.2, 6.3, 6.4.
Chapter 7: Section 7.4.
Chapter 8: Sections 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.7.
Chapter 9: Sections 9.1, 9.2, 9.3, 9.4, 9.5, 9.6.
Chapter 10: Sections 10.1, 10.2.
Chapter 11: Sections 11.1, 11.2, 11.3, 11.4.
Chapter 12: Sections 12.1, 12.2.
Chapter 13: Sections 13.1, 13.2, 13.4, if time permits.

What's Not in the Book

This book is not meant to be a comprehensive text in cryptography. There are certain topics covered very briefly or omitted entirely. For instance, there is no significant history of cryptography in the text. For a well-written, concise account of some history, see [33, Chapter 1, Section 1].

Moreover, I did not include any quantum cryptography. For a discussion of the quantum key distribution protocol, see [37, Chapter 4, Section 4.4.5].

Acknowledgments

This book has its origins in notes for a one-semester course in cryptography, which is part of a master's degree program in cyber security at Auburn University at Montgomery in Montgomery, Alabama.

I would like to thank current and former colleagues for their support and encouragement during the writing of this book: Dr. Babak Rahbarinia, (now at Salesforce), who read an early draft, Dr. Luis Cueva-Parra (now at the University of North Georgia), Dr. Yi Wang, Dr. Lei Wu, Dr. Matthew Ragland, Dr. Semih Dinc, Dr. Patrick Pape (now at the University of Alabama in Huntsville), Dr. Enoch Lee, and the many cryptography students at AUM who have used the course notes.

I also want to thank Elizabeth Loew at Springer for her interest in this project and her encouragement and guidance in helping to develop the manuscript.

I thank the reviewers for their helpful comments and suggestions, which have improved the material in this book. I am indebted to readers of the first draft who kindly agreed to read a second draft and provided many useful comments on the manuscript.

I thank my wife Rebecca and my son Andre for accepting the many hours that have been devoted to this book. I acknowledge that the musical works of the Grateful Dead, along with the literary efforts of Richard Brautigan, have been an influence and inspiration.

Montgomery, AL, USA Robert G. Underwood

Contents

Chapter 1
Introduction to Cryptography

1.1 Introduction to Cryptography

Cryptography, from the Greek meaning "secret writing," is the science of making messages unintelligible to all except those for whom the messages are intended. In this way, sensitive or private messages are kept secure and protected from unauthorized or unwanted access by persons or entities other than the intended recipients.

The message intended to be sent is the **plaintext**. An example of plaintext is

`Eyes of the World.`

The communications are kept secure by converting the plaintext into **ciphertext** by a process called **encryption**. Encryption is achieved through the use of an **encryption transformation**. An example of an encryption transformation is

e = replace each letter of the plaintext with the letter that is k places to the right in the alphabet, where k is an integer that satisfies $k \in \{0, 1, 2, \ldots, 25\}$.

(Note: in the encryption transformation e, the alphabet wraps around, and thus, if $k = 2$, then the letter `y` is replaced with `a`, `z` is replaced with `b`, and so on.)

The person sending the plaintext can only encrypt if she has chosen a value for k, which is the "key" to encryption. Generally, an **encryption key** is the additional information required so that plaintext can be converted to ciphertext using an encryption transformation. Given an encryption transformation, the set of all possible encryption keys is the **encryption keyspace**. For the encryption transformation e given above, the keyspace is $\{0, 1, 2, \ldots, 25\}$.

With e as above and encryption key $k = 2$, the encryption of the plaintext `Eyes of the World` is the ciphertext

`Gagu qh vjg Yqtnf;`

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_1

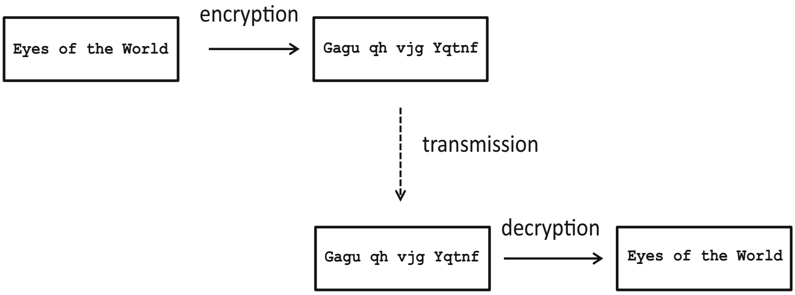


Fig. 1.1 Encryption–decryption process, $k = 2$

if $k = 17$, then the ciphertext is

`Vpvj fw kyv Nficu.`

The sender of the message then transmits the ciphertext to the recipient, who converts it back into the original plaintext by a process called **decryption**. Decryption is achieved using a **decryption transformation**. Decryption of ciphertext should only be possible if the recipient is in possession of the key for decryption, called the **decryption key**. For a given decryption transformation, the set of all possible decryption keys is the **decryption keyspace**.

For the example above, the decryption transformation is

> d = replace each letter of the ciphertext with the letter that is k places to the *left* in the alphabet, where k is an integer that satisfies $k \in \{0, 1, 2, \ldots, 25\}$,

and the decryption keyspace is $\{0, 1, 2, \ldots, 25\}$. With decryption key $k = 2$, the decryption of `Gagu qh vjg Yqtnf` is

`Eyes of the World.`

The encryption–decryption process is illustrated in Figure 1.1.
In the example of Figure 1.1, both the sender and the recipient share the same encryption–decryption key, namely, $k = 2$. This encryption–decryption process is an example of a **symmetric (or shared-key) cryptosystem**.

1.2 The Players in the Game

We introduce some formal notation for a cryptosystem. We assume the person sending plaintext message M is "Alice," denoted by A, and the intended recipient is "Bob," denoted by B. Let e denote an encryption transformation and let k_e be an encryption key. Let d denote the corresponding decryption transformation with decryption key k_d. Let

$$C = e(M, k_e)$$

denote the encryption of M into ciphertext C using transformation e and key k_e. Then

$$M = d(C, k_d) = d(e(M, k_e), k_d) \tag{1.1}$$

denotes the decryption of C back to M. In the example of Section 1.1,

$$M = \texttt{Eyes of the World},$$

$e =$ replace each letter of the plaintext with the letter that is k places to the right in the alphabet where $k \in \{0, 1, 2, \ldots, 25\}$,

$$k_e = 2,$$

and

$$C = \texttt{Gagu qh vjg Yqtnf}.$$

Thus in formal notation, the encryption–decryption process in Figure 1.1 can be written as

$$\texttt{Gagu qh vjg Yqtnf} = e(\texttt{Eyes of the World}, 2),$$

$$\texttt{Eyes of the World} = d(\texttt{Gagu qh vjg Yqtnf}, 2).$$

Definition 1.2.1 A **cryptosystem** is a system

$$\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle,$$

where $\mathcal{M}$ is a collection of plaintext messages, $\mathcal{C}$ is the corresponding set of ciphertext, e is an encryption transformation with keyspace $\mathcal{K}_e$, and d is the decryption transformation with keyspace $\mathcal{K}_d$.

Security of a cryptosystem relies on the fact that the decryption key $k_d \in \mathcal{K}_d$ is known only to Bob (and possibly Alice or a trusted third party). As we have seen, a symmetric cryptosystem is one in which both Alice and Bob have a shared secret key for encryption and decryption: $k_e = k_d$. A cryptosystem in which $k_e \neq k_d$ is an **asymmetric cryptosystem**.

In an asymmetric cryptosystem, Alice does not know Bob's secret key for decryption, k_d; only Bob knows his key for decryption. The key for encrypting messages to Bob, k_e, however, is made public (it is not secret). For this reason, an asymmetric cryptosystem is also called a **public key cryptosystem**. In public key cryptography, we write the encryption–decryption relationship as

$$C = e(M), \tag{1.2}$$

$$M = d(C, k_d) = d(e(M), k_d). \tag{1.3}$$

In Chapter 8, we will review the major symmetric key cryptosystems, and in Chapter 9, we will consider public key cryptosystems.

Suppose Alice intends to send messages to Bob using the cryptosystem $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle$. There is another player in this game, "Malice" (sometimes also called " Eve"), whose goal is to intercept ciphertext, obtain knowledge of the transformations e and d and the keys k_d and k_e (if not public), and use this information to decipher ciphertext by some method or process.

Consequently, Malice eavesdrops on as many conversations between Alice and Bob as possible. Malice will also pose as Alice and send messages to Bob, with Bob thinking they are coming from Alice, or pose as Bob to receive messages from Alice that Alice thinks are going to Bob. The result: Total breakdown of secure communications between Alice and Bob!

Malice's activities are known as **attacks** on the cryptosystem. If Malice only has some ciphertext C, then the attack is known as a **ciphertext only attack**. If Malice has obtained a pairing $M, C = e(M, k_e)$ for some $M \in \mathcal{M}$, then the attack is a **known plaintext attack**. If Malice has a pairing $M, C = e(M, k_e)$, where M is selected by Malice, then the attack is a **chosen plaintext attack**.

Generally, Malice is engaging in cryptanalysis of the cryptosystem. The science and art of obtaining plaintext from its corresponding ciphertext *without a priori knowledge of* e, d, k_e, *or* k_d is **cryptanalysis**. Cryptography and cryptanalysis together are known as **cryptology**.

1.3 Ciphertext Only Attack: An Example

Suppose Alice and Bob have agreed to use the symmetric cryptosystem $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ where

> $e =$ replace each letter of the plaintext with the letter that is k places to the right in the alphabet where $k \in \{0, 1, 2, \ldots, 25\}$.

(Here, we assume that $\mathcal{K} = \mathcal{K}_e = \mathcal{K}_d$ since this is a symmetric cryptosystem.) They have communicated beforehand and have agreed to use the shared key $k = 18$ which they have kept as a secret—only Alice and Bob know the value of k.

Suppose Alice encrypts a plaintext message, and in the transmission of the ciphertext to Bob, Malice obtains some of the ciphertext. Malice then engages in a ciphertext only attack to obtain the key by decrypting the ciphertext with every possible key. Only one key, the correct key k, should result in a decryption that is legible, legitimate plaintext. This method of attack is called **key trial**. Key trial is an example of a **brute-force** method of cryptanalysis since *every possible* key is tested.

Example 1.3.1 Alice encrypts the message

$$M = \texttt{Move supplies north}$$

as

$$C = e(M, 18) = \texttt{Egnw kmhhdawk fgjlz}.$$

Malice is able to intercept the ciphertext Egnw and uses key trial to obtain the key and some plaintext. He guesses that the ciphertext has been encrypted using e, and so the decryption transformation is

$d =$ replace each letter of the ciphertext with the letter that is k places to the left in the alphabet where $k \in \{0, 1, 2, \ldots, 25\}$.

Malice computes $d(\texttt{Egnw}, k)$ for all possible keys $k = 0, 1, 2, \ldots, 25$:

k	$d(\texttt{Egnw}, k)$	k	$d(\texttt{Egnw}, k)$
0	Egnw	13	Rtaj
1	Dfmv	14	Qszi
2	Celu	15	Pryh
3	Bdkt	16	Oqxg
4	Acjs	17	Npwf
5	Zbir	18	Move
6	Yahq	19	Lnud
7	Xzgp	20	Kmtc
8	Wyfo	21	Jlsb
9	Vxen	22	Ikra
10	Uwdm	23	Hjqz
11	Tvcl	24	Gipy
12	Subk	25	Fhox

Malice observes that Move is the only legible English word in the list and so deduces that $k = 18$.

□

The success of the cryptanalysis technique of key trial depends on two factors:

1. The existence of a relatively small number of keys, making exhaustive testing of each key feasible
2. The unlikelihood that two different keys produce recognizable plaintext after decryption

Example 1.3.2 In this example, Alice encrypts the plaintext

$$M = \texttt{IBM COMPUTERS}$$

as

$$C = e(M, 4) = \texttt{MFQ GSQTYXIVW}.$$

In this case, Malice is able to intercept the ciphertext MFQ and again uses key trial to find the key:

k	$d(\texttt{MFQ}, k)$	k	$d(\texttt{MFQ}, k)$
0	MFQ	13	ZSD
1	LEP	14	YRC
2	KDO	15	XQB
3	JCN	16	WPA
4	IBM	17	VOZ
5	HAL	18	UNY
6	GZK	19	TMX
7	FYJ	20	SLW
8	EXI	21	RKV
9	DWH	22	QJU
10	CVG	23	PIT
11	BUF	24	OHS
12	ATE	25	NGR

There are at least four legible words produced: IBM, HAL, ATE, and PIT. So Malice can only conclude that the key is either 4, 5, 12, or 23.

□

In Example 1.3.2, none of the keys 5, 12, and 23 are the correct key, yet they produce legible plaintext. These keys are "spurious" keys.

More formally, let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ be a symmetric cryptosystem with shared secret key k. Let C be the encryption of plaintext message M using k, i.e., $C = e(M, k)$. A key $l \in \mathcal{K}$, other than k, for which the decryption of C with l results in legible, legitimate, plaintext is a **spurious key** for C. Said differently, a spurious key is a key $l \neq k$ for which $d(C, l) \in \mathcal{M}$.

In Example 1.3.2, the spurious keys for MFQ are 5, 12, 23; in Example 1.3.1, there are no spurious keys for Engw.

As the number of characters in the ciphertext increases to ∞, the number of spurious key decreases to 0. For a sufficiently long string of ciphertext, there are no spurious keys; only the correct key k results in a legitimate decryption.

For a given (symmetric) cryptosystem $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$, the minimum length of ciphertext that guarantees no spurious keys is difficult to compute. We can, however, compute a lower bound for this minimum value; this lower bound is the **unicity distance** of the cryptosystem.

More precisely, the unicity distance is a lower bound for the size n_0 of encrypted words so that there is a unique key $k \in \mathcal{K}$ that maps $\mathcal{C}$ (consisting of ciphertext of length n_0) into $\mathcal{M}$.

The unicity distance is a theoretical measure of the ability of the cryptosystem to withstand a ciphertext only attack.

For the type of "right shift" transformations we have seen in our examples, it has been computed that the unicity distance is

$$\log_2(26)/3.2 = 4.7/3.2 = 1.47$$

characters of ciphertext, see Section 3.4.

In Example 1.3.1, the length of the ciphertext is 4, which is greater than the unicity distance 1.47. We had no spurious keys.

In Example 1.3.2, the length of the ciphertext is 3, which is greater than 1.47, yet there were 3 spurious keys (the correct key is not uniquely determined). This finding is not a contradiction: as we recall, the unicity distance is only a *lower bound* for the minimal length of ciphertext needed to guarantee no spurious keys.

We will discuss spurious keys and unicity distance in detail in Section 3.4.

1.4 Exercises

1. Let e be the encryption transformation defined as

   $e =$ replace each letter of the plaintext with the letter that is k places to the right in the alphabet, where k is an integer that satisfies $k \in \{0, 1, 2, \ldots, 25\}$.

   Suppose that $k_e = 5$. Compute $C = e(\texttt{Montgomery}, 5)$.
2. Let d be the decryption transformation defined as

   $d =$ replace each letter of the ciphertext with the letter that is k places to the left in the alphabet, where k is an integer that satisfies $k \in \{0, 1, 2, \ldots, 25\}$.

   Suppose that $k_d = 12$. Compute $M = d(\texttt{mxsqndm}, 12)$.
3. Suppose that Alice sends Bob the ciphertext $C = \texttt{ZLUKOLSW}$ using the encryption transformation e of Exercise 1. During transmission, Malice obtains the partial ciphertext `ZLUK`. Find the value of k_e and k_d. What is the plaintext message? What type of attack is Malice engaged in?
4. Suppose that Malice knows that Alice and Bob are using e and d as in Section 1.1 to communicate securely. Suppose that Malice has obtained the plaintext,ciphertext pairing `secret`, `bnlanc`. Explain how Malice can obtain the encryption key k_e. What is the value of k_e?
5. Consider the following variation of the cryptosystem given in Section 1.1. In this case, the alphabet is the three-element set of letters $\{a, b, c\}$, and plaintext

messages are finite sequences of a's, b's, and c's. Thus the message space $\mathcal{M} = \{a, b, c\}^*$. Let e be the encryption transformation defined as

> $e =$ replace each letter of the plaintext with the letter that is k places to the right in the alphabet, where k is an integer that satisfies $k \in \{0, 1, 2\}$.

Using this cryptosystem, compute the following:

(a) $C = e(\texttt{accbbac}, 2)$
(b) $M = d(\texttt{abcacbc}, 1)$

6. Suppose that Alice is using the "right shift" encryption transformation given in Section 1.1 to encrypt plaintext messages. Alice encrypts a two-letter plaintext word as `tw`. Compute the number of spurious keys for `tw`.
7. Suppose that Alice is using the "right shift" encryption transformation given in Section 1.1 to compute the ciphertext $C = e(M, k)$, where $M \in \mathcal{M}$, $k \in \mathcal{K}$, $k \neq 0$. If $C \in \mathcal{M}$, prove that there is at least one spurious key.

Chapter 2
Introduction to Probability

In this chapter we review some basic notions of probability that we will need in the chapters that follow.

2.1 Introduction to Probability

Probability concerns the outcomes of experiments. A **nondeterministic experiment** is an experiment in which the outcome cannot be predicted. For example, the following are nondeterministic experiments:

E_1 = a fair die is cast and the number of spots shown on the uppermost face is recorded.
E_2 = a card is selected at random from a standard deck of playing cards and the suit is recorded.
E_3 = a coin is flipped three times and the occurrences of “heads” and “tails” are recorded.

The reader should make the distinction between nondeterministic and deterministic experiments. For instance, given the ordinary calculus function $f(x) = 2x + 3$, the experiment “the input $x = -1$ is selected and the value of the function $f(x)$ is recorded” is *deterministic* in that we can easily predict its outcome, in this case $f(-1) = 1$.

Let E be a nondeterministic experiment. A **sample space** Ω associated to E is a set containing all possible mutually exclusive outcomes of E. For example, a sample space for experiment E_1 is $\Omega_1 = \{1, 2, 3, 4, 5, 6\}$, a sample space for E_2 is $\Omega_2 = \{\clubsuit, \diamondsuit, \heartsuit, \spadesuit\}$, and a sample space for E_3 is

$$\Omega_3 = \{TTT, TTH, THT, HTT, THH, HTH, HHT, HHH\}.$$

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_2

A sample space for an experiment is not unique: indeed

$$\Omega = \{\text{exactly zero heads, exactly one head, exactly two heads,}$$
$$\text{exactly three heads}\}$$

is also a sample space for experiment E_3.

Let E be a nondeterministic experiment with sample space Ω. An **event** A is a subset of Ω.

For experiment E_3 and sample space Ω_3, $A = \{THH, HTH, HHT\} \subseteq \Omega_3$ is the event "a coin is tossed three times and exactly two heads occur." For experiment E_2 and sample space Ω_2, $A = \{\clubsuit, \heartsuit\} \subseteq \Omega_2$ is the event "a card is selected at random and it is a club or a heart."

Event A **occurs** if the result of E is an element of A. For example, if the result of experiment E_2 is the card 10$\heartsuit$, then A has occurred. If the card drawn is K$\diamondsuit$, then A has not occurred.

Definition 2.1.1 Let E be a nondeterministic experiment with sample space Ω, and let $A \subseteq \Omega$ be an event. Suppose that E is repeated N times. The **relative frequency** of event A is

$$f_N(A) = \frac{n(A)}{N}$$

where $n(A)$ is the number of times that event A occurs in the N repetitions.

For example, consider experiment

E_1 = a fair die is cast and the number of spots shown on the uppermost face is recorded.

with sample space $\Omega_1 = \{1, 2, 3, 4, 5, 6\}$. Let $A = \{4\}$ be the event "a fair die is cast and the result is 4." Let $B = \{5, 6\}$ be the event "a fair die is cast and the result is at least 5." Suppose that experiment E_1 is repeated 20 times with following results.

Trial	Outcome	Trial	Outcome
1	5	11	2
2	6	12	5
3	2	13	5
4	3	14	2
5	4	15	4
6	1	16	5
7	4	17	1
8	6	18	4
9	6	19	6
10	2	20	2

Then $f_{20}(A) = 4/20 = 1/5$ and $f_{20}(B) = 8/20 = 2/5$.

As we shall see in Section 2.6, the relative frequency of an event is a good approximation of the probability that an event occurs.

2.1.1 Abstract Probability Spaces

Let Ω be a non-empty set, and let A, B be subsets of Ω. The **union** of A, B is the set

$$A \cup B = \{x \in \Omega : \ x \in A \text{ or } x \in B\};$$

the **intersection** is the set

$$A \cap B = \{x \in \Omega : \ x \in A, x \in B\}.$$

Let $\mathcal{A}$ be a (non-empty) collection of subsets of Ω. Then $\mathcal{A}$ is an **algebra** if the following conditions hold:

(i) $A \cup B$ is in $\mathcal{A}$ whenever $A, B \in \mathcal{A}$.
(ii) $\overline{A}$ is in $\mathcal{A}$ whenever $A \in \mathcal{A}$. (Here, $\overline{A} = \Omega A$).

An algebra $\mathcal{A}$ is a σ-**algebra** if the union $\bigcup_{i=1}^{\infty} A_i$ is in $\mathcal{A}$ whenever $\{A_i\}_{i=1}^{\infty}$ is a countable collection of subsets of $\mathcal{A}$.

Proposition 2.1.2 *Let $\mathcal{A}$ be a σ-algebra of subsets of Ω. Then $\Omega \in \mathcal{A}$ and $\emptyset \in \mathcal{A}$.*

Proof Let $A \in \mathcal{A}$. Since $\mathcal{A}$ is an algebra, $\overline{A} = \Omega \backslash A \in \mathcal{A}$. Thus, $\Omega = A \cup \overline{A} \in \mathcal{A}$. Now, since $\Omega \in \mathcal{A}$, $\overline{\Omega} = \emptyset \in \mathcal{A}$. □

Definition 2.1.3 An **abstract probability space** is a triple $(\Omega, \mathcal{A}, \text{Pr})$, where Ω is a non-empty set, $\mathcal{A}$ is a σ-algebra of subsets of Ω, and Pr is a function $\text{Pr} : \mathcal{A} \to [0, 1]$ that satisfies:

(i) $\text{Pr}(\Omega) = 1$.
(ii) $\text{Pr}(A \cup B) = \text{Pr}(A) + \text{Pr}(B)$ whenever $A \cap B = \emptyset$.

If $(\Omega, \mathcal{A}, \text{Pr})$ is an abstract probability space, then Ω is the **sample space**, $\mathcal{A}$ is the **set of events**, and Pr is the **probability function**. For event $A \in \mathcal{A}$, $\text{Pr}(A)$ is the **probability that event** A **occurs**. From Definition 2.1.3(i), we have $\text{Pr}(\Omega) = 1$. Thus from Definition 2.1.3(ii) we conclude that $\text{Pr}(\emptyset) = 0$.

Let $(\Omega, \mathcal{A}, \text{Pr})$ be an abstract probability space. For $A, B \in \mathcal{A}$,

$$\text{Pr}(A \cup B) = \text{Pr}(A) + \text{Pr}(B) - \text{Pr}(A \cap B). \tag{2.1}$$

Suppose that $A_1, A_2, \ldots, A_n$ is a finite collection of n events with $A_i \cap A_j = \emptyset, \forall i, j$, and $\bigcup_{i=1}^{n} A_i = \Omega$ (that is, suppose that $A_1, A_2, \ldots, A_n$ is a **partition** of

Ω). Then

$$\sum_{i=1}^{n} \Pr(A_i) = 1. \tag{2.2}$$

Note that (2.1) and (2.2) can be proved using Definition 2.1.3.

There is a very special abstract probability space that we will use in cryptography.

Definition 2.1.4 (Classical Definition of Probability) Let Ω to be a finite set of n elements:

$$\Omega = \{a_1, a_2, \ldots, a_n\},$$

and let $\mathcal{A}$ be the **power set** of Ω, that is, $\mathcal{A}$ is the collection of all subsets of Ω. We define a set function $\Pr : \mathcal{A} \to [0, 1]$ by the rule

$$\Pr(A) = \frac{|A|}{n},$$

where $|A|$ is the number of elements in $A \in \mathcal{A}$. For an event $A \subseteq \Omega$, $\Pr(A)$ is the probability that event A occurs. The triple $(\Omega, \mathcal{A}, \Pr)$ is an abstract probability space called the **classical probability space**; the probability function Pr is the **classical probability function**.

The classical probability space was introduced by Laplace (1749-1827).

Remark 2.1.5 In Definition 2.1.4, the probability of each singleton point set event $\{a_i\}$, $1 \le i \le n$, is $1/n$; each "elementary" event is equally likely. The probability function yields the uniform distribution function (see Example 2.4.1).

The classical definition of probability is used to compute the probabilities associated to nondeterministic experiments.

Example 2.1.6 Consider again the experiment and sample space:

> E_1 = a fair die is cast and the number of spots shown on the uppermost face is recorded; $\Omega_1 = \{1, 2, 3, 4, 5, 6\}$. Let $(\Omega_1, \mathcal{A}_1, \Pr)$ be the classical probability space.

Let $A = \{4\}$. Then $\Pr(A) = 1/6$. Let $B = \{5, 6\}$. Then $\Pr(B) = 1/3$. Also,

$$\Pr(A \cup B) = \Pr(A) + \Pr(B) = 1/6 + 1/3 = 1/2.$$

□

Example 2.1.7 In this example, we take

> E_2 = a card is selected at random from a standard deck of playing cards and the suit is recorded; $\Omega_2 = \{\clubsuit, \diamondsuit, \heartsuit, \spadesuit\}$

Let A = the selected card is not a ♠, that is, $A = \{♣, ♢, ♡\}$. Then $\Pr(A) = 3/4$ and $\Pr(\overline{A}) = 1/4$. □

Example 2.1.8 We next consider:

E_3 = A fair coin is flipped three times and the occurrences of "heads" (H) and "tails" (T) are recorded;

$$\Omega_3 = \{TTT, TTH, THT, HTT, THH, HTH, HHT, HHH\}.$$

Let A be the event "the last flip is H," and let B be the event "at least two H have occurred." Then $\Pr(A) = 1/2$ and $\Pr(A \cup B) = 5/8$. □

Sometimes we have to be very careful how we use Definition 2.1.4 to compute probabilities. For example, consider the experiment

E = Two fair die are cast and the sum of the spots shown on the uppermost faces are recorded, with sample space

$$\Omega = \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}.$$

In this case, a naive use of Definition 2.1.4 will lead us astray: for instance, $\Pr(\{5\}) \neq \frac{1}{11}$.

The issue is that the elementary events (singleton subsets) are *not* equally likely; the probability distribution is not uniform. This can be easily seen by repeating E a large number of times and computing the relative frequencies of the singleton subsets of Ω.

If we think of E as two *independent* events (one die comes to a stop an instant before the other die) and use the sample space

$$\Omega' = \{(1,1), (1,2), (1,3), (1,4), (1,5), (1,6),$$

$$(2,1), (2,2), (2,3), (2,4), (2,5), (2,6)$$

$$(3,1), (3,2), (3,3), (3,4), (3,5), (3,6),$$

$$(4,1), (4,2), (4,3), (4,4), (4,5), (4,6),$$

$$(5,1), (5,2), (5,3), (5,4), (5,5), (5,6),$$

$$(6,1), (6,2), (6,3), (6,4), (6,5), (6,6)\},$$

then we can use Definition 2.1.4 to compute $\Pr(\{(i, j)\}) = 1/36$ for $1 \le i, j \le 6$. In fact, then $\{5\} = \{(1, 4), (2, 3), (3, 2), (4, 1)\}$ and so $\Pr(\{5\}) = 4/36 = 1/9$.

2.2 Conditional Probability

Let $(\Omega, \mathcal{A}, \Pr)$ be a probability space, and let $A, B \in \mathcal{A}$. Assume $\Pr(A) \neq 0$. The probability that B occurs given that A has occurred is the **conditional probability** of B given A, denoted by $\Pr(B|A)$. One has

$$\Pr(B|A) = \frac{\Pr(A \cap B)}{\Pr(A)}.$$

For example, consider again the experiment and sample space:

$E_3 =$ A fair coin is flipped three times and the occurrences of "heads" and "tails" are recorded;

$$\Omega_3 = \{TTT, TTH, THT, HTT, THH, HTH, HHT, HHH\}.$$

Let $A =$ the last flip is H, and let $B =$ at least two H have occurred. Then one computes

$$\begin{aligned}\Pr(B|A) &= \frac{\Pr(A \cap B)}{\Pr(A)} \\ &= \frac{3/8}{4/8} \\ &= 3/4.\end{aligned}$$

Intuitively, if $\Pr(B|A) = \Pr(B)$, then the occurrence of B does not depend on the occurrence of A. Thus B is independent of A. Moreover,

$$\Pr(A \cap B) = \Pr(B|A)\Pr(A) = \Pr(B)\Pr(A).$$

This motivates the following definition. The events A, B are **independent** if

$$\Pr(A \cap B) = \Pr(A)\Pr(B).$$

Observe that in experiment E_3 with sample space Ω_3, the events A, B are not independent since

$$\Pr(B|A) = 3/4 \neq \Pr(B) = 1/2.$$

Here is an important proposition involving conditional probabilities.

Proposition 2.2.1 (Total Probability Theorem) *Suppose that $A_1, A_2, \ldots, A_n$ is a partition of Ω, and let A be any event. Then*

$$\Pr(A) = \sum_{i=1}^{n} \Pr(A|A_i)\Pr(A_i).$$

Proof We have

$$A = \Omega \cap A = \left(\bigcup_{i=1}^{n} A_i\right) \cap A = \bigcup_{i=1}^{n}(A_i \cap A).$$

Thus

$$\begin{aligned}\Pr(A) &= \Pr\left(\bigcup_{i=1}^{n}(A_i \cap A)\right)\\ &= \sum_{i=1}^{n}\Pr(A_i \cap A)\\ &= \sum_{i=1}^{n}\Pr(A|A_i)\Pr(A_i).\end{aligned}$$

□

2.3 Collision Theorems

Collision theorems concern probabilities about whether function values collide or are the same, and are used in various attacks on cryptosystems.

We present two different collision models.

Suppose S is a finite non-empty set of elements with $N = |S|$. We think of S as a sample space Ω of N elements. We assume that $(\Omega, \mathcal{A}, \Pr)$ is the classical probability space, where Pr is the classical definition of probability. To simplify notation, we write y for the singleton subset $\{y\} \subseteq S$. We have $\Pr(y) = 1/N$ for all $y \in S$.

Let $m \geq 2$ be an integer, and let

$$f : \{1, 2, 3, \ldots, m\} \to S$$

be a function with $f(i) = y_i \in S$ for $1 \leq i \leq m$. We consider the function f as a sequence of m terms (function values)

$$y_1, y_2, y_3, \ldots, y_m$$

randomly chosen from S (with replacement); y_1 is the first term of the sequence and represents the first choice, y_2 is the second term of the sequence representing the second choice, and so on. As with all sequences, we could have $y_i = y_j$ for some $i \neq j$.

What is the probability that all of the terms are distinct? (Equivalently, what is the probability that f is an injection?)

Let $(y_i = y_j)$ denote the event "y_i is equal to y_j," $1 \leq i, j \leq m$. We have $\Pr(y_1 = y_2) = \frac{1}{N}$, thus $\Pr(y_1 \neq y_2) = 1 - \frac{1}{N}$. So the probability that the first two terms of the sequence are distinct is $1 - \frac{1}{N}$.

We next consider the first three terms y_1, y_2, y_3. We have the conditional probability

$$\Pr(((y_1 = y_3) \cup (y_2 = y_3))|(y_1 \neq y_2)) = \frac{1}{N} + \frac{1}{N} = \frac{2}{N},$$

since

$$(y_1 = y_3) \cap (y_2 = y_3) = \emptyset,$$

whenever $y_1 \neq y_2$. Thus

$$\Pr(((y_1 \neq y_3) \cap (y_2 \neq y_3))|(y_1 \neq y_2)) = 1 - \frac{2}{N}.$$

Now,

$$\Pr((y_1 \neq y_3) \cap (y_2 \neq y_3) \cap (y_1 \neq y_2)) = \left(1 - \frac{2}{N}\right)\left(1 - \frac{1}{N}\right),$$

and so the probability that the first three terms are distinct is

$$\Pr(y_1, y_2, y_3 \text{ distinct}) = \left(1 - \frac{1}{N}\right)\left(1 - \frac{2}{N}\right).$$

Continuing in this manner, we find that the probability that the m terms are distinct is

$$\begin{aligned}\Pr(y_1, y_2, \dots y_m \text{ distinct}) &= \left(1 - \frac{1}{N}\right)\left(1 - \frac{2}{N}\right)\cdots\left(1 - \frac{m-1}{N}\right)\\ &= \prod_{i=1}^{m-1}\left(1 - \frac{i}{N}\right).\end{aligned}$$

We have proved our first "collision" theorem.

Proposition 2.3.1 *Let S be a finite non-empty set with $N = |S|$. Let $y_1, y_2, \dots y_m$ be a sequence of $m \geq 2$ terms chosen at random from S (with replacement). Then the probability that there is a "collision," that is, the probability that at least two of the terms are the same, $y_i = y_j$ for some $i \neq j$, is*

$$\Pr(y_i = y_j \textit{ for some } i \neq j) = 1 - \prod_{i=1}^{m-1} \left(1 - \frac{i}{N}\right).$$

□

We can obtain a lower bound for the probability of a collision. From the inequality $e^x \geq 1+x$, valid for all $x \in \mathbb{R}$, we obtain $e^{-\frac{i}{N}} \geq 1-\frac{i}{N}$ for $1 \leq i \leq m-1$. And so

$$\begin{aligned}\prod_{i=1}^{m-1} \left(1 - \frac{i}{N}\right) &\leq \prod_{i=1}^{m-1} e^{-\frac{i}{N}} \\ &= e^{-\frac{1}{N}\sum_{i=1}^{m-1} i} \\ &= e^{\frac{-m(m-1)}{2N}}.\end{aligned}$$

Thus

$$\Pr(y_i = y_j \text{ for some } i \neq j) \geq 1 - e^{\frac{-m(m-1)}{2N}}. \tag{2.3}$$

Example 2.3.2 (Birthday Paradox) Let S be the set of days in a non-leap year, with $N = |S| = 365$. Let $m \geq 2$ and randomly select a group of $m \geq 2$ people, denoted as $P_1, P_2, \dots, P_m$. Let

$$B : \{P_1, P_2, \dots, P_m\} \to S$$

be the "birthday function," where $y_i = B(P_i)$ is the birthday of person P_i. The birthday function corresponds to a sequence of birthdays $y_1, y_2, \dots, y_m$.

We compute the minimum value for m so that it is likely that at least two people in the group of m people have the same birthday. In other words, we find the smallest value of m so that

$$\Pr(y_i = y_j \text{ for some } i \neq j) > \frac{1}{2}.$$

For $x \in \mathbb{R}$, let $\lceil x \rceil$ denote the smallest integer $\geq x$.

From (2.3), we seek the smallest m so that $1 - e^{\frac{-m(m-1)}{730}} > \frac{1}{2}$, or

$$\frac{1}{2} > e^{\frac{-m(m-1)}{730}},$$

or

$$730 \ln(0.5) > -m(m-1),$$

or

$$m(m-1) \geq \lceil -730 \ln(0.5) \rceil = 506,$$

thus $m = 23$.

So our conclusion is the following: if we choose at least 23 people at random, then it is likely that two of them will have the same birthday. The fact that we obtain a "collision" of birthdays by choosing as few as 23 people is somewhat surprising, hence this phenomenon is known as the **birthday paradox**.

□

Example 2.3.3 (Square Root Attack) If $m = 1 + \lceil \sqrt{1.4N} \rceil$ in Proposition 2.3.1, then $\frac{m(m-1)}{2N} > \frac{\sqrt{1.4N}\sqrt{1.4N}}{2N} = 0.7$, thus $\frac{-m(m-1)}{2N} < -0.7$, hence $e^{\frac{-m(m-1)}{2N}} < e^{-0.7}$, or

$$1 - e^{\frac{-m(m-1)}{2N}} > 1 - e^{-0.7} > 50\%.$$

Thus choosing a sequence of at least $1 + \lceil \sqrt{1.4N} \rceil \approx \sqrt{1.4N}$ terms in S ensures that a collision is likely to occur (i.e., a collision occurs with probability $> 50\%$).

Choosing $m \geq 1 + \lceil \sqrt{1.4N} \rceil \approx \sqrt{1.4N}$ to ensure a likely collision is therefore called the **square root attack**.

□

The square root attack is used in the Pollard ρ algorithm (Algorithm 9.3.6) to attack the RSA public key cryptosystem. It is also used in our discussion of cryptographic hash functions (Section 10.5).

Here is the second collision model. Again, S is a finite non-empty set of elements $N = |S|$ and we assume that $(\Omega, \mathcal{A}, \Pr)$ is the classical probability space, where $\Omega = S$ and Pr is the classical probability function.

Let n be an integer, $1 \leq n \leq N$, and let $T = \{x_1, x_2, \ldots, x_n\}$ be a subset of S. Let $m \geq 1$, and let

$$y_1, y_2, \ldots, y_m$$

be a sequence of m random terms of S. The probability that y_1 does not match any of the elements of T is $1 - \frac{n}{N}$. The probability that y_2 does not match any of the elements of T is $1 - \frac{n}{N}$, and the probability that neither y_1 nor y_2 matches any element of T is $(1 - \frac{n}{N})^2$. The probability that none of the random terms match any element of T is $(1 - \frac{n}{N})^m$. Thus

$$\Pr(y_i = x_j \text{ for some } i, j) = 1 - \left(1 - \frac{n}{N}\right)^m .$$

So we have the second collision theorem.

Proposition 2.3.4 *Let S be a finite non-empty set, $N = |S|$. Let n be an integer, $1 \le n \le N$, and let $T = \{x_1, x_2, \ldots, x_n\} \subseteq S$. Let $y_1, y_2, \ldots y_m$ be a sequence of $m \ge 1$ terms chosen at random from S (with replacement). Then the probability that there is a "collision," that is, there is at least one term of the sequence that matches some element of T is*

$$\Pr(y_i = x_j \textit{ for some } i, j) = 1 - \left(1 - \frac{n}{N}\right)^m .$$

□

Proposition 2.3.4 is used in the Baby Step/Giant Step attack on the Diffie–Hellman Key Exchange protocol (see Algorithm 12.2.11).

2.4 Random Variables

Random variables will be used to compute the entropy of plaintext English (Section 3.2) and in the computation of the unicity distance of a cryptosystem (Section 3.4).

Let $(\Omega, \mathcal{A}, \Pr)$ be a probability space. Let $S = \{s_1, s_2, \ldots, s_m\}$ be a finite set of elements. A **discrete random variable** is a function $X : \Omega \to S$. Let $s_i \in S$. By the notation $(X = s_i)$, we mean the event

$$(X = s_i) = \{\omega \in \Omega : \ X(\omega) = s_i\}.$$

Put $p_i = \Pr(X = s_i)$. Then $\sum_{i=1}^m p_i = 1$. The function

$$f_X : S \to [0, 1],$$

defined as

$$f_X(s_i) = \Pr(X = s_i) = p_i$$

is the **distribution function** for the random variable X. Here are two important examples.

Example 2.4.1 (Uniform Distribution Function) Let $\Omega = \{a_1, a_2, \dots, a_n\}$, and let $\mathcal{A}$ be the power set of Ω. Define $\Pr(A) = |A|/n$ for $A \in \mathcal{A}$. (Note: this is the classical definition of probability.) Let $S = \{1, 2, \dots, n\}$, and let $X : \Omega \to S$ be defined as

$$X(a_i) = i,$$

for $1 \le i \le n$. Then the event

$$(X = i) = \{a \in \Omega : X(a) = i\} = \{a_i\}.$$

Thus

$$\Pr(X = i) = \Pr(\{a_i\}) = \frac{1}{n}, \text{ for } i = 1, \dots, n.$$

The function $f_X : S \to [0, 1]$ defined as

$$f_X(i) = \frac{1}{n}, \ \forall i$$

is the **uniform distribution function**.

□

For instance, take $\Omega = \{a_1, a_2, a_3, a_4, a_5\}$, $S = \{1, 2, 3, 4, 5\}$ with random variable $X : \Omega \to S$ defined as $X(a_i) = i$. Then the uniform distribution function $f_X : S \to [0, 1]$ appears as in Figure 2.1.

Let Ω be the sample space defined as $\Omega = \{\text{success, failure}\}$, and let $\mathcal{A}$ be the power set of Ω. Define $\Pr : \mathcal{A} \to [0, 1]$ by $\Pr(\emptyset) = 0$, $\Pr(\{\text{success}\}) = p$, $0 \le p \le 1$, $\Pr(\{\text{failure}\}) = q = 1 - p$, $\Pr(\{\text{success, failure}\}) = 1$. (Note that $\{\text{success}\} \cup \{\text{failure}\} = \{\text{success,failure}\}$.) Then $(\Omega, \mathcal{A}, \Pr)$ is a probability space.

Let $S = \{0, 1\}$, and define a random variable $\xi : \Omega \to S$ by the rule $\xi(\text{success}) = 1$, $\xi(\text{failure}) = 0$. Then the distribution function $f_\xi : S \to [0, 1]$ is defined as $f_\xi(1) = \Pr(\xi = 1) = \Pr(\{\omega \in \Omega : \xi(\omega) = 1\}) = \Pr(\{\text{success}\}) = p$, $f_\xi(0) = \Pr(\xi = 0) = \Pr(\{\omega \in \Omega : \xi(\omega) = 0\}) = \Pr(\{\text{failure}\}) = 1 - p = q$.

We repeat an experiment with sample space

$$\Omega = \{\text{success, failure}\}$$

n times. The possible outcomes are

$$\Omega^n = \{\text{success, failure}\}^n,$$

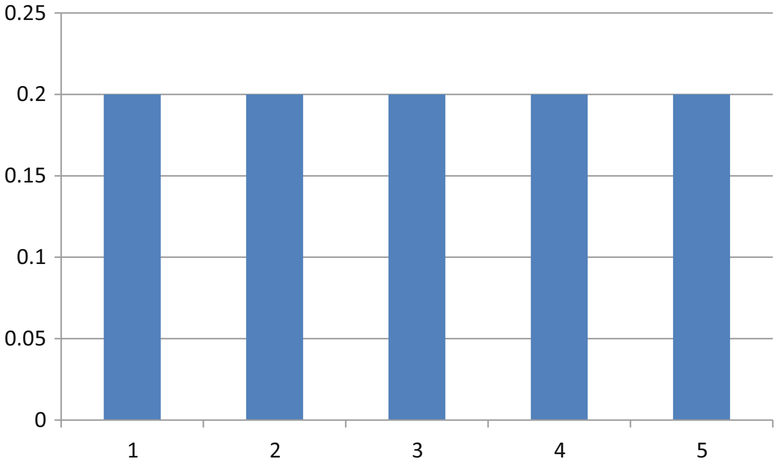


Fig. 2.1 Uniform distribution function $f_X : S \to [0, 1]$

which consists of all possible sequences of "success" and "failure" of length n. For example, for $n = 5$,

$$\omega = \text{success, failure, success, failure, failure}$$

is such a sequence.

Example 2.4.2 (Binomial Distribution Function) With the notation as above, let $S = \{0, 1, 2, 3, \ldots, n\}$ and define a random variable

$$\xi_n : \Omega^n \to S$$

by the rule

$$\begin{aligned}\xi_n(\omega) &= \text{the number of occurrences of "success" in sequence} \\ &\quad \omega \in \Omega^n \\ &= \text{the number of favorable outcomes in conducting the} \\ &\quad \text{ experiment } n \text{ times}\end{aligned}$$

ξ_n is the **binomial random variable**. For $0 \leq k \leq n$, $0 < p < 1$, the event

$$(\xi_n = k) = \{\omega \in \Omega^n : \text{ there are } k \text{ occurrences of "success" in } \omega\}$$

has probability

$$\Pr(\xi_n = k) = \binom{n}{k} p^k q^{n-k} = \binom{n}{k} p^k (1 - p)^{n-k}.$$

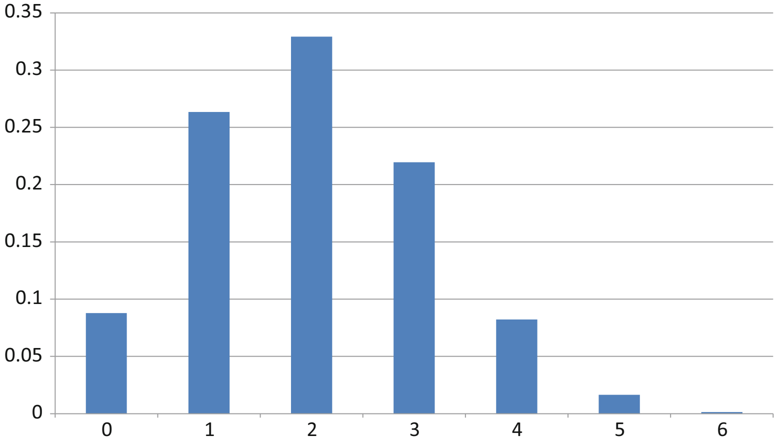


Fig. 2.2 Binomial distribution function $f_{\xi_6} : S \to [0, 1]$

The corresponding distribution function

$$f_{\xi_n} : S \to [0, 1]$$

defined as

$$f_{\xi_n}(k) = \Pr(\xi_n = k) = \binom{n}{k} p^k (1 - p)^{n-k},$$

for $0 \le k \le n$, is the **binomial distribution function**.

□

For example, if $n = 6$, $p = 1/3$, $S = \{0, 1, 2, 3, 4, 5, 6\}$, we obtain the binomial distribution $f_{\xi_6} : S \to [0, 1]$, with $f_{\xi_6}(k) = \binom{6}{k}(1/3)^k(2/3)^{6-k}$. (See Figure 2.2.)

2.5 2-Dimensional Random Variables

Let $(\Omega, \mathcal{A}, \Pr)$ be an abstract probability space. A 2-**dimensional random variable** is a pairing (X, Y) of random variables $X : \Omega \to S$, $Y : \Omega \to T$, $S = \{s_1, s_2, \ldots, s_m\}$, $T = \{t_1, t_2, \ldots, t_n\}$. The notation $(X = s_i \cap Y = t_j)$ denotes the event

$$\{\omega \in \Omega : X(\omega) = s_i \text{ and } Y(\omega) = t_j\},$$

or more simply, both $(X = s_i)$ and $(Y = t_j)$ occur. The **joint probability** is $\Pr(X = s_i \cap Y = t_j)$. The conditional probability is $\Pr(X = s_i | Y = t_j)$. The function

$$f_{X,Y}(s_i, t_j) = \Pr(X = s_i \cap Y = t_j)$$

is the **joint distribution function** for the 2-dimensional random variable (X, Y). Note that $\sum_{i=1}^{m} \sum_{j=1}^{n} f_{X,Y}(s_i, t_j) = 1$.

Example 2.5.1 Let $(\Omega, \mathcal{A}, \Pr)$ be an abstract probability space. Let E be the non-deterministic experiment "a pair of fair dice are cast and number of spots on the uppermost face of the first die and the uppermost face on the second die is recorded." Let $X : \Omega \to \{1, 2, 3, 4, 5, 6\}$, $Y : \Omega \to \{1, 2, 3, 4, 5, 6\}$ be random variables, where the event $(X = i)$ is "the uppermost face of the first die has i spots," and the event $(Y = j)$ is "the uppermost face on the second die has j spots." Then the joint probability is $\Pr(X = i \cap Y = j) = \frac{1}{36}$ for $1 \le i, j \le 6$. □

Example 2.5.2 Let $(\Omega, \mathcal{A}, \Pr)$ be an abstract probability space. Let E be the non-deterministic experiment "a single die is cast and number of spots on the uppermost face is recorded." Let $X : \Omega \to \{1, 2, 3, 4, 5, 6\}$, $Y : \Omega \to \{\text{odd,even}\}$ be random variables, where the event $(X = i)$ is "the uppermost face of the die has i spots," and the event $(Y = \text{odd})$ is "i is odd," and the event $(Y = \text{even})$ is "i is even." Then the joint probability satisfies

$$\Pr(X = i \cap Y = \text{odd}) = \begin{cases} \frac{1}{6} & \text{if } i \text{ is odd} \\ 0 & \text{if } i \text{ is even} \end{cases}$$

$$\Pr(X = i \cap Y = \text{even}) = \begin{cases} 0 & \text{if } i \text{ is odd} \\ \frac{1}{6} & \text{if } i \text{ is even} \end{cases}$$

□

Let (X, Y) be a 2-dimensional random variable. Then by the events $(X = s_i)$, $(Y = t_j)$ we mean

$$(X = s_i) = \bigcup_{j=1}^{n} (X = s_i \cap Y = t_j),$$

$$(Y = t_j) = \bigcup_{i=1}^{m} (Y = t_j \cap X = s_i).$$

Since

$$(X = s_i \cap Y = t_j) \cap (X = s_i \cap Y = t_k) = \emptyset,$$

whenever $j \neq k$, we have

$$\begin{aligned}\Pr(X = s_i) &= \Pr\Big(\bigcup_{j=1}^{n} (X = s_i \cap Y = t_j)\Big)\\ &= \sum_{j=1}^{n} \Pr(X = s_i \cap Y = t_j)\\ &= \sum_{j=1}^{n} f_{X,Y}(s_i, t_j),\end{aligned}$$

Likewise,

$$\Pr(Y = t_j) = \sum_{i=1}^{m} f_{X,Y}(s_i, t_j).$$

The **marginal distribution functions** are defined as

$$f_1(s_i) = \Pr(X = s_i) = \sum_{j=1}^{n} f_{X,Y}(s_i, t_j) = \sum_{j=1}^{n} \Pr(X = s_i \cap Y = t_j),$$

$$f_2(t_j) = \Pr(Y = t_j) = \sum_{i=1}^{m} f_{X,Y}(s_i, t_j) = \sum_{i=1}^{m} \Pr(Y = t_j \cap X = s_i).$$

The **product of marginals distribution function** is defined as

$$p(s_i, t_j) = f_1(s_i) f_2(t_j).$$

As one can check,

$$\sum_{i=1}^{m} \sum_{j=1}^{n} f_1(s_i) f_2(t_j) = 1.$$

The random variables X, Y are **independent** if

$$f_{X,Y}(s_i, t_j) = f_1(s_i) f_2(t_j),$$

for all s_i, t_j, otherwise, X and Y are **dependent**.

The random variables in Example 2.5.1 are independent, while the random variables of Example 2.5.2 are dependent.

2.6 Bernoulli's Theorem

We close this chapter with Bernoulli's theorem, also known as the **Law of Large Numbers**. Bernoulli's theorem reconciles the relative frequency of an event and the classical definition of the probability of that event occurring.

Let $(\Omega, \mathcal{A}, \Pr)$ denote the probability space defined in Example 2.4.2. Let $n \geq 1$, and let

$$\xi_n : \Omega^n \to S = \{0, 1, 2, \ldots, n\}$$

denote the binomial random variable. For an integer k, $0 \leq k \leq n$, we have

$$\Pr(\xi_n = k) = \binom{n}{k} p^k (1-p)^{n-k},$$

where p is the probability that the event {success} occurs, $0 < p < 1$. Note that the relative frequency of the event {success} given n trials is precisely

$$f_n(\{\text{success}\}) = \frac{\xi_n}{n}.$$

At the same time, from the probability space $(\Omega, \mathcal{A}, \Pr)$, we have

$$\Pr(\{\text{success}\}) = p.$$

We relate these two definitions of the probability of the event {success}. We claim that $\frac{\xi_n}{n}$ approximates p in the sense that for any $\epsilon > 0$, and sufficiently large n, the event

$$\left(\left|\frac{\xi_n}{n} - p\right| < \epsilon\right) = \left\{\omega \in \Omega^n : \left|\frac{\xi_n(\omega)}{n} - p\right| < \epsilon\right\}$$

is very nearly certain to occur. That is,

$$\Pr\left(\left|\frac{\xi_n}{n} - p\right| < \epsilon\right) \approx 1,$$

for n very large. More precisely, we have the following theorem.

Proposition 2.6.1 (Bernoulli's Theorem) *Let* $\epsilon > 0$. *Then*

$$\lim_{n\to\infty} \Pr\left(\left|\frac{\xi_n}{n} - p\right| < \epsilon\right) = 1.$$

Proof For a proof, see [37, Chapter 3, Section 3.5.3]. □

Here is an application of Bernoulli's Theorem. We take the experiment and sample space:

> E_1 = a fair die is cast and the number of spots shown on the uppermost face is recorded; $\Omega_1 = \{1, 2, 3, 4, 5, 6\}$.

Let $A = \{4\}$. Define "success" to be "E_1 is conducted and event A occurs." Then $\Pr(\{\text{success}\}) = 1/6$. Let ξ_n be the binomial random variable defined as: ξ_n is the number of occurrences of {success} in conducting experiment E_1 n times. Bernoulli's Theorem then says that for each $\epsilon > 0$,

$$\lim_{n\to\infty} \Pr\left(\left|\frac{\xi_n}{n} - \frac{1}{6}\right| < \epsilon\right) = 1.$$

In other words,

$$\frac{\xi_n}{n} \approx \frac{1}{6}$$

for very large n.

2.7 Exercises

1. Let E be the nondeterministic experiment:

   > E = A pair of fair dice are rolled and the sum of the number of spots shown on the uppermost faces is recorded.

   Let $\Omega = \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$ denote the sample space of E. Suppose experiment E is repeated 20 times with the following results:

   Define events as follows: A = the sum is at least 8, $B = \{6, 7, 11\}$, $C = \{10\}$.

   (a) Compute the relative frequency that A occurs.
   (b) Compute the relative frequency that A or B occurs.
   (c) Compute the relative frequency that C does not occur.

Trial	Outcome	Trial	Outcome
1	3	11	4
2	6	12	7
3	6	13	7
4	8	14	11
5	7	15	2
6	12	16	5
7	12	17	6
8	9	18	5
9	5	19	7
10	8	20	9

2. Let E be the nondeterministic experiment:

   $E =$ A coin is flipped four times and the occurrences “heads” (H) and “tails” (T) are recorded.

   Let $\Omega =$
   $$\{HHHH, HHHT, HHTH, HHTT, HTHH, HTHT, HTTH, HTTT, THHH, THHT, THTH, THTT, TTHH, TTHT, TTTH, TTTT\}$$
   denote the sample space of E.

   Define events as follows: $A =$ the first flip is H, $B =$ at least two T have occurred, $C = \{HHHH, HHTT, TTTT\}$.

   Compute the following probabilities using the classical definition of probability:

   (a) $\Pr(A)$
   (b) $\Pr(A \cup B)$
   (c) $\Pr(B \cap C)$
   (d) $\Pr(C|B)$

3. Referring to Exercise 2: Are the events A and B independent? Are the events B and C independent?
4. Let $(\Omega, \mathcal{A}, \Pr)$ be an abstract probability space. Prove formulas (2.1) and (2.2).
5. Let $(\Omega, \mathcal{A}, \Pr)$ be an abstract probability space, and let A, B be events. Prove that $\Pr(B|A) + \Pr(\overline{B}|A) = 1$.
6. Consider the plaintext message $M =$
   ```
   the storyteller makes no choice soon you will not
   hear his voice his job is to shed light and not to
   master
   ```
   Compute the relative frequency that the letter a occurs in M. Compute the relative frequency that the letter e or s occurs.
7. In Example 1.3.1 and Example 1.3.2, Malice uses a method of cryptanalysis called key trial to determine the key and some plaintext. Let A be the event: “given ciphertext C, key trial determines the correct key.” Show that $\Pr(A) = \frac{1}{|W(C)|}$, where

$$W(C) = \{l \in \mathcal{K} : \ d(C, l) \in \mathcal{M}\}$$

(cf. Section 3.4).

8. Let $S = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ be a set of integers. Suppose that a sequence of 5 elements of S are chosen at random (with replacement). What is the probability that at least 2 of the terms of the sequence are identical?
9. Prove the formula: $e^x \geq 1 + x, \forall x \in \mathbb{R}$.
10. What is the probability that in a random group of 10 people at least two have the same birthday?
11. Consider the experiment:

    $E =$ Two fair die are cast and the sum of the spots shown on the uppermost faces are recorded, with sample space

    $$\Omega = \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}.$$

    Prove directly that the elementary events (singleton subsets) are *not* equally likely by repeating E a large number of times and computing the relative frequencies of the singleton subsets of Ω.
12. Let $S = \{0, 1, 2, \ldots, 10\}$. Compute the binomial distribution function $f_{\xi_{10}} : S \to [0, 1]$, assuming that the success probability is $\frac{1}{2}$. Illustrate the probability distribution using a bar graph.
13. Let $X : \Omega \to \{s_1, s_2, s_3\}$, $Y : \Omega \to \{t_1, t_2, t_3\}$ be random variables. Suppose that the joint probability distribution of the random vector (X, Y) is given by the table

	t_1	t_2	t_3
s_1	1/18	1/9	1/6
s_2	1/9	1/9	1/18
s_3	1/18	1/18	5/18

Compute the following probabilities:

(a) $\Pr(X = s_3 \cap Y = t_1)$
(b) $\Pr(X = s_2)$
(c) $\Pr(Y = t_1)$
(d) $\Pr(X = s_2 | Y = t_1)$
(e) $\Pr(X = s_1 \cup Y = t_3)$

Chapter 3
Information Theory and Entropy

3.1 Entropy

Let $(\Omega, \mathcal{A}, \Pr)$ be a fixed probability space. Let $S = \{s_1, s_2, s_3, \ldots, s_n\}$ be a finite set, and let $X : \Omega \to S$ be a random variable with distribution function $f_X : S \to [0, 1]$, $f_X(s_i) = \Pr(X = s_i)$.

Definition 3.1.1 Let $X : \Omega \to S$ be a random variable with probability distribution function $f_X : S \to [0, 1]$. Then the **entropy** of X is

$$H(X) = -\sum_{i=1}^{n} f_X(s_i) \log_2(f_X(s_i)).$$

Entropy is a measure of the amount of information in the random variable X. To see that entropy is a measure of information, we look at a few examples.

Suppose Alice and Bob want to share a single bit, 0 or 1. In the first scenario, Alice always chooses 0 and Bob knows that this will always be her choice. In this case, when Alice sends 0 to Bob, there is no information conveyed from Alice to Bob. From Bob's viewpoint, there is nothing noteworthy, interesting, new, or surprising from this communication with Alice; he knows that Alice will always send 0, and she does send 0.

The first scenario can be modeled by the random variable $X : \Omega \to \{0, 1\}$, where $f_X(0) = \Pr(X = 0) = 1$ and $f_X(1) = \Pr(X = 1) = 0$. According to Definition 3.1.1, the entropy of X is

$$\begin{aligned} H(X) &= -f_X(0) \cdot \log_2(f_X(0)) - f_X(1) \cdot \log_2(f_X(1)) \\ &= -1 \cdot \log_2(1) - 0 \cdot \log_2(0) \\ &= 0. \end{aligned}$$

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_3

(Note: we take the value of $0 \cdot \log_2(0)$ to be 0.) The computed value $H(X) = 0$ is consistent with what we expect in this scenario; it matches what we intuitively expect to be the amount of information in X, i.e., no information.

In the second scenario, Alice chooses 0 exactly half of the time and chooses 1 exactly half of the time. Bob has no idea which choice Alice will make beforehand. When Bob receives Alice's bit, he has received some information; he is aware of something new or interesting that he did not know before: the value of Alice's bit. In this case, information is conveyed by Alice in her communication with Bob. But how much information?

This second scenario is modeled by the random variable $Y : \Omega \to \{0, 1\}$, with $f_X(0) = \Pr(X = 0) = 1/2$ and $f_X(1) = \Pr(X = 1) = 1/2$. We have

$$\begin{aligned} H(Y) &= -f_X(0) \cdot \log_2(f_X(0)) - f_X(1) \cdot \log_2(f_X(1)) \\ &= -\frac{1}{2} \cdot \log_2\left(\frac{1}{2}\right) - \frac{1}{2} \cdot \log_2\left(\frac{1}{2}\right) \\ &= 1. \end{aligned}$$

Now, the computed value $H(Y) = 1$ is consistent with what we expect in this scenario; it matches what we intuitively expect to be the amount of information in Y: 1 bit of information is conveyed by Alice.

Whether Alice's bit is 0 or 1 does not matter; regardless, there is 1 bit of information in Y.

These two scenarios show that information is measured in bits.

Here is another interpretation of the entropy formula. Let X be a discrete random variable taking values in the set $S = \{s_1, s_2, \ldots, s_n\}$. If an event $(X = s_1)$ has high probability of occurrence, then we should expect it to contain a small amount of information (it is not surprising or noteworthy when it occurs). On the other hand, if an event $(X = s_i)$ has a low probability, then we expect it to be attached to a large amount of information (it is surprising or noteworthy when it occurs).

Observe that the function $\log_2(\frac{1}{x}) = -\log_2(x)$ is strictly decreasing on the interval $(0, 1]$; see Figure 3.1.

With $x = \Pr(X = s_i)$, the value

$$-\log_2(\Pr(X = s_i)) = -\log_2(f_X(s_i))$$

is thus suited to represent the amount of **information** in the event $(X = s_i)$ as measured in bits. The entropy of the random variable

$$H(X) = -\sum_{i=1}^{n} f_X(s_i) \log_2(f_X(s_i))$$

is the **average information in** X.

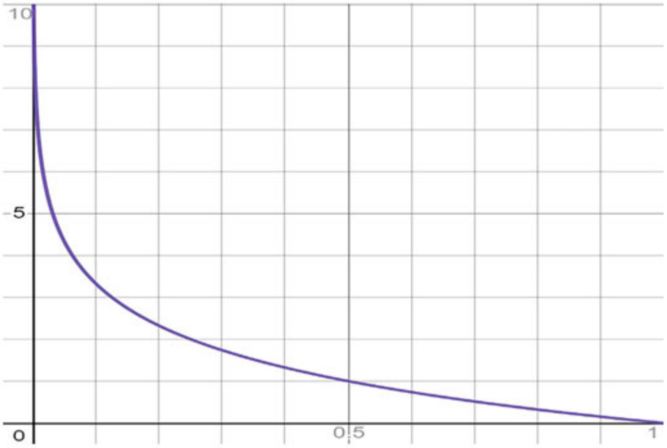


Fig. 3.1 Graph of $y = -\log_2(x)$ on the interval (0, 1]

Here is an example. Suppose 1000 raffle tickets are sold for \$1.00 each to 1000 different people, including Alice. One winner is selected at random from the 1000 ticket holders. The raffle can be modeled by the random variable $X : \Omega \to \{Alice\ wins, Alice\ does\ not\ win\}$. Now, $\Pr(X = \text{Alice wins}) = 1/1000$ and $\Pr(X = \text{Alice does not win}) = 999/1000$.

The amount of information in the event $(X = Alice\ wins)$ is

$$-\log_2(1/1000) = \log_2(1000) = 9.96578\,\text{bits}.$$

In Alice's viewpoint, this is a large amount of information; it would be quite newsworthy and surprising to Alice if she won. The amount of information in the event $(X = Alice\ does\ not\ win)$ is

$$-\log_2(999/1000) = \log_2(1000/999) = 0.001443\,\text{bits}.$$

This is a small amount of information, which makes sense since it is expected that Alice will not win. Overall, the average amount of information in the raffle X is

$$\frac{1}{1000} \cdot 9.96578 + \frac{999}{1000} \cdot 0.001443 = 0.011407\ \text{bits}.$$

3.1.1 Entropy and Randomness: Jensen's Inequality

Given a discrete random variable X taking values in the finite set S, the amount of information in X is the entropy $H(X)$ in bits. $H(X)$ is also the measure of the degree of randomness in the distribution $f_X : S \to [0, 1]$. To support this notion, we compute the entropy of two special distributions.

Example 3.1.2 (Minimal Entropy) Let X be a discrete random variable taking values in the set $S = \{s_1, s_2, \cdots, s_n\}$, $n \geq 1$. Suppose that $f_X(s_1) = 1$ and $f_X(s_i) = 0$ for $2 \leq i \leq n$. There is no randomness or uncertainty in the distribution: it is certain that the output will be s_1.

We calculate the entropy to obtain

$$\begin{aligned} H(X) &= -\sum_{i=1}^{n} f_X(s_i)\log_2(f_X(s_i)) \\ &= -1 \cdot \log_2(1) + \sum_{i=2}^{n} 0 \cdot \log_2(0) \\ &= 0. \end{aligned}$$

The result $H(X) = 0$ is consistent with our observation that X contains no randomness; the amount of randomness in X is 0. □

Example 3.1.3 (Maximal Entropy) Let X be a discrete random variable taking values in the set $S = \{s_1, s_2, \cdots, s_n\}$, $n \geq 1$. Suppose that X is the uniform random variable, i.e., suppose that $f_X(s_i) = \frac{1}{n}$ for $1 \leq i \leq n$. Intuitively, there is a large amount of randomness or uncertainty in the distribution: each outcome is equally likely to occur; in fact, the randomness of the uniform random variable is maximal.

We calculate the entropy to obtain

$$\begin{aligned} H(X) &= -\sum_{i=1}^{n} f_X(s_i)\log_2(f_X(s_i)) \\ &= -\sum_{i=1}^{n} \frac{1}{n} \cdot \log_2\left(\frac{1}{n}\right) \\ &= \log_2(n). \end{aligned}$$

The result $H(X) = \log_2(n)$ is consistent with our observation that X contains a large amount of randomness.

□

All of the other random variables X taking values in $S = \{s_1, s_2, \cdots s_n\}$ have entropy (randomness) $H(X)$ between these extremal values; we show that

$$0 \leq H(X) \leq \log_2(n).$$

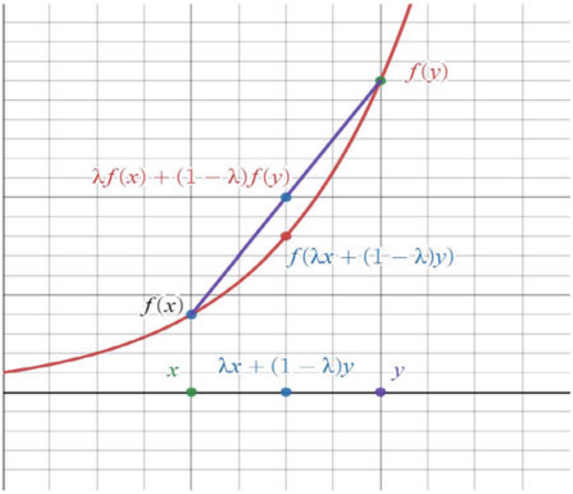


Fig. 3.2 Graph of convex function $y = f(x)$; we have $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$

To this end, we introduce Jensen's Inequality.

A function $f : \mathbb{R} \to \mathbb{R}$ is **convex** on the interval (a, b) if for all $x, y \in (a, b)$, $0 < \lambda < 1$,

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y).$$

For instance, $f(x) = 2^x$ is convex on $(-\infty, \infty)$, cf. Figure 3.2.

A function $f(x)$ is **concave** on (a, b) if $-f(x)$ is convex on (a, b). For example, $f(x) = \log_2(x)$ is concave on $(0, \infty)$ as is $f(x) = \sqrt{x}$.

Proposition 3.1.4 (Jensen's Inequality) *Let* $f : \mathbb{R} \to \mathbb{R}$ *be convex on* $(0, \infty)$. *Suppose that* $\sum_{i=1}^{n} a_i = 1$ *for* $a_i > 0$, $1 \leq i \leq n$, *and that* $x_i > 0$ *for* $1 \leq i \leq n$. *Then*

$$\sum_{i=1}^{n} a_i f(x_i) \geq f(\sum_{i=1}^{n} a_i x_i).$$

Proof Our proof is by induction on n, with the trivial case being $n = 2$.

Trivial Case $n = 2$ Assume that $a_1 + a_2 = 1$, $a_i > 0$, thus $a_2 = 1 - a_1$. Since f is convex on $(0, \infty)$,

$$a_1 f(x_1) + a_2 f(x_2) \geq f(a_1 x_1 + a_2 x_2),$$

for $x_1, x_2 \in (0, \infty)$. So the trivial case holds.

Induction Step Assume that Jensen's inequality holds for n and suppose that $\sum_{i=1}^{n+1} a_i = 1, a_i > 0$. Then for $x_i > 0, 1 \le i \le n+1$,

$$\begin{aligned} f\left(\sum_{i=1}^{n+1} a_i x_i\right) &= f\left(a_{n+1}x_{n+1} + \sum_{i=1}^{n} a_i x_i\right) \\ &= f(a_{n+1}x_{n+1} + (1-a_{n+1})\frac{1}{1-a_{n+1}}\sum_{i=1}^{n} a_i x_i) \\ &\le a_{n+1}f(x_{n+1}) + (1-a_{n+1})f\left(\frac{1}{1-a_{n+1}}\sum_{i=1}^{n} a_i x_i\right), \\ &= a_{n+1}f(x_{n+1}) + (1-a_{n+1})f\left(\sum_{i=1}^{n} \frac{a_i}{1-a_{n+1}} x_i\right), \end{aligned}$$

by the trivial step. Since $\sum_{i=1}^{n} \frac{a_i}{1-a_{n+1}} = 1$,

$$f\left(\sum_{i=1}^{n} \frac{a_i}{1-a_{n+1}} x_i\right) \le \sum_{i=1}^{n} \frac{a_i}{1-a_{n+1}} f(x_i),$$

and so

$$\begin{aligned} a_{n+1}f(x_{n+1}) &+ (1-a_{n+1})f\left(\sum_{i=1}^{n} \frac{a_i}{1-a_{n+1}} x_i\right) \\ &\le a_{n+1}f(x_{n+1}) + (1-a_{n+1})\sum_{i=1}^{n} \frac{a_i}{1-a_{n+1}} f(x_i) \\ &= a_{n+1}f(x_{n+1}) + \sum_{i=1}^{n} a_i f(x_i) \\ &= \sum_{i=1}^{n+1} a_i f(x_i), \end{aligned}$$

which completes the induction proof.

□

Proposition 3.1.5 *Let X be a discrete random variable taking values in the set $S = \{s_1, s_2, \ldots, s_n\}$. Then* $0 \le H(X) \le \log_2(n)$.

Proof Note that $\sum_{i=1}^{n} f_X(s_i) = 1$ with $f_X(s_i) = \Pr(X = s_i) \geq 0$ for $1 \leq i \leq n$. Thus $H(X) \geq 0$. Moreover,

$$\begin{aligned} -H(X) &= \sum_{i=1}^{n} f_X(s_i) \log_2(f_X(s_i)) \\ &= \sum_{i=1}^{n} f_X(s_i) g(\frac{1}{f_X(s_i)}), \end{aligned}$$

where $g(x) = -\log_2(x)$ is convex on $(0, \infty)$. Now by Proposition 3.1.4,

$$-H(X) \geq g\left(\sum_{i=1}^{n} f_X(s_i)\frac{1}{f_X(s_i)}\right) = g(n) = -\log_2(n),$$

and hence $H(X) \leq \log_2(n)$. □

Example 3.1.6 Let X be a random variable taking values in $S = \{s_1, s_2, s_3, s_4\}$ with probability distribution

$$f_{X_2}(s_1) = \frac{1}{4}, \quad f_{X_2}(s_2) = \frac{1}{4}, \quad f_{X_2}(s_3) = \frac{1}{10}, \quad f_{X_2}(s_4) = \frac{4}{10}.$$

Then

$$\begin{aligned} H(X) &= -\sum_{i=1}^{4} f_{X_2}(s_i) \log_2(f_{X_2}(s_i)) \\ &= -\left(\frac{1}{4}\log_2\left(\frac{1}{4}\right) + \frac{1}{4}\log_2\left(\frac{1}{4}\right) + \frac{1}{10}\log_2\left(\frac{1}{10}\right)\right. \\ &\qquad \left. + \frac{4}{10}\log_2\left(\frac{4}{10}\right)\right) \\ &= 1.86096. \end{aligned}$$

Note that a uniform random variable taking values in S has maximum entropy $\log_2(4) = 2$.

□

3.2 Entropy of Plaintext English

English plaintext consists of words over an alphabet A of 26 letters:

$$A = \{\mathtt{A}, \mathtt{B}, \mathtt{C}, \mathtt{D}, \mathtt{E}, \mathtt{F}, \mathtt{G}, \mathtt{H}, \mathtt{I}, \mathtt{J}, \mathtt{K}, \mathtt{L}, \mathtt{M}, \mathtt{N}, \mathtt{O}, \mathtt{P}, \mathtt{Q}, \mathtt{R}, \mathtt{S}, \mathtt{T}, \mathtt{U}, \mathtt{V}, \mathtt{W}, \mathtt{X}, \mathtt{Y}, \mathtt{Z}\}.$$

Let $n \geq 0$. An n-**gram** is a sequence of n letters in A. For example, IBM is a 3-gram, but so is XVQ. There is only one 0-gram, the **empty word** of length 0, usually denoted by ε. For $n \geq 0$, let L_n denote the collection of all n-grams over the alphabet A. Then $L_0 = \{\varepsilon\}$,

$L_1 = \{$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$\}$
$L_2 = \{$AA,AB,AC,AD,AE,AF,AG,AH,AI,AJ,AK, AL,AM,AN,AO,AP,AQ,AR,AS
AT,AU,AV,AW,AX,AY,AZ,BA,BB,BC,BD,BE,BF,BG,BH,BI,BJ,BK, BL
BM,BN,BO,BP,BQ,BR,BS,...,ZX,ZY,ZZ$\}$
$L_3 = \{$AAA,AAB, AAC,...,ZXZ, ZYZ,ZZZ$\}$, and so on.

Note that $|L_0| = 1$, $|L_1| = 26$, $|L_2| = 26^2 = 676$, $|L_3| = 26^3 = 17576$, and in general, for $n \geq 0$,

$$|L_n| = 26^n.$$

For $n \geq 1$, let E_n be the nondeterministic experiment: "an n-gram is observed in English plaintext and it is recorded." Choose L_n for the sample space of E_n. Let $w \in L_n$ and let $\{w\}$ be the event: the n-gram w appears in English plaintext.

We repeat experiment E_n many times and compute the relative frequency $f_n(w)$ that event $\{w\}$ occurs. We do this by obtaining a very large sample T of English plaintext. Then

$$f_n(w) = \frac{\text{number of times that } w \text{ occurs in } T}{\text{number of } n\text{-grams in } T}.$$

In this manner, for each $n \geq 1$, we obtain the n-gram relative frequency distributions:

$$f_n : L_n \to [0, 1].$$

For example, if $n = 1$, we get the 1-gram relative frequency distribution:

$$f_1(\mathtt{A}),\ f_1(\mathtt{B}),\ f_1(\mathtt{C}),\ \ldots,\ f_1(\mathtt{Z}).$$

In fact, f_1 has been calculated for a large sample of (typical) English plaintext T (Figure 3.3).

In table form, the distribution f_1 is given as
The largest 2-gram relative frequencies are given in Figure 3.4. (For a complete listing of all 676 2-gram relative frequencies, see [35, Table 2.3.4]).

For $n \geq 1$, let H_n denote the entropy of the n-gram relative frequency distribution; that is,

$$H_n = -\sum_{w \in L_n} f_n(w) \log_2(f_n(w)).$$

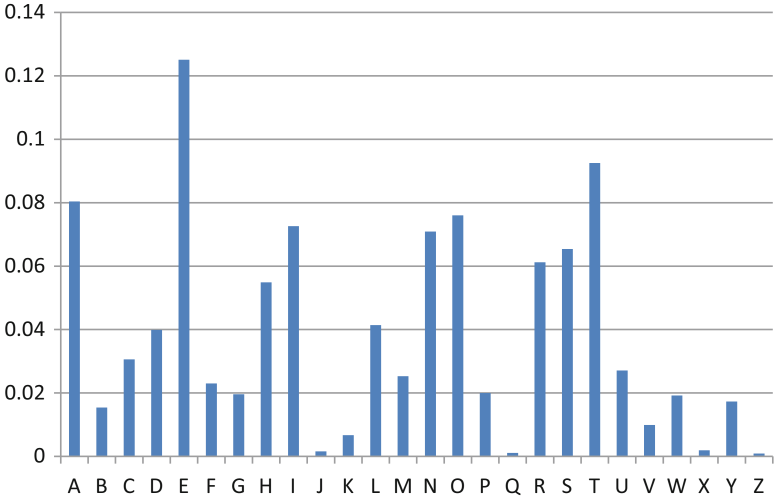


Fig. 3.3 1-gram relative frequency distribution $f_1 : L_1 \rightarrow [0, 1]$

Letter	Prob.	Letter	Prob.
A	0.0804	N	0.0709
B	0.0154	O	0.0760
C	0.0306	P	0.0200
D	0.0399	Q	0.0011
E	0.1251	R	0.0612
F	0.0230	S	0.0654
G	0.0196	T	0.0925
H	0.0549	U	0.0271
I	0.0726	V	0.0099
J	0.0016	W	0.0192
K	0.0067	X	0.0019
L	0.0414	Y	0.0173
M	0.0253	Z	0.0009

Using the distribution of Figure 3.3, one computes 1-gram entropy to be

$$H_1 = -\sum_{w \in L_1} f_1(w) \log_2(f_1(w)) = 4.14.$$

Employing the 2-gram relative frequency distribution $f_2 : L_2 \rightarrow [0, 1]$ [35, Table 2.3.4], one obtains

$$H_2 = -\sum_{w \in L_2} f_2(w) \log_2(f_2(w)) = 7.12.$$

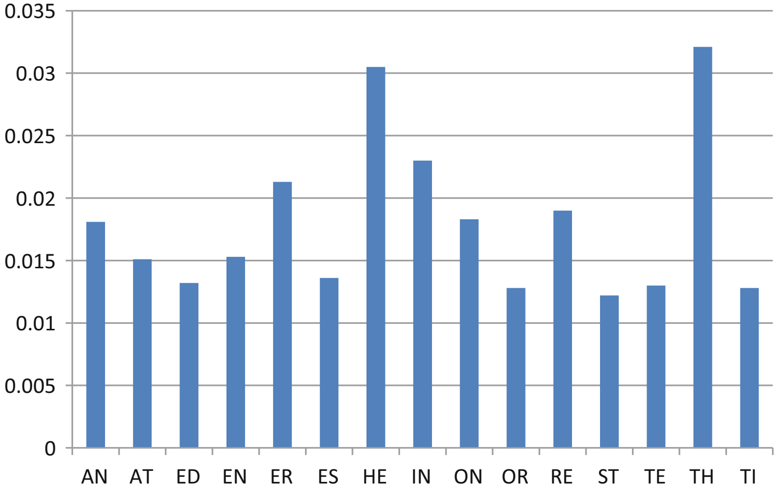


Fig. 3.4 Largest 2-gram relative frequencies $f_2 : L_2 \to [0, 1]$

For each $n \geq 1$, we let H_n/n denote the n**-gram (relative frequency distribution) entropy per letter (entropy rate)**. Consider the sequence of entropy rates:

$$H_1/1, H_2/2, H_3/3, \ldots.$$

Shannon [54, 55] has found that the limit $\lim_{n\to\infty} \frac{H_n}{n}$ exists and has determined its value to be

$$H_\infty = \lim_{n\to\infty} \frac{H_n}{n} \approx 1.5 \text{ bits per letter.}$$

This says that on average English plaintext contains 1.5 bits of information per letter; when we write English it is as if we are using only $2^{1.5} \approx 3$ characters with equal probability of occurrence. Said differently, English plaintext is quite redundant; we need $\log_2(26) = 4.70044$ bits to represent each letter, yet each letter contains only 1.5 bits of information.

From Example 3.1.3, we see that the maximum amount of information per letter in a language of words over an alphabet of 26 letters is

$$\log_2(26) = 4.70044.$$

The **redundancy per letter** of plaintext English is thus defined to be the difference between the maximum information rate and the information rate of English:

$$4.7 - 1.5 = 3.2 \text{ bits per letter.}$$

Since the minimum entropy per letter is 0, the maximum redundancy rate per letter is 4.7. The **redundancy ratio** of plaintext English is therefore

$$\frac{3.2}{4.7} = 68.09\%.$$

This says that plaintext English can be compressed by 68% without losing any information.

3.2.1 ASCII Encoding

In order to process data containing plaintext English, a digital computer must first convert English into a finite sequence of bits or a "block" of bits. The standard way to do this is to use **ASCII (American Standard Code for Information Interchange)**. ASCII is an alphabet consisting of $2^8 = 256$ characters numbered from 0 to 255 and written in base 2 as bytes (8-bit strings). The uppercase letters A,B,...,Z correspond to numbers $65, 66, \ldots 90$ written as bytes $01000001, 01000010, \ldots, 01011010$. Thus,

$$\texttt{A} \leftrightarrow 01000001, \quad \texttt{B} \leftrightarrow 01000010, \quad \texttt{C} \leftrightarrow 01000011,$$

and so on.

The lowercase letters a–z correspond to numbers 97–122 written as bytes 01100001–01111010.

We want to compute the redundancy rate and redundancy ratio of English when it is encoded in ASCII. We assume that English is written in uppercase only.

Let Σ denote the ASCII alphabet, and let L_n denote the collection of all n-grams over Σ, $n \geq 1$. For instance,

$$L_1 = \{00000000, 00000001, 00000010, \ldots, 11111111\},$$

$$L_2 = \{00000000\ 00000000, 00000000\ 00000001, 00000000\ 00000010,$$

$$\ldots, 11111110\ 11111111, 11111111\ 11111111\}.$$

The collection of ordinary 26 uppercase letters of English (encoded as bytes) is a proper subset of L_1; the collection of $26^2 = 676$ 2-grams of English letters is a proper subset of L_2, and so on.

Let $f_1 : L_1 \rightarrow [0, 1]$ denote the 1-gram relative frequency distribution of plaintext English encoded in ASCII as bytes. We may assume that $f_1(b) = 0$ if b does not correspond to an uppercase English letter. Otherwise, if b corresponds to an uppercase English letter, then $f_1(b)$ is equal to the accepted relative frequency of that letter as given in Figure 3.3. Thus $f_1(01000001) = 0.0804$, $f_1(01000010) = 0.0154$, and so on.

The 2-gram relative frequency distribution $f_2 : L_2 \to [0, 1]$ is given similarly using the accepted relative frequency distribution of 2-grams, see Figure 3.4 and [35, Table 2.3.4].

Now,

$$H_1 = -\sum_{w \in L_1} f_1(w) \log_2(f_1(w)) = 4.14$$

and

$$H_2 = -\sum_{w \in L_2} f_2(w) \log_2(f_2(w)) = 7.12,$$

as computed previously. Moreover,

$$H_\infty = \lim_{n \to \infty} \frac{H_n}{n} \approx 1.5 \text{ bits per letter.}$$

Of course, now a letter is a byte.

Since $|\Sigma| = 2^8$, we find that the maximum information rate is $\log_2(2^8) = 8$, and so the redundancy rate of English as encoded in ASCII is

$$8 - 1.5 = 6.5 \text{ bits per letter.}$$

The maximum redundancy rate is 8 and so the redundancy ratio of ASCII encoded English is

$$\frac{6.5}{8} = 81.24\% > 68.09\%.$$

We conclude that encoding English in ASCII increases the redundancy ratio of English.

3.3 Joint and Conditional Entropy

In this section we include some technical material that is needed in our discussion of unicity distance (Section 3.4).

Let $(\Omega, \mathcal{A}, \Pr)$ be an abstract probability space, and let $X : \Omega \to S$, $Y : \Omega \to T$, $S = \{s_1, s_2, \dots, s_m\}$, $T = \{t_1, t_2, \dots, t_n\}$ be random variables. The **joint probability distribution** for X, Y is

$$p_{i,j} = \Pr(X = s_i \cap Y = t_j),$$

for $1 \leq i \leq m$, $1 \leq j \leq n$ The **joint entropy** of X,Y is

$$H(X, Y) = -\sum_{i=1}^{m}\sum_{j=1}^{n} p_{i,j}\log_2(p_{i,j}).$$

The **conditional probability distribution** for $X = s_i$ given $Y = t_j$ is

$$\Pr(X = s_i|Y = t_j),$$

for $1 \leq i \leq m$, $1 \leq j \leq n$.

For fixed j, the **conditional entropy of** X **given** $Y = t_j$ is

$$H(X|Y = t_j) = -\sum_{i=1}^{m}\Pr(X = s_i|Y = t_j)\log_2(\Pr(X = s_i|Y = t_j)),$$

and the **conditional entropy of** X **given** Y is

$$\begin{aligned} H(X|Y) &= \sum_{j=1}^{n}\Pr(Y = t_j)H(X|Y = t_j) \\ &= -\sum_{i=1}^{m}\sum_{j=1}^{n}\Pr(Y = t_j)\Pr(X = s_i|Y = t_j) \\ &\quad\cdot \log_2(\Pr(X = s_i|Y = t_j)). \end{aligned}$$

The conditional entropy of X given Y is an average conditional entropy.

Proposition 3.3.1 *$H(X) \geq H(X|Y)$, with equality holding if and only if X and Y are independent.*

Proof We show that $H(X) - H(X|Y) \geq 0$. By the Total Probability Theorem (Proposition 2.2.1), $\sum_{j=1}^{n}\Pr(Y = t_j|X = s_i) = 1$. Thus

$$\begin{aligned} H(X) &= -\sum_{i=1}^{m}\Pr(X = s_i)\log_2(\Pr(X = s_i)) \\ &= -\sum_{i=1}^{m}\sum_{j=1}^{n}\Pr(X = s_i)\Pr(Y = t_j|X = s_i)\log_2(\Pr(X = s_i)). \end{aligned}$$

Thus $H(X) - H(X|Y)$

$$
\begin{aligned}
&= -\sum_{i=1}^{m}\sum_{j=1}^{n} \Pr(X = s_i)\Pr(Y = t_j|X = s_i)\log_2(\Pr(X = s_i)) \\
&\quad + \sum_{i=1}^{m}\sum_{j=1}^{n} \Pr(Y = t_j)\Pr(X = s_i|Y = t_j)\log_2(\Pr(X = s_i|Y = t_j)) \\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} \Pr(X - s_i \cap Y = s_j)(\log_2(\Pr(X = s_i|Y = t_j)) \\
&\quad - \log_2(\Pr(X = s_i))) \\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} \Pr(X - s_i \cap Y = s_j)\log_2\left(\frac{\Pr(X = s_i|Y = t_j)}{\Pr(X = s_i)}\right) \\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} \Pr(X - s_i \cap Y = s_j)\log_2\left(\frac{\Pr(X = s_i \cap Y = t_j)}{\Pr(X = s_i)\Pr(Y - t_j)}\right). \\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} \Pr(X - s_i \cap Y = s_j)g\left(\frac{\Pr(X = s_i)\Pr(Y - t_j)}{\Pr(X = s_i \cap Y = t_j)}\right),
\end{aligned}
$$

where $g(x) = -\log_2(x)$. Now by Proposition 3.1.4,

$$
\begin{aligned}
H(X) - H(X|Y) &\geq g\left(\sum_{i=1}^{m}\sum_{j=1}^{n} \Pr(X - s_i \cap Y = s_j)\frac{\Pr(X = s_i)\Pr(Y - t_j)}{\Pr(X = s_i \cap Y = t_j)}\right) \\
&= g\left(\sum_{i=1}^{m}\sum_{j=1}^{n} \Pr(X = s_i)\Pr(Y - t_j)\right) \\
&= g(1) \\
&= 0.
\end{aligned}
$$

□

The non-negative quantity $H(X) - H(X|Y)$ is the **mutual information** between X and Y, denoted as $I(X; Y)$. It is the reduction in the randomness of X due to the knowledge of Y; $I(X; Y) = 0$ if and only if X and Y are independent.

Example 3.3.2 Let X and Y be the random variables of Example 2.5.1. Then $I(X; Y) = 0$ since the random variables are independent. □

Example 3.3.3 Let X and Y be the random variables of Example 2.5.2. Then

$$H(X) = \log_2(6),$$

$$\begin{aligned} H(X|Y = \text{odd}) &= -\sum_{i=1}^{6} \Pr(X = i|Y = \text{odd}) \log_2(\Pr(X = i|Y = \text{odd})) \\ &= \log_2(3), \end{aligned}$$

$$\begin{aligned} H(X|Y = \text{even}) &= -\sum_{i=1}^{6} \Pr(X = i|Y = \text{even}) \log_2(\Pr(X = i|Y = \text{even})) \\ &= \log_2(3), \end{aligned}$$

and

$$\begin{aligned} H(X|Y) &= \Pr(Y = \text{odd}) H(X|Y = \text{odd}) + \Pr(Y = \text{even}) H(X|Y = \text{even}) \\ &= \frac{1}{2} \log_2(3) + \frac{1}{2} \log_2(3) \\ &= \log_2(3). \end{aligned}$$

And so the amount of mutual information between X and Y is

$$I(X; Y) = \log_2(6) - \log_2(3) = \log_2(2) = 1.$$

□

Proposition 3.3.4 (The Chain Rule) $H(X, Y) = H(Y) + H(X|Y)$.

Proof By the Total Probability Theorem (Proposition 2.2.1), $\sum_{i=1}^{m} \Pr(X = s_i|Y = t_j) = 1$. Thus

$$\begin{aligned} H(Y) &= -\sum_{j=1}^{n} \Pr(Y = t_j) \log_2(\Pr(Y = t_j)) \\ &= -\sum_{i=1}^{m} \sum_{j=1}^{n} \Pr(Y = t_j) \Pr(X = s_i|Y = t_j) \log_2(\Pr(Y = t_j)). \end{aligned}$$

And so, $H(Y) + H(X|Y)$

$$\begin{aligned}
&= -\sum_{i=1}^{m}\sum_{j=1}^{n}\Pr(Y = t_j)\Pr(X = s_i|Y = t_j)\log_2(\Pr(Y = t_j))\\
&\quad -\sum_{i=1}^{m}\sum_{j=1}^{n}\Pr(Y = t_j)\Pr(X = s_i|Y = t_j)\log_2(\Pr(X = s_i|Y = t_j))\\
&= -\sum_{i=1}^{m}\sum_{j=1}^{n}\Pr(Y = t_j)\Pr(X = s_i|Y = t_j)\\
&\quad \cdot (\log_2(\Pr(Y = t_j) + \log_2(\Pr(X = s_i|Y = t_j)))\\
&= -\sum_{i=1}^{m}\sum_{j=1}^{n}\Pr(Y = t_j)\Pr(X = s_i|Y = t_j)\\
&\quad \cdot \log_2(\Pr(Y = t_j)\Pr(X = s_i|Y = t_j))\\
&= H(X, Y).
\end{aligned}$$

□

Proposition 3.3.5 *$H(X) + H(Y) \geq H(X, Y)$, with equality holding if and only if X and Y are independent.*

Proof By Proposition 3.3.1, $H(X) \geq H(X|Y)$. Thus by Proposition 3.3.4, $H(X) \geq H(X, Y) - H(Y)$. □

3.4 Unicity Distance

Suppose Alice is communicating with Bob using the symmetric cryptosystem

$$\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}\rangle.$$

Here $\mathcal{M}$ is plaintext English, $\mathcal{C}$ is the collection of all possible ciphertext messages, and $\mathcal{K}$ is the keyspace, which is the collection of all possible keys used for encryption and decryption.

Malice would like to break the cryptosystem with a ciphertext only attack, using the brute-force method of key trial (Section 1.3), i.e., Malice will compute $d(C, k)$ for *every possible key k*.

Malice wonders: how much ciphertext needs to be captured in order to uniquely determine the key? What is the minimal length of ciphertext that guarantees that there are no spurious keys?

As we saw in Example 1.3.1, Malice's interception of 4 characters of ciphertext `Egnw` is enough to determine the key $k = 18$. On the other hand, in Example 1.3.2, Malice's knowledge of 3 characters of ciphertext `MFQ` is not enough to uniquely determine the key, though Malice has learned that the key must be either 4, 5, 12, or 13. The correct key is $k = 4$ and 5, 12, and 13 are spurious keys.

Essentially, Malice wants to know how much ciphertext to capture so that the number of spurious keys goes to 0. Our goal in this section is to obtain a lower bound for this value; this lower bound is called the unicity distance of the cryptosystem.

We need to introduce some formal notation.

Let $(\Omega, \mathcal{A}, \Pr)$ be an abstract probability space. Let $n \geq 1$ and let $M_n : \Omega \to L_n \cap \mathcal{M}$, $C_n : \Omega \to L_n \cap \mathcal{C}$, and $K : \Omega \to \mathcal{K}$ denote random variables. Then

$$(M_n = M) = \{\omega \in \Omega : \ M_n(\omega) = M\},$$

$M \in L_n \cap \mathcal{M}$, denotes the event "the plaintext message is the n-gram M"; $\Pr(M_n = M)$ is the probability that event $(M_n = M)$ occurs, i.e., $\Pr(M_n = M)$ is the probability that the plaintext message is the n-gram M.

Likewise,

$$(C_n = C) = \{\omega \in \Omega : \ C_n(\omega) = C\},$$

$C \in L_n \cap \mathcal{C}$, denotes the event "the ciphertext is the n-gram C"; $\Pr(C_n = C)$ is the probability that event $(C_n = C)$ occurs. In other words, $\Pr(C_n = C)$ is the probability that the ciphertext is the n-gram C. Finally,

$$(K = k) = \{\omega \in \Omega : \ K(\omega) = k\},$$

$k \in \mathcal{K}$, denotes the event "Alice and Bob have chosen k as the encryption–decryption key"; $\Pr(K = k)$ is the probability that the encryption–decryption key is k.

Let $C \in L_n \cap \mathcal{C}$ be ciphertext of length n and let

$$W(C) = \{k \in \mathcal{K} : \ d(C, k) \in \mathcal{M}\}.$$

Then $W(C)$ is the set of keys k for which the decryption of C with k results in a meaningful or legible word (or words) in plaintext. Clearly, $|W(C)| \geq 1$, since there is always a correct key k with $M = d(C, k)$, where M is the original intended plaintext.

A **spurious key** is a key $k \in \mathcal{K}$ for which $d(C, k) = M \in \mathcal{M}$, where M is legitimate plaintext other than the original, intended plaintext. The number of spurious keys is $|W(C)| - 1$.

For instance, in Example 1.3.1, $W(\texttt{Egnw}) = \{18\}$ and there are no spurious keys. In Example 1.3.2, $W(\texttt{MFQ}) = \{4, 5, 12, 13\}$ and $\{5, 12, 13\}$ are spurious keys.

The average number of spurious keys over all $C \in L_n \cap \mathcal{C}$ is

$$\text{Spur}(n) = \sum_{C \in L_n \cap \mathcal{C}} \Pr(C_n = C)(|W(C)| - 1).$$

Intuitively,

$$\lim_{n \to \infty} \text{Spur}(n) = 0,$$

and thus (theoretically) there exists a smallest integer $n_0 \geq 1$ for which

$$\text{Spur}(n_0) = 0.$$

Thus, n_0 is the smallest positive integer so that

$$|W(C)| - 1 = 0$$

for all $C \in L_{n_0} \cap \mathcal{C}$. That is, n_0 is the smallest positive integer so that for each $C \in \mathcal{C}$ of length n_0, there is a unique key k that maps C back to a message $M \in \mathcal{M}$.

We seek a lower bound for n_0; this lower bound will be the unicity distance of the cryptosystem.

We first find a lower bound for Spur(n). To this end, we prove two lemmas.

Lemma 3.4.1 $H(K, C_n) = H(K) + H(M_n)$.

Proof We view (K, M_n) as a 2-dimensional random variable. By Proposition 3.3.4,

$$H(C_n, (K, M_n)) = H(K, M_n) + H(C_n|(K, M_n)).$$

We have $H(C_n|(K, M_n)) = 0$ because the ciphertext is determined by the key and the plaintext. Thus, $H(C_n, (K, M_n)) = H(K, M_n)$. Of course, K and M_n are independent, and so,

$$H(C_n, (K, M_n)) = H(K) + H(M_n)$$

by Proposition 3.3.5.

We next view (K, C_n) as a 2-dimensional random variable. By Proposition 3.3.4,

$$H(M_n, (K, C_n)) = H(K, C_n) + H(M_n|(K, C_n)).$$

We have $H(M_n|(K, C_n)) = 0$ because knowledge of the key and the ciphertext yields the correct plaintext (the cryptosystem works). Thus,

$$H(M_n, (K, C_n)) = H(K, C_n).$$

The result follows. □

The **key equivocation** is the conditional entropy $H(K|C_n)$. Key equivocation is a measure of the randomness of the key given that an n-gram of cyphertext has been provided.

Lemma 3.4.2 $H(K|C_n) = H(K) + H(M_n) - H(C_n)$.

Proof By the Chain Rule (Proposition 3.3.4), $H(K|C_n) + H(C_n) = H(K, C_n)$. By Lemma 3.4.1, $H(K|C_n) + H(C_n) = H(K) + H(M_n)$, which gives the result. □

Here is the lower bound on Spur(n).

Proposition 3.4.3 *Let Spur(n) be the average number of spurious keys over all $C \in L_n \cap \mathcal{C}$. Then*

$$Spur(n) \geq \frac{|\mathcal{K}|}{2^{3.2n}} - 1,$$

for $n \geq 1$.

Proof For $n \geq 1$,

$$\begin{aligned}\text{Spur}(n) &= \sum_{C \in L_n \cap \mathcal{C}} \Pr(C_n = C)(|W(C)| - 1)\\ &= \sum_{C \in L_n \cap \mathcal{C}} \Pr(C_n = C)|W(C)| - \sum_{C \in L_n \cap \mathcal{C}} \Pr(C_n = C)\\ &= \sum_{C \in L_n \cap \mathcal{C}} \Pr(C_n = C)|W(C)| - 1,\end{aligned}$$

so that

$$\text{Spur}(n) + 1 = \sum_{C \in L_n \cap \mathcal{C}} \Pr(C_n = C)|W(C)|,$$

thus

$$\log_2(\text{Spur}(n) + 1) = \log_2\left(\sum_{C \in L_n \cap \mathcal{C}} \Pr(C_n = C)|W(C)|\right).$$

By Proposition 3.1.4,

$$\log_2(\text{Spur}(n) + 1) \geq \sum_{C \in L_n \cap \mathcal{C}} \Pr(C_n = C) \log_2(|W(C)|).$$

An application of Proposition 3.1.5 yields $\log_2(|W(C)|) \geq H(K|C_n = C)$ (since $K|(C_n = C)$ is a random variable taking values in $W(C)$), thus

$$\log_2(\mathrm{Spur}(n) + 1) \geq \sum_{C \in L_n \cap \mathcal{C}} \Pr(C_n = C) H(K|C_n = C).$$

Now,

$$\begin{aligned}\log_2(\mathrm{Spur}(n) + 1) &\geq \sum_{C \in L_n \cap \mathcal{C}} \Pr(C_n = C) H(K|C_n = C) \\ &= H(K|C_n) \\ &= H(K) + H(M_n) - H(C_n),\end{aligned}$$

by Proposition 3.4.2.

In Section 3.2, we computed

$$H_\infty = \lim_{n \to \infty} \frac{H_n}{n} = \lim_{n \to \infty} \frac{H(M_n)}{n}.$$

Thus, for large n, $H(M_n) \approx nH_\infty$. Moreover, we may assume that

$$H(K) = \log_2(|\mathcal{K}|).$$

Consequently,

$$H(K) + H(M_n) - H(C_n) \approx \log_2(|\mathcal{K}|) + nH_\infty - H(C_n).$$

Now by Proposition 3.1.5,

$$\begin{aligned}\log_2(|\mathcal{K}|) + nH_\infty - H(C_n) &\geq \log_2(|\mathcal{K}|) + nH_\infty - n\log_2(26) \\ &= \log_2(|\mathcal{K}|) - n(\log_2(26) - H_\infty).\end{aligned}$$

Thus,

$$\log_2(\mathrm{Spur}(n) + 1) \geq \log_2(|\mathcal{K}|) - n(\log_2(26) - H_\infty),$$

and so,

$$\mathrm{Spur}(n) \geq \frac{|\mathcal{K}|}{2^{n(\log_2(26) - H_\infty)}} = \frac{|\mathcal{K}|}{2^{3.2n}} - 1$$

(recall $H_\infty = 1.5$, as computed in Section 3.2). □

By definition, $n_0 \geq 1$ is the smallest integer for which $\text{Spur}(n_0) = 0$. To obtain a lower bound for n_0, we find the unique $m \in \mathbb{R}$ for which

$$0 = \frac{|\mathcal{K}|}{2^{3.2m}} - 1,$$

thus

$$m = \frac{\log_2(|\mathcal{K}|)}{3.2}. \tag{3.1}$$

We then have

$$n_0 \geq m,$$

for if not, then

$$\text{Spur}(n_0) \geq \frac{|\mathcal{K}|}{2^{3.2n_0}} - 1 > 0.$$

Definition 3.4.4 Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ be a symmetric key cryptosystem that encrypts plaintext English. The value

$$m = \frac{\log_2(|\mathcal{K}|)}{3.2}$$

is the **unicity distance** of the cryptosystem.

This value for m is the best lower bound possible for the smallest integer n_0 with $\text{Spur}(n_0) = 0$. Any integer larger than m would not be optimal, and an integer m' smaller than m would result in the inequality

$$\text{Spur}(m') \geq \frac{|\mathcal{K}|}{2^{3.2m'}} - 1 > 0,$$

indicating at least one spurious key.

So, if $|\mathcal{K}| = 26$ (as in our right shift cryptosystem), we compute

$$\begin{aligned} \text{unicity distance} &= \frac{\log_2(26)}{3.2} \\ &= \frac{4.7}{3.2} \\ &= 1.47 \text{ characters of ciphertext.} \end{aligned}$$

For the shift cipher, we therefore deduce that $n_0 \geq 1.47$, and as we have seen in Example 1.3.2, the actual value for n_0 is larger than 1.47.

Now suppose Alice and Bob are using the simple substitution cryptosystem (see Example 8.1.2) with $26! \approx 4 \cdot 10^{26}$ keys. Then the unicity distance is

$$\frac{\log_2(26!)}{3.2} \approx 28 \text{ characters of ciphertext.}$$

Thus for the simple substitution cryptosystem, we conclude that $n_0 \geq 28$; again, the exact value for n_0 might be significantly larger than 28. If Malice captures 28 characters of ciphertext C and performs a brute-force key trial, there *may still be* spurious keys for C.

On the other hand, suppose that the plaintext $\mathcal{M}$ consists of words from a language with a redundancy of 0 bits per letter. Then, assuming $|\mathcal{K}|$ finite,

$$\text{unicity distance} = \frac{\log_2(|\mathcal{K}|)}{0} = \infty.$$

Consequently, $n_0 = \infty$, and it is impossible to uniquely determine the key using key trial.

Since English has redundancy > 0, every cryptosystem with a finite keyspace has a finite unicity distance.

3.5 Exercises

1. Suppose that Alice and Bob wish to exchange a single bit, either 0 or 1. The probability that Alice selects 0 is $3/4$, and the probability that Alice selects 1 is $1/4$. After choosing her bit, Alice then sends the bit to Bob. We model this exchange using a random variable $X : \Omega \to \{0, 1\}$, where $f_X(0) = \Pr(X = 0) = 3/4$ and $f_X(1) = \Pr(X = 1) = 1/4$.

   Compute the entropy $H(X)$ of the random variable X, i.e., compute the amount of information in the transmission of Alice's bit.
2. Suppose that Alice and Bob wish to exchange one of the four letters, either $\mathtt{a}$, $\mathtt{b}$, $\mathtt{c}$, or $\mathtt{d}$. The probability that Alice selects $\mathtt{a}$ is $9/10$, the probability that Alice selects $\mathtt{b}$ is $1/30$, the probability that Alice selects $\mathtt{c}$ is $1/30$, and the probability that Alice selects $\mathtt{d}$ is $1/30$.

   After choosing her letter, Alice then sends the letter to Bob. We model this exchange using a random variable $X : \Omega \to \{\mathtt{a,b,c,d}\}$, where $f_X(\mathtt{a}) = \Pr(X = \mathtt{a}) = 9/10$, $f_X(\mathtt{b}) = \Pr(X = \mathtt{b}) = 1/30$, $f_X(\mathtt{c}) = \Pr(X = \mathtt{c}) = 1/30$, and $f_X(\mathtt{d}) = \Pr(X = \mathtt{d}) = 1/30$.

   Compute the amount of information (entropy) in the random variable X.
3. Let $S = \{0, 1, 2, 3, 4, 5, 6\}$, and let $\xi_6 : \Omega_6 \to S$ denote the binomial random variable with binomial distribution function $f_{\xi_6} : S \to [0, 1]$ defined as

$$f_{\xi_6}(k) = \binom{6}{k} \left(\frac{1}{3}\right)^k \left(\frac{2}{3}\right)^{6-k}.$$

(Note that the probability of "success" is $\frac{1}{3}$.) Compute the entropy $H(\xi_6)$ of the random variable ξ_6.

4. Let $f_1 : L_1 \to [0, 1]$ be the 1-gram relative frequency distribution of plaintext English, as given in Section 3.2. Let

$$H_1 = -\sum_{w \in L_1} f_1(w) \log_2(f_1(w))$$

denote the entropy of the distribution. Write a computer program that computes H_1.

5. Let r be a real number, $0 \leq r \leq 1$. Let $X : \Omega \to \{s_1, s_2\}$ be a random variable with $f_X(s_1) = p$, $f_X(s_2) = 1 - p$, for some p, $0 \leq p \leq 1$. Show that there exists a value for p so that $H(X) = r$.

6. Let $X : \Omega \to \{s_1, s_2, s_3\}$ and $Y : \Omega \to \{t_1, t_2, t_3\}$ be random variables. Suppose that the joint probability distribution of the random vector (X, Y) is given by the table

	t_1	t_2	t_3
s_1	1/18	1/9	1/6
s_2	1/9	1/9	1/18
s_2	1/18	1/18	5/18

Compute the following:

(a) $H(X)$
(b) $H(X|Y = t_1)$
(c) $H(X|Y)$
(d) The mutual information, $I(X; Y)$

7. Suppose plaintext English is encrypted using the cryptosystem $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ where $|\mathcal{K}| = 4096$. Compute the unicity distance of this cryptosystem.

8. Suppose the redundancy of natural language L is 1.2 bits per letter. Plaintext messages written in L are encrypted using cryptosystem $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$. In which scenario below does Malice have a better chance of defeating the cryptosystem with a ciphertext only attack (assuming infinite computing power)?

   Scenario 1: Malice intercepts 20 characters of ciphertext encrypted using a cryptosystem with $|\mathcal{K}| = 33554432$.

   Scenario 2: Malice intercepts 10 characters of ciphertext encrypted using a cryptosystem with $|\mathcal{K}| = 1024$.

9. Suppose that the plaintext English alphabet $\Sigma = \{\mathtt{A}, \mathtt{B}, \mathtt{C}, \ldots, \mathtt{Z}\}$ is encoded in as 5-bit blocks as follows: $\mathtt{A} \leftrightarrow 00000$, $\mathtt{B} \leftrightarrow 00001$, $\mathtt{C} \leftrightarrow 00010, \ldots$, $\mathtt{Z} \leftrightarrow 11001$. As a result of this encoding, plaintext English can be viewed as a collection of words over the alphabet $\{0, 1\}$; an n-gram is a bit string of length n.

(a) Compute the information rate per bit of plaintext English encoded as above, i.e., compute $\lim_{n\to\infty} \frac{H_n}{n}$, where H_n is the entropy of the n-gram relative frequency distribution. *Hint: we know that* $\lim_{n\to\infty} \frac{H_{5n}}{n} = 1.5$. *Use this fact to compute* $\lim_{n\to\infty} \frac{H_{5n}}{5n}$, *and from this deduce* $\lim_{n\to\infty} \frac{H_n}{n}$.

(b) Use part (a) to find the redundancy rate per bit and the redundancy ratio. How does this compare with the standard redundancy ratio of 68%?

Chapter 4
Introduction to Complexity Theory

4.1 Basics of Complexity Theory

Suppose Alice and Bob are using the cryptosystem $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle$ to communicate securely. Recall from Definition 1.2.1 that a cryptosystem is a system

$$\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle,$$

where $\mathcal{M}$ is a collection of plaintext messages, $\mathcal{C}$ is the corresponding set of ciphertext, e is an encryption transformation with keyspace $\mathcal{K}_e$, and d is the decryption transformation with keyspace $\mathcal{K}_d$.

Encryption of messages in $\mathcal{M}$ can be viewed as a function $f : \mathcal{M} \rightarrow \mathcal{C}$; the ciphertext is given as $f(M) = C$ where $M \in \mathcal{M}$ is a message.

In order for the cryptosystem to be secure it should be very hard for Malice to decrypt the ciphertext C *without knowledge of the decryption key* k_d. In other words, without knowledge of k_d, it should be hard to find an inverse function f^{-1} for which $f^{-1}(C) = M$.

Complexity theory seeks to determine the inherent difficulty of a problem by measuring the amount of computer resources (e.g., time or space) that are required for a computer to solve a problem.

The problems we are interested in solving involve the computation of function values, e.g., $f(M) = C$, $f^{-1}(C) = M$. We write algorithms to compute functions. Generally, an **algorithm** is a method, expressed as a finite list of well-defined instructions, which is used to compute a function value.

We want to measure how efficiently an algorithm computes the value of a function. The **running time** on input x of an algorithm is the number of steps the algorithm takes to compute the function value $f(x)$. The running time of an algorithm usually depends on the size of the input.

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_4

When evaluating the performance of an algorithm in terms of its running time, we always consider the worst possible case (we consider inputs which result in the maximum run time). Moreover, since our algorithms will be implemented on a digital computer, we assume that the size of the input is given in terms of bits.

Let $\mathbb{R}$ denote the set of real numbers. For $x \in \mathbb{R}$, let $\lfloor x \rfloor$ denote the largest integer $\leq x$ ($\lfloor x \rfloor$ is the **floor function**). If the input of an algorithm is an integer $n \geq 1$, then its size in terms of bits is

$$m = \lfloor \log_2(n) \rfloor + 1.$$

For example, if the input is $n = 17$, then its size in terms of bits is

$$m = \lfloor \log_2(17) \rfloor + 1 = 4 + 1 = 5 \text{ bits},$$

indeed, $(17)_2 = 10001$. (Note: if n is a decimal integer, we sometimes denote its binary representation as $(n)_2$.)

A convenient way to measure the running time of an algorithm is to use **order notation**.

Definition 4.1.1 (Order Notation) Let $\mathbb{R}_+ = \{x \in \mathbb{R} : x > 0\}$ denote the set of positive real numbers. Let $f : \mathbb{R}_+ \to \mathbb{R}_+$ and $g : \mathbb{R}_+ \to \mathbb{R}_+$ be functions. Then f is of **order** g, denoted as

$$f(x) = O(g(x)),$$

if there exist real numbers $M, N \in \mathbb{R}_+$ for which $f(x) \leq Mg(x)$ for all $x \geq N$.

For example, the polynomial function $p(n) = n^2 + 2n + 3$ is of order n^2, that is, $p(n) = O(n^2)$, since $p(n) \leq 3n^2$ for $n \geq 2$.

In the next section we consider "polynomial time" algorithms, algorithms whose running time is of the order of a polynomial, usually of relatively low degree. Such algorithms are considered to be practical since they can be implemented efficiently on a digital computer.

4.2 Polynomial Time Algorithms

Definition 4.2.1 An algorithm is a **polynomial time algorithm** if its running time $t(m)$ satisfies $t(m) = O(p(m))$ for some polynomial $p(m)$ with integer coefficients, where m is the size of the input of the algorithm as measured in bits.

The fundamental operations of arithmetic ($+$, $-$, $\times$, $\div$) are given by algorithms that run in polynomial time, when the input size is measured in bits. This shows that these operations can be performed efficiently on a computer.

In the algorithms below we use the notation $x \leftarrow y$ to denote that x is assigned the value y.

Algorithm 4.2.2 (BIN_INT_ADD)

```
Input:     integers $a \geq b \geq 0$ written in binary as $a = a_m \cdots a_2 a_1$,
           $b = b_m \cdots b_2 b_1$
Output:    the function value $f(a, b) = a + b = 1d_m d_{m-1} \cdots d_1$
           or $f(a, b) = a + b = d_m d_{m-1} \cdots d_1$ in binary
Algorithm:
           $c \leftarrow 0$
           for $i = 1$ to $m$
              if $a_i + b_i + c = 1$ or 3
                 then $d_i \leftarrow 1$
                 else $d_i \leftarrow 0$
              if $a_i + b_i + c \geq 2$
                 then $c \leftarrow 1$
                 else $c \leftarrow 0$
           next $i$
           if $c = 1$
              then output $1d_m d_{m-1} \cdots d_1$
              else output $d_m d_{m-1} \cdots d_1$
```

□

To see that Algorithm 4.2.2 works, we compute $7 + 5$, or in binary, $111 + 101$.

In this case, $m = 3$. We have $a_1 + b_1 + c = 2$ and so, $d_1 = 0$ and $c = 1$. Next, $a_2 + b_2 + c = 2$, and so $d_2 = 0$ and $c = 1$. Finally, $a_3 + b_3 + c = 3$, thus $d_3 = 1$, $c = 1$. Since $c = 1$, we output 1100, which is the correct sum in binary.

Proposition 4.2.3 *The running time of Algorithm 4.2.2 is $O(m)$ where $m = \lfloor \log_2(a) \rfloor + 1$.*

Proof Recall that $m = 1 + \lfloor \log_2(a) \rfloor$ is the number of bits in the binary representation of $a \geq b$.

The algorithm performs m iterations of the for-next loop. On each iteration the algorithm performs one bit operation. Additionally, we have three more steps: $c \leftarrow 0$, checking if $c = 1$, and the output. Thus the total number of steps is $m + 3$. Since $m + 3 \leq 2m$ for $m \geq 3$, the result follows. □

We next consider subtraction. Let $b = b_m b_{m-1} \ldots b_2 b_1$ be a binary integer, i.e., b is a non-negative decimal integer written in binary. The **1's complement** of b is the binary integer $\overline{b} = \overline{b}_m \overline{b}_{m-1} \ldots \overline{b}_2 \overline{b}_1$ where

$$\overline{b_i} = \begin{cases} 0 \text{ if } b_i = 1 \\ 1 \text{ if } b_i = 0. \end{cases}$$

The 2's **complement** of b is $\overline{b}+1$. We denote the 2's complement of b as $\text{COMP}(b)$. For example, COMP(10111) = 01000 + 1 = 01001 and COMP(0000) = 10000.

Suppose $a = a_m \dots a_1$, $b = b_m \dots b_1$ are binary integers with $a \geq b$. Then, as one can check,

$$a - b = \text{the rightmost } m \text{ bits of } (a + \text{COMP}(b)).$$

For example,

$$\begin{aligned} 101010 - 010111 &= \text{rightmost 6 bits of } (101010 + \text{COMP}(010111)) \\ &= \text{rightmost 6 bits of } (101010 + 101001) \\ &= 010011, \end{aligned}$$

which is correct.

Here is an algorithm that computes $a - b$ assuming that a, b are binary integers with $a \geq b$.

Algorithm 4.2.4 (BIN_INT_SUB)

```
Input:      integers $a \geq b \geq 0$ written in binary as $a = a_m \cdots a_2 a_1$,
            $b = b_m \cdots b_2 b_1$
Output:     $a - b$ in binary
Algorithm:
            $c \leftarrow \text{COMP}(b)$
            $d \leftarrow a + c$
            output $d_m d_{m-1} \dots d_2 d_1$
```

□

Proposition 4.2.5 *The running time of Algorithm 4.2.4 is $O(m)$ where $m = \lfloor \log_2(a) \rfloor + 1$.*

Proof Each step of Algorithm 4.2.4 runs in time $O(m)$, thus the total run time is $O(m)$. □

Let $a = a_m \cdots a_2 a_1$ be the binary representation of the (decimal) integer a. Then adding a to itself (doubling a), denoted as $2a$, can be achieved in polynomial time $O(m)$. The operation $2a$ is performed by shifting the string a to the left one digit and appending a 0 at the end.

Here is an algorithm that multiplies two binary integers.

Algorithm 4.2.6 (BIN_INT_MULT)

```
Input:     integers $a \geq b \geq 0$ written in binary as $a = a_m \cdots a_2 a_1$,
           $b = b_m \cdots b_2 b_1$
Output:    $ab$ in binary
Algorithm:
           $c \leftarrow 0$
           for $i = 1$ to $m$
              if $b_i = 1$ then $c \leftarrow c + a$
              $a \leftarrow 2a$
           next $i$
           output $c$
```

□

To see how this algorithm works, we multiply $3 \cdot 5$. In this case $a = (3)_2 = 011$, $b = (5)_2 = 101$, so $m = 3$. We compute $5 \cdot 3 = 101 \cdot 011$. On the first iteration, c becomes 011, a is 0110. On the second iteration, c remains 011, a is 01100. On the third iteration, c is 01111, which is then output as the correct answer, 15.

Proposition 4.2.7 *The running time of Algorithm 4.2.6 is $O(m^2)$ where $m = \lfloor \log_2(a) \rfloor + 1$.*

Proof The algorithm performs m iterations of the for-next loop. On each iteration the algorithm performs at most $2 \cdot O(m)$ bit operations. Thus the running time is $O(m) \cdot (2 \cdot O(m)) = O(m^2)$. □

We next consider division of (decimal) integers. Given integer $a, n, a \geq 0, n > 0$, we write an algorithm that computes $\lfloor a/n \rfloor$.

Algorithm 4.2.8 (a_DIV_n)

```
Input:     integers $a \geq 0, n > 0$, with $(a)_2 = a_m \ldots a_2 a_1$
Output:    $\lfloor a/n \rfloor$
Algorithm:
           $c \leftarrow 0$
           while $a \geq n$
              for $i = 1$ to $m$
                 if $2^{i-1}n \leq a < 2^i n$
                 then $a \leftarrow a - 2^{i-1}n$ and $c \leftarrow c + 2^{i-1}$
              next $i$
           end-while
           output $c$
```

□

To show how Algorithm 4.2.8 works, we use it to compute $\lfloor 8/5 \rfloor = 1$. We have $a = 8$, $n = 5$, and $(8)_2 = 1000$, hence $m = 4$. Now, with $i = 1$, we have $5 \le 8 < 10$, and so $a = 3$ and $c = 1$. The next three iterations of the for-next loop leave a and c unchanged. Since $3 \not\ge 5$, we output $c = 1$, as required.

Of course, if a is divided by n using the "long division" algorithm of elementary arithmetic, we obtain the division statement

$$a = nq + r,$$

for unique integers q, r with $0 \le r < n$. The value of q is precisely $\lfloor a/n \rfloor$.

In Section 5.3.2, we will see that r coincides with the least non-negative residue $(a \bmod n)$.

Proposition 4.2.9 *The running time of Algorithm 4.2.8 is $O(m^2)$ where $m = \lfloor \log_2(a) \rfloor + 1$.*

Proof The while loop is iterated at most m times and on each iteration we perform m steps each costing $O(1)$. Thus the running time is $O(m^2)$. □

4.3 Non-polynomial Time Algorithms

We next discuss algorithms whose running times are not the order of any polynomial. These "non-polynomial" time algorithms cannot be considered practical or efficient on inputs that are very large.

As an example, we consider an algorithm that decides whether a given integer $n \ge 2$ is prime or composite. In other words, our algorithm computes the function

$$f : \{2, 3, 4, 5, \dots\} \to \{\text{YES,NO}\}$$

defined as

$$f(n) = \begin{cases} \text{YES if } n \text{ is a prime number} \\ \text{NO} \quad \text{if } n \text{ is a composite number.} \end{cases}$$

The algorithm is based on the following well-known fact from number theory.

Proposition 4.3.1 *Let n be a composite number. Then n has an integer factor $\le \sqrt{n}$.*

Proof Write $n = ab$ for $1 < a < n$, $1 < b < n$. We can assume without loss of generality that $a \le b$. Suppose that $a > \sqrt{n}$. Then $b > \sqrt{n}$, thus $ab > \sqrt{n} \cdot \sqrt{n} = n$, a contradiction. □

Thus if n has no integer factors $\le \sqrt{n}$, then it is prime.

Algorithm 4.3.2 (PRIME)

```
Input:     an integer $n \geq 2$
Output:    YES if $n$ is prime, NO if $n$ is composite
Algorithm:
           $d \leftarrow 2$
           $p \leftarrow$ YES
           while $p$ is YES and $d \leq \sqrt{n}$
              if $d \mid n$
              then $p \leftarrow$ NO
              else $d \leftarrow d + 1$
           end-while
           output $p$
```

□

In order to compute the running time for Algorithm 4.3.2, we first need measure the size of the integer input n in terms of bits. We know that the integer n is of size

$$m = \lfloor \log_2(n) \rfloor + 1$$

as measured in bits.

Proposition 4.3.3 *The running time for PRIME is $O(2^{m/2})$, where $m = \lfloor \log_2(n) \rfloor + 1$.*

Proof The steps "$d \leftarrow 2$" and "$p \leftarrow$ YES" count as 2 steps. Moreover, if n is prime, then the while loop is repeated $\lfloor \sqrt{n} \rfloor - 1$ times. This is the maximum number of times it will be repeated as a function of input n. Finally, the "output" step counts as an additional step. Thus the maximal run time is

$$2 + (\lfloor \sqrt{n} \rfloor - 1) + 1 = \lfloor \sqrt{n} \rfloor + 2.$$

Now, since $m \geq \log_2(n)$,

$$\lfloor \sqrt{n} \rfloor + 2 = \lfloor \sqrt{2^{\log_2(n)}} \rfloor + 2 \leq \sqrt{2^m} + 2.$$

Since $\sqrt{2^m} + 2 \leq 2\sqrt{2^m}$ for $m \geq 2$, the running time for PRIME is $O(2^{m/2})$. □

We can show that the running time for PRIME is non-polynomial; the running time $O(2^{m/2})$ is the *best* possible for PRIME.

To this end, observe that

$$2^{(m-1)/2} \leq \lfloor \sqrt{n} \rfloor + 2,$$

for $m \geq 2$. Now, if $\lfloor \sqrt{n} \rfloor + 2 = O(p(m))$ for some polynomial $p(m)$, then $2^{(m-1)/2} = O(p(m))$. Thus, there exists $M, N \in \mathbb{R}_+$ for which

$$2^{(m-1)/2} \leq Mp(m),$$

for all $m \geq N$. Consequently,

$$\lim_{m\to\infty} \frac{2^{(m-1)/2}}{p(m)} = M,$$

which is impossible since

$$\lim_{m\to\infty} \frac{2^{(m-1)/2}}{p(m)} = \infty,$$

by L'Hôpital's rule from calculus.

Thus the running time for PRIME grows exponentially with the size of the input when the input is measured in bits; the running time is the order of the exponential function $2^{m/2}$. Algorithm PRIME is an "exponential time" algorithm.

Definition 4.3.4 An algorithm is an **exponential time algorithm** if its running time $t(m)$ satisfies $t(m) = O(g(m))$ where $g(m)$ is an exponential function of m, and where m is the size of the input as measured in bits.

An algorithm that has exponential running time cannot be considered practical (or efficient) if the input is large.

In between polynomial and exponential time is "subexponential time".

Definition 4.3.5 Let $n \geq 1$ be an integer and let $m = 1 + \lfloor \log_2(n) \rfloor \approx \log_2(n)$, so that polynomial time can be written in powers of $O(m)$ and exponential time is written in powers of $O(n) = O(2^m)$. An algorithm runs in **subexponential time** if its running time is

$$O(2^{\beta(\log_2(n))^\alpha(\log_2(\log_2(n)))^{1-\alpha}}),$$

where $0 < \alpha < 1$ and $\beta > 0$.

If $\alpha = 0$, then

$$O(2^{\beta(\log_2(n))^\alpha(\log_2(\log_2(n)))^{1-\alpha}}) = O(2^{\beta\log_2(\log_2(n))}) = O(\log_2(n)^\beta) = O(m^\beta),$$

which is polynomial time. If $\alpha = 1$, then

$$O(2^{\beta(\log_2(n))^\alpha(\log_2(\log_2(n)))^{1-\alpha}}) = O(2^{\beta\log_2(n)}) = O(2^{\beta m}),$$

and we have exponential time. If α is near 0, then the subexponential time is close to polynomial time, if α is near 1, then the subexponential time is close to exponential time. We will see some subexponential time algorithms in Sections 9.3 and 12.2.

We conclude that the problem of computing a function value is "easy" if there is a polynomial time algorithm that computes the function value. Likewise, the problem of computing a function value is "hard" if there is no polynomial time algorithm that computes the function value.

4.4 Complexity Classes P, PP, BPP

Definition 4.4.1 A **decision problem** $\mathcal{D}$ is a problem whose solution is either "yes" or "no". An **instance** of the decision problem is an input for the problem that needs to be decided. If S is the set of instances of decision problem $\mathcal{D}$, then $\mathcal{D}$ can be viewed as a function $f : S \to \{\text{YES}, \text{NO}\}$; f is the **function associated** to $\mathcal{D}$.

Example 4.4.2 Determining whether a given integer is prime or composite is a decision problem,

$\mathcal{D}$: given integer $n \geq 2$, decide whether n is prime or composite

The set of instances for $\mathcal{D}$ is $S = \{2, 3, 4, \dots\}$, the associated function is

$$f(n) = \begin{cases} \text{YES if } n \text{ is a prime number} \\ \text{NO} \quad \text{if } n \text{ is a composite number} \end{cases}$$

□

Algorithms are used to solve (or decide) decision problems; an algorithm that solves a decision problem actually computes its associated function. For instance, PRIME solves the decision problem $\mathcal{D}$ of Example 4.4.2; PRIME computes the function $f(n)$ of Example 4.4.2.

Decision problems that can be solved by efficient, practical algorithms form a special subclass of problems.

Definition 4.4.3 A decision problem that can be solved in polynomial time (that is, by using a polynomial time algorithm) is a **polynomial time decidable (or solvable) decision problem**. The class of all decision problems that are decidable in polynomial time is denoted as P.

The decision problem which determines whether an integer is prime or composite is in P. Note: this cannot be established using Algorithm 4.3.2 since this algorithm is not polynomial time. See [59, Theorems 3.17 and 3.18].

4.4.1 Probabilistic Polynomial Time

Suppose that we want to write a practical (that is, polynomial time) algorithm to solve a decision problem. Suppose that the best we can do is to devise an algorithm that most of the time computes the correct answer but sometimes yields an incorrect result. Is this kind of algorithm good enough?

For example, let $\mathcal{D}$ be a decision problem with associated function f, $f(n) \in \{\text{YES}, \text{NO}\}$ for each instance n of the problem $\mathcal{D}$.

Suppose that there is no polynomial time algorithm that will solve this decision problem.

What we do have, however, is a polynomial time algorithm A, with input n and output $A(n) \in \{\text{YES}, \text{NO}\}$, with the property that if $f(n) = \text{NO}$, then A will always output NO, and if $f(n) = \text{YES}$, then A will likely output YES. That is, A satisfies the conditional probabilities

$$\Pr(A(n) = \text{YES}|f(n) = \text{YES}) > \frac{1}{2}, \tag{4.1}$$

$$\Pr(A(n) = \text{YES}|f(n) = \text{NO}) = 0. \tag{4.2}$$

Thus, more than half of the time, the polynomial time algorithm A computes the right answer, but there is a chance that it will output the wrong answer. As we will show in Proposition 4.4.5, one can devise a new algorithm that satisfies

$$\Pr(\text{output} = \text{YES}|f(n) = \text{YES}) > 1 - \epsilon,$$

for $\epsilon > 0$. And so, we can consider A an acceptable, practical algorithm for computation.

Our hypothetical algorithm A is a “probabilistic” polynomial time algorithm.

A probabilistic algorithm is allowed to employ a random process in one or more of its steps.

Decision problems which can be solved with probabilistic algorithms that run in polynomial time define a broader class of decision problems than P.

Definition 4.4.4 Let $\mathcal{D}$ be a decision problem with instance n and associated function f, $f(n) \in \{\text{YES}, \text{NO}\}$. Then $\mathcal{D}$ is **decidable (or solvable) in probabilistic polynomial time** if there exists a probabilistic polynomial time algorithm A so that

(i) $\Pr(A(n) = \text{YES}|f(n) = \text{YES}) > \frac{1}{2}$,
(ii) $\Pr(A(n) = \text{YES}|f(n) = \text{NO}) = 0$.

The class of all decision problems that are decidable in probabilistic polynomial time is denoted as PP.

Let $p(x)$ be a **positive polynomial**, that is, $p(x)$ is a polynomial with integer coefficients, for which $p(m) \geq 1$, whenever $m \geq 1$.

Proposition 4.4.5 *Let $\mathcal{D}$ be a decision problem in PP with function f. Suppose n is an instance of $\mathcal{D}$ of size m in bits and let $p(x)$ be a positive polynomial. Then there exists a probabilistic polynomial time algorithm A' so that*

(i) $\Pr(A'(n) = YES|f(n) = YES) > 1 - 2^{-p(m)}$,
(ii) $\Pr(A'(n) = YES|f(n) = NO) = 0$.

Proof Since $\mathcal{D} \in$ PP, there exists a probabilistic polynomial time algorithm A so that

$$\Pr(A(n) = \text{YES}|f(n) = \text{YES}) > \frac{1}{2}$$

and

$$\Pr(A(n) = \text{YES}|f(n) = \text{NO}) = 0.$$

Note that

$$\begin{aligned}\Pr(A(n) = \text{NO}|f(n) = \text{YES}) &= 1 - \Pr(A(n) = \text{YES}|f(n) = \text{YES})\\ &< \frac{1}{2}.\end{aligned}$$

We devise a new algorithm A' as follows: Repeat A $p(m)$ times; if $A(n) = $ YES on any repetition, then set $A'(n) = $ YES, else $A'(n) = $ NO. Thus,

$$\begin{aligned}\Pr(A'(n) = \text{NO}|f(n) = \text{YES}) &= (\Pr(A(n) = \text{NO}|f(n) = \text{YES})^{p(m)}\\ &< 2^{-p(m)}.\end{aligned}$$

Thus $\Pr(A'(n) = \text{YES}|f(n) = \text{YES}) > 1 - 2^{-p(m)}$. Also, if $f(n) = $ NO, then $\Pr(A(n) = \text{YES}) = 0$, thus, $\Pr(A'(n) = \text{YES}|f(n) = \text{NO}) = 0$. □

There is a broader class of decision problems.

Definition 4.4.6 Let $\mathcal{D}$ be a decision problem with instance n and associated function f, $f(n) \in \{\text{YES}, \text{NO}\}$. Then $\mathcal{D}$ is **bounded-error probabilistic polynomial time decidable** if there exists a probabilistic polynomial time algorithm A so that

(i) $\Pr(A(n) = \text{YES}|f(n) = \text{YES}) > \frac{1}{2}$,
(ii) $\Pr(A(n) = \text{YES}|f(n) = \text{NO}) < \frac{1}{2}$.

The class of all decision problem that are decidable in bounded-error probabilistic polynomial time is denoted as BPP.

Proposition 4.4.7 *Let $\mathcal{D}$ be a decision problem in BPP with function f. Suppose n is an instance of $\mathcal{D}$ of size m in bits and let $p(x)$ be a positive polynomial. Then there exists a probabilistic polynomial time algorithm A' so that*

(i) $\Pr(A'(n) = YES|f(n) = YES) > 1 - 2^{-p(m)}$,
(ii) $\Pr(A'(n) = YES|f(n) = NO) < 2^{-p(m)}$.

Proof Since $\mathcal{D} \in$ BPP, there exists a probabilistic polynomial time algorithm A so that

$$\Pr(A(n) = \text{YES}|f(n) = \text{YES}) = \frac{1}{2} + \epsilon$$

and

$$\Pr(A(n) = \text{YES}|f(n) = \text{NO}) = \frac{1}{2} - \delta,$$

for $0 < \epsilon, \delta \le \frac{1}{2}$. □

Without loss of generality, we assume that $\epsilon \le \delta$. Since $4\epsilon^2 > 0, 0 \le 1-4\epsilon^2 < 1$ and we have

$$\lim_{x\to\infty} (x+1)(1-4\epsilon^2)^x = 0.$$

Thus there exists an integer c so that

$$(c+1)(1-4\epsilon^2)^c < \frac{1}{2},$$

and so, $(c+1)^{p(m)}(1-4\epsilon^2)^{cp(m)} < 2^{-p(m)}$. But

$$(cp(m)+1)(1-4\epsilon^2)^{cp(m)} \le (c+1)^{p(m)}(1-4\epsilon^2)^{cp(m)},$$

and so,

$$(cp(m)+1)(1-4\epsilon^2)^{cp(m)} < 2^{-p(m)}.$$

Now $\epsilon \le \delta$ implies $1-4\epsilon^2 \ge 1-4\delta^2$, thus

$$(cp(m)+1)(1-4\delta^2)^{cp(m)} \le (cp(m)+1)(1-4\epsilon^2)^{cp(m)} < 2^{-p(m)}. \tag{4.3}$$

We now prove (i). We devise a new algorithm A' as follows: Repeat A $2cp(m)+1$ times; if $A(n) = $ YES on a majority of the repetitions (at least $cp(m)+1$ times), then $A'(n) = $ YES, otherwise, $A'(n) = $ NO. From the binomial distribution function with success probability $\frac{1}{2} + \epsilon$ (see Example 2.4.2),

$$\Pr(A'(n) = \text{NO} | f(n) = \text{YES})$$

$$\begin{aligned}
&= \sum_{i=0}^{cp(m)} \binom{2cp(m)+1}{i} \left(\frac{1+2\epsilon}{2}\right)^{i} \left(\frac{1-2\epsilon}{2}\right)^{2cp(m)+1-i} \\
&\leq (cp(m)+1)\binom{2cp(m)+1}{cp(m)} \left(\frac{1+2\epsilon}{2}\right)^{cp(m)} \\
&\qquad \cdot \left(\frac{1-2\epsilon}{2}\right)^{cp(m)} \left(\frac{1-2\epsilon}{2}\right) \\
&= \frac{(cp(m)+1)(1-2\epsilon)}{2}\binom{2cp(m)+1}{cp(m)} \left(\frac{1-4\epsilon^2}{4}\right)^{cp(m)} \\
&\leq \frac{(cp(m)+1)}{2}(2^{2cp(m)+1}) \left(\frac{1-4\epsilon^2}{4}\right)^{cp(m)} \\
&= (cp(m)+1)(1-4\epsilon^2)^{cp(m)} \\
&< 2^{-p(m)} \quad \text{by (4.3).}
\end{aligned}$$

Thus,

$$\Pr(A'(n) = \text{YES} | f(n) = \text{YES}) > 1 - 2^{-p(m)}.$$

Since A runs in polynomial time and A' repeats A a polynomial number of times, A' is polynomial time. Thus (i) follows.

For (ii): $\Pr(A'(n) = \text{YES} | f(n) = \text{NO})$

$$\begin{aligned}
&= \sum_{i=cp(m)+1}^{2cp(m)+1} \binom{2cp(m)+1}{i} \left(\frac{1-2\delta}{2}\right)^{i} \left(\frac{1+2\delta}{2}\right)^{2cp(m)+1-i} \\
&\leq (cp(m)+1)\binom{2cp(m)+1}{cp(m)+1} \left(\frac{1-2\delta}{2}\right)^{cp(m)+1} \left(\frac{1+2\delta}{2}\right)^{cp(m)} \\
&= \frac{(cp(m)+1)(1-2\delta)}{2}\binom{2cp(m)+1}{cp(m)+1} \left(\frac{1-4\delta^2}{4}\right)^{cp(m)} \\
&\leq \frac{(cp(m)+1)}{2}(2^{2cp(m)+1}) \left(\frac{1-4\delta^2}{4}\right)^{cp(m)} \\
&= (cp(m)+1)(1-4\delta^2)^{cp(m)} \\
&< 2^{-p(m)} \quad \text{by (4.3).}
\end{aligned}$$

Proposition 4.4.8 *$P \subseteq PP \subseteq BPP$.*

Proof This follows from the definitions of P, PP, and BPP. □

Remark 4.4.9 If A is any computer algorithm, then a **Turing machine** capable of simulating the logical structure of A can be constructed. This is a special case of the **Church–Turing thesis** , which states that anything that can be computed can be computed by some Turing machine. (A Turing machine is a special type of finite automaton, see [59, Section 2.2], [27, Chapter 2, Chapter 8].)

Thus in the definition of complexity classes P, PP, BPP given above, one can replace "probabilistic polynomial time algorithm" with "probabilistic Turing machine", see [59, Chapter 4]. □

4.4.2 An Example

We give an example of a decision problem in BPP. Suppose Alice and Bob are using the right shift cryptosystem $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ with symmetric key k (Section 1.1). Here $\mathcal{M}$ consists of all 3-grams over the alphabet $\{\mathtt{A} \dots \mathtt{Z}\}$ that are recognizable 3-letter words in plaintext English and $\mathcal{C}$ consists of all encryptions of words in $\mathcal{M}$ using the correct key k, i.e., $\mathcal{C} = e(\mathcal{M}, k)$.

For $M \in \mathcal{M}$, let $C = e(M, k)$ denote the encryption of M. Then

$$\Pr(d(C, k) \notin \mathcal{M}) = 0.$$

For l, $0 \le l \le 25$, with $l \neq k$,

$$\Pr(d(C, l) \notin \mathcal{M}) > \frac{1}{2}.$$

Let $\mathcal{D}$ be the decision problem:

Given an integer l, $0 \le l \le 25$, decide whether l is the key.

The associated function for $\mathcal{D}$ is $f : \{0, 1, 2, \dots, 25\} \to \{\text{YES}, \text{NO}\}$ with

$$f(l) = \begin{cases} \text{YES if } l = k \\ \text{NO} \quad \text{if } l \neq k. \end{cases}$$

Proposition 4.4.10 *$\mathcal{D} \in BPP$.*

Proof We have to show that there is a probabilistic polynomial time algorithm A so that, for instance, l of $\mathcal{D}$,

$$\Pr(A(l) = \text{YES} | f(l) = \text{YES}) > \frac{1}{2},$$

and

$$\Pr(A(l) = \text{YES} | f(l) = \text{NO}) < \frac{1}{2}.$$

To this end, we consider the algorithm

Algorithm (IS_KEY)

```
Input:      an integer $l \in \{0, 1, 2, \ldots, 25\}$
Algorithm:
            choose $C$ at random from $\mathcal{C}$
            $M \leftarrow d(C, l)$
            if $M \in \mathcal{M}$ then output YES
            else output NO
```

□

If $l = k$, i.e., if $f(l) = \text{YES}$, then $\Pr(A(l) = \text{YES}) = 1$. Thus

$$\Pr(A(l) = \text{YES} | f(l) = \text{YES}) = 1 > \frac{1}{2}.$$

On the other hand, if $l \neq k$, then

$$\Pr(d(C, l) \in \mathcal{M}) = 1 - \Pr(d(C, l) \notin \mathcal{M}) < \frac{1}{2}.$$

Thus, if $f(l) = \text{NO}$, then $\Pr(A(l) = \text{YES}) < \frac{1}{2}$, and so,

$$\Pr(A(l) = \text{YES} | f(l) = \text{NO}) < \frac{1}{2}.$$

Computation of $d(C, l)$ uses the basic operations $+$, $-$, and so the algorithm runs in polynomial time. Thus $\mathcal{D} \in \text{BPP}$. □

Here is an instance where IS_KEY returns the wrong answer. Suppose that the correct key is $k = 4$. Then

$$\texttt{MFQ} = e(\texttt{IBM}, 4),$$

and so $\texttt{MFQ} \in \mathcal{C}$. Suppose that the input to IS_KEY is $l = 12$. Then $f(12) = \text{NO}$. We now run the algorithm IS_KEY on input $l = 12$. Suppose that the randomly chosen element of $\mathcal{C}$ is $\texttt{MFQ}$. Then

$$\texttt{ATE} = d(\texttt{MFQ}, 12),$$

with $\texttt{ATE} \in \mathcal{M}$. Consequently, $A(12) = \text{YES}$, which is incorrect.

Of course, by Proposition 4.4.5, IS_KEY can be modified to give an algorithm which is nearly always correct: Let l be an integer, $0 \le l \le 25$, of input size $m = \lfloor \log_2(l) \rfloor + 1$, and let $p(m)$ be a positive polynomial. By Proposition 4.4.5 there exists a probabilistic polynomial time algorithm A' with

$$\Pr(A'(l) = \text{YES} | f(l) = \text{YES}) = 1 > 1 - 2^{-p(m)},$$

$$\Pr(A'(n) = \text{YES} | f(n) = \text{NO}) < 2^{-p(m)}.$$

4.5 Probabilistic Algorithms for Functions

Suppose that $\mathcal{D}$ is a decision problem in BPP with associated function f. Then there exists a polynomial time algorithm A so that

$$\Pr(A(n) = \text{YES} | f(n) = \text{YES}) = \frac{1}{2} + \epsilon,$$

$$\Pr(A(n) = \text{YES} | f(n) = \text{NO}) = \frac{1}{2} - \delta,$$

for some $\epsilon, \delta > 0$. Thus

$$\Pr(A(n) = \text{NO} | f(n) = \text{NO}) = \frac{1}{2} + \delta.$$

Proposition 4.5.1 *Let $\mathcal{D} \in BPP$ and let n be an instance of $\mathcal{D}$. Then*

$$\Pr(A(n) = f(n)) > \frac{1}{2}.$$

Proof Suppose that $r = \Pr(f(n) = \text{YES})$, so that $\Pr(f(n) = \text{NO}) = 1 - r$. Then

$$\begin{aligned}\Pr(A(n) = f(n)) &= r \Pr(A(n) = \text{YES} | f(n) = \text{YES}) \\ &\quad + (1 - r) \Pr(A(n) = \text{NO} | f(n) = \text{NO}) \\ &= r(\frac{1}{2} + \epsilon) + (1 - r)\left(\frac{1}{2} + \delta\right) \\ &= \frac{r}{2} + r\epsilon + \frac{1}{2} + \delta - \frac{r}{2} - r\delta\end{aligned}$$

$$= \frac{1}{2} + r\epsilon + (1-r)\delta$$
$$> \frac{1}{2}.$$

□

From Proposition 4.5.1 we see that A is a polynomial time algorithm that computes f in the sense that *it is likely to compute* f for an arbitrary instance. We say that A is a **probabilistic polynomial time algorithm** that computes the function f.

In Chapter 9 we will encounter probabilistic polynomial time algorithms that attempt to compute other functions.

4.6 Exercises

1. How may bits are required to represent the decimal integer $n = 237$ in binary?
2. Compute the size of the decimal integer $n = 39$ in bits. Compute $(39)_2$.
3. Let A be an algorithm with running time $O(n^3)$ where n is an integer. Compute the running time as a function of bits. Is A a polynomial time algorithm?
4. Use the algorithm a_DIV_n to compute $\lfloor 45/4 \rfloor$.
5. Assume that n is an even integer, $n \geq 0$. Write an algorithm that computes the function $f(n) = n/2$ and determine whether the algorithm is efficient.
6. Assume that n is an integer, $n \geq 0$. Write an algorithm that computes the function $f(n) = 2^n$ and determine the running time of the algorithm.
7. In this exercise we consider another algorithm that adds integers. The **unary representation** of an integer $a \geq 0$, is a string of 1's of length a. For example, 8 in unary is

$$11111111.$$

Suppose a is written in unary. Let $a+$ denote the unary representation formed by appending a 1 to a and let $a-$ denote the unary representation formed by deleting a 1 from the end of a.

Algorithm (UN_INT_ADD)

```
Input:      integers $a \geq b \geq 0$ encoded in unary
Output:     $a + b$ encoded in unary
Algorithm:
            while $b \neq 0$ do
              $a \leftarrow a+$
              $b \leftarrow b-$
```

```
    end-while
    output $a$
```

□

Compute the running time of **UN_INT_ADD**.

8. Let $n \geq 1$ be an integer and let S be a set of n positive integers. The well-known algorithm MERGE_SORT sorts the elements of S from smallest to largest [65].

   It is known that the running time of MERGE_SORT is $O(n \log_2(n))$, where the input n is the size of the set S to be sorted. Show that MERGE_SORT is non-polynomial time.

9. The **bit-wise truth table for exclusive AND** (XAND) is given as

x	y	XAND(x, y)
0	0	1
0	1	0
1	0	0
1	1	1

Let $[0, 1]^m$ denote the set of all binary strings of length m and let

$$f : [0, 1]^m \times [0, 1]^m \to [0, 1]^m$$

be the function defined as

$$f(a, b) = c,$$

where $a = a_m a_{m-1} \cdots a_2 a_1$, $b = b_m b_{m-1} \cdots b_2 b_1$ are binary strings of length m and and $c = c_m c_{m-1} \cdots c_2 c_1$ is the binary string where $c_i = \text{XAND}(a_i, b_i)$, for $1 \leq i \leq m$.

The following algorithm computes f:

Algorithm

```
Input:      two strings in binary $a = a_m a_{m-1} \cdots a_2 a_1$,
            $b = b_m b_{m-1} \cdots b_2 b_1$
Algorithm:
            for $i = 1$ to $m$
                if $a_i = b_i$
                then $c_i \leftarrow 1$
                else $c_i \leftarrow 0$
            next $i$
            output $c_m c_{m-1} \cdots c_2 c_1$
```

(a) Compute the output of the algorithm if the input is $a = 1100$, $b = 0100$.
(b) Determine the running time of the algorithm.

10. Let $f : \mathbb{R}_+ \to \mathbb{R}_+$, $g : \mathbb{R}_+ \to \mathbb{R}_+$ be functions with $f(x) = O(x)$, and $g(x) = O(x^2)$. Let $h(x) = f(x)g(x)$. Show that $h(x) = O(x^3)$.
11. Let A be an algorithm that runs in time $O(m^2)$. Show that A runs in time $O(m^r)$ for $r \geq 2$.
12. Let A be an algorithm that runs in subexponential time $O(2^{\sqrt{m}\sqrt{\log_2(m)}})$, where the input has length m in bits. Show that A is not polynomial time.
13. Let $\mathcal{D}$ be the decision problem:

    Given two integers $a \geq 0$, $n > 0$, determine whether n divides a evenly (denoted as $n \mid a$).

    Show that $\mathcal{D} \in$ P.
14. Let $\mathcal{D}$ be the decision problem:

    Let $p(x)$ be a polynomial of degree d integer coefficients. Determine whether $p(x)$ is constant, i.e., $p(x) = c$ for some integer c.

    Show that $\mathcal{D} \in$ PP.
15. Let $\mathcal{D}$ be a decision problem with function f. Let n be an instance of $\mathcal{D}$ and suppose there exists a probabilistic polynomial time algorithm A so that

    (i) $\Pr(A(n) = \text{YES} | f(n) = \text{YES}) \geq c$,
    (ii) $\Pr(A(n) = \text{YES} | f(n) = \text{NO}) = 0$,

    where $0 < c \leq 1$. Show that $\mathcal{D} \in$ PP.
16. Let $\mathcal{D}$ be a decision problem with function f and let n be an instance of $\mathcal{D}$ of size m. Let $p(x)$ be a positive polynomial for which $p(m) \geq 2$. Suppose there exists a probabilistic polynomial time algorithm A so that

    (i) $\Pr(A(n) = \text{YES} | f(n) = \text{YES}) \geq \frac{1}{p(m)}$,
    (ii) $\Pr(A(n) = \text{YES} | f(n) = \text{NO}) = 0$.

    Show that $\mathcal{D} \in$ PP.

Chapter 5
Algebraic Foundations: Groups

Algebraic concepts such as groups, rings, and fields are essential for the study of symmetric key and public key cryptography.

5.1 Introduction to Groups

Let S be a non-empty set of elements. A **binary operation** on S is a function

$$B : S \times S \to S.$$

We denote the image $B(a, b)$ by ab. A binary operation is **commutative** if for all $a, b \in S$,

$$ab = ba;$$

it is **associative** if for all $a, b, c \in S$,

$$a(bc) = (ab)c.$$

For example, let

$$\mathbb{Z} = \{\cdots - 3, -2, -1, 0, 1, 2, 3, \dots\}$$

denote the set of integers. Then ordinary addition,

$$+ : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}, \quad (a, b) \mapsto a + b,$$

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_5

is a binary operation on $\mathbb{Z}$. Since $a + b = b + a$, and $a + (b + c) = (a + b) + c, \forall a, b, c \in \mathbb{Z}$, ordinary addition is a commutative and associative binary operation on $\mathbb{Z}$.

A **semigroup** is a non-empty set S together with an associative binary operation $S \times S \to S$. For example, the set $\mathbb{Z}$ together with ordinary addition (sometimes denoted as $\langle \mathbb{Z}, + \rangle$) is a semigroup.

A **monoid** is a semigroup S for which there exists an element $e \in S$ with

$$ea = a = ae,$$

$\forall a \in S$. The element e is an **identity element** for the monoid. If we take $e = 0$, then the semigroup $\langle \mathbb{Z}, + \rangle$ is also a monoid, since $\forall a \in \mathbb{Z}, a + 0 = a = 0 + a$.

Here is an important example of a monoid. An **alphabet** is a finite set $\Sigma = \{s_1, s_2, \ldots, s_k\}$ whose elements are the **letters** of the alphabet. A **word** over Σ is a finite sequence of letters in Σ. A word over Σ can be written as

$$x = a_1 a_2 a_3 \ldots a_l,$$

where $a_i \in \Sigma$ for $1 \le i \le l$; the **length** of word x is the number of letter in x.

Let Σ be an alphabet. The **closure** of Σ, denoted by Σ^*, is the collection of all words over Σ. The closure Σ^* contains a unique word of length 0, called the **empty word**, which is denoted as ε.

On Σ^* there is a binary operation called **concatenation**, denoted by $\cdot$, which is defined as follows. Let $x = a_1 a_2 \ldots a_l$, $y = b_1 b_2 \ldots b_m$ be two words in Σ^*. Then

$$x \cdot y = xy = a_1 a_2 \ldots a_l b_1 b_2 \ldots b_m.$$

One easily shows that $\cdot$ is an associative binary operation: For words $x, y, z \in \Sigma^*$, we have

$$\begin{aligned} x \cdot (y \cdot z) &= x \cdot yz \\ &= xyz \\ &= xy \cdot z \\ &= (x \cdot y) \cdot z, \end{aligned}$$

thus $\langle \Sigma^*, \cdot \rangle$ is a semigroup. In fact, if we take e to be the empty word ε, then $\langle \Sigma^*, \cdot \rangle$ is a monoid, which is the **monoid of words** over Σ.

We have seen this monoid before. For our alphabet, take

$$\Sigma = \{\mathtt{A}, \mathtt{B}, \mathtt{C}, \mathtt{D}, \mathtt{E}, \mathtt{F}, \mathtt{G}, \mathtt{H}, \mathtt{I}, \mathtt{J}, \mathtt{K}, \mathtt{L}, \mathtt{M}, \mathtt{N}, \mathtt{O}, \mathtt{P}, \mathtt{Q}, \mathtt{R}, \mathtt{S}, \mathtt{T}, \mathtt{U}, \mathtt{V}, \mathtt{W}, \mathtt{X}, \mathtt{Y}, \mathtt{Z}\}.$$

Then

$$\Sigma^* = \bigcup_{n \ge 0} L_n = L_0 \cup L_1 \cup L_2 \cup \cdots,$$

where L_n is the collection of all n-grams over Σ; $\langle \Sigma^*, \cdot, \varepsilon \rangle$ is a monoid. In this monoid, for instance,

$$\texttt{SCARLET} \cdot \texttt{FIRE} = \texttt{SCARLETFIRE}.$$

The set of plaintext English words is a small subset of Σ^*; given a cryptosystem, the set of all possible ciphertext encryptions of these plaintext English words is some other subset of Σ^*.

A group is a monoid with an additional property.

Definition 5.1.1 A **group** is a non-empty set G together with a binary operation $G \times G \to G$ for which

(i) the binary operation is associative;
(ii) there exists an element $e \in G$ for which $ea = a = ae$, for all $a \in G$;
(iii) for each $a \in G$, there exists an element $c \in G$ for which $ca = e = ac$.

An element e satisfying (ii) is an **identity element** for G; an element c satisfying (iii) is an **inverse element** of a and is denoted by a^{-1}.

The **order** $|G|$ of a group G is the number of elements in G. If G is a finite set, i.e., $|G| = n < \infty$, then G is a **finite** group. If G is not finite, then G is an **infinite** group.

A group G in which the binary operation is commutative is an **abelian** group.

The monoid Σ^* of words over the alphabet Σ is not abelian and is not a group.

5.2 Examples of Infinite Groups

Example 5.2.1 $\langle \mathbb{Z}, + \rangle$ is an infinite group. To prove this we check that conditions (i), (ii), (iii) of Definition 5.1.1 hold. For $a, b, c \in \mathbb{Z}$,

$$a + (b + c) = (a + b) + c,$$

thus $+$ is an associative binary operation, so (i) holds. Setting $e = 0$, we see that

$$a + 0 = a = 0 + a,$$

so 0 is an identity element, so (ii) holds, and finally, for each $a \in \mathbb{Z}$, let $c = -a$, then

$$a + (-a) = 0 = (-a) + a,$$

so $-a$ is an inverse for a, and so (iii) holds.

Clearly, $\mathbb{Z}$ is an infinite set.

□

Since

$$a + b = b + a,$$

for all $a, b \in \mathbb{Z}$, $\langle \mathbb{Z}, + \rangle$ is an infinite abelian group with an "additive" binary operation.

Let $\mathbb{Q}$ denote the set of rational numbers, and let $\mathbb{R}$ denote the set of real numbers. Then $\langle \mathbb{Q}, + \rangle$ and $\langle \mathbb{R}, + \rangle$ are infinite abelian additive groups.

Example 5.2.2 Let $\mathbb{Q}_+$ denote the set of positive rational numbers, that is,

$$\mathbb{Q}_+ = \{a \in \mathbb{Q} \mid a > 0\}.$$

Let $\cdot$ denote ordinary multiplication. Then $\langle \mathbb{Q}_+, \cdot \rangle$ is an infinite group. To prove this, we show that conditions (i), (ii), and (iii) of Definition 5.1.1 hold: For $a, b, c \in \mathbb{Q}_+$,

$$a \cdot (b \cdot c) = (a \cdot b) \cdot c,$$

thus $\cdot$ is an associative binary operation, so (i) holds. Setting $e = 1$, we see that

$$a \cdot 1 = a = 1 \cdot a,$$

so 1 is an identity element, so (ii) holds, and for each $a \in \mathbb{Q}_+$, let $c = \frac{1}{a}$, then

$$a \cdot \frac{1}{a} = 1 = \frac{1}{a} \cdot a,$$

so $\frac{1}{a}$ is an inverse for a, and so (iii) holds. Finally, $\mathbb{Q}$ is not finite, thus $\langle \mathbb{Q}_+, \cdot \rangle$ is an infinite group.

Since

$$a \cdot b = b \cdot a, \quad \forall a, b \in \mathbb{Q}_+,$$

$\langle \mathbb{Q}_+, \cdot \rangle$ is an infinite abelian group with a "multiplicative" binary operation.

□

Example 5.2.3 Let $\mathbb{R}^\times$ denote the set of non-zero real numbers, i.e.,

$$\mathbb{R}^\times = \{x \in \mathbb{R} \mid x \neq 0\}.$$

Then one can easily show that $\langle \mathbb{R}^\times, \cdot \rangle$ is an infinite abelian group.

□

Example 5.2.4 Let $\mathrm{GL}_2(\mathbb{R})$ denote the collection of all invertible 2×2 matrices with entries in $\mathbb{R}$. Let $\cdot$ denote ordinary matrix multiplication. Then $\langle \mathrm{GL}_2(\mathbb{R}), \cdot \rangle$ is an infinite multiplicative group which is not abelian.

Proof Recall from linear algebra that matrix multiplication is associative, moreover $e = I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ serves as an identity element. Since $A \in \mathrm{GL}_2(\mathbb{R})$ is invertible, there exists $A^{-1} \in \mathrm{GL}_2(\mathbb{R})$ with

$$AA^{-1} = I_2 = A^{-1}A,$$

thus $\langle \mathrm{GL}_2(\mathbb{R}), \cdot \rangle$ is an infinite multiplicative group. Since

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \neq \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix},$$

$\mathrm{GL}_2(\mathbb{R})$ is not abelian. □

More generally, the collection of $l \times l$ invertible matrices over $\mathbb{R}$, denoted as $\mathrm{GL}_l(\mathbb{R})$ is a group under matrix multiplication. In the case $l = 1$, we identify $\mathrm{GL}_1(\mathbb{R})$ with $\mathbb{R}^\times$.

When studying group theory, non-examples are just as important as examples! For instance, let Σ be the alphabet $\{0, 1\}$. Then $\Sigma^* = \{0, 1\}^*$ is the collection of all finite sequences of 0's and 1's; $\{0, 1\}^*$ is infinite.

If we take e to be the empty string, then $\{0, 1\}^*$ is a monoid under the binary operation concatenation. However, $\{0, 1\}^*$ fails to be a group since no element other than e has an inverse (Definition 5.1.1(iii) fails).

5.3 Examples of Finite Groups

Recall that a group G is finite if $|G| = n < \infty$. We give two examples of finite groups that are important in cryptography.

5.3.1 The Symmetric Group on n Letters

Let $\Sigma = \{0, 1, 2, 3, \ldots, n-1\}$ be an alphabet of n letters. A **permutation** of Σ is a function $\sigma : \Sigma \to \Sigma$ that is both 1-1 and onto (equivalently: σ is a one-to-one correspondence, or σ is a bijection).

Let $\sigma : \Sigma \to \Sigma$ be a permutation of Σ. For $i \in \Sigma$, we let $\sigma(i) \in \Sigma$ denote the image of i under σ. We can write σ in convenient **permutation notation**:

$$\sigma = \begin{pmatrix} 0 & 1 & 2 & 3 & \cdots & n-2 & n-1 \\ \sigma(0) & \sigma(1) & \sigma(2) & \sigma(3) & \cdots & \sigma(n-2) & \sigma(n-1) \end{pmatrix},$$

in which the domain of σ, $\Sigma = \{0, 1, 2, 3, \ldots, n-1\}$, is written along the top row, and under each domain value i, we write its image $\sigma(i)$, forming the bottom row. For instance, if $\Sigma = \{0, 1, 2\}$, and $\sigma : \Sigma \to \Sigma$ is the permutation defined as $\sigma(0) = 1, \sigma(1) = 2, \sigma(2) = 0$, then

$$\sigma = \begin{pmatrix} 0\ 1\ 2 \\ 1\ 2\ 0 \end{pmatrix}.$$

There are

$$n! = n \times (n-1) \times (n-2) \times \cdots \times 3 \times 2 \times 1$$

possible permutations of the set Σ of n letters.

Let S_n denote the collection of all permutations of Σ, $|S_n| = n!$ We define a binary operation on S_n

$$\circ : S_n \times S_n \to S_n$$

by the rule: for $\sigma, \tau \in S_n, i \in \Sigma$,

$$(\sigma \circ \tau)(i) = \sigma(\tau(i)).$$

Note that $\sigma \circ \tau \in S_n$; $\circ$ is ordinary function composition.

Proposition 5.3.1 *$\langle S_n, \circ \rangle$ is a group.*

Proof We show that conditions (i), (ii), (iii) of Definition 5.1.1 hold. Let $\sigma, \tau, \rho \in S_n$. Then $\sigma \circ (\tau \circ \rho) = (\sigma \circ \tau) \circ \rho$, since function composition is always associative.

Next, let e be the identity permutation:

$$e = \begin{pmatrix} 0\ 1\ 2\ 3 \cdots n-2\ n-1 \\ 0\ 1\ 2\ 3 \cdots n-2\ n-1 \end{pmatrix}.$$

Then $e \circ \sigma = \sigma = \sigma \circ e$, so (ii) holds. Finally, let $\sigma \in S_n$. Then $\sigma : \Sigma \to \Sigma$ is a bijection, thus there exists an inverse function $\sigma^{-1} : \Sigma \to \Sigma$, which is also a permutation. Moreover, $\sigma^{-1} \circ \sigma = e = \sigma \circ \sigma^{-1}$, thus (iii) holds.

It follows that $\langle S_n, \circ \rangle$ is a group of order $|S_n| = n!$ □

The group $\langle S_n, \circ \rangle$ is the **symmetric group** on n letters.

Example 5.3.2 Let $n = 3$, so that $\Sigma = \{0, 1, 2\}$. Then $\langle S_3, \circ \rangle$ is the symmetric group on 3 letters; S_3 contains $3! = 6$ permutations of the 3 letter set $\{0, 1, 2\}$. These 6 permutations can be listed as:

$$\sigma_1 = \begin{pmatrix} 0\ 1\ 2 \\ 0\ 1\ 2 \end{pmatrix}, \quad \sigma_2 = \begin{pmatrix} 0\ 1\ 2 \\ 2\ 0\ 1 \end{pmatrix}, \quad \sigma_3 = \begin{pmatrix} 0\ 1\ 2 \\ 1\ 2\ 0 \end{pmatrix},$$

$$\sigma_4 = \begin{pmatrix} 0 & 1 & 2 \\ 0 & 2 & 1 \end{pmatrix}, \quad \sigma_5 = \begin{pmatrix} 0 & 1 & 2 \\ 2 & 1 & 0 \end{pmatrix}, \quad \sigma_6 = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 2 \end{pmatrix}.$$

□

The permutation σ_1 is an identity element in S_3. In S_3,

$$\sigma_2 \circ \sigma_5 = \sigma_6$$

since

$$\begin{aligned}
(\sigma_2 \circ \sigma_5)(0) &= \sigma_2(\sigma_5(0)) = \sigma_2(2) = 1 = \sigma_6(0), \\
(\sigma_2 \circ \sigma_5)(1) &= \sigma_2(\sigma_5(1)) = \sigma_2(1) = 0 = \sigma_6(1), \\
(\sigma_2 \circ \sigma_5)(2) &= \sigma_2(\sigma_5(2)) = \sigma_2(0) = 2 = \sigma_6(2).
\end{aligned}$$

The group product $\sigma_2 \circ \sigma_5 = \sigma_6$ can also be computed using "right-to-left" permutation multiplication:

$$\xleftarrow{\hspace{3cm}}$$

$$\begin{pmatrix} 0 & 1 & 2 \\ 2 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 2 \\ 2 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 2 \end{pmatrix},$$

in which reading right-to-left, $2 \mapsto 0 \mapsto 2$, $1 \mapsto 1 \mapsto 0$, and $0 \mapsto 2 \mapsto 1$.

The complete table of binary operations in S_3 is:

$\circ$	σ_1	σ_2	σ_3	σ_4	σ_5	σ_6
σ_1	σ_1	σ_2	σ_3	σ_4	σ_5	σ_6
σ_2	σ_2	σ_3	σ_1	σ_5	σ_6	σ_4
σ_3	σ_3	σ_1	σ_2	σ_6	σ_4	σ_5
σ_4	σ_4	σ_6	σ_5	σ_1	σ_3	σ_2
σ_5	σ_5	σ_4	σ_6	σ_2	σ_1	σ_3
σ_6	σ_6	σ_5	σ_4	σ_3	σ_2	σ_1

From the table, one finds that $\sigma_3^{-1} = \sigma_2$ since $\sigma_2 \circ \sigma_3 = \sigma_1 = \sigma_3 \circ \sigma_2$. Moreover, S_3 is non-abelian since $\sigma_4 \circ \sigma_3 \neq \sigma_3 \circ \sigma_4$.

Here is another example of "right-to-left" permutation multiplication. Let S_5 be the symmetric group on 5 letters and let

$$\sigma = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 3 & 4 & 2 \end{pmatrix}, \quad \tau = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 0 & 4 \end{pmatrix}.$$

be elements of S_5. Then the right-to-left permutation multiplication yields

$$\sigma \circ \tau = \overleftarrow{\begin{pmatrix} 0\,1\,2\,3\,4 \\ 1\,0\,3\,4\,2 \end{pmatrix} \begin{pmatrix} 0\,1\,2\,3\,4 \\ 3\,2\,1\,0\,4 \end{pmatrix}} = \begin{pmatrix} 0\,1\,2\,3\,4 \\ 4\,3\,0\,1\,2 \end{pmatrix}$$

since (reading right-to-left) $4 \mapsto 4 \mapsto 2$, $3 \mapsto 0 \mapsto 1$, and so on.

Cycle Decomposition

Let $\Sigma = \{0, 1, 2, \ldots, n-1\}$ be the set of n letters, let $\{a_1, a_2, \ldots, a_l\} \subseteq \Sigma$ be a subset of letters. A **cycle** is a permutation of the form $(a_1, a_2, a_3, \ldots, a_{l-1}, a_l)$ that denotes the permutation

$$a_1 \mapsto a_2 \mapsto a_3 \mapsto \cdots \mapsto a_{l-1} \mapsto a_l \mapsto a_1,$$

which fixes all of the other letters in Σ. The **length** of the cycle $(a_1, a_2, \ldots, a_l)$ is l.

Every permutation in S_n can be written as a product of cycles yielding the **cycle decomposition** of the permutation. For example, let $\sigma \in S_5$ be the permutation given above. We have

$$0 \overset{\sigma}{\mapsto} 1 \overset{\sigma}{\mapsto} 0,$$

and

$$2 \overset{\sigma}{\mapsto} 3 \overset{\sigma}{\mapsto} 4 \overset{\sigma}{\mapsto} 2.$$

Thus σ factors as

$$\sigma = (0, 1)(2, 3, 4),$$

where the cycles $(0, 1)$ and $(2, 3, 4)$ can be written in standard notation as

$$(0, 1) = \begin{pmatrix} 0\,1\,2\,3\,4 \\ 1\,0\,2\,3\,4 \end{pmatrix}, \quad \text{and} \quad (2, 3, 4) = \begin{pmatrix} 0\,1\,2\,3\,4 \\ 0\,1\,3\,4\,2 \end{pmatrix}.$$

Moreover, for $\tau = \begin{pmatrix} 0\,1\,2\,3\,4 \\ 3\,2\,1\,0\,4 \end{pmatrix} \in S_5$, we obtain

$$0 \overset{\tau}{\mapsto} 3 \overset{\tau}{\mapsto} 0,$$

$$1 \overset{\tau}{\mapsto} 2 \overset{\tau}{\mapsto} 1,$$

and

$$4 \overset{\tau}{\mapsto} 4.$$

Thus

$$\tau = (0,3)(1,2)(4),$$

where

$$(0,3) = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 3 & 1 & 2 & 0 & 4 \end{pmatrix}, \quad (1,2) = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 0 & 2 & 1 & 3 & 4 \end{pmatrix},$$

and

$$(4) = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 3 & 4 \end{pmatrix}.$$

A cycle of length 2 is a **transposition**. Every permutation in S_n, $n \geq 2$, can be written as a product of transpositions by first obtaining its cycle decomposition and then using the cycle decomposition formula

$$(a_1, a_2, \ldots, a_l) = (a_1, a_l)(a_1, a_{l-1}) \cdots (a_1, a_2).$$

For instance, for σ, τ in S_5 as above,

$$\sigma = (0,1)(2,4)(2,3) \text{ and } \tau = (0,3)(1,2).$$

Note that cycles of length one may be omitted from the cycle decomposition.

If $\sigma \in S_n, n \geq 2$, can be written as a product of an even number of transpositions, then it is an **even permutation**, if σ factors into an odd number of transpositions, then σ is an **odd permutation**. In our example, $\sigma \in S_5$ is odd, while τ is even.

A permutation in S_n, $n \geq 2$, cannot be both even and odd. See Section 5.8, Exercise 7.

5.3.2 The Group of Residues Modulo n

Let n, a be integers with $n > 0$. A **residue of** a **modulo** n is any integer r for which $a = nq + r$ for some $q \in \mathbb{Z}$. For instance, if $n = 3$, $a = 8$, then 11 is a residue of 8 modulo 3 since $8 = 3(-1) + 11$, but so is 2 since $8 = 3(2) + 2$.

The **least non-negative residue** of a modulo n is the smallest non-negative number r for which $a = nq + r$. The possible least non-negative residues of a modulo n are $0, 1, 2, \ldots, n-1$. The least non-negative residue of a modulo n is denoted as **(a mod n)**. For example, $(8 \bmod 3) = 2$, moreover, $(-3 \bmod 4) = 1$ and $(11 \bmod 4) = (3 \bmod 4) = 3$.

For $n, a \in \mathbb{Z}$, $n > 0$, $a \geq 0$, the value of $(a \bmod n)$ coincides with the value of r obtained when we divide a by n using the long division algorithm, yielding the division statement

$$a = nq + r, \quad r = (a \bmod n).$$

In fact, as we saw in Algorithm 4.2.8, $q = \lfloor a/n \rfloor$, thus

$$a = n\lfloor a/n \rfloor + (a \bmod n),$$

$0 \leq (a \bmod n) < n$.

We say that two integers a, b are **congruent modulo** n if $(a \bmod n) = (b \bmod n)$, and we write $a \equiv b \pmod{n}$. Let a, n be integers with $n > 0$. Then n **divides** a, denoted by $n \mid a$, if there exists an integer k for which $a = nk$.

Proposition 5.3.3 *Let $a, b, n \in \mathbb{Z}$, $n > 0$. Then $a \equiv b \pmod{n}$ if and only if $n \mid (a - b)$.*

Proof To prove the "only if" part, assume that $a \equiv b \pmod{n}$. Then $(a \bmod n) = (b \bmod n)$, so there exist integers l, m for which $a = nm + r$ and $b = nl + r$ with $r = (a \bmod n) = (b \bmod n)$. Thus $a - b = n(m - l)$. For the "if" part, assume that $a - b = nk$ for some k. Then $(nm + (a \bmod n)) - (nl + (b \bmod n)) = nk$ for some $m, l \in \mathbb{Z}$, so that n divides $(a \bmod n) - (b \bmod n)$. Consequently, $(a \bmod n) - (b \bmod n) = 0$, hence $a \equiv b \pmod{n}$. □

Proposition 5.3.3 can help us compute $(a \bmod n)$. For instance,

$$(-14 \bmod 17) = (3 \bmod 17) = 3$$

since $17 \mid (-14 - 3)$. Likewise $(-226 \bmod 17) = (12 \bmod 17) = 12$ since $17 \mid (-226 - 12)$.

The standard way to compute $(a \bmod n)$ for $a \geq 0$ is by using the formula

$$(a \bmod n) = a - \lfloor a/n \rfloor n.$$

For example, $\lfloor 226/17 \rfloor = 13$, thus $(226 \bmod 17) = 226 - 13 \cdot 17 = 5$. The following algorithm computes $(a \bmod n)$ using Algorithm 4.2.8.

Algorithm 5.3.4 (*a*_**MOD**_*n*)

```
Input:      integers $a \geq 0$, $n > 0$, with $(a)_2 = a_m \dots a_2 a_1$,
            i.e., $a_m \dots a_2 a_1$ is the binary representation of $a$
Output:     $(a \bmod n)$
Algorithm:
            $d \leftarrow$ a_DIV_n
```

```
        $r \leftarrow a - dn$
        output $r$
```

□

Proposition 5.3.5 *a_MOD_n runs in time $O(m^2)$.*

Proof Each step of the algorithm runs in time $O(m^2)$. □

For $n > 0$ consider the set $J = \{0, 1, 2, 3, \dots, n-1\}$ of least non-negative residues modulo n. Note that $a = (a \bmod n), \forall a \in J$. On J we define a binary operation $+_n$ as follows: for $a, b \in J$,

$$(a \bmod n) +_n (b \bmod n) = ((a + b) \bmod n).$$

Proposition 5.3.6 *$\langle J, +_n \rangle$ is a finite abelian group of order n.*

Proof Exercise. □

The group $\langle J, +_n \rangle$ is the **group of residues modulo** n. We denote this group by $\mathbb{Z}_n$; $|\mathbb{Z}_n| = n$. In the case $n = 5$, $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$ is the group of residues modulo 5. In $\mathbb{Z}_5$, for instance, $2 +_5 3 = ((2+3) \bmod 5) = (5 \bmod 5) = 0$. The complete group table (binary operation table) for $\mathbb{Z}_5$ is:

$+_5$	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

For another example, consider the group $\mathbb{Z}_8 = \{0, 1, 2, 3, 4, 5, 6, 7\}$. In this group, $4 +_8 5 = ((4 + 5) \bmod 8) = (9 \bmod 8) = 1$. The group table appears as

$+_8$	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

5.4 Direct Product of Groups

We can construct a new group from a finite set of groups.

Let $S_1, S_2, \ldots, S_k$ be a finite collection of sets. The **cartesian product** $\prod_{i=1}^{k} S_i$ is the collection of all k-tuples $\{(a_1, a_2, \ldots, a_k) : a_i \in S_i\}$. For example, if $S_1 = S_2 = \mathbb{R}$, then the cartesian product $\prod_{i=1}^{2} \mathbb{R} = \mathbb{R} \times \mathbb{R}$ is the familiar xy-coordinate system that you have seen in calculus class.

Proposition 5.4.1 *Let G_i, $i = 1, \ldots, k$, be a finite collection of groups. Then the cartesian product $\prod_{i=1}^{k} G_i$ is a group under the binary operation defined as*

$$(a_1, a_2, \ldots, a_k) \cdot (b_1, b_2, \ldots, b_k) = (a_1b_1, a_2b_2, \ldots, a_kb_k),$$

where a_ib_i is the group product in G_i for $1 \le i \le k$.

Proof We show that the conditions of Definition 5.1.1 hold. For associativity, let $(a_1, a_2, \ldots, a_k)$, $(b_1, b_2, \ldots, b_k)$, $(c_1, c_2, \ldots, c_k)$ be elements of $\prod_{i=1}^{k} G_i$. Then

$$\begin{aligned}
&(a_1, a_2, \ldots, a_k) \cdot ((b_1, b_2, \ldots, b_k) \cdot (c_1, c_2, \ldots, c_k)) \\
&= (a_1, a_2, \ldots, a_k) \cdot (b_1c_1, b_2c_2, \ldots, b_kc_k) \\
&= (a_1(b_1c_1), a_2(b_2c_2), \ldots, a_k(b_kc_k)) \\
&= ((a_1b_1)c_1), (a_2b_2)c_2), \ldots, (a_kb_k)c_k)) \\
&= (a_1b_1, a_2b_2, \ldots, a_kb_k) \cdot (c_1, c_2, \ldots, c_k) \\
&= ((a_1, a_2, \ldots, a_k) \cdot (b_1, b_2, \ldots, b_k)) \cdot (c_1, c_2, \ldots, c_k)
\end{aligned}$$

and so $\cdot$ is associative. For an identity element we take $e = (e_1, e_2, \ldots, e_k)$ where e_i is an identity in G_i. Then

$$\begin{aligned}
(e_1, e_2, \ldots, e_k) \cdot (a_1, a_2, \ldots, a_k) &= (e_1a_1, e_2a_2, \ldots, e_ka_k) \\
&= (a_1, a_2, \ldots, a_k) \\
&= (a_1e_1, a_2e_2, \ldots, a_ke_k) \\
&= (a_1, a_2, \ldots, a_k) \cdot (e_1, e_2, \ldots, e_k)
\end{aligned}$$

as required.

Lastly, for each k-tuple $(a_1, a_2, \ldots, a_k)$ one has

$$(a_1, a_2, \ldots, a_k)^{-1} = (a_1^{-1}, a_2^{-1}, \ldots, a_k^{-1}).$$

We leave it to the reader to verify that $(a_1^{-1}, a_2^{-1}, \ldots, a_k^{-1})$ is a left and right inverse for the element $(a_1, a_2, \ldots, a_k)$. □

The group $\prod_{i=1}^{k} G_i$ of Proposition 5.4.1 is the **direct product group**. As an illustration, we take $G_1 = \mathbb{Z}_4$ and $G_2 = \mathbb{Z}_5$ and form the direct product group $\mathbb{Z}_4 \times \mathbb{Z}_5$. In $\mathbb{Z}_4 \times \mathbb{Z}_5$, for instance,

$$(3, 2) \cdot (1, 4) = (3 +_4 1, 2 +_5 4) = (0, 1).$$

Note that $|\mathbb{Z}_4 \times \mathbb{Z}_5| = 20$.

As another example, we take $G_1 = G_2 = G_3 = \mathbb{R}$, where $\mathbb{R}$ is the additive group of real numbers. Then $\prod_1^3 \mathbb{R} = \mathbb{R}^3$ is the underlying group of Euclidean 3-space consisting of 3-dimensional vectors of the form $v = (a_1, a_2, a_3)$ where $a_1, a_2, a_3 \in \mathbb{R}$. The group operation is the familiar vector addition: for $v = (a_1, a_2, a_3)$, $w = (b_1, b_2, b_3) \in \mathbb{R}^3$,

$$v + w = (a_1, a_2, a_3) + (b_1, b_2, b_3) = (a_1 + b_1, a_2 + b_2, a_3 + b_3).$$

5.5 Subgroups

In this section we consider subgroups, which are the analogs for groups of subsets of a set. We determine the collection of left and right cosets of a subgroup in a group, give the partition theorem and Lagrange's Theorem.

Let H be a subset of a group G. Then the binary operation B on G restricts to a function $B|_H : H \times H \to G$. If $B|_H(H \times H) \subseteq H$, then H is **closed** under the binary operation B. In other words, H is closed under B if $B(a, b) = ab \in H$ for all $a, b \in H$. If H is closed under B, then $B|_H$ is a binary operation on H. Closure is fundamental to the next definition.

Definition 5.5.1 Let H be a subset of a group G that satisfies the following conditions.

(i) H is closed under the binary operation of G,
(ii) $e \in H$,
(iii) for all $a \in H$, $a^{-1} \in H$.

Then H is a **subgroup** of G, which we denote by $H \leq G$.

For example, $2\mathbb{Z} = \{2n \mid n \in \mathbb{Z}\} \leq \mathbb{Z}$. The subset $\{0, 3, 6, 9\}$ is a subgroup of $\mathbb{Z}_{12}$. The subset $\{\sigma_1, \sigma_4\}$ is a subgroup of S_3. The set of integers $\mathbb{Z}$ is a subgroup of the additive group $\mathbb{R}$.

Every group G admits at least two subgroups: the **trivial subgroup** $\{e\} \leq G$, and the group G which is a subgroup of itself. If $H \leq G$ and H is a proper subset of G, then H is a **proper subgroup** of G and we write $H < G$. If H is a subgroup of G, then H is a group under the restricted binary operation of G.

Definition 5.5.2 Let H be a subgroup of G with $a \in G$. The set of group products $aH = \{ah \mid h \in H\}$ is the **left coset of** H **in** G **represented by** a. The collection $Ha = \{ha \mid h \in H\}$ is the **right coset of** H **in** G **represented by** a.

Let aH be a left coset. The element $x \in G$ is a **representative of** aH if $xH = aH$.

Since $eH = He = H$, the subgroup H is always a left and right coset of itself in G represented by e. Observe that $1+\{0,3,6,9\} = \{1,4,7,10\}$ and $2+\{0,3,6,9\} = \{2,5,7,11\}$ are left cosets of $\{0,3,6,9\}$ in $\mathbb{Z}_{12}$ represented by 1 and 2, respectively. Also, $\{\sigma_1,\sigma_4\}\sigma_1 = \{\sigma_1,\sigma_4\}$ and $\{\sigma_1,\sigma_4\}\sigma_6 = \{\sigma_1\sigma_6,\sigma_4\sigma_6\} = \{\sigma_6,\sigma_2\}$ are right cosets of $\{\sigma_1,\sigma_4\}$ in S_3 represented by σ_1 and σ_6, respectively.

Proposition 5.5.3 *Let $H \le G$, and let aH, bH be left cosets. Then there exists a bijection $\phi : aH \to bH$ defined as $\phi(ah) = bh$ for $h \in H$.*

Proof To show that ϕ is 1-1 (one-to-one), suppose that $\phi(ah_1) = \phi(ah_2)$, for $h_1, h_2 \in H$. Then $bh_1 = bh_2$, so that $h_1 = h_2$, and consequently, $ah_1 = ah_2$. Next let $bh \in bH$. Then clearly, $\phi(ah) = bh$, so that ϕ is onto. □

The following corollary is immediate.

Corollary 5.5.4 *Suppose G is a finite group. Then $|aH| = |bH| = |H|$ for all $a, b \in G$.*

Let G be a group, let H be a subgroup of G. A subset $S = \{a_\eta\}_{\eta\in I}$ of G is a **left transversal** of H if the family $\{a_\eta H\}_{\eta\in I}$ constitutes the collection of all distinct left cosets of H in G.

Proposition 5.5.5 *Let G be a group, let H be a subgroup of G, and let $S = \{a_\eta\}_{\eta\in I}$ be a left transversal of H. Then the collection of distinct left cosets $\{a_\eta H\}_{\eta\in I}$ of H in G forms a partition of the set G, i.e.,*

$$G = \bigcup_{\eta\in I} a_\eta H,$$

with $a_\eta H \cap a_\gamma H = \emptyset$ whenever $a_\eta H \neq a_\gamma H$.

Proof Let $g \in G$. Then $gH = a_\eta H$ for some $\eta \in I$, thus $G \subseteq \bigcup_{\eta\in I} a_\eta H$. Clearly, $\bigcup_{\eta\in I} a_\eta H \subseteq G$, and so, $G = \bigcup_{\eta\in I} a_\eta H$. Suppose there exists an element $x \in a_\eta H \cap a_\gamma H$ for $\eta, \gamma \in I$. Then $x = a_\eta h_1 = a_\gamma h_2$ for some $h_1, h_2 \in H$. Consequently, $a_\eta = a_\gamma h_2 h_1^{-1} \in a_\gamma H$. Now, for any $h \in H$, $a_\eta h = a_\gamma h_1 h_2^{-1} h \in a_\gamma H$, and so, $a_\eta H \subseteq a_\gamma H$. By a similar argument $a_\gamma H \subseteq a_\eta H$, and so, $a_\eta H = a_\gamma H$. Thus the collection $\{a_\eta H\}_{\eta\in I}$ is a partition of G. □

To illustrate Proposition 5.5.5, let $G = S_3$, $H = \{\sigma_1,\sigma_4\}$. Then $H = \sigma_1\{\sigma_1,\sigma_4\}$, $\sigma_2 H = \{\sigma_2,\sigma_5\}$ and $\sigma_3 H = \{\sigma_3,\sigma_6\}$ are the distinct left cosets of H which form the partition of S_3,

$$\{H, \sigma_2 H, \sigma_3 H\}.$$

For another example, let $G = \mathbb{Z}$, $H = 3\mathbb{Z}$. Then the collection of distinct left cosets is $\{3\mathbb{Z}, 1 + 3\mathbb{Z}, 2 + 3\mathbb{Z}\}$ which forms a partition of $\mathbb{Z}$.

In many cases, even if the group G is infinite, there may be only a finite number of left cosets. When this occurs we define the number of left cosets of H in G to be the **index** $[G : H]$ **of** H **in** G. For instance, $[Z : 3\mathbb{Z}] = 3$.

If the group G is finite, we have the following classical result attributed to Lagrange.

Proposition 5.5.6 (Lagrange's Theorem) *Suppose* $H \leq G$ *with* $|G| < \infty$. *Then* $|H|$ *divides* $|G|$.

Proof By Corollary 5.5.4 any two left cosets have the same number of elements, and this number is $|H|$. Moreover, the number of left cosets of H in G is $[G : H]$. Since the left cosets partition G, we have

$$|H|[G : H] = |G|.$$

□

5.6 Homomorphisms of Groups

If A and B are sets, then the functions $f : A \to B$ are the basic maps between A and B. In this section we introduce group homomorphisms: functions preserving group structure which are the basic maps between groups.

Definition 5.6.1 Let G, G' be groups. A map $\psi : G \to G'$ is a **homomorphism of groups** if

$$\psi(ab) = \psi(a)\psi(b)$$

for all $a, b \in G$.

In additive notation, the homomorphism condition is given as

$$\psi(a + b) = \psi(a) + \psi(b).$$

For example, the map $\psi : \mathbb{Z} \to \mathbb{Z}_n$ given by $\psi(a) = (a \bmod n)$ is a homomorphism of groups since

$$\psi(a + b) = ((a + b) \bmod n) = (a \bmod n) + (b \bmod n) = \psi(a) + \psi(b)$$

for all $a, b \in \mathbb{Z}$. The map $\psi : \mathrm{GL}_n(\mathbb{R}) \to \mathbb{R}^\times$ defined as $\psi(A) = \det(A)$ is a homomorphism of groups since by a familiar property of determinants,

$$\psi(AB) = \det(AB) = \det(A)\det(B) = \psi(A)\psi(B).$$

Another example comes from elementary calculus: Let $\mathbb{R}_+ = \{x \in \mathbb{R} \mid x > 0\}$ denote the multiplicative group of positive real numbers. The map $\psi : \mathbb{R}_+ \to \mathbb{R}$, defined by $\psi(x) = \ln(x)$ is a homomorphism of groups; it is an example of a homomorphism of a multiplicative group into an additive group.

Definition 5.6.2 A homomorphism of groups $\psi : G \to G'$ which is both injective and surjective is an **isomorphism of groups.** Two groups G, G' are **isomorphic** if there exists an isomorphism $\psi : G \to G'$. We then write $G \cong G'$.

Proposition 5.6.3 *Let $n > 0$. Then $\psi : \mathbb{Z} \to n\mathbb{Z}$ with $\psi(a) = na$ is an isomorphism of additive groups.*

Proof Note that

$$\psi(a + b) = n(a + b) = na + nb = \psi(a) + \psi(b),$$

$\forall a, b \in \mathbb{Z}$, thus ψ is a homomorphism. Now suppose $na = nb$. Then $na + (-nb) = n(a + (-b)) = 0$, hence $a = b$. Thus ψ is an injection. Clearly ψ is surjective. □

A map can of course be a bijection without being a group isomorphism. For example, there are exactly $6! = 120$ functions from S_3 to $\mathbb{Z}_6$ that are bijections, yet none is group isomorphisms. Likewise, the groups $\mathbb{Z}_4$ and $\mathbb{Z}_2 \times \mathbb{Z}_2$ have the same number of elements, but are not isomorphic.

Let G be any group. The collection of all groups G' for which $G' \cong G$ is the **isomorphism class of groups represented by** G. Essentially, two groups are contained in the same isomorphism class if and only if they are isomorphic. Our observation above shows that as groups S_3 and $\mathbb{Z}_6$ are in different isomorphism classes.

5.7 Group Structure

Let G be a group with binary operation $G \times G \to G$, $(a, b) \mapsto ab$. Let $a \in G$, and let $n > 0$ be a positive integer. Then by the notation a^n we mean

$$a^n = \underbrace{aaa \cdots a}_{n \text{ times}}.$$

For $n < 0$, we write

$$a^n = \underbrace{a^{-1}a^{-1}a^{-1} \cdots a^{-1}}_{|n| \text{ times}}.$$

If $n = 0$, we set

$$a^0 = e, \text{ the identity element of the group.}$$

Now assume that G is an "additive" group, i.e., a group in which the binary operation is written additively as $+$. Let $a \in G$ and let $n > 0$ be a positive integer. Then by the notation na we mean

$$na = \underbrace{a + a + a + \cdots + a}_{n \text{ times}}.$$

For $n < 0$, we write

$$na = \underbrace{(-a) + (-a) + (-a) + \cdots (-a)}_{|n| \text{ times}},$$

where $-a$ is the inverse of a. If $n = 0$, we set

$$0a = 0, \text{ the identity element of the group.}$$

Let $S \subseteq G$ be a subset of G. Then G is **generated** by S if every element of G can be written as a finite group product over the (possibly infinite) alphabet $\Sigma = \{a^n : a \in S, n \in \mathbb{Z}\}$. If G is generated by S we write $G = \langle S \rangle$. For example, if $G = S_3$, $S = \{\sigma_2, \sigma_4\}$, then S_3 is generated by S since

$$\sigma_2^0 = \sigma_1, \ \sigma_2^1 = \sigma_2, \ \sigma_2^2 = \sigma_3, \ \sigma_4^1 = \sigma_4, \ \sigma_2\sigma_4 = \sigma_5, \ \sigma_2^2\sigma_4 = \sigma_6.$$

Every group is generated by some subset of the group. If necessary one could choose the set G as a generating set for itself, $G = \langle G \rangle$.

A group G is **finitely generated** if $G = \langle S \rangle$ where S is a finite subset of G. Any finite group is finitely generated. A group G is **cyclic** if it is generated by a singleton subset $\{a\}$, $a \in G$. If G is cyclic, then there exists an element $a \in G$ for which

$$G = \{a^n : n \in \mathbb{Z}\}.$$

The element a is a **generator** for the cyclic group G and we write $G = \langle a \rangle$.

In additive notation, a group G is cyclic if there exists $a \in G$ for which

$$G = \{na : n \in \mathbb{Z}\}.$$

For example, the additive group $\mathbb{Z}_5$ is cyclic, generated by 1. To see this note that

$$\begin{aligned} 1 &= 1, \\ 1 +_5 1 &= 2, \\ 1 +_5 1 +_5 1 &= 3, \\ 1 +_5 1 +_5 1 +_5 1 &= 4 \\ 1 +_5 1 +_5 1 +_5 1 +_5 1 &= 0. \end{aligned}$$

Thus $\{n1 \,:\, n \in \mathbb{Z}\} = \mathbb{Z}_5$. Are there any other generators for $\mathbb{Z}_5$?

The additive group $\mathbb{Z}$ is cyclic on the generator 1, $\langle 1\rangle = \{n1 \,:\, n \in \mathbb{Z}\} = \mathbb{Z}$. Note that -1 is also a generator for $\mathbb{Z}$.

On the other hand, S_3 is not cyclic: there is no permutation $\sigma \in S_3$ for which

$$S_3 = \{\sigma^n \,:\, n \in \mathbb{Z}\}.$$

For instance, $\sigma_2 = \begin{pmatrix} 0 & 1 & 2 \\ 2 & 0 & 1 \end{pmatrix}$ is not a generator of S_3 since

$$\begin{aligned}
\sigma_2^1 &= \sigma_2, \\
\sigma_2^2 &= \sigma_2 \circ \sigma_2 = \sigma_3, \\
\sigma_2^3 &= \sigma_2 \circ \sigma_2 \circ \sigma_2 = \sigma_1, \\
\sigma_2^4 &= \sigma_2 \circ \sigma_2 \circ \sigma_2 \circ \sigma_2 = \sigma_2, \\
\sigma_2^5 &= \sigma_2 \circ \sigma_2 \circ \sigma_2 \circ \sigma_2 \circ \sigma_2 = \sigma_3, \\
\sigma_2^6 &= \sigma_2 \circ \sigma_2 \circ \sigma_2 \circ \sigma_2 \circ \sigma_2 \circ \sigma_2 = \sigma_1,
\end{aligned}$$

and so on. In fact, $\{\sigma_2^n \,:\, n \in \mathbb{Z}\} = \{\sigma_1, \sigma_2, \sigma_3\} \neq S_3$.

Proposition 5.7.1 *Every subgroup of a cyclic group is cyclic.*

Proof Let $G = \langle a\rangle$ be cyclic, and let $H \leq G$. If $H = \{e\}$, then H is cyclic and the proposition is proved. So we assume that H has at least two elements e and $b \neq e$. Since b is non-trivial and $H \leq \langle a\rangle$, there exists a positive integer k so that $a^k \in H$. By the well-ordering principle for natural numbers, there exists a smallest positive integer m for which $a^m \in H$.

Let $h \in H$. Then $h = a^n$ for some integer n. Now by the division algorithm for integers, $n = mq + r$ for integers q, r with $0 \leq r < m$. Thus $h = (a^m)^q a^r$, and so, $a^r = h(a^m)^{-q} \in H$. But this says that $r = 0$ since m was chosen to be minimal. Consequently, $h = (a^m)^q \in \langle a^m\rangle$, and so $H = \langle a^m\rangle$. □

Let G be any group and let $g \in G$. Then the set

$$\{g^n \,:\, n \in \mathbb{Z}\}$$

is a subgroup of G, which we call the **cyclic subgroup of** G generated by g. We denote the cyclic subgroup generated by g as $\langle g\rangle$. For instance, in S_3, $\langle\sigma_2\rangle = \{\sigma_1, \sigma_2, \sigma_3\}$. In $\mathbb{Q}_+$, under ordinary multiplication,

$$\langle 2\rangle = \{2^n \,:\, n \in \mathbb{Z}\} = \left\{\ldots, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2, 4, 8, \ldots\right\} \leq \mathbb{Q}_+.$$

For an additive example, take $G = \mathbb{Z}$. Then

$$\langle -3 \rangle = \{n(-3) : n \in \mathbb{Z}\} = \{\dots, 9, 6, 3, 0, -3, -6, -9 \dots\} \leq \mathbb{Z}.$$

Let G be a group and let $g \in G$. The **order** of g is the order of the cyclic subgroup $\langle g \rangle \leq G$.

Proposition 5.7.2 *Let G be a finite group and let $g \in G$. Then the order of the group $\langle g \rangle$ is the smallest positive integer m for which $g^m = e$.*

Proof Let $m = |\langle g \rangle|$ and let

$$e = g^0, g^1, g^2, \dots, g^{m-1} \tag{5.1}$$

be the list of powers of g. We claim that (5.1) contains m distinct powers of g. If not, let s be the largest integer $1 \leq s \leq m - 1$ for which the list

$$e = g^0, g^1, g^2, \dots, g^{s-1} \tag{5.2}$$

contains distinct powers of g. Now, $g^s = g^i$ for some i, $0 \leq i \leq s - 1$. Thus $g^{s-i} = e$. If $i \geq 1$, then $1 \leq s - i \leq s - 1$ with $g^{s-i} = g^0 = e$, which contradicts our assumption that list (5.2) contains distinct powers of g. Thus $i = 0$ and so, $g^s = e$. Let $g^t \in \langle g \rangle$, $t \in \mathbb{Z}$. By the division algorithm, $t = sq + r$ for some $q, r \in \mathbb{Z}$ with $0 \leq r < s$. Thus $g^t = g^{sq+r} = (g^s)^q g^r = g^r$, and so, g^t is one of the powers in (5.2). Hence the list (5.2) constitutes the cyclic subgroup $\langle g \rangle$, but this says that $|\langle g \rangle| = s < m$, a contradiction. We conclude that the list (5.1) contains distinct powers of g.

Now $g^m = g^i$ for some $0 \leq i \leq m - 1$ and arguing as above, we conclude that $i = 0$. Thus $g^m = e$. Now suppose there exists $0 < m' < m$ with $g^{m'} = e$. Then (5.1) does not contain distinct powers of g, a contradiction. □

Thus, if G is finite, then the order of $g \in G$ is the smallest positive integer m for which $g^m = e$.

Proposition 5.7.3 *Let G be a group and let $g \in G$ be an element of finite order m. Then $g^s = e$ if and only if s is a multiple of m.*

Proof Suppose $g^s = e$. Write $s = mq + r$, $0 \leq r < m$. Then $e = g^s = (g^m)^q g^r = g^r$. We cannot have $r > 0$ since m is the smallest positive integer with $g^m = e$. Thus $mq = s$. The converse is immediate. □

Proposition 5.7.4 *Suppose G is finite with $n = |G|$. Let $g \in G$. Then $g^n = e$.*

Proof Let $m = |\langle g \rangle|$. By Proposition 5.7.2, $g^m = e$. By Lagrange's Theorem $m \mid n$, that is, $ml = n$ for some integer l. Hence $g^n = g^{ml} = (g^m)^l = e$. □

Let G be a finite abelian group and let $\mathbb{Z}_+ = \{n \in \mathbb{Z} : n > 0\}$. Let T be the set of positive integers defined as

$$T = \{t \in \mathbb{Z}_+ : g^t = e, \forall g \in G\}.$$

Then by Proposition 5.7.4, T is non-empty (since $|G| \in T$). Hence by the well-ordering principle, T contains a smallest element f, which is the **exponent** of G. The exponent of G is the smallest positive integer f for which $g^f = e$ for all $g \in G$; one has $f \leq |G|$.

For example, the exponent of the group product $\mathbb{Z}_4 \times \mathbb{Z}_6$ (a finite abelian group) is 12. Note that $12 \leq |\mathbb{Z}_4 \times \mathbb{Z}_6| = 24$. The exponent of a finite abelian group can equal its order. For instance, let G be a finite group and let $\langle g \rangle$ be the cyclic subgroup of G generated by g. Then by Proposition 5.7.2, the exponent of $\langle g \rangle$ is $|\langle g \rangle|$.

Ultimately, we will show that if G is a finite abelian group of exponent f, then G contains an element of order f (Proposition 5.7.13).

To this end, we introduce some number theory.

5.7.1 Some Number Theory

Definition 5.7.5 Let $a, b \in \mathbb{Z}$ and assume that a, b are not both 0. The **greatest common divisor** of a, b is the unique positive integer d that satisfies

(i) d divides a and d divides b,
(ii) c divides d whenever c is a common divisor of a and b.

We denote the greatest common divisor as $d = \gcd(a, b)$. The famous Euclidean algorithm computes $\gcd(a, b)$ using Algorithm 5.3.4.

Algorithm 5.7.6 (EUCLID)

```
Input:     integers $a \geq b \geq 1$
Output:    $\gcd(a, b)$
Algorithm:
           $r_0 \leftarrow a$
           $r_1 \leftarrow b$
           $i \leftarrow 1$
           while $r_i \neq 0$ do
              $i \leftarrow i + 1$
              $r_i \leftarrow (r_{i-2} \bmod r_{i-1})$
           end-while
           output $r_{i-1}$
```

□

Proposition 5.7.7 *EUCLID is a polynomial time algorithm.*

Proof Since Algorithm 5.3.4 runs in time $O(m^2)$, EUCLID runs in time $O(m^3)$. □

Let us illustrate how EUCLID works to compute gcd(63, 36). On the inputs $a = 63$, $b = 36$, the algorithm computes gcd(63, 36) by executing three iterations of the while loop:

$$\begin{aligned} r_2 &= (63 \bmod 36) = 27 \qquad \text{(iteration 1)} \\ r_3 &= (36 \bmod 27) = 9 \qquad \text{(iteration 2)} \\ r_4 &= (27 \bmod 9) = 0. \end{aligned}$$

The algorithm stops with $i = 4$ and outputs $r_3 = \gcd(63, 36) = 9$.

The greatest common divisor has the following property.

Proposition 5.7.8 (Bezout's Lemma) *Let a, b be integers, not both zero, and suppose that $d = \gcd(a, b)$. Then there exist integers $x, y \in \mathbb{Z}$ so that*

$$ax + by = d.$$

Proof The set of integers

$$a\mathbb{Z} + b\mathbb{Z} = \{am + bn : \ m, n \in \mathbb{Z}\}$$

is a subgroup of the cyclic group $\mathbb{Z}$. Thus by Proposition 5.7.1, $a\mathbb{Z} + b\mathbb{Z} = d'\mathbb{Z}$ for some integer $d' > 0$. Thus there exist $x, y \in \mathbb{Z}$ with $ax + by = d'$. We claim that $d' = d$. Since $a \in d'\mathbb{Z}$ and $b \in d'\mathbb{Z}$, d' divides both a and b. Now suppose that $cm = a$ and $cn = b$ for c, m, n. Then $cmx + cny = c(mx + ny) = d'$, so $c \mid d'$. It follows that $d = d'$. □

For example, we know that $9 = \gcd(63, 36)$. Let us find $x, y \in \mathbb{Z}$ so that

$$63x + 36y = 9.$$

From iteration 1 and iteration 2, we compute $63 = 36 \cdot 1 + 27$ and $36 = 27 \cdot 1 + 9$. Thus,

$$\begin{aligned} 9 &= 36 - 27 \cdot 1 \\ &= 36 - (63 - 36 \cdot 1) \cdot 1 \\ &= 36 - 63 + 36 \\ &= 63 \cdot (-1) + 36 \cdot 2. \end{aligned}$$

And so $x = -1$, $y = 2$.

Integers $a \geq b \geq 1$ are **relatively prime** if $\gcd(a, b) = 1$. For instance, $a = 20$, $b = 11$ are relatively prime since

$$\begin{aligned} r_2 &= (20 \bmod 11) = 9, \\ r_3 &= (11 \bmod 9) = 2, \\ r_4 &= (9 \bmod 2) = 1, \\ r_5 &= (2 \bmod 1) = 0, \end{aligned}$$

so that $\gcd(20, 11) = 1$.

A common multiple of the non-zero integers a, b is an integer $m > 0$ for which $a \mid m$ and $b \mid m$. Let S be the set of all common multiples of a, b. Then by the well-ordering principle S has the smallest element which is the **least common multiple** of a, b, denoted as $\mathrm{lcm}(a, b)$.

Proposition 5.7.9 *Let a, b be non-zero integers. Then*

$$lcm(a, b) = \frac{|ab|}{\gcd(a, b)}.$$

Proof Let $d = \gcd(a, b)$. Then $\gcd(a/d, b/d) = 1$. Note that $|ab|/d^2$ is a common multiple of a/d and b/d, and so $\mathrm{lcm}(a/d, b/d) \leq |ab|/d^2$. By the division algorithm

$$|ab|/d^2 = \mathrm{lcm}(a/d, b/d) \cdot q + r, \quad 0 \leq r < \mathrm{lcm}(a/d, b/d),$$

for unique q, r. If $r > 0$, then r is a common multiple of a/d and b/d, which is impossible. Thus

$$|ab|/d^2 = \mathrm{lcm}(a/d, b/d) \cdot q,$$

and so there exist integers m, l with

$$|ab|/d^2 = mqa/d = lqb/d.$$

Thus q divides both a/d and b/d, hence $q = 1$, which gives

$$\mathrm{lcm}(a/d, b/d) = |ab|/d^2,$$

thus

$$d \cdot \mathrm{lcm}(a/d, b/d) = mqa = lqb.$$

Consequently, $d \cdot \mathrm{lcm}(a/d, b/d)$ is a common multiple of a, b and so

$$\mathrm{lcm}(a, b) \leq d \cdot \mathrm{lcm}(a/d, b/d).$$

There exist integers s, t so that $sa = tb = \text{lcm}(a, b)$, so $sa/d = tb/d = \text{lcm}(a, b)/d$. Thus $\text{lcm}(a, b)/d$ is a common multiple of $a/d, b/d$. Hence

$$\text{lcm}(a/d, b/d) \leq \text{lcm}(a, b)/d,$$

which yields

$$d \cdot \text{lcm}(a/d, b/d) = \text{lcm}(a, b),$$

and so $\text{lcm}(a, b) = |ab|/d$. □

Proposition 5.7.10 *Let $G = \langle a \rangle$ be cyclic of order n and let s be an integer. Then the order of the cyclic subgroup of G generated by a^s is n/d where $d = \text{gcd}(n, s)$.*

Proof Note that n is the smallest positive integer for which $a^n = e$. The order of $\langle a^s \rangle$ is the smallest positive integer m for which $(a^s)^m = a^{sm} = e$. Note that $sm \geq n$, but even more: n must divide sm. For if not, there exist positive integers v, w with $sm = nv + w$ with $0 < w < n$. Thus $e = a^{sm} = a^{nv}a^w = a^w$, which contradicts the minimality of n.

Thus the order of $\langle a^s \rangle$ is the smallest positive integer m for which n divides sm. We compute the value of m as follows. Since n divides sm, sm is a common multiple of s and n, hence $sm \geq \text{lcm}(s, n)$. By Proposition 5.7.9, $m \geq \text{lcm}(s, n)/s = n/d$, where $d = \text{gcd}(n, s)$. Now, $(n/d)s = ns/d = \text{lcm}(n, s)$, which is a multiple of n, and so $(a^s)^{(n/d)} = e$. It follows that $m = n/d$. □

Corollary 5.7.11 *Let $G = \langle a \rangle$ be cyclic of order n, and let s be an integer. Then $G = \langle a^s \rangle$ if and only if $gcd(n, s) = 1$.*

Proof Exercise. □

We can extend the notion of lcm. Let $a_1, a_2, \dots, a_k$, $k \geq 2$, be non-zero integers. Then the **least common multiple** of $a_1, a_2, \dots, a_k$, denoted as $\text{lcm}(a_1, a_2, \dots, a_k)$, is the smallest positive integer $m > 0$ which is a multiple of a_i for all i, $1 \leq i \leq k$. For $k \geq 3$, we can define $\text{lcm}(a_1, a_2, \dots, a_k)$ inductively as follows:

$$\text{lcm}(a_1, a_2, \dots, a_k) = \text{lcm}(\text{lcm}(a_1, a_2, \dots, a_{k-1}), a_k).$$

Proposition 5.7.12 *Let G be a finite abelian group. Then the exponent of G is the lcm of the orders of the elements of G.*

Proof Let $g \in G$, let m_g denote the order of g and let f be the exponent of G. By Proposition 5.7.3, f is a multiple of m_g, thus f is a common multiple of the set $\{m_h\}_{h \in G}$. Thus $\text{lcm}(\{m_h\}_{h \in G}) \leq f$. Note that $g^{\text{lcm}(\{m_h\}_{h \in G})} = e$ for each $g \in G$, thus $\text{lcm}(\{m_h\}_{h \in G}) = f$ by definition of the exponent. □

Proposition 5.7.13 *Let G be a finite abelian group with exponent f. Then there exists an element of G whose order is f.*

Proof Let $n = |G|$ and list the elements of G as $e = g_0, g_1, \dots, g_{n-1}$. Let m_{g_i} be the order of g_i for $0 \le i \le n-1$. Clearly, $m_{g_0} = 1$. Consider the elements g_1, g_2 and let $d = \gcd(m_{g_1}, m_{g_2})$. Then m_{g_1} and m_{g_2}/d are coprime. Hence the element $g_1 g_2^d$ has order $m_{g_1} m_{g_2}/d = \mathrm{lcm}(m_{g_1}, m_{g_2})$.

Next, consider the elements $g_1 g_2^d$ and g_3. Let

$$d' = \gcd(\mathrm{lcm}(m_{g_1}, m_{g_2}), m_{g_3}).$$

Then $g_1 g_2^d g_3^{d'}$ has order

$$\mathrm{lcm}(\mathrm{lcm}(m_{g_1}, m_{g_2}), m_{g_3}) = \mathrm{lcm}(m_{g_1}, m_{g_2}, m_{g_3}).$$

Continuing in this manner we can construct an element of order $\mathrm{lcm}(m_{g_1}, m_{g_2}, \dots, m_{g_{n-1}})$, which is equal to f by Proposition 5.7.12. □

Example 5.7.14 Consider the direct product group

$$\mathbb{Z}_2 \times \mathbb{Z}_3 = \{(0,0), (0,1), (0,2), (1,0), (1,1), (1,2)\}$$

whose elements have orders

$$1, 3, 3, 2, 6, 6,$$

respectively. By Proposition 5.7.12, the exponent of $\mathbb{Z}_2 \times \mathbb{Z}_3$ is $\mathrm{lcm}(1, 3, 3, 2, 6, 6) = 6$ and $(1, 2)$ has order 6. Note that $\mathbb{Z}_2 \times \mathbb{Z}_3 = \langle (1, 2) \rangle$ and so $\mathbb{Z}_2 \times \mathbb{Z}_3$ is a finite cyclic group of order 6.

We close the chapter with a classical theorem of number theory.

Theorem 5.7.15 (The Chinese Remainder Theorem) *Let n_1, n_2 be integers that satisfy $n_1, n_2 > 0$ and $\gcd(n_1, n_2) = 1$ and let a_1, a_2 be any integers. Then the system of congruences*

$$\begin{cases} x \equiv a_1 \pmod{n_1} \\ x \equiv a_2 \pmod{n_2} \end{cases}$$

has a unique solution modulo $n_1 n_2$.

Proof By Bezout's lemma, there exist integers x_1, x_2 so that $n_1 x_1 + n_2 x_2 = 1$. Then

$$(a_1 - a_2)(n_1 x_1 + n_2 x_2) = (a_1 - a_2),$$

and

$$x = a_1 + n_1 x_1 (a_2 - a_1) = a_2 + n_2 x_2 (a_1 - a_2),$$

is a solution modulo $n_1 n_2$.

To show that this solution is unique modulo n_1n_2, let x' be some other solution. We may assume without loss of generality that $x > x'$. Then $n_1 \mid (x - a_1)$ and $n_1 \mid (x' - a_1)$, and so $n_1 \mid (x - x')$. Likewise, $n_2 \mid (x - x')$. Thus $x - x'$ is a common multiple of n_1 and n_2 and so, $\mathrm{lcm}(n_1, n_2) \le x - x'$. By the division algorithm,

$$x - x' = \mathrm{lcm}(n_1, n_2) \cdot q + r$$

for some q, r with $0 \le r < \mathrm{lcm}(n_1, n_2)$. But then $r = (x - x') - \mathrm{lcm}(n_1, n_2)$ is a common multiple of n_1 and n_2 and so $r = 0$. Thus $\mathrm{lcm}(n_1, n_2) \mid (x - x')$. By Proposition 5.7.9, $n_1n_2 = \mathrm{lcm}(n_1, n_2)$, thus $x' \equiv x \pmod{n_1n_2}$. □

Example 5.7.16 Let $n_1 = 2, n_2 = 3, a_1 = 1, a_2 = 2$. Then the Chinese Remainder theorem applies to show that there is a unique residue x in $\mathbb{Z}_6$ for which

$$\begin{cases} x \equiv 1 \pmod{2} \\ x \equiv 2 \pmod{3}. \end{cases}$$

In fact, as one can check, $x = 5$. Observe that the residue $x = 5$ has order 6 in $\mathbb{Z}_6$ and the element $(a_1, a_2) = (1, 2)$ has order 6 in the group product $\mathbb{Z}_2 \times \mathbb{Z}_3$. We have $\mathbb{Z}_6 = \langle 5 \rangle$ and $\mathbb{Z}_2 \times \mathbb{Z}_3 = \langle (1, 2) \rangle$. Both $\mathbb{Z}_6$ and $\mathbb{Z}_2 \times \mathbb{Z}_3$ are cyclic groups of order 6; there is an isomorphism of groups

$$\psi : \mathbb{Z}_6 \to \mathbb{Z}_2 \times \mathbb{Z}_3$$

defined by $5 \mapsto (1, 2)$.

□

The Chinese Remainder theorem (CRT) plays a central role in our development of cryptographic systems in subsequent chapters. For instance, the CRT is critical to the definition of the RSA public key cryptosystem (Section 9.2) and the RSA Digital Signature scheme (Section 10.2). It is also used in the construction of the Blum-Blum-Shub pseudorandom sequence in Section 11.4.5.

5.8 Exercises

1. Let $S = \{a, b, c\}$, and let $\star$ be the binary operation on S defined by the following table:

$\star$	a	b	c
a	c	c	a
b	b	a	a
c	a	b	b

(a) Compute $(b \star (a \star a)) \star (b \star c)$.
(b) Determine whether $\langle S, \star \rangle$ is a semigroup.

2. Let G be a group. Prove that the identity element e is unique. Prove that for $a \in G$, there exists a unique inverse element a^{-1} with $aa^{-1} = e = a^{-1}a$.
3. Let $\mathrm{GL}_2(\mathbb{R})$ be the group of invertible 2×2 matrices over $\mathbb{R}$.

(a) Show that $A = \begin{pmatrix} 4 & 5 \\ 2 & 3 \end{pmatrix}$ is in $\mathrm{GL}_2(\mathbb{R})$.
(b) Compute A^{-1}.

4. Let S_5 denote the symmetric group on the 5 letter set $\Sigma = \{0, 1, 2, 3, 4\}$, and let

$$\sigma = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 0 & 4 \end{pmatrix} \quad \text{and} \quad \tau = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 3 & 4 & 2 \end{pmatrix}$$

be elements of S_5.

Compute the following.

(a) τ^{-1}
(b) $\sigma \circ \tau$
(c) $\tau \circ \sigma$
(d) $|S_5|$

5. Let S_2 denote the symmetric group on the 2 letter set $\Sigma = \{0, 1\}$.

(a) List the elements of S_2 in permutation notation.
(b) Compute the group table for S_2.

6. Let $\sigma \in S_5$ be the permutation given in Exercise 4. Decompose σ into a product of transpositions. Is σ even or odd?
7. Prove that a permutation in S_n, $n \geq 2$, cannot be both even and odd.
8. Compute the following.

(a) $(17 \bmod 6)$
(b) $(256 \bmod 31)$
(c) $(-2245 \bmod 7)$

9. Compute the binary operation table for the group of residues $\langle \mathbb{Z}_6, + \rangle$.
10. Compute $(5, 7) \cdot (2, 13)$ in the direct product group $\mathbb{Z}_8 \times \mathbb{Z}_{20}$.
11. List all of the subgroups of the group $\langle \mathbb{Z}_{20}, + \rangle$.
12. Let G be a group and H be a finite non-empty subset of G that is closed under the binary operation of G. Prove that H is a subgroup of G.
13. Let $\psi : \mathbb{Z}_6 \to \mathbb{Z}_5$ be the map defined as $a \mapsto (a \bmod 5)$ for $a \in \mathbb{Z}_6$. Determine whether ψ is a homomorphism of groups.
14. Compute the cyclic subgroup of $\mathbb{R}_+$ generated by π.
15. Compute the cyclic subgroup of $\mathbb{Z}$ generated by -1.

16. Let $\mathbb{Q}^\times = \{x \in \mathbb{Q} \mid x \neq 0\}$ denote the multiplicative group of non-zero rationals. Determine whether $\mathbb{Q}^\times$ is finitely generated.
17. Compute $d = \gcd(28, 124)$. Find integers x, y so that $28x + 124y = d$.
18. Find the order of the element 3 in $\mathbb{Z}_{15}$.
19. Find the order of the element $\sigma = \begin{pmatrix} 0 & 1 & 2 & 3 \\ 3 & 2 & 1 & 0 \end{pmatrix}$ in S_4.
20. Find the order of the element (2, 3) in the direct product group $\mathbb{Z}_4 \times \mathbb{Z}_8$.
21. Compute the exponent and the order of the group $\mathbb{Z}_{10} \times \mathbb{Z}_{12}$.
22. Let $d = \gcd(a, b)$. Prove that $\gcd(a/d, b/d) = 1$.
23. Find the unique solution in $\mathbb{Z}_{4300}$ of the system of congruences

$$\begin{cases} x \equiv 30 \pmod{43} \\ x \equiv 87 \pmod{100}. \end{cases}$$

Chapter 6
Algebraic Foundations: Rings and Fields

6.1 Introduction to Rings and Fields

In contrast to a group, a ring is a set together with two binary operations.

Definition 6.1.1 A **ring** is a non-empty set R together with two binary operations, addition, $+$, and multiplication, $\cdot$, that satisfy

(i) $\langle R, +\rangle$ is an abelian group with identity element 0_R
(ii) For all $a, b, c \in R$,

$$a \cdot (b \cdot c) = (a \cdot b) \cdot c$$

(iii) For all $a, b, c \in R$,

$$a \cdot (b + c) = a \cdot b + a \cdot c,$$

and

$$(a + b) \cdot c = a \cdot c + b \cdot c$$

It is often more convenient to denote the multiplication $a \cdot b$ by ab.

The integers $\mathbb{Z}$, together with ordinary addition $+$ and ordinary multiplication $\cdot$, are a ring called the **ring of integers**. Likewise, $\langle \mathbb{Q}, +, \cdot\rangle$ and $\langle \mathbb{R}, +, \cdot\rangle$ are rings. The collection of residues $\mathbb{Z}_n$ modulo n is a ring under addition $+_n$ and multiplication $\cdot_n$ defined as

$$(a \bmod n) +_n (b \bmod n) = ((a + b) \bmod n),$$

$$(a \bmod n) \cdot_n (b \bmod n) = ((ab) \bmod n).$$

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_6

For instance, $\mathbb{Z}_4$ is a ring with binary operations given by the tables

$+_4$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

$\cdot_4$	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	1

Here are some ways to construct a new ring from a given ring.

Example 6.1.2 Let R be a ring, and let x be an indeterminate. The collection of all polynomials

$$a_0 + a_1x + a_2x^2 + \cdots + a_{n-1}x^{n-1} + a_nx^n, \quad a_i \in R,$$

is a ring under the usual polynomial addition and multiplication, thus

$$\sum_{i=0}^{n} a_ix^i + \sum_{i=0}^{n} b_ix^i = \sum_{i=0}^{n} (a_i + b_i)x^i,$$

$$\left(\sum_{i=0}^{m} a_ix^i\right)\left(\sum_{j=0}^{n} b_jx^j\right) = \sum_{i=0}^{m}\sum_{j=0}^{n} a_ib_jx^{i+j}.$$

We denote this **ring of polynomials** by $R[x]$. If $a_n \neq 0$, then the **degree** of $p(x)$ is n. We assume that the degree of the zero polynomial $p(x) = 0$ is $-\infty$. □

If we take the ring R to be the polynomial ring $R[x]$ and let y be an indeterminate, then we can form the ring of polynomials in y over $R[x]$, which is denoted as $R[x][y]$ (or alternatively, as $R[x, y]$). This is the ring of polynomials in two indeterminates x and y. In $R[x, y]$, x and y commute: $xy = yx$.

This construction can be extended inductively: let R be a ring, and let $\{x_1, x_2, \ldots, x_n\}$ denote a set of indeterminates. The set of all polynomials in $\{x_1, x_2, \ldots, x_n\}$ over R is a ring with the obvious polynomial addition and multiplication; the x_i commute. This ring of polynomials is denoted as $R[x_1, x_2, \ldots, x_n]$.

Example 6.1.3 Let R be a ring and let $\mathrm{Mat}_n(R)$ denote the collection of all $n \times n$ matrices with entries in R. Then $\mathrm{Mat}_n(R)$ is a ring under matrix addition and multiplication. The ring $\mathrm{Mat}_n(R)$ is the **ring of** $n \times n$ **matrices** over R. □

For instance, if we take $R = \mathbb{R}$, then $\mathrm{Mat}_n(\mathbb{R})$ is the ring of $n \times n$ matrices with entries in $\mathbb{R}$, which is quite familiar to students of linear algebra.

We may also take $n = 2$ and $R = \mathbb{Z}_{10}$ to obtain $\mathrm{Mat}_2(\mathbb{Z}_{10})$, which is the ring of 2×2 matrices over $\mathbb{Z}_{10}$. In $\mathrm{Mat}_2(\mathbb{Z}_{10})$, we have

$$\begin{pmatrix} 3 & 7 \\ 6 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 9 \\ 4 & 1 \end{pmatrix} = \begin{pmatrix} 3+0 & 7+9 \\ 6+4 & 0+1 \end{pmatrix} = \begin{pmatrix} 3 & 6 \\ 0 & 1 \end{pmatrix},$$

$$\begin{pmatrix} 3 & 7 \\ 6 & 0 \end{pmatrix}\begin{pmatrix} 0 & 9 \\ 4 & 1 \end{pmatrix} = \begin{pmatrix} 3\cdot 0+7\cdot 4 & 3\cdot 9+7\cdot 1 \\ 6\cdot 0+0\cdot 4 & 6\cdot 9+0\cdot 1 \end{pmatrix} = \begin{pmatrix} 8 & 4 \\ 0 & 4 \end{pmatrix}.$$

(Note: entries are reduced modulo 10.)

Just as we did for groups (see Section 5.4), we can form the direct product of a collection of rings.

Example 6.1.4 If R_i, for $i = 1, \ldots, k$, is a collection of rings, then the Cartesian product $\displaystyle\prod_{i=1}^{k} R_i$ is a ring, where

$$(a_1, a_2, \ldots, a_k) + (b_1, b_2, \ldots, b_k) = (a_1 + b_1, a_2 + b_2, \ldots, a_k + b_k),$$

and

$$(a_1, a_2, \ldots, a_k)(b_1, b_2, \ldots, b_k) = (a_1b_1, a_2b_2, \ldots, a_kb_k).$$

Here, $a_i + b_i$ denotes the addition in the component ring R_i, and a_ib_i denotes the multiplication in the component ring R_i, $1 \le i \le k$. □

To illustrate a direct product, we take $k = 2$ and $R_1 = R_2 = \mathbb{Z}$ and consider the ring $\mathbb{Z} \times \mathbb{Z}$. In $\mathbb{Z} \times \mathbb{Z}$, $(5, 4) + (-3, 1) = (2, 5)$ and $(3, 2)(-1, 6) = (-3, 12)$.

A ring R is **commutative** if

$$ab = ba$$

for all $a, b \in R$. A **ring with unity** is a ring R for which there exists a multiplicative identity element, denoted by 1_R and distinct from 0_R that satisfies

$$1_R a = a = a 1_R$$

for all $a \in R$.

A **subring** of a ring with unity R is a subset of S of R which is a ring under the binary operations of R restricted to S. If S is a subring of R, then S is a ring with unity $0_S = 0_R$, $1_S = 1_R$. For example, $\mathbb{Z}$ is a subring of $\mathbb{Q}$.

Suppose R is a ring with unity. A **unit** of R is an element $u \in R$ for which there exists an element $a \in R$ that satisfies

$$au = 1_R = ua.$$

The element a is a **multiplicative inverse** of u and is denoted by u^{-1}.

For example, the ring $\mathbb{Z}$ is a ring with unity with identity element $1_{\mathbb{Z}} = 1$, and the only units are 1 and -1. On the other hand, in the ring $\mathbb{Q}$, $1_{\mathbb{Q}} = 1$, and every non-zero element is a unit.

Let R be a ring. A **zero divisor** in R is a non-zero element $a \in R$ for which there exists a non-zero element $b \in R$ so that $ab = 0$. For example 4 and 6 are zero divisors in the ring $\mathbb{Z}_8$ since $4 \cdot 6 = 0$ in $\mathbb{Z}_8$ (equivalently, $(4 \cdot 6) \equiv 0 \pmod 8$). A commutative ring with unity in which there are no zero divisors is an **integral domain**. For instance, $\mathbb{Z}$ is an integral domain.

A **division ring** is a ring R with unity in which every non-zero element is a unit. A commutative division ring is a **field**. For example, the ring $\mathbb{Q}$ is a field, as is $\mathbb{R}$. It is not hard to show that every field is an integral domain, and however, the converse is false; see Section 6.5, Exercise 2.

We note that if F is a field, then the ring of polynomials $F[x]$ is an integral domain.

Let F be a field. A **subfield** of F is a subring E of F, which is a field under the binary operations of F restricted to E. For example, $\mathbb{Q}$ is a subfield of $\mathbb{R}$.

Let E and F be fields. If E is a subfield of F, then E is a **field extension** of F and we write E/F. For instance, $\mathbb{R}$ is a field extension of $\mathbb{Q}$.

6.1.1 Polynomials in $F[x]$

Let F be a field and let $F[x]$ be the ring of polynomials over F.

Proposition 6.1.5 (Division Theorem) *Let $f(x)$ and $g(x)$ be polynomials in $F[x]$ with $f(x) \neq 0$. There exist unique polynomials $q(x)$ and $r(x)$ for which*

$$g(x) = f(x)q(x) + r(x)$$

where $\deg(r(x)) < \deg(f(x))$.

Proof Our proof is by induction on the degree of $g(x)$. If $\deg(g(x)) < \deg(f(x))$, then $g(x) = f(x) \cdot 0 + g(x)$, as required by the Division Theorem. So we assume that $\deg(g(x)) \geq \deg(f(x))$. Let $f(x) = a_0 + a_1x + \cdot + a_mx^m$, $a_m \neq 0$, $g(x) = b_0 + b_1x + \cdots + b_{m+n}x^{m+n}$, $b_{m+n} \neq 0$, $n \geq 0$. For the trivial case, suppose that $\deg(g(x)) = 0$. Then $g(x) = c$, $f(x) = d$, with c, d non-zero, and so, $c = (c/d)d + 0$, which proves the theorem.

Next, put

$$h(x) = g(x) - \left(\frac{b_{m+n}}{a_m}\right) x^n f(x).$$

Then $\deg(h(x)) < \deg(g(x))$, and thus by the induction hypothesis, there exist $q_1(x)$ and $r(x)$ so that

$$h(x) = f(x)q_1(x) + r(x),$$

with $\deg(r(x)) < f(x)$. Now,

$$g(x) = f(x)\left(q_1(x) + \left(\frac{b_{m+n}}{a_m}\right)x^n\right) + r(x),$$

which proves the theorem.

We leave it to the reader to prove the uniqueness of $q(x)$ and $r(x)$. □

Let $f(x) \in F[x]$. A **zero** (or **root**) of $f(x)$ in F is an element $a \in F$ for which $f(a) = 0$.

Proposition 6.1.6 (Factor Theorem) *Let $f(x) \in F[x]$ be a polynomial and let $a \in F$ be a zero of $f(x)$. Then $x - a$ divides $f(x)$.*

Proof By the Division Theorem, there exist polynomials $q(x), r(x) \in F[x]$ for which

$$f(x) = (x - a)q(x) + r(x),$$

with $\deg(r(x)) < \deg(x - a) = 1$, and so, $r(x) = r$ for some $r \in F$. Now,

$$0 = f(a) = (a - a)q(a) + r = 0 \cdot q(a) + r = r,$$

and so, $r(x) = 0$, and consequently, $x - a \mid f(x)$. □

Proposition 6.1.7 *Let $f(x) \in F(x)$ be a polynomial of degree $n \geq 0$. Then $f(x)$ can have at most n roots in F.*

Proof If $f(x)$ has degree 0, then $f(x) = c$, $c \neq 0$, and so, $f(x)$ has no zeros in F. So we assume that $\deg(f(x)) = n \geq 1$. If $a_1 \in F$ is a zero of $f(x)$, then by the Factor Theorem,

$$f(x) = (x - a_1)g_1(x),$$

where $g_1(x)$ is a polynomial in $F[x]$ of degree $n - 1$. If $a_2 \in F$ is a zero of $g_1(x)$, then again by the Factor Theorem,

$$f(x) = (x - a_1)(x - a_2)g_2(x),$$

with $g_2(x) \in F[x]$ with $\deg(g_2(x)) = n - 2$. We continue in this manner until we arrive at the factorization

$$f(x) = (x - a_1)(x - a_2)(x - a_3) \cdots (x - a_k)g_k(x),$$

where $g_k(x)$ has no roots in F and $\deg(g_k) = n - k$. Now, $a_1, a_2, \dots, a_k$ is a list of k zeros of $f(x)$ in F. Note that $k \le n$ since the degrees on the left-hand side and the right-hand side must be equal.

We claim that the a_i are all of the zeros of $f(x)$ in F. By way of contradiction, suppose that $b \in F$ satisfies $b \ne a_i$ for $1 \le i \le k$, with $f(b) = 0$. Then

$$0 = f(b) = (b - a_1)(b - a_2)(b - a_3) \cdots (b - a_k)g_k(b),$$

with none of the factors on the right-hand side equal to 0. Thus F has zero divisors, which is impossible since F is an integral domain. □

6.2 The Group of Units of $\mathbb{Z}_n$

Let R be a ring with unity 1_R. As a ring, R is provided with two binary operations: addition and multiplication. Under the addition, R is an abelian group. Is there a group arising from the multiplication of R?

Proposition 6.2.1 *Let R be a ring with unity. Let $U(R)$ denote the collection of units of R. Then $U(R)$ together with the ring multiplication is a group.*

Proof We only have to notice that ab is a unit whenever a and b are. □

The group $\langle U(R), \cdot \rangle$ is the **group of units** of R. For example, $\mathbb{Z}$ is a (commutative) ring with unity. The group of units $U(\mathbb{Z})$ is the (abelian) group $\{1, -1\}$ with group table:

$\cdot$	1	-1
1	1	-1
-1	-1	1

The ring of residues $\mathbb{Z}_n$ is a commutative ring with unity. Our goal in this section is to compute $U(\mathbb{Z}_n)$.

Proposition 6.2.2 *The units of the ring $\mathbb{Z}_n$ are precisely those residues m, $1 \le m \le n - 1$, that satisfy* $\gcd(n, m) = 1$.

Proof If $\gcd(n, m) = 1$, then by Proposition 5.7.8, there exist x, y so that $mx + ny = 1$. Thus $1 - mx = ny$, which says that n divides $1 - mx$. Hence $mx \equiv 1 \pmod{n}$. Consequently, m is a unit with $m^{-1} \equiv x \pmod{n}$.

For the converse, suppose that m is a unit in $\mathbb{Z}_n$. Then there exists $x \in \mathbb{Z}_n$ so that $mx = 1$ in $\mathbb{Z}_n$. Thus $(mx \bmod n) = 1$ so that $mx = nq + 1$ for some $q \in \mathbb{Z}$. Thus the algorithm EUCLID yields $\gcd(n, mx) = 1$, which implies that $\gcd(n, m) = 1$. □

For example, by Proposition 6.2.2, $U(\mathbb{Z}_8) = \{1, 3, 5, 7\}$. The group table for $U(\mathbb{Z}_8)$ is

$\cdot_8$	1	3	5	7
1	1	3	5	7
3	3	1	7	5
5	5	7	1	3
7	7	5	3	1

For another example, we see that $U(\mathbb{Z}_5) = \{1, 2, 3, 4\}$. The group table for $U(\mathbb{Z}_5)$ is

$\cdot_5$	1	2	3	4
1	1	2	3	4
2	2	4	1	3
3	3	1	4	2
4	4	3	2	1

In $U(\mathbb{Z}_5)$, observe that $2 \cdot 3 = 1$, and thus $2^{-1} = 3$.

Proposition 6.2.3 *Let p be a prime number. Then $U(\mathbb{Z}_p) = \{1, 2, 3, \dots, p-1\}$. Consequently, $|U(\mathbb{Z}_p)| = p - 1$.*

Proof $\{1, 2, 3, \dots, p-1\}$ are the residues in $\mathbb{Z}_p$ that are relatively prime to p. □

Corollary 6.2.4 *Let p be a prime number. Then $\mathbb{Z}_p$ is a field.*

Proof Certainly, $\mathbb{Z}_p$ is a commutative ring with unity. By Proposition 6.2.3, it is a division ring. □

Let $\mathbb{Z}_+$ denote the set of positive integers.

Definition 6.2.5 (Euler's Function) Let $\phi : \mathbb{Z}_+ \to \mathbb{Z}$ be the function defined by the rule: $\phi(n)$ is the number of integers $1 \le m \le n-1$ that satisfy $\gcd(m, n) = 1$. Then ϕ is **Euler's function**.

For instance, $\phi(5) = 4$, $\phi(8) = 4$. If p is prime, then $\phi(p) = p - 1$. Proposition 6.2.2 says that $\phi(n) = |U(\mathbb{Z}_n)|$.

Theorem 6.2.6 (Fermat's Little Theorem) *Let p be a prime number and let a be an integer with* $\gcd(a, p) = 1$. *Then*

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

Proof Let $a' = (a \bmod p)$. Then $\gcd(a', p) = 1$, and so $a' \in U(\mathbb{Z}_p)$. By Proposition 6.2.3, $|U(\mathbb{Z}_p)| = p - 1$, and so by Proposition 5.7.4, $(a')^{p-1} \equiv 1 \pmod{p}$.

Now, $(a' \bmod p) = (a \bmod p)$ and so we may write $a = a' + pm$ for some integer m. Then by the binomial theorem,

$$a^{p-1} = (a' + pm)^{p-1} = (a')^{p-1} + \sum_{i=1}^{p-1} \binom{p-1}{i} (a')^{p-1-i} (pm)^i,$$

and so,

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

The result follows. □

Corollary 6.2.7 *Let p be a prime number and let a be any integer. Then*

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

Proof If $\gcd(a, p) = 1$, then $a^{p-1} \equiv 1 \pmod{p}$ by Proposition 6.2.6. Thus $a \cdot a^{p-1} = a^p \equiv a \pmod{p}$. If $p \mid a$, then $a \equiv 0 \pmod{p}$, and hence $a^p \equiv 0^p \equiv 0 \equiv a \pmod{p}$. □

Proposition 6.2.8 *Let p be prime and let $a \in U(\mathbb{Z}_p)$. Then a is its own inverse in $U(\mathbb{Z}_p)$ if and only if $a \equiv 1 \pmod{p}$ or $a \equiv -1 \pmod{p}$.*

Proof If $a \equiv \pm 1 \pmod{p}$, then $a^2 \equiv 1 \pmod{p}$, so $a \equiv a^{-1} \pmod{p}$. Conversely, suppose that $a^2 \equiv 1 \pmod{p}$. Then $(a + 1)(a - 1) \equiv 0 \pmod{p}$. Since p is prime, this implies that $p \mid (a + 1)$ or $p \mid (a - 1)$, and so $a \equiv \pm 1 \pmod{p}$. □

Theorem 6.2.9 (Wilson's Theorem) *Let p be prime. Then $(p - 1)! \equiv -1 \pmod{p}$.*

Proof If $p = 2$, then $(p - 1)! \equiv 1! \equiv -1 \pmod{2}$. So we assume that $p > 2$. Consider the list of residues

$$1, 2, 3 \ldots, p - 2, p - 1.$$

Each of these residues has a unique inverse in the list. By Proposition 6.2.8, the inverse of 1 is 1, the inverse of $p - 1$ is $p - 1$, and all other residues have distinct inverses in the list $2, 3, \ldots, p - 2$. Thus,

$$(p - 1)! = \prod_{i=1}^{p-1} i \equiv p - 1 \equiv -1 \pmod{p}.$$

□

Euler has given a generalization of Fermat's Little Theorem.

Theorem 6.2.10 (Euler's Theorem) *Let $n > 1$ be an integer and let a be an integer with* $\gcd(a, n) = 1$. *Then*

$$a^{\phi(n)} \equiv 1 \pmod{n}.$$

Proof Let $a' = (a \bmod n)$. Since $\gcd(a, n) = 1$, $\gcd(a', n) = 1$, and so $a' \in U(\mathbb{Z}_n)$. Thus $(a')^{\phi(n)} \equiv 1 \pmod{n}$ by Proposition 5.7.4. It follows that $a^{\phi(n)} \equiv 1 \pmod{n}$. □

6.2.1 A Formula for Euler's Function

As we have seen, Euler's function ϕ counts the number of units in $\mathbb{Z}_n$. Our goal here is to derive a formula for ϕ. We already know that $\phi(p) = p - 1$ for p prime. We begin with a generalization of this formula.

Proposition 6.2.11 *Let p be a prime number and let $a \geq 1$ be an integer. Then*

$$\phi(p^a) = p^a - p^{a-1}.$$

Proof The integers m with $1 \leq m \leq p^a$ that are not relatively prime to p^a are precisely those of the form $m = kp$ for $1 \leq k \leq p^{a-1}$ and there are exactly p^{a-1} such integers. So the number of integers m, $1 \leq m \leq p^a$, that are relatively prime to p^a is $p^a - p^{a-1}$. Thus $\phi(p^a) = p^a - p^{a-1}$. □

Proposition 6.2.12 *Suppose that $m, n \geq 1$ are integers with $gcd(m, n) = 1$. Then $\phi(mn) = \phi(m)\phi(n)$.*

Proof We begin by writing all integers k, $1 \leq k \leq mn$, in the following arrangement (m rows × n columns):

$$\begin{array}{ccccc}
\hline
1 & m+1 & 2m+1 & \cdots & (n-1)m+1 \\
2 & m+2 & 2m+2 & \cdots & (n-1)m+2 \\
3 & m+3 & 2m+3 & \cdots & (n-1)m+3 \\
\vdots & \vdots & \vdots & & \vdots \\
m & 2m & 3m & \cdots & mn \\
\hline
\end{array}$$

Now, exactly $\phi(m)$ rows of this matrix contain integers that are relatively prime mn, and each of these $\phi(m)$ rows contains exactly $\phi(n)$ integers that are relatively prime to mn. Thus the total number of integers $1 \leq k \leq mn$ that are relatively prime to mn is $\phi(m)\phi(n)$. □

Proposition 6.2.13 *Let $n \geq 2$ and let $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ be the prime factor decomposition of n. Then*

$$\phi(n) = n\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\cdots\left(1 - \frac{1}{p_k}\right).$$

Proof Since $\gcd(p_i^{e_i}, p_j^{e_j}) = 1$ for $1 \leq i, j \leq k$, $i \neq j$, we have $\phi(n) = \phi(p_1^{e_1})\phi(p_2^{e_2})\cdots\phi(p_k^{e_k})$ by Proposition 6.2.12. Thus

$$\begin{aligned}
\phi(n) &= (p_1^{e_1} - p_1^{e_1-1})(p_2^{e_2} - p_2^{e_2-1})\cdots(p_k^{e_k} - p^{e_k-1}) \quad \text{by Proposition 6.2.11} \\
&= p_1^{e_1}\left(1 - \frac{1}{p_1}\right)p_2^{e_2}\left(1 - \frac{1}{p_2}\right)\cdots p_k^{e_k}\left(1 - \frac{1}{p_k}\right) \\
&= (p_1^{e_1} p_2^{e_2}\cdots p_k^{e_k})\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\cdots\left(1 - \frac{1}{p_k}\right) \\
&= n\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\cdots\left(1 - \frac{1}{p_k}\right).
\end{aligned}$$

□

For example, $\phi(2200) = \phi(2^3 \cdot 5^2 \cdot 11) = 2200(1 - \frac{1}{2})(1 - \frac{1}{5})(1 - \frac{1}{11}) = 800.$

6.3 $U(\mathbb{Z}_p)$ Is Cyclic

We prove a theorem that is of fundamental importance in cryptography.

Theorem 6.3.1 *Let p be a prime number. Then the group of units $U(\mathbb{Z}_p)$ is cyclic, i.e., there exists an element $g \in U(\mathbb{Z}_p)$ for which $\{g^n : n \in \mathbb{Z}\} = U(\mathbb{Z}_p)$.*

Proof Since $\mathbb{Z}_p$ is a commutative ring with unity, $U(\mathbb{Z}_p)$ is a finite abelian group whose identity element is the residue 1. Let f be the exponent of $U(\mathbb{Z}_p)$. We have $f \leq |U(\mathbb{Z}_p)| = p - 1$.

By Corollary 6.2.4, $\mathbb{Z}_p$ is a field. Consider the polynomial $x^f - 1$ in $\mathbb{Z}_p[x]$. By the definition of exponent, $g^f = 1$ for all $g \in U(\mathbb{Z}_p)$ and so $x^f - 1$ has $p - 1$ distinct zeros in $\mathbb{Z}_p$. Thus $f = p - 1$ by Proposition 6.1.7.

By Proposition 5.7.13, there exists an element $g \in U(\mathbb{Z}_p)$ which has order $p - 1$. But then $|\langle g\rangle| = p - 1$ so that $\langle g\rangle = U(\mathbb{Z}_p)$. Thus $U(\mathbb{Z}_p)$ is cyclic. □

An element $g \in U(\mathbb{Z}_p)$ that generates $U(\mathbb{Z}_p)$ is a **primitive root** modulo p. For example, 2 is a primitive root modulo 5 since 2 generates $U(\mathbb{Z}_5)$, and 3 is a primitive root modulo 7 since 3 generates $U(\mathbb{Z}_7)$.

For n not prime, $U(\mathbb{Z}_n)$ may not be cyclic. For example, $U(\mathbb{Z}_8)$ is not cyclic.

Theorem 6.3.2 *$U(\mathbb{Z}_n)$ is cyclic if and only if $n = 2, 4, p^e, 2p^e$, where p is an odd prime and $e \geq 1$.*

Proof For a proof, see [28, Proposition 4.1.3] or [47, Theorem 9.15]. □

Let $n > 0$ and let $a \in U(\mathbb{Z}_n)$. Then the order of a in $U(\mathbb{Z}_n)$ is the order of the cyclic subgroup $\langle a \rangle \leq U(\mathbb{Z}_n)$; it is the smallest positive integer m for which $a^m = 1$ in $U(\mathbb{Z}_n)$, or equivalently, $a^m \equiv 1 \pmod{n}$ (Proposition 5.7.2).

Theorem 6.3.1 says that $U(\mathbb{Z}_p)$, p prime, contains an element of order $p - 1$. For composite moduli, we have the following.

Proposition 6.3.3 *Let $n = n_1 n_2$, $\gcd(n_1, n_2) = 1$. Let $a \in U(\mathbb{Z}_n)$, $a_1 = (a \bmod n_1)$, $a_2 = (a \bmod n_2)$. Let l be the order of a_1 in $U(\mathbb{Z}_{n_1})$ and let m be the order of a_2 in $U(\mathbb{Z}_{n_2})$. Then the order of a in $U(\mathbb{Z}_n)$ is $\mathrm{lcm}(l, m)$.*

Proof We have $a_1^{\mathrm{lcm}(l,m)} \equiv 1 \pmod{n_1}$ and $a_2^{\mathrm{lcm}(l,m)} \equiv 1 \pmod{n_2}$ since $l \mid \mathrm{lcm}(l, m)$ and $m \mid \mathrm{lcm}(m, l)$. By the binomial theorem,

$$a^{\mathrm{lcm}(l,m)} \equiv a_1^{\mathrm{lcm}(l,m)} \pmod{n_1} \quad \text{and} \quad a^{\mathrm{lcm}(l,m)} \equiv a_2^{\mathrm{lcm}(l,m)} \pmod{n_2}.$$

So $n_1 \mid (a^{\mathrm{lcm}(l,m)} - 1)$ and $n_2 \mid (a^{\mathrm{lcm}(l,m)} - 1)$, and thus $a^{\mathrm{lcm}(l,m)} - 1$ is a common multiple of n_1, n_2. By the division algorithm, $\mathrm{lcm}(n_1, n_2) \mid (a^{\mathrm{lcm}(l,m)} - 1)$. Since $\gcd(n_1, n_2) = 1$, $\mathrm{lcm}(n_1, n_2) = n_1 n_2 = n$, and so, $n \mid (a^{\mathrm{lcm}(l,m)} - 1)$, and hence $a^{\mathrm{lcm}(l,m)} \equiv 1 \pmod{n}$.

We show that $\mathrm{lcm}(l, m)$ is the order of a in $U(\mathbb{Z}_n)$. Suppose $a^k \equiv 1 \pmod{n}$ for some k. Then $a^k \equiv a_1^k \equiv 1 \pmod{n_1}$. Write $k = lq + r$, $0 \leq r < l$. Then $r = 0$, hence $l \mid k$. Likewise, $m \mid k$, and so, k is a common multiple of l, m which yields $k \geq \mathrm{lcm}(l, m)$. □

For example, the order of 2 in $U(\mathbb{Z}_{35})$ can be computed as follows. The order of 2 in $U(\mathbb{Z}_5)$ is 4, and the order of 2 in $U(\mathbb{Z}_7)$ is 3. Thus the order of 2 in $U(\mathbb{Z}_{35})$ is $\mathrm{lcm}(4, 3) = 12$.

In Section 11.4, we will apply Proposition 6.3.3 to the case $n = pq$, where p and q are distinct primes.

6.4 Exponentiation in $\mathbb{Z}_n$

Many important cryptosystems depend on the computation of

$$(a^b \bmod n),$$

where a, b, and n are integers with $a > 0$, $b \geq 0$, and $n > 0$. Here is an algorithm that computes $(a^b \bmod n)$.

Algorithm 6.4.1 (EXP_MOD_n)

```
Input:     integers $a, b, n, n > a \geq 0, n > b \geq 0, n > 0, m = \lfloor n \rfloor + 1$,
           $b$ encoded in binary as $b = b_m b_{m-1} \cdots b_1$
Output:    $(a^b \bmod n)$
Algorithm:
           $c \leftarrow 1$
           for $i = 1$ to $m$
              if $b_i = 1$ then $c \leftarrow (ca \bmod n)$
              $a \leftarrow (a^2 \bmod n)$
           next $i$
           output $c$
```

□

To see how EXP_MOD_n works, we compute $(3^9 \bmod 11)$. Here $a = 3$, $b = 9$, and $n = 11$, so $m = 4$. In binary, $b = (9)_2 = 1001$. On the first iteration of the loop, c becomes 3, and a is 9. On the second iteration, c remains the same, and a becomes $(81 \bmod 11) = 4$. On the third iteration, c remains the same, and a is $(16 \bmod 11) = 5$. On the fourth iteration, c becomes $((3 \cdot 5) \bmod 11) = 4$, which is the correct output.

Proposition 6.4.2 EXP_MOD_n *has running time* $O(m^3)$, *where* $m = \lfloor \log_2(n) \rfloor + 1$.

Proof The algorithm performs m iterations of the for-next loop. On each iteration, the algorithm performs at most $2 \cdot O(m^2)$ steps (Proposition 4.2.6). Thus the running time is $O(m^3)$. □

Remark 6.4.3 Algorithm 6.4.1 is the analog of Algorithm 4.2.6 with a^2 playing the role of $2a$. □

Remark 6.4.4 Algorithm 6.4.1 is not feasible unless we reduce modulo n; a^b is of size $m = \lfloor \log_2(a^b) \rfloor + 1 \approx b \lfloor \log_2(a) \rfloor$. Thus a^b grows too fast for a computer, unless we reduce modulo n. This is also why we require *finite* cyclic groups for use in the Diffie–Hellman key exchange protocol (see Section 12.1). □

6.4.1 Quadratic Residues

An element $a \in U(\mathbb{Z}_n)$ is a **quadratic residue** modulo n if there exists $x \in U(\mathbb{Z}_n)$ for which $x^2 \equiv a \pmod{n}$. For example, 9 is a quadratic residue modulo 10 since $7^2 \equiv 9 \pmod{10}$.

Let $n = p$ be prime. We define the **Legendre symbol** as follows:

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{if } a \text{ is a quadratic residue modulo } p \\ -1 & \text{otherwise.} \end{cases}$$

Proposition 6.4.5 (Euler's Criterion) *Let p be an odd prime and let $a \in \mathbb{Z}_p$, $a \not\equiv 0 \pmod{p}$. Then*

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod{p}.$$

Proof Suppose that $\left(\frac{a}{p}\right) = 1$. Then $x^2 \equiv a \pmod{p}$ for some $x \in \mathbb{Z}_p$ with $x \not\equiv 0 \pmod{p}$. Thus

$$a^{(p-1)/2} \equiv (x^2)^{(p-1)/2} \equiv x^{p-1} \equiv 1 \pmod{p},$$

where the last equivalence is by Fermat's Little Theorem.

On the other hand, suppose that $\left(\frac{a}{p}\right) = -1$. Then $x^2 \equiv a \pmod{p}$ has no solution in $\mathbb{Z}_p$. Thus for each residue i in $U(\mathbb{Z}_p)$, there is a unique residue $j \neq i \in U(\mathbb{Z}_p)$ with $ij \equiv a \pmod{p}$, in fact, $j = i^{-1}a$. Consequently,

$$\prod_{i=1}^{p-1} i \equiv (p-1)! \equiv a^{(p-1)/2} \pmod{p}.$$

By Wilson's theorem (Theorem 6.2.9), $(p-1)! \equiv -1 \pmod{p}$ and the result follows. □

Proposition 6.4.6 *Let p be an odd prime and let $a, b \in U(\mathbb{Z}_p)$. Then*

$$\left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$$

Proof This follows from Euler's criterion (Proposition 6.4.5). □

The Jacobi symbol generalizes the Legendre symbol. Let $n \geq 2$ be an integer with

$$n = p_1^{e_1} p_2^{e_2} \cdots p_m^{e_m},$$

for distinct primes p_i and $e_i \geq 1$. Then for $a \in U(\mathbb{Z}_n)$, the **Jacobi symbol** is defined as

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right)^{e_1} \left(\frac{a}{p_2}\right)^{e_2} \cdots \left(\frac{a}{p_m}\right)^{e_m},$$

where we take $a = (a \bmod p_i)$ in the Legendre symbol $\left(\frac{a}{p_i}\right)$ for $1 \leq i \leq m$.

The Jacobi symbol takes on the values ±1. However, in contrast to the Legendre symbol, $\left(\frac{a}{n}\right) = 1$ does not imply that a is a quadratic residue modulo n. For example, $\left(\frac{2}{15}\right) = \left(\frac{2}{3}\right)\left(\frac{2}{5}\right) = (-1)(-1) = 1$, yet 2 is not a quadratic residue modulo 15.

For $n \geq 2$, let QR_n denote the set of quadratic residues modulo n. Let

$$J_n^{(1)} = \left\{a \in U(\mathbb{Z}_n) \mid \left(\frac{a}{n}\right) = 1\right\},$$

$$J_n^{(-1)} = \left\{a \in U(\mathbb{Z}_n) \mid \left(\frac{a}{n}\right) = -1\right\}.$$

We have $QR_n \subseteq J_n^{(1)}$. To see this, suppose $a \equiv x^2 \pmod{n}$ for some integer x. Then $a \equiv x^2 \pmod{p}$ for any prime p dividing n. Thus $\left(\frac{a}{p}\right) = 1$, and so by the definition of the Jacobi symbol, $a \in J_n^{(1)}$. Of course, if $a \in J_n^{(-1)}$, then $a \notin QR_n$.

For the case $n = p$, a prime, there are $\phi(p) = p - 1$ elements in $U(\mathbb{Z}_p)$. If $p = 2$, then $J_2^{(1)} = \{1\}$, $J_2^{(-1)} = \emptyset$. So, we assume that p is an odd prime.

Proposition 6.4.7 *Let p be an odd prime. Then $|J_p^{(1)}| = |J_p^{(-1)}| = (p-1)/2$.*

Proof Note that $J_p^{(1)}$ is a subgroup of $U(\mathbb{Z}_p)$ by Proposition 6.4.6.

Let g be a primitive root modulo p. Then g is not a quadratic residue modulo p since $g^{(p-1)/2} \not\equiv 1 \pmod{p}$.

Since $\gcd(2, p-1) = 2$, the order of g^2 is $(p-1)/2$ by Proposition 5.7.10. Thus g^2 is a quadratic residue modulo p by Euler's criterion. Since $g^2 \in J_p^{(1)}$, $|J_p^{(1)}| \geq (p-1)/2$, but since $J_p^{(1)}$ is a proper subgroup of $U(\mathbb{Z}_p)$, we conclude that $|J_p^{(1)}| = (p-1)/2$. Since $\left(\frac{a}{p}\right) = \pm 1$ for all $a \in U(\mathbb{Z}_p)$, $|J_p^{(-1)}| = (p-1)/2$. □

Proposition 6.4.8 *Let $n = pq$ be a product of distinct odd primes. Then*

(i) $|J_n^{(1)}| = |J_n^{(-1)}| = (p-1)(q-1)/2$.

(ii) Exactly half of the elements of $J_n^{(1)}$ are quadratic residues modulo n, i.e., $QR_n \subseteq J_n^{(1)}$, $|QR_n| = (p-1)(q-1)/4$, $|J_n^{(1)} \backslash QR_n| = (p-1)(q-1)/4$.

Proof See Section 6.5, Exercise 19. □

Proposition 6.4.9 *Let $n = pq$ be the product of distinct odd primes. Then every element of QR_n has exactly four square roots modulo n: x, $-x$, y, and $-y$.*

Proof Let $a \in QR_n$. Then the congruence $x^2 \equiv a \pmod{n}$ has a solution $x \in \mathbb{Z}_n$. Let $x' = (x \bmod p)$ and $x'' = (x \bmod q)$. Then the congruence $x^2 \equiv a \pmod{p}$ has exactly two solutions $x = x'$ and $x = p - x'$ in $\mathbb{Z}_p$. Likewise, the congruence $x^2 \equiv a \pmod{q}$ has exactly two solutions $x = x''$ and $x = q - x''$ in $\mathbb{Z}_q$.

Next, apply the Chinese Remainder Theorem (Theorem 5.7.15) to the four systems of congruences

$$(i)\ \begin{cases} x \equiv x' \pmod{p} \\ x \equiv x'' \pmod{q} \end{cases} \qquad (ii)\ \begin{cases} x \equiv x' \pmod{p} \\ x \equiv q - x'' \pmod{q} \end{cases}$$

$$(iii)\ \begin{cases} x \equiv p - x' \pmod{p} \\ x \equiv x'' \pmod{q} \end{cases} \qquad (iv)\ \begin{cases} x \equiv p - x' \pmod{p} \\ x \equiv q - x'' \pmod{q} \end{cases}$$

to obtain four non-congruent solutions to $x^2 \equiv a \pmod{n}$: $x, -x, y$, and $-y$, where x and y are the unique solutions modulo n to (i) and (ii), respectively. □

How many (if any) of these square roots are in QR_n? We can answer this question if we specialize to Blum primes. A **Blum prime** is a prime satisfying $p \equiv 3 \pmod{4}$.

Proposition 6.4.10 *Let $n = pq$ be the product of distinct Blum primes. Let $a \in QR_n$. Then a has exactly one square root modulo n in QR_n.*

Proof By Proposition 6.4.9, there are exactly four square roots of a modulo n: $\pm x$, $\pm y$. Since p is Blum, $p = 4m + 3$, for some m, hence $(-1)^{(p-1)/2} = (-1)^{2m+1} \equiv -1 \pmod{p}$, and thus $\left(\frac{-1}{p}\right) = -1$ by Proposition 6.4.5. Likewise, $\left(\frac{-1}{q}\right) = -1$. Note this implies $\left(\frac{-x}{p}\right) = -\left(\frac{x}{p}\right)$, $\left(\frac{-x}{q}\right) = -\left(\frac{x}{q}\right)$, $\left(\frac{-y}{p}\right) = -\left(\frac{y}{p}\right)$, and $\left(\frac{-y}{q}\right) = -\left(\frac{y}{q}\right)$, by Proposition 6.4.6.

Thus

$$\begin{aligned}\left(\frac{-x}{n}\right) &= \left(\frac{-x}{p}\right)\left(\frac{-x}{q}\right) \\ &= \left(-\left(\frac{x}{p}\right)\right)\left(-\left(\frac{x}{q}\right)\right) \\ &= \left(\frac{x}{n}\right).\end{aligned}$$

In a similar manner, we obtain $\left(\frac{-y}{n}\right) = \left(\frac{y}{n}\right)$.

Now, $x^2 - y^2 = (x + y)(x - y) \equiv 0 \pmod{n}$, and so, $nm = pqm = (x + y)(x - y)$, for some m. Thus $p \mid (x + y)(x - y)$, and since p is prime, $p \mid (x + y)$ or $p \mid (x - y)$. Moreover, $q \mid (x + y)$ or $q \mid (x - y)$. We cannot have both p and q dividing the same factor, else n divides the same factor, which implies that the square roots $\pm x$ and $\pm y$ are not distinct. So we either have $p \mid (x + y)$ and $q \mid (x - y)$ or $p \mid (x - y)$ and $q \mid (x + y)$.

Without loss of generality, we assume the first case. Then $x \equiv -y \pmod{p}$, thus $x = -y$ in $U(\mathbb{Z}_p)$, and so, $\left(\frac{x}{p}\right) = \left(\frac{-y}{p}\right) = -\left(\frac{y}{p}\right)$. Moreover, $x = y$ in $U(\mathbb{Z}_q)$, and thus $\left(\frac{x}{q}\right) = \left(\frac{y}{q}\right)$. Thus $\left(\frac{x}{n}\right) \neq \left(\frac{y}{n}\right)$.

We conclude that either $\left(\frac{\pm x}{n}\right) = 1$ and $\left(\frac{\pm y}{n}\right) = -1$ or $\left(\frac{\pm x}{n}\right) = -1$ and $\left(\frac{\pm y}{n}\right) = 1$. Let us assume the first case. Then $\pm y$ are not quadratic residues modulo n, also

$\left(\frac{x}{n}\right) = \left(\frac{-x}{n}\right) = 1$. Now, $\left(\frac{x}{n}\right) = 1$ implies that $\left(\frac{x}{p}\right) = \left(\frac{x}{q}\right) = 1$ or that $\left(\frac{x}{p}\right) = \left(\frac{x}{q}\right) = -1$, and $\left(\frac{-x}{n}\right) = 1$ implies that $\left(\frac{-x}{p}\right) = \left(\frac{-x}{q}\right) = 1$ or that $\left(\frac{-x}{p}\right) = \left(\frac{-x}{q}\right) = -1$. Now if $\left(\frac{x}{p}\right) = 1$, then $\left(\frac{-x}{p}\right) = -1$, or vice versa. Consequently, exactly, one of $\pm x$, say x, is a quadratic residue modulo both p and q and hence n. This x is the only square root of a that is contained in QR_n. □

6.5 Exercises

1. Let $\langle \mathbb{Z}_{100}, +, \cdot \rangle$ denote the ring of residues modulo 100.

   (a) Compute $67 \cdot (3 + 72)$.
   (b) Show that $\mathbb{Z}_{100}$ is a commutative ring with unity.

2. Find an example of an integral domain that is not a field.
3. Determine whether 3 is a unit in the ring $\langle \mathbb{Q}, +, \cdot \rangle$.
4. Determine whether $A = \begin{pmatrix} 5 & 1 \\ 6 & 3 \end{pmatrix}$ is a unit in $\mathrm{Mat}_2(\mathbb{R})$. Is A a unit in $\mathrm{Mat}_2(\mathbb{Z})$?
5. Compute $\begin{pmatrix} 3 & 6 \\ 9 & 0 \end{pmatrix} \begin{pmatrix} 1 & 4 \\ 7 & 1 \end{pmatrix}$ in $\mathrm{Mat}_2(\mathbb{Z}_{11})$.
6. Let $\mathbb{Z}_{20}$ denote the ring of residues modulo 20.

   (a) Find the complete set of units in $\mathbb{Z}_{20}$.
   (b) Compute 7^{-1} in $\mathbb{Z}_{20}$.
   (c) Compute the group table for $U(\mathbb{Z}_{20})$.

7. Use Fermat's Little Theorem to compute $(20^{12} \bmod 11)$.
8. Let ϕ denote Euler's function. Compute $\phi(500)$.
9. Use Euler's theorem to compute $(43^{129} \bmod 256)$.
10. Let $n > 4$ be a composite number. Prove that $(n-1)! \equiv 0 \pmod{n}$.
11. Find a primitive root modulo 13, that is, find a generator for the cyclic group $U(\mathbb{Z}_{13})$.
12. Show that 5 is a primitive root modulo 23. Use this fact to find all of the primitive roots modulo 23.
13. Use the method of the algorithm EXP_MOD_n to compute $2^{28} \bmod 31$.
14. Show that 2 is a quadratic residue modulo 23.
15. Compute the Legendre symbol $\left(\frac{a}{11}\right)$ for $a = 2, 3, 4$.
16. Compute the Jacobi symbol $\left(\frac{a}{35}\right)$ for $a = 2, 10, 30$.
17. Find all four of the square roots of 4 modulo 35.
18. Let $n = 21$. Compute $J_{21}^{(1)}$, $J_{21}^{(-1)}$, and QR_{21}.
19. Prove Proposition 6.4.8.

Chapter 7
Advanced Topics in Algebra

7.1 Quotient Rings and Ring Homomorphisms

Throughout this section all rings are assumed to be commutative rings with unity. In this section we introduce ideals, quotient rings, and ring homomorphisms, which are advanced topics in ring theory needed in computer science and cryptography.

7.1.1 Quotient Rings

Let R be a commutative ring with unity. An **ideal** of R is a subgroup N of the additive group $\langle R, +\rangle$ for which $rN \subseteq N$ for all $r \in R$. Every ring has at least two ideals: the trivial ideal $\{0\}$ and R. For example, in the ring $\mathbb{Z}$, the subgroup $4\mathbb{Z}$ is an ideal since

$$n \cdot 4\mathbb{Z} = \{n \cdot 4m : m \in \mathbb{Z}\} = \{4nm : m \in \mathbb{Z}\} \subseteq 4\mathbb{Z}$$

for all $n \in \mathbb{Z}$.

Let a be any element of R. Then we can always form the ideal

$$\{ra : r \in R\}$$

of R-multiples of a; this is the **principal ideal** of R generated by a, denoted as (a). For example, (4) is the principal ideal of $\mathbb{Z}$ generated by 4; we have

$$(4) = \{m \cdot 4 : m \in \mathbb{Z}\} = 4\mathbb{Z}.$$

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_7

In the ring $\mathbb{Q}[x]$,

$$(x^2 - 2) = \{r(x)(x^2 - 2) : r(x) \in \mathbb{Q}[x]\}$$

is the principal ideal of $\mathbb{Q}[x]$ generated by $x^2 - 2$.

It is a consequence of Proposition 5.7.1 that every ideal of the ring of integers $\mathbb{Z}$ is principal. This is also true for the ring of polynomials over a field.

Proposition 7.1.1 *Let F be a field. Then every ideal of $F[x]$ is principal.*

Proof Let N be a non-zero ideal of $F[x]$ (clearly, the zero ideal is principal). Let $p(x)$ be a non-zero polynomial of minimal degree in N. Then every element of N is a multiple of $p(x)$, for if $f(x)$ is in N, then by the Division Theorem, there exist polynomials $q(x)$ and $r(x)$ in $F[x]$ for which

$$f(x) = p(x)q(x) + r(x),$$

where $\deg(r(x)) < \deg(p(x))$. Thus $r(x) = f(x) - p(x)q(x) \in I$, and so $r(x) = 0$. □

Let N be an ideal of R and let R/N denote the collection of all additive left cosets of N in R:

$$R/N = \{r + N : r \in R\}.$$

One can endow R/N with the structure of a ring by defining addition and multiplication on the left cosets. Addition on R/N is defined as

$$(a + N) + (b + N) = (a + b) + N, \tag{7.1}$$

and multiplication is given as

$$(a + N) \cdot (b + N) = ab + N \tag{7.2}$$

for left cosets $a + N, b + N \in R/N$.

Remark 7.1.2 The reader should verify that these operations are well-defined on left cosets, i.e., if $x+N = a+N$ and $y+N = b+N$, then $(x+N)+(y+N) = (a+b)+N$ and $(x + N) \cdot (y + N) = ab + N$; see Section 7.5, Exercise 1. □

The ring R/N endowed with the binary operations (7.1) and (7.2) is the **quotient ring of R by N**. The quotient ring R/N is a commutative ring with unity; the quotient ring R/N inherits these properties from R; the 0 element in R/N is the coset $0 + N = N$, and the unity element is the coset $1 + N$.

Example 7.1.3 Let $R = \mathbb{Z}$, $N = (4)$. Then the quotient ring $\mathbb{Z}/(4)$ consists of the cosets $\{(4), 1+(4), 2+(4), 3+(4)\}$, together with coset addition and multiplication. For instance, the sum of cosets $2 + (4)$, $3 + (4)$ is defined as

$$(2 + (4)) + (3 + (4)) = (2 + 3) + (4) = 5 + (4),$$

but note that the left coset $5 + (4)$ is equal to the left coset $1 + (4)$, thus

$$(2 + (4)) + (3 + (4)) = 1 + (4).$$

Likewise, the product of cosets is given as

$$(2 + (4)) \cdot (3 + (4)) = (2 \cdot 3) + (4) = 6 + (4) = 2 + (4).$$

The complete binary operation tables for $\mathbb{Z}/(4)$ are as follows:

$+$	(4)	$1 + (4)$	$2 + (4)$	$3 + (4)$
(4)	(4)	$1 + (4)$	$2 + (4)$	$3 + (4)$
$1 + (4)$	$1 + (4)$	$2 + (4)$	$3 + (4)$	(4)
$2 + (4)$	$2 + (4)$	$3 + (4)$	(4)	$1 + (4)$
$3 + (4)$	$3 + (4)$	(4)	$1 + (4)$	$2 + (4)$

$\cdot$	(4)	$1 + (4)$	$2 + (4)$	$3 + (4)$
(4)	(4)	(4)	(4)	(4)
$1 + (4)$	(4)	$1 + (4)$	$2 + (4)$	$3 + (4)$
$2 + (4)$	(4)	$2 + (4)$	(4)	$2 + (4)$
$3 + (4)$	(4)	$3 + (4)$	$2 + (4)$	$1 + (4)$

□

Example 7.1.4 Let $R = \mathbb{Q}[x]$ and let $N = (x^2 - 2)$ be the principal ideal of $\mathbb{Q}[x]$ generated by $x^2 - 2$. The elements of the quotient ring $\mathbb{Q}[x]/(x^2 - 2)$ consist of left cosets computed as follows. Let $f(x) \in \mathbb{Q}[x]$. By the Division Theorem, there exist polynomials $q(x)$ and $r(x)$ for which

$$f(x) = q(x)(x^2 - 2) + r(x),$$

with $\deg(r(x)) < \deg(x^2 - 2) = 2$. Thus $r(x) = a + bx$ for some $a, b \in \mathbb{Q}$. It follows that the elements of $\mathbb{Q}[x]/(x^2 - 2)$ are $\{a + bx + (x^2 - 2) : a, b \in \mathbb{Q}\}$. Note that addition in $\mathbb{Q}[x]/N$, $N = (x^2 - 2)$, is given as

$$(a + bx + N) + (c + dx + N) = a + c + (b + d)x + N,$$

while multiplication is given as

$$\begin{aligned}(a+bx+N)\cdot(c+dx+N) &= (a+bx)(c+dx)+N\\ &= ac+(ad+bc)x+bdx^2+N\\ &= ac+(ad+bc)x+2bd-2bd+bdx^2+N\\ &= ac+2bd+(ad+bc)x+bd(x^2-2)+N\\ &= ac+2bd+(ad+bc)x+N.\end{aligned}$$

□

Let N and M be ideals of R. Then the **sum of ideals** is an ideal of R defined as $N+M=\{a+b:\ a\in M, b\in N\}$.

Proposition 7.1.5 *Let R be a commutative ring with unity, let N be a proper ideal of R, and let $a\in R$. Then $a+N$ is a unit of R/N if and only if $(a)+N=R$.*

Proof Suppose $(a)+N=R$. Since $1\in R$, there exist elements $r\in R$ and $n\in N$ so that $ra+n=1$, and hence $ra=1-n$. Now

$$\begin{aligned}(r+N)(a+N) &= ra+N\\ &= (1-n)+N\\ &= (1+N)+(-n+N)\\ &= (1+N)+N\\ &= 1+N,\end{aligned}$$

and thus $r+N=(a+N)^{-1}$.

Conversely, suppose $a+N$ is a unit of R/N. Then $1+N=ar+N$ for some $r\in R$, and so, $1\in ar+N$. Thus $R\subseteq(a)+N$. Since $(a)+N\subseteq R$, one has $(a)+N=R$. □

A **maximal ideal** is a proper ideal M of R for which there is no proper ideal N with $M\subset N\subset R$. For instance, (3) is a maximal ideal of $\mathbb{Z}$, and (4) is not a maximal ideal of $\mathbb{Z}$ since $(4)\subset(2)\subset\mathbb{Z}$.

Proposition 7.1.6 *Let R be a commutative ring with unity. Then M is a maximal ideal of R if and only if R/M is a field.*

Proof Suppose that R/M is a field and let N be a proper ideal of R with $M\subseteq N\subset R$. If $M\neq N$, then there exists an element $a\in N\backslash M$, and hence $a+M$ is a non-zero element of the field R/M. Consequently, $a+M$ is a unit in R/M, and so, by Proposition 7.1.5, $R=(a)+M\subseteq N$. Thus $N=R$, which is a contradiction.

For the converse, we suppose that M is maximal. Since R is a commutative ring with unity, so is R/M. So it remains to show that every non-zero element of R/M is a unit. Let $a + M \in R/M$, $a \notin M$. Then $(a) + M$ is an ideal of R with $M \subseteq (a)+M \subseteq R$. But M is maximal, so either $(a)+M = M$ or $(a)+M = R$. In the former case, $a \in M$, which is a contradiction. Thus $(a) + M = R$, which says that $a + M$ is a unit in R/M. □

We can use Proposition 7.1.6 to “invent" zeros of polynomials. Let F be a field and let $F[x]$ denote the ring of polynomials over F. Then $F[x]$ is an integral domain (and hence a commutative ring with unity). The units of $F[x]$ consist of all degree 0 polynomials, i.e., polynomials of the form $f(x) = c$, $c \neq 0$.

A non-zero non-unit polynomial $p(x) \in F[x]$ is **irreducible** over F if the factorization

$$p(x) = f(x)g(x)$$

in $F[x]$ implies that either $f(x)$ or $g(x)$ is a unit in $F[x]$. The non-zero non-unit polynomial $p(x)$ is **reducible** if it is not irreducible.

For example, x^2-2 is an irreducible polynomial in $\mathbb{Q}[x]$, while x^2-1 is reducible over $\mathbb{Q}$ since

$$x^2 - 1 = (x + 1)(x - 1),$$

where neither $x + 1$ nor $x - 1$ is a unit in $\mathbb{Q}[x]$.

Proposition 7.1.7 *Let F be any field and let $q(x)$ be an irreducible polynomial in $F[x]$. Then the principal ideal $(q(x))$ is maximal.*

Proof Suppose there exists a proper ideal N of $F[x]$ for which $(q(x)) \subset N \subset F[x]$. By Proposition 7.1.1, $N = (f(x))$ for some $f(x) \in F[x]$. Thus $q(x) = r(x)f(x)$ where neither $r(x)$ nor $f(x)$ is a unit of $F[x]$, and thus $q(x)$ is reducible. □

Proposition 7.1.8 *There exists a field extension E/F that contains a zero of $q(x)$.*

Proof By Proposition 7.1.7, $(q(x))$ is a maximal ideal of $F[x]$, and by Proposition 7.1.6, $E = F[x]/(q(x))$ is a field. We identify F with the collection of cosets $\{r + (q(x)) : r \in F\}$, and in this way, $F \subseteq E$. Thus E is a field extension of F.

Now let $\alpha = x + (q(x)) \in E$. Write

$$q(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_2x^2 + a_1x + a_0$$

for $a_i \in F$, $1 \leq i \leq n$, $a_n \neq 0$. Then

$$\begin{aligned} q(\alpha) &= q(x + (q(x))) \\ &= a_n(x + (q(x)))^n + a_{n-1}(x + (q(x)))^{n-1} + \cdots + a_2(x + (q(x)))^2 \\ &\quad + a_1(x + (q(x))) + a_0 \end{aligned}$$

$$\begin{aligned}&= a_n(x^n + (q(x))) + a_{n-1}(x^{n-1} + (q(x))) + \cdots + a_2(x^2 + (q(x)))\\ &\qquad + a_1(x + (q(x))) + a_0\\ &= q(x) + (q(x))\\ &= (q(x))\end{aligned}$$

and so $q(\alpha) = 0$ in E. □

We call the coset α an **invented root** of $q(x)$.

Proposition 7.1.9 *Let $f(x) \in F[x]$ with $\deg(f(x)) = d \geq 1$. Then there exists a field extension E/F so that $f(x)$ factors into a product of linear factors in $E[x]$.*

Proof We prove this by induction on $d = \deg(f(x))$.

The Trivial Case: $d = 1$ In this case $f(x) = ax + b \in F[x]$, so we may take $E = F$.

The Induction Step We assume that the proposition is true for polynomials of degree $d - 1$. The polynomial $f(x)$ factors into a product of irreducible polynomials

$$f(x) = q_1(x)q_2(x)q_3(x)\cdots q_k(x).$$

If $\deg(q_i(x)) = 1$ for $i = 1, 2, \dots, k$, then we can take $E = F$. Else, let j be the smallest index with $\deg(q_j(x)) > 1$. By Proposition 7.1.8, there exists an invented root α of $q_j(x)$ in some field extension L/F. Over L, $f(x)$ factors as

$$f(x) = q_1(x)q_2(x)\cdots q_{j-1}(x)(x - \alpha)r(x)q_{j+1}\cdots q_k(x),$$

for $r(x) \in L[x]$.

Put

$$g(x) = q_1(x)q_2(x)\cdots q_{j-1}(x)r(x)q_{j+1}\cdots q_k(x).$$

Then $\deg(g(x)) = d - 1$. By the induction hypothesis, there exists a field extension E/L so that $g(x)$ factors into a product of linear factors in $E[x]$. Since $f(x) = g(x)(x - \alpha)$, $f(x)$ factors into a product of linear factors in $E[x]$.

□

Proposition 7.1.9 says that E/F contains d roots of $f(x)$ which may or may not be distinct. These are the only possible zeros of $f(x)$ since a degree d polynomial over a field can have at most d roots in the field (Proposition 6.1.7).

7.1.2 Ring Homomorphisms

Definition 7.1.10 Let R and R' be commutative rings with unity elements, $1_R \in R$, $1_{R'} \in R'$. A function $f : R \to R'$ is a **ring homomorphism** if

(i) $f(1_R) = 1_{R'}$
(ii) $f(a+b) = f(a) + f(b)$
(iii) $f(ab) = f(a)f(b)$

for all $a, b \in R$.

For example, let $n \geq 1$ be an integer and let (n) be the principal ideal of $\mathbb{Z}$ generated by n. The map $f : \mathbb{Z} \to \mathbb{Z}/(n)$ defined as $f(a) = a + (n)$ is a homomorphism of commutative rings with unity. To see this, we show that the conditions of Definition 7.1.10 hold. Indeed, $f(1) = 1 + (n)$, so (i) holds. For (ii), $f(a+b) = a + b + (n) = a + (n) + b + (n) = f(a) + f(b)$. For (iii), $f(ab) = ab + (n) = (a + (n)) \cdot (b + (n)) = f(a)f(b)$.

Here is another example of a ring homomorphism.

Proposition 7.1.11 *Let E/F be a field extension and let $\alpha \in E$. Then the function $f_\alpha : F[x] \to E$ defined by $f_\alpha(p(x)) = p(\alpha)$ is a ring homomorphism.*

Proof Let $p(x) = \sum_{i=0}^m a_i x^i$ and $q(x) = \sum_{j=0}^n b_j x^j$ be polynomials in $F[x]$. Then

$$\begin{aligned} f_\alpha(p(x) + q(x)) &= f_\alpha\Big(\sum_{i=0}^m a_i x^i + \sum_{j=0}^n b_j x^j\Big) \\ &= \sum_{i=0}^m a_i \alpha^i + \sum_{j=0}^n b_j \alpha^j \\ &= f_\alpha\Big(\sum_{i=0}^m a_i x^i\Big) + f_\alpha\Big(\sum_{j=0}^n b_j x^j\Big) \end{aligned}$$

and

$$\begin{aligned} f_\alpha(p(x)q(x)) &= f_\alpha\Big(\sum_{i=0}^m \sum_{j=0}^n a_i b_j x^{i+j}\Big) \\ &= \sum_{i=0}^m \sum_{j=0}^n a_i b_j \alpha^{i+j} \\ &= f_\alpha\Big(\sum_{i=0}^m a_i x^i\Big) f_\alpha\Big(\sum_{j=0}^n b_j x^j\Big). \end{aligned}$$

Moreover, $f_\alpha(1_{F[x]}) = 1_F = 1_E$, and so f_α is a homomorphism of rings with unity. □

The homomorphism of Proposition 7.1.11 is called the **evaluation homomorphism**. For an example, let $F = \mathbb{Q}$, $E = \mathbb{R}$, and $\alpha = \sqrt{2}$. Then the evaluation homomorphism $\phi_{\sqrt{2}} : \mathbb{Q}[x] \to \mathbb{R}$ is given by $p(x) \mapsto p(\sqrt{2})$.

Definition 7.1.12 The **kernel** of the ring homomorphism $f : R \to R'$ is the subset of R defined as $\ker(f) = \{a \in R : f(a) = 0\}$.

Proposition 7.1.13 *The kernel of a ring homomorphism $f : R \to R'$ is an ideal of R.*

Proof Let $N = \ker(f)$. Then N is an additive subgroup of R (proof?), so we need to only show that $aN \subseteq N$ for all $a \in R$. But this condition follows since $\phi(an) = \phi(a)\phi(n) = 0$ for $an \in aN$. □

Definition 7.1.14 A ring homomorphism $f : R \to R'$ is an **isomorphism** of rings if f is a bijection. The rings R and R' are **isomorphic** if there exists an isomorphism $f : R \to R'$. We then write $R \cong R'$.

Theorem 7.1.15 *Let $f : R \to R'$ be a homomorphism of rings. Let ker$(f) = N$. Then $g : R/N \to f(R)$ defined by $g(a + N) = f(a)$ is an isomorphism of rings.*

Proof One first checks that $f(R)$ is a ring. We then check that g is well-defined on cosets. Let $a + N = b + N$. Then $a = b + n$ for some $n \in N$. Hence $f(a) = f(b+n) = f(b)+f(n) = f(b)$, and so $g(a+N) = g(b+N)$. Thus g is a function on cosets.

Now, $g(1+N) = f(1_R) = 1_{R'}$, $g(a+N+b+N) = g(a+b+N) = f(a+b) = f(a) + f(b) = g(a + N) + g(b + N)$, and

$$\begin{aligned} g((a + N)(b + N)) &= g(ab + N) \\ &= f(ab) \\ &= f(a)f(b) \\ &= g(a + N)g(b + N). \end{aligned}$$

□

For example, the surjective (onto) ring homomorphism $f : \mathbb{Z} \to \mathbb{Z}_n$, $a \mapsto (a \bmod n)$ induces the ring isomorphism $g : \mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}_n$, $a + n\mathbb{Z} \mapsto (a \bmod n)$.

Here is an important example of a ring isomorphism.

Proposition 7.1.16 *Let $n_1, n_2 > 0$ be integers with* $\gcd(n_1, n_2) = 1$. *Then there is an isomorphism of rings $\mathbb{Z}_{n_1n_2} \to \mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2}$.*

Proof Define a map $\psi : \mathbb{Z} \to \mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2}$ by the rule

$$\psi(a) = ((a \bmod n_1), (a \bmod n_2)).$$

Then ψ is a ring homomorphism since, for $a, b \in \mathbb{Z}$,

$$\begin{aligned}\psi(a+b) &= (((a+b) \bmod n_1), ((a+b) \bmod n_2))\\ &= ((a \bmod n_1) + (b \bmod n_1), (a \bmod n_2) + (b \bmod n_2))\\ &= ((a \bmod n_1), (a \bmod n_2)) + ((b \bmod n_1), (b \bmod n_2))\\ &= \psi(a) + \psi(b).\end{aligned}$$

Moreover,

$$\psi(ab) = \psi(a)\psi(b).$$

Now, $\ker(\psi)$ is a cyclic subgroup of $\mathbb{Z}$ generated by a common multiple of n_1, n_2, which in fact is the least common multiple $\text{lcm}(n_1, n_2)$. Since $\gcd(n_1, n_2) = 1$, $\text{lcm}(n_1, n_2) = n_1 n_2$ by Proposition 5.7.9.

By Theorem 7.1.15, there is an injective ring homomorphism

$$g : \mathbb{Z}/n_1n_2\mathbb{Z} \to \mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2}$$

defined as $g(a + n_1n_2\mathbb{Z}) = ((a \bmod n_1), (a \bmod n_2))$.

Observe that $\mathbb{Z}/n_1n_2\mathbb{Z} \cong \mathbb{Z}_{n_1n_2}$ and so we can write this injective homomorphism as $g : \mathbb{Z}_{n_1n_2} \to \mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2}$.

Since $\mathbb{Z}_{n_1n_2}$ and $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2}$ have the same number of elements, g is a ring isomorphism. □

Proposition 7.1.17 *Let R be a commutative ring with unity, 1_R. Then the map $\varrho : \mathbb{Z} \to R$ defined as $\varrho(n) = n1_R = \underbrace{1_R + 1_R + \cdots + 1_R}_{n}$ is a homomorphism of commutative rings with unity.*

Proof For $m, n \in \mathbb{Z}$, $\varrho(m+n) = (m+n)1_R = m1_R + n1_R = \varrho(m) + \varrho(n)$, and $\varrho(mn) = mn1_R = m(\varrho(n)) = m(1_R\varrho(n)) = (m1_R)\varrho(n) = \varrho(m)\varrho(n)$. Also, $\varrho(1_\mathbb{Z}) = 1_Z 1_R = 1_R$, and so, ϱ is a homomorphism of commutative rings with unity. □

The kernel of $\varrho : \mathbb{Z} \to R$ is an ideal of $\mathbb{Z}$ of the form $r\mathbb{Z}$ for some integer $r \geq 0$. The integer r is the **characteristic** of the ring R and is denoted as $\text{char}(R)$.

Corollary 7.1.18 *Let R be a ring with unity with $r = char(R)$. If $r = 0$, then R contains a (sub)ring isomorphic to $\mathbb{Z}$. If $r > 0$, then R contains a subring isomorphic to $\mathbb{Z}_r$.*

Proof Note that $\varrho(\mathbb{Z})$ is a subring of R. If $r = \text{char}(R) = 0$, then $\mathbb{Z} \cong \mathbb{Z}/\{0\}$ is isomorphic to $\varrho(\mathbb{Z})$ by Theorem 7.1.15. If $r = \text{char}(R) > 0$, then $\mathbb{Z}_r \cong \mathbb{Z}/r\mathbb{Z}$ is isomorphic to $\varrho(R)$ (again by Theorem 7.1.15). □

7.2 Simple Algebraic Extensions

Theorem 7.1.15 can be applied in the following important way.

Let F be a field contained in some larger extension field E (perhaps $F = \mathbb{Q}$ and $E = \mathbb{C}$). Let $F[x]$ be the ring of polynomials over F and let α be an element of E that is a zero of some polynomial $g(x)$ in $F[x]$ with $\deg(g(x)) \geq 1$.

Let $f_\alpha : F[x] \to E$ be the evaluation homomorphism, $q(x) \mapsto q(\alpha) \in E$. The kernel of f_α is the ideal N of $F[x]$ consisting of all polynomials in $F[x]$ for which α is a zero. By Proposition 7.1.1, N is a principal ideal of the form $N = (p(x))$ for some $p(x) \in F[x]$. In fact, $p(x)$ is the monic irreducible polynomial of smallest degree with $p(\alpha) = 0$. Since $p(x)$ is irreducible, $(p(x))$ is a maximal ideal of $F[x]$.

By Proposition 7.1.6, $F[x]/(p(x))$ is a field, and by Theorem 7.1.15 there exists an isomorphism

$$g : F[x]/(p(x)) \to f_\alpha(F[x]),$$

defined as $h(x) + (p(x)) \mapsto f_\alpha(h(x)) = h(\alpha)$, for $h(x) \in F[x]$.

Thus the image $f_\alpha(F[x])$ is a field. In the case that $h(x) = c$ for $c \in F$, we have $g(c + (p(x)) = c$, $c \in F$, and so, F is a subfield of $f_\alpha(F[x])$, and $f_\alpha(F[x])$ is a field extension of F.

Definition 7.2.1 The field $f_\alpha(F[x])$, denoted as $F(\alpha)$, is the **simple algebraic extension** of F by α. The **degree** of $F(\alpha)$ is the degree of the irreducible polynomial $p(x)$.

We want to give an explicit description of the field $F(\alpha)$.

Proposition 7.2.2 *The simple algebraic field extension $F(\alpha)$ of F is a vector space over F of dimension equal to the degree of $F(\alpha)$. An F-basis for $F(\alpha)$ is $\{1, \alpha, \alpha^2, \ldots, \alpha^{n-1}\}$.*

Proof Let $n = \deg(p(x))$. Let $h(x) \in F[x]$. By Proposition 6.1.5, we have

$$h(x) = p(x)q(x) + r(x)$$

for $q(x)$ and $r(x)$ with $\deg(r(x)) < p(x)$. Write

$$r(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{n-1}x^{n-1},$$

$a_i \in F$. Then the coset $h(x) + (p(x))$ can be written as the coset $r(x) + (p(x))$. Thus, every element of $F(\alpha)$ can be written in the form

$$r(\alpha) = a_0 + a_1\alpha + a_2\alpha^2 + \cdots + a_{n-1}\alpha^{n-1},$$

for $a_i \in F$. This says that the F-span of the vectors

$$B = \{1, \alpha, \alpha^2, \dots, \alpha^{n-1}\}$$

is $F(\alpha)$.

Now, the set B is linearly independent. For if not, then there exist an integer m, $1 \le m \le n-1$ and elements a_i, $0 \le i \le m-1$, not all zero, for which

$$\alpha^m = a_{m-1}\alpha^{m-1} + a_{m-2}\alpha^{m-2} + \cdots + a_1\alpha + a_0.$$

But this says that the kernel of f_α contains a non-zero polynomial of degree $< n$, a contradiction.

Thus B is a linearly independent spanning set for $F(\alpha)$ and consequently is an F-basis for $F(\alpha)$. □

Example 7.2.3 We take $F = \mathbb{Q}$, $E = \mathbb{R}$, and $\alpha = \sqrt{2}$; $\sqrt{2}$ is a zero of the polynomial $x^2 - 2$ over $\mathbb{Q}$. We have the evaluation homomorphism

$$f_{\sqrt{2}} : \mathbb{Q}[x] \to \mathbb{R},$$

whose kernel is $(x^2 - 2)$. Thus,

$$\mathbb{Q}[x]/(x^2 - 2) \cong f_\alpha(\mathbb{Q}[x]) = \mathbb{Q}(\sqrt{2}).$$

A “power” basis for the simple algebraic field extension $\mathbb{Q}(\sqrt{2})$ is $\{1, \sqrt{2}\}$; the degree of $\mathbb{Q}(\sqrt{})$ over $\mathbb{Q}$ is 2. We have

$$\mathbb{Q}(\sqrt{2}) = \{a + b\sqrt{2} : \ a, b \in \mathbb{Q}\}.$$

□

Example 7.2.4 We take $F = \mathbb{R}$, $E = \mathbb{C}$, and $\alpha = i = \sqrt{-1}$; i is a zero of the polynomial $x^2 + 1$ over $\mathbb{R}$. We have the evaluation homomorphism

$$f_i : \mathbb{R}[x] \to \mathbb{C},$$

whose kernel is $(x^2 + 1)$. Thus,

$$\mathbb{R}[x]/(x^2 + 1) \cong f_\alpha(\mathbb{R}[x]) = \mathbb{R}(i).$$

A power basis for the simple algebraic field extension $\mathbb{R}(i)$ is $\{1, i\}$; the degree of $\mathbb{R}(i)$ over $\mathbb{R}$ is 2. We have

$$\mathbb{R}(i) = \{a + bi : \ a, b \in \mathbb{R}\},$$

which is well-known to be the field of complex numbers $\mathbb{C}$. □

7.2.1 Algebraic Closure

Let E be an extension field of F. An element $\alpha \in E$ is **algebraic** over F if α is a zero of a polynomial $g(x) \in F[x]$. The extension E/F is an **algebraic extension** of F if every element of E is algebraic over F. For instance, $\mathbb{Q}(\sqrt{2})$ is an algebraic extension of $\mathbb{Q}$ (can you prove this?)

A field F is **algebraically closed** if every polynomial $g(x) \in F[x]$ has a zero in F. One familiar example of an algebraically closed field is $\mathbb{C}$.

An **algebraic closure** of F is an algebraic extension of F that is algebraically closed. Every field F has (essentially) a unique algebraic closure which we denote as $\overline{F}$; it is the largest algebraic extension of F. For example, if $F = \mathbb{R}$, then $\overline{\mathbb{R}} = \mathbb{C}$.

7.3 Finite Fields

A **finite field** is a field with a finite number of elements. If p is a prime number, then there exists a field with exactly p elements.

Proposition 7.3.1 *Let p be prime. Then $\mathbb{Z}_p$ is a field.*

Proof Certainly, $\mathbb{Z}_p$ is a commutative ring with unity, 1. By Proposition 6.2.3, $\mathbb{Z}_p$ is a division ring. □

It turns out that the number of elements in any finite field is always a power of a prime number.

Proposition 7.3.2 *Let F be a field with a finite number of elements. Then $|F| = p^n$, where p is a prime number and $n \geq 1$ is an integer.*

Proof Since F is finite, $r = \text{char}(F) > 0$, and hence by Corollary 7.1.18, F contains a subring B isomorphic to $\mathbb{Z}_r$. Henceforth, we identify B with $\mathbb{Z}_r$. Since F is a field, r must be a prime number p, and hence F contains the field $\mathbb{Z}_p$. As F is finite, it is certainly a finite-dimensional vector space over $\mathbb{Z}_p$, with scalar multiplication $\mathbb{Z}_p \times F \to F$ given by multiplication in F. Thus $F = \underbrace{\mathbb{Z}_p \oplus \mathbb{Z}_p \oplus \cdots \oplus \mathbb{Z}_p}_{n}$, where $n = \dim(F)$, and hence $|F| = p^n$. □

In Theorem 6.3.1, we showed that $U(\mathbb{Z}_p)$ is cyclic. This can be extended to finite fields.

Proposition 7.3.3 *Let F be a field with a finite number of elements. Then the multiplicative group of units of F, $U(F) = F^{\times}$, is cyclic.*

Proof Since F is a commutative ring with unity, $U(F)$ is a finite abelian group whose identity element is 1. Let f be the exponent of $U(F)$. We have $f \leq |U(F)| = p^n - 1$, for some $n \geq 1$. Consider the polynomial $x^f - 1$ in $F[x]$. By the definition of exponent, $g^f = 1$ for all $g \in U(F)$ and so $x^f - 1$ has $p^n - 1$ distinct zeros in F. Thus $f = p^n - 1$ by Proposition 6.1.7.

By Proposition 5.7.13, there exists an element $g \in U(F)$ which has order $p^n - 1$. But then $|\langle g \rangle| = p^n - 1$ so that $\langle g \rangle = U(F)$. Thus $U(F)$ is cyclic. □

Proposition 7.3.4 *Let F be a field with a finite number of elements. Then F is isomorphic to a simple algebraic extension of $\mathbb{Z}_p$ for some prime number p, and $|F| = p^n$, where n is the degree of the simple algebraic extension of $\mathbb{Z}_p$.*

Proof By Proposition 7.3.2, $|F| = p^n$ for some prime p and integer $n \geq 1$; F contains the subfield $\mathbb{Z}_p$. By Proposition 7.3.3, the group of units of F, $F^\times$, is cyclic of order $p^n - 1$ generated by α.

Let $f_\alpha : \mathbb{Z}_p[x] \to F$ denote the evaluation homomorphism with $\ker(f_\alpha) = (q(x))$, where $q(x)$ is monic, irreducible of degree m in $\mathbb{Z}_p[x]$, with root α. Then $\mathbb{Z}_p[x]/(q(x)) \cong \mathbb{Z}_p(\alpha)$ is a simple algebraic extension of $\mathbb{Z}_p$ of degree m over $\mathbb{Z}_p$.

Let $g : \mathbb{Z}_p[x]/(q(x)) \to F$ be defined by $r(x) + (q(x)) \mapsto r(\alpha)$. Since $r(x) - s(x) \in (q(x))$ implies that $r(\alpha) = s(\alpha)$, g is well-defined on cosets. Clearly, g is onto, and since $\mathbb{Z}_p[x]/(q(x))$ is a field, g is 1–1. Hence g is an isomorphism of fields. The set $\{1, \overline{x}, \overline{x}^2, \ldots, \overline{x}^{m-1}\}$ is a $\mathbb{Z}_p$-basis for $\mathbb{Z}_p[x]/(q(x))$ and so $|F| = p^m$. But we already know that $|F| = p^n$, and hence $m = n$. □

Let p be a prime and let $n \geq 1$. In what follows, we address the *existence* of finite fields of order p^n; we show how to explicitly construct a finite field with exactly p^n elements.

In the case $n = 1$, there exists a field with exactly $p^1 = p$ elements, namely, $\mathbb{Z}_p$. For $n \geq 1$, let

$$f(x) = x^{p^n} - x \in \mathbb{Z}_p[x].$$

By Proposition 7.1.9, there exists a field extension $E/\mathbb{Z}_p$ that contains all p^n zeros of $f(x)$ (counting multiplicities).

Proposition 7.3.5 *The zeros of $f(x) = x^{p^n} - x$ in E are distinct.*

Proof Let $F = \{\alpha_i\}$, $1 \leq i \leq p^n$, be the set of roots of $f(x)$, and suppose that some root α_i has multiplicity ≥ 2. Then $f'(\alpha_i) = 0$. But this is impossible since the formal derivative $f'(x) = -1$ in $\mathbb{Z}_p[x]$. □

Proposition 7.3.6 *Let $F = \{\alpha_i\}$, $1 \leq i \leq p^n$, be the set of roots of $f(x) = x^{p^n} - x$. Then F is a field, with operations induced from E.*

Proof By Corollary 6.2.7, the elements of $\mathbb{Z}_p$ are roots of $f(x)$. Thus $\mathbb{Z}_p \subseteq F$ and $\text{char}(F) = p$. Let $\alpha_i, \alpha_j \in F$. Then $(\alpha_i + \alpha_j)^{p^n} = \alpha_i^{p^n} + \alpha_j^{p^n} = \alpha_i + \alpha_j$ (use the binomial theorem and Corollary 6.2.7). Thus F is closed under addition. Moreover, $(-\alpha_i)^{p^n} = (-1)^{p^n}\alpha_i^{p^n} = -\alpha_i$, since $(-1)^{p^n} \equiv -1 \bmod p$ by Corollary 6.2.7. Hence F is an additive subgroup of E. Also, $(\alpha_i\alpha_j)^{p^n} = \alpha_i^{p^n}\alpha_j^{p^n} = \alpha_i\alpha_j$, so that F

is closed under multiplication. Thus F is a commutative ring with unity. For $\alpha_i \in F$ non-zero, $\alpha_i^{-1} \in E$. But also, $(\alpha_i^{-1})^{p^n} = \alpha_i^{-1}$. Thus F is a field. □

By Proposition 7.3.6, there is a field F of order p^n, consisting of the p^n distinct roots of $x^{p^n} - x$. The following proposition shows that there is essentially only one finite field of order p^n.

Proposition 7.3.7 *Let F_1 and F_2 be finite fields of order p^n. Then $F_1 \cong F_2$.*

Proof We show that $F_1 \cong F$, where F is the field constructed in Proposition 7.3.6 consisting of the roots of $x^{p^n} - x \in \mathbb{Z}_p[x]$. By Proposition 7.3.4, F_1 is isomorphic to a simple algebraic extension of $\mathbb{Z}_p$, $\mathbb{Z}_p[x]/(q(x)) \cong \mathbb{Z}_p(\alpha)$, where $\deg(q(x)) = n$, and α is a root of both $q(x)$ and $x(x^{p^n-1} - 1) = x^{p^n} - x$. Hence $\alpha \in F$. Define $f : \mathbb{Z}_p[x]/(q(x)) \to F$ by the rule $r(x) + (q(x)) \mapsto r(\alpha)$. Then f is well-defined on cosets (f is a function). Moreover, f is 1–1 since $\mathbb{Z}_p[x]/(q(x))$ is a field and onto since both $\mathbb{Z}_p[x]/(q(x))$ and F have the same number of elements. Thus f is an isomorphism. It follows that $F_1 \cong F$. Similarly, $F_2 \cong F$, which proves the result. □

The unique (up to isomorphism) finite field of order p^n is called the **Galois field** of order p^n and is denoted by $\mathrm{GF}(p^n)$, or more simply, by $\mathbb{F}_{p^n}$. For a prime number p, $\mathrm{GF}(p) = \mathbb{F}_p = \mathbb{Z}_p$.

We next discuss polynomials over the Galois field $\mathbb{F}_{p^n}$.

Proposition 7.3.8 *Let $f(x) \in \mathbb{F}_{p^n}[x]$ with $\deg(f(x)) = k \geq 1$. Assume that $f(0) \neq 0$. Then there exists an integer t, $1 \leq t \leq p^{nk} - 1$, for which $f(x) \mid x^t - 1$.*

Proof First note that the quotient ring $\mathbb{F}_{p^n}[x]/(f(x))$ contains $p^{nk} - 1$ left cosets other than $(f(x))$. The collection $\{x^i + (f(x)) : i = 0, 1, 2, \ldots, p^{nk} - 1\}$ is a set of p^{nk} left cosets not containing $(f(x))$. Thus

$$x^i + (f(x)) = x^j + (f(x)),$$

for some i, j, $0 \leq i < j \leq p^{nk} - 1$. Since $f(0) \neq 0$, there exist polynomials $r(x), s(x) \in \mathbb{F}_{p^n}[x]$ so that

$$xr(x) + f(x)s(x) = 1.$$

Consequently, $x^i + (f(x))$ is a unit in $\mathbb{F}_{p^n}[x]/(f(x))$. Set $t = j - i$. It follows that $x^t \in 1 + (f(x))$, and hence $f(x) \mid x^t - 1$ with $1 \leq t \leq p^{nk} - 1$. □

The smallest positive integer e for which $f(x) \mid (x^e - 1)$ is the **order of** $f(x)$, denoted as $\mathrm{ord}(f(x))$. For example, over $\mathbb{F}_2$, $\mathrm{ord}(x^4 + x + 1) = 15$ and $\mathrm{ord}(x^4 + x^3 + x^2 + x + 1) = 5$. Over $\mathbb{F}_9 = \mathbb{F}_3(\alpha)$, $\alpha^2 + 1 = 0$, $\mathrm{ord}(x - \alpha) = 4$ since

$$x^4 - 1 = (x - 1)(x + 1)(x - \alpha)(x + \alpha)$$

over $\mathbb{F}_9$.

Proposition 7.3.9 *Let $f(x)$ be an irreducible polynomial in $\mathbb{F}_{p^n}[x]$ of degree k. Then $f(x)$ divides $x^{p^{nm}} - x$ if and only if $k \mid m$.*

Proof Suppose that $f(x)$ divides $x^{p^{nm}} - x$. By Proposition 7.1.8, there exists a field extension $E/\mathbb{F}_{p^n}$ that contains a zero α of $f(x)$. Thus $\alpha^{p^{nm}} - \alpha = 0$, and so, $\alpha \in \mathbb{F}_{p^{nm}}$. Moreover, the elements of $\mathbb{F}_{p^n}$ are precisely the zeros of $x^{p^n} - x$ and any zero of $x^{p^n} - x$ is also a zero of $x^{p^{nm}} - x$. Thus, $F = \mathbb{F}_{p^n}(\alpha)$ is a subfield of $\mathbb{F}_{p^{nm}}$. Note that F has p^{nk} elements.

Now let β be a generator of the cyclic group $\mathbb{F}_{p^{nm}}^{\times}$ and let $q(x)$ be the irreducible polynomial of β over F. Let $t = \deg(q(x))$. Then $F(\beta) = \mathbb{F}_{p^{nm}}$. Now $F(\beta)$ has p^{nkt} elements and so $p^{nkt} = p^{nm}$. Hence $kt = m$, that is, $k \mid m$.

For the converse, suppose that $k \mid m$. Let α be a zero of $f(x)$ in some extension field $E/\mathbb{F}_{p^n}$. Then $\mathbb{F}_{p^n}(\alpha)$ is a field with p^{nk} elements, and α satisfies the relation $\alpha^{p^{nk}} - \alpha = 0$. Let s be so that $ks = m$. Then $\alpha^{p^{nks}} - \alpha = 0$, and hence, $\alpha^{p^{nm}} - \alpha = 0$. Consequently, α is a root of $x^{p^{nm}} - x$. It follows that $x^{p^{nm}} - x$ is in $(f(x))$, the kernel of the evaluation homomorphism $\phi_\alpha : \mathbb{F}_{p^n}[x] \to E$, and so $f(x) \mid x^{p^{nm}} - x$. □

Proposition 7.3.10 *Let $f(x)$ be an irreducible polynomial in $\mathbb{F}_{p^n}[x]$ of degree k. Let α be a zero of $f(x)$ in an extension field $E/\mathbb{F}_{p^n}$. Then the zeros of $f(x)$ are of the form $\alpha, \alpha^{p^n}, \alpha^{p^{2n}}, \dots, \alpha^{p^{(k-1)n}}$.*

Proof We already know that $\alpha \in E$ is one zero of $f(x)$. Since the characteristic of $\mathbb{F}_{p^n}$ is p,

$$f(\alpha^{p^{jn}}) = f(\alpha)^{p^{jn}} = 0,$$

for $1 \le j \le k - 1$. □

Proposition 7.3.11 *Let $f(x)$ be an irreducible polynomial in $\mathbb{F}_{p^n}[x]$ of degree k. Let α be a zero of $f(x)$ in an extension field $E/\mathbb{F}_{p^n}$. The smallest field extension containing all of the zeros of $f(x)$ is $\mathbb{F}_{p^n}(\alpha)$, which is isomorphic to the Galois field $\mathbb{F}_{p^{nk}}$.*

Proof By Proposition 7.3.10, $\mathbb{F}_{p^n}(\alpha)$ is the smallest field containing all of the zeros of $f(x)$. We have $|\mathbb{F}_{p^n}(\alpha)| = p^{nk}$, and thus by Proposition 7.3.7, $\mathbb{F}_{p^n}(\alpha) \cong \mathbb{F}_{p^{nk}}$. □

Let $f(x)$ be an irreducible polynomial in $\mathbb{F}_{p^n}[x]$ of degree $k \ge 1$, and let α be a zero of $f(x)$. As we have seen in Proposition 7.3.11, $\mathbb{F}_{p^n}(\alpha)$ contains all of the roots of $f(x)$. The following proposition computes $\mathrm{ord}(f(x))$.

Proposition 7.3.12 *Let $f(x)$ be an irreducible polynomial in $\mathbb{F}_{p^n}[x]$ of degree $k \ge 1$, and let α be a zero of $f(x)$ in an extension field E. Then* $\mathrm{ord}(f(x))$ *equals the order of any root of $f(x)$ in the group of units of $\mathbb{F}_{p^n}(\alpha)$.*

Proof By Proposition 7.3.3, $\mathbb{F}_{p^n}(\alpha)^\times$ is cyclic of order $p^{nk}-1$, generated by some element β. Put $\alpha = \beta^l$ for some integer l. Now a typical zero of $f(x)$ can be written $\beta^{lp^{mn}}$ for $0 \le m \le k-1$. By Proposition 5.7.10,

$$|\langle \beta^{lp^{mn}} \rangle| = \frac{p^{nk}-1}{\gcd(p^{nk}-1, lp^{mn})}.$$

Since $\gcd(p^{nk}-1, p^{mn}) = 1$, the right-hand side above only depends on l, and so each zero of $f(x)$ has the same order.

Let e be the order of α in $\mathbb{F}_{p^n}(\alpha)^\times$ (the smallest positive integer e so that $\alpha^e = 1$). Then α is a zero of $x^e - 1$. Thus $x^e - 1 \in (f(x))$ since $(f(x))$ is the kernel of the evaluation homomorphism

$$\phi_\alpha : \mathbb{F}_{p^n}[x] \to E.$$

Thus $f(x) \mid x^e - 1$. It follows that $e = \text{order}(f(x))$. □

An irreducible polynomial $f(x)$ of degree k in $\mathbb{F}_{p^n}[x]$ is **primitive** if $\text{order}(f(x)) = p^{nk}-1$. Equivalently, an irreducible, degree k polynomial $f(x)$ is primitive if there exists a root α of $f(x)$ for which $\langle \alpha \rangle = \mathbb{F}_{p^n}(\alpha)^\times$. For example, in $\mathbb{F}_2$, every irreducible polynomial of degree $k = 3, 5, 7$ is primitive since the group of units $\mathbb{F}_{2^k}^\times$ has prime order for $k = 3, 5, 7$.

We give some examples of Galois fields of order p^n and primitive and non-primitive polynomials.

Example 7.3.13 Let $p = 5$ and $n = 1$. The Galois field $\mathbb{F}_5 = \mathbb{Z}_5$ contains all of the distinct roots of $x^5 - x \in \mathbb{F}_5[x]$ by Corollary 6.2.7. Indeed, over $\mathbb{F}_5$,

$$x^5 - x = x(x-1)(x-2)(x-3)(x-4).$$

Observe that 2 has order 4 in $U(\mathbb{F}_5) = \mathbb{F}_5^\times$ and 4 has order 2 in $\mathbb{F}_5^\times$. Hence $\text{order}(x - 2) = 4$ and $\text{order}(x-4) = 2$; $x - 2$ is a primitive polynomial over $\mathbb{F}_5$. □

Example 7.3.14 Let $p = 3$ and $n = 2$. Then $\mathbb{F}_9$ consists of all of the roots of $x^9 - x$. In $\mathbb{F}_3[x]$, $x^9 - x$ factors into irreducible elements as

$$x^9 - x = x(x-1)(x+1)(x^2+1)(x^2-x-1)(x^2+x-1).$$

We take α to be a root of $x^2 + 1$ (the other root is $\alpha^3 = 2\alpha$). Thus

$$\mathbb{F}_9 \cong \mathbb{F}_3[x]/(x^2+1) \cong \mathbb{F}_3(\alpha).$$

An $\mathbb{F}_3$-basis for $\mathbb{F}_3(\alpha)$ is $\{1, \alpha\}$. The 9 elements of $\mathbb{F}_9$ are thus

$$0 = 0 \cdot 1 + 0 \cdot \alpha,$$

$$\alpha = 0 \cdot 1 + 1 \cdot \alpha,$$

$$\begin{aligned}
2\alpha &= 0\cdot 1 + 2\cdot\alpha,\\
1 &= 1\cdot 1 + 0\cdot\alpha,\\
1+\alpha &= 1\cdot 1 + 1\cdot\alpha,\\
1+2\alpha &= 1\cdot 1 + 2\cdot\alpha,\\
2 &= 2\cdot 1 + 0\cdot\alpha,\\
2+\alpha &= 2\cdot 1 + 1\cdot\alpha,\\
2+2\alpha &= 2\cdot 1 + 2\cdot\alpha.
\end{aligned}$$

Both α and α^3 have order 4 in $\mathbb{F}_9^\times$, and so $\text{order}(x^2+1) = 4$. Thus x^2+1 is a nonprimitive polynomial over $\mathbb{F}_3$. On the other hand, the root β of $x^2 - x - 1 \in \mathbb{F}_3[x]$ has order 8 in $\mathbb{F}_9^\times$, and so $x^2 - x - 1$ is a primitive polynomial over $\mathbb{F}_3$. □

Example 7.3.15 In this example, we take $\mathbb{F}_9 = \mathbb{F}_3(\alpha)$, $\alpha^2+1 = 0$, as our base field. Let $f(x) = x^2 + x + \beta \in \mathbb{F}_9[x]$ with β as in Example 7.3.14. Then one checks directly that $f(x)$ is irreducible over $\mathbb{F}_9$. By Proposition 7.3.10, the (distinct) roots of $f(x)$ are γ, γ^9; $\mathbb{F}_9(\gamma) = \mathbb{F}_{81}$. We have

$$(x-\gamma)(x-\gamma^9) = x^2 + x + \beta,$$

so that $\gamma^{10} = \beta$. Since β has order 8 in $\mathbb{F}_9$, γ has order 80 in $\mathbb{F}_{81}^\times$. Thus $f(x)$ is primitive over $\mathbb{F}_9$. □

The finite fields $\mathbb{F}_{2^n}$, $n \geq 1$, are of use in computer science since their base field $\mathbb{F}_2 = \{0, 1\}$ represents the collection of binary digits (bits).

Example 7.3.16 Consider $\mathbb{F}_{16}$. The polynomial $x^{16} - x \in \mathbb{F}_2[x]$ factors into irreducibles as

$$x(x+1)(x^2+x+1)(x^4+x+1)(x^4+x^3+1)(x^4+x^3+x^2+x+1). \quad (7.3)$$

So $\mathbb{F}_{16}$ can be constructed as $\mathbb{F}_2[x]/(x^4+x+1) = \mathbb{F}_2(\alpha)$, where α is a root of x^4+x+1. An $\mathbb{F}_2$-basis for $\mathbb{F}_{16}$ is $\{1, \alpha, \alpha^2, \alpha^3\}$ and so the 16 elements are

$$\begin{aligned}
0 &= 0\cdot 1 + 0\cdot\alpha + 0\cdot\alpha^2 + 0\cdot\alpha^3,\\
\alpha^3 &= 0\cdot 1 + 0\cdot\alpha + 0\cdot\alpha^2 + 1\cdot\alpha^3,\\
\alpha^2 &= 0\cdot 1 + 0\cdot\alpha + 1\cdot\alpha^2 + 0\cdot\alpha^3,\\
\alpha^2+\alpha^3 &= 0\cdot 1 + 0\cdot\alpha + 1\cdot\alpha^2 + 1\cdot\alpha^3,\\
\alpha &= 0\cdot 1 + 1\cdot\alpha + 0\cdot\alpha^2 + 0\cdot\alpha^3,\\
\alpha+\alpha^3 &= 0\cdot 1 + 1\cdot\alpha + 0\cdot\alpha^2 + 1\cdot\alpha^3,\\
\alpha+\alpha^2 &= 0\cdot 1 + 1\cdot\alpha + 1\cdot\alpha^2 + 0\cdot\alpha^3,
\end{aligned}$$

$$\begin{aligned}
\alpha+\alpha^2+\alpha^3 &= 0\cdot 1+1\cdot\alpha+1\cdot\alpha^2+1\cdot\alpha^3,\\
1 &= 1\cdot 1+0\cdot\alpha+0\cdot\alpha^2+0\cdot\alpha^3,\\
1+\alpha^3 &= 1\cdot 1+0\cdot\alpha+0\cdot\alpha^2+1\cdot\alpha^3,\\
1+\alpha^2 &= 1\cdot 1+0\cdot\alpha+1\cdot\alpha^2+0\cdot\alpha^3,\\
1+\alpha^2+\alpha^3 &= 1\cdot 1+0\cdot\alpha+1\cdot\alpha^2+1\cdot\alpha^3,\\
1+\alpha &= 1\cdot 1+1\cdot\alpha+0\cdot\alpha^2+0\cdot\alpha^3,\\
1+\alpha+\alpha^3 &= 1\cdot 1+1\cdot\alpha+0\cdot\alpha^2+1\cdot\alpha^3,\\
1+\alpha+\alpha^2 &= 1\cdot 1+1\cdot\alpha+1\cdot\alpha^2+0\cdot\alpha^3,\\
1+\alpha+\alpha^2+\alpha^3 &= 1\cdot 1+1\cdot\alpha+1\cdot\alpha^2+1\cdot\alpha^3.
\end{aligned}$$

In fact, $f(x) = x^4 + x + 1$ is a primitive polynomial over $\mathbb{F}_2$: from the factorization (7.3), $f(x)$ is irreducible. By Proposition 7.3.12, $\text{order}(f(x)) = 3$ or 5 or 15. But clearly, $f(x) \nmid x^3 - 1$ and $f(x) \nmid x^5 - 1$, and thus $\text{order}(f(x)) = 15$ which says that $f(x)$ is primitive. □

The Galois field $\mathbb{F}_{16}$ is the field consisting of all possible half-bytes (strings of 0's and 1's of length 4). The addition is given by bit-wise addition modulo 2, and the multiplication is induced by the relation $\alpha^4 = 1 + \alpha$. For example,

$$0110 + 1100 = 1010,$$

and

$$0110 \cdot 1001 = 1100,$$

since $(\alpha + \alpha^2)(1 + \alpha^3) = 1 + \alpha$.

7.4 Invertible Matrices over $\mathbb{Z}_{pq}$

We close the chapter with some material needed in the construction of the Hill cipher (Section 8.3).

Let $p, q > 0$ be distinct primes and let $\mathbb{Z}_{pq}$ be the ring of residues. As shown in Section 6.1, the set of $n \times n$ matrices $\text{Mat}_n(\mathbb{Z}_{pq})$ is a ring with unity under ordinary matrix addition and multiplication. The unity in $\text{Mat}_n(\mathbb{Z}_{pq})$ is the $n \times n$ identity matrix I_n. The group of units of $\text{Mat}_n(\mathbb{Z}_{pq})$, $U(\text{Mat}_n(\mathbb{Z}_{pq}))$, is the group of invertible $n \times n$ matrices $\text{GL}_n(\mathbb{Z}_{pq})$.

In this section, we compute the number of elements in $\text{GL}_n(\mathbb{Z}_{pq})$.

Lemma 7.4.1 *Let p be prime and let $\mathrm{GL}_n(\mathbb{Z}_p)$ denote the (group of) invertible $n \times n$ matrices over $\mathbb{Z}_p$. Then*

$$|\mathrm{GL}_n(\mathbb{Z}_p)| = p^{n^2} \prod_{i=1}^{n} \left(1 - \frac{1}{p^i}\right).$$

Proof We view the matrix ring $\mathrm{Mat}_n(\mathbb{Z}_p)$ as a $\mathbb{Z}_p$-vector space W of dimension n^2.

We construct an invertible matrix column by column and count the possibilities for each column. Since $\mathbb{Z}_p$ is a field, the first column is an arbitrary non-zero vector over $\mathbb{Z}_p$. This yields $p^n - 1$ choices for the first column. The first column spans a one-dimensional subspace W_1 of W containing p elements.

The second column must be chosen so that the first and second columns are linearly independent and thus span a two-dimensional subspace $W_2 \subseteq W$ containing p^2 elements. The second column must be chosen from $W \backslash W_1$. Hence there are $p^n - p$ choices for the second column. Continuing in this manner, we see that there are $p^n - p^2$ choices for the third column, and so on. It follows that

$$|\mathrm{GL}_n(\mathbb{Z}_p)| = \prod_{i=0}^{n-1} (p^n - p^i) = p^{n^2} \prod_{i=1}^{n} \left(1 - \frac{1}{p^i}\right).$$

□

Example 7.4.2 Let $n = 1$. Then we may identify $\mathrm{GL}_1(\mathbb{Z}_p)$ with $U(\mathbb{Z}_p)$. The formula yields

$$|\mathrm{GL}_1(\mathbb{Z}_p)| = p - 1$$

as expected. □

Proposition 7.4.3 *Let p and q be distinct primes. Then*

$$|\mathrm{GL}_n(\mathbb{Z}_{pq})| = (pq)^{n^2} \prod_{i=1}^{n} \left(1 - \frac{1}{p^i}\right)\left(1 - \frac{1}{q^i}\right).$$

Proof By Proposition 7.1.16, $\mathbb{Z}_{pq} \cong \mathbb{Z}_p \times \mathbb{Z}_q$, as rings. Thus there is a bijection

$$\mathrm{Mat}_n(\mathbb{Z}_{pq}) \to \mathrm{Mat}_n(\mathbb{Z}_p) \times \mathrm{Mat}_n(\mathbb{Z}_q).$$

It follows that

$$\begin{aligned} |\mathrm{Mat}_n(\mathbb{Z}_{pq})| &= |\mathrm{Mat}_n(\mathbb{Z}_p) \times \mathrm{Mat}_n(\mathbb{Z}_q)| \\ &= p^{n^2} \prod_{i=1}^{n} \left(1 - \frac{1}{p^i}\right) q^{n^2} \prod_{i=1}^{n} \left(1 - \frac{1}{q^i}\right) \end{aligned}$$

$$= (pq)^{n^2} \prod_{i=1}^{n} \left(1 - \frac{1}{p^i}\right)\left(1 - \frac{1}{q^i}\right)$$

by Lemma 7.4.1. □

Example 7.4.4 Let $n = 2$ and $p = 2, q = 13$. Then the formula yields

$$|\mathrm{GL}_2(\mathbb{Z}_{26})| = (26)^4 \prod_{i=1}^{2} \left(1 - \frac{1}{2^i}\right)\left(1 - \frac{1}{13^i}\right) = 157,248.$$

□

So there are 157, 248 elements in the group of units of the matrix ring $\mathrm{Mat}_2(\mathbb{Z}_{26})$. One of these units is $A = \begin{pmatrix} 10 & 7 \\ 1 & 5 \end{pmatrix}$; indeed, using the familiar formula

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix} = (ad - bc)I_2,$$

one obtains

$$A^{-1} = \begin{pmatrix} 11 & 21 \\ 3 & 22 \end{pmatrix}.$$

We can also compute A^{-1} using GAP:

```
gap> A:=[[10,7],[1,5]];
[ [ 10, 7 ], [ 1, 5 ] ]
gap> Inverse(A) mod 26;
[ [ 11, 21 ], [ 3, 22 ] ].
```

7.5 Exercises

1. As in Remark 7.1.2, verify that the coset operations on the quotient ring R/N are well-defined on left cosets, i.e., if $x + N = a + N$ and $y + N = b + N$, then $(x + N) + (y + N) = (a + b) + N$ and $(x + N) \cdot (y + N) = ab + N$.
2. Let R be a commutative ring with unity, and let N be an ideal of R. Show that the map $f : R \to R/N$ defined as $a \mapsto a + N$ is a surjective homomorphism of rings.
3. Let $\psi : R \to R'$ be a homomorphism of commutative rings with unity. Let $U(R)$ and $U(R')$ denote the groups of units, respectively. Prove that ψ restricted to $U(R)$ determines a homomorphism of groups $\psi : U(R) \to U(R')$.

4. Let $\psi : \mathbb{Z}/5\mathbb{Z} \to \mathbb{Z}_5$ be defined as $a + 5\mathbb{Z} \mapsto (a \bmod 5)$. Show that ψ is an isomorphism of rings.
5. Let R_1 and R_2 be rings and let $R_1 \times R_2$ denote the direct product of rings. Prove that
$$\mathrm{Mat}_n(R_1 \times R_2) \cong \mathrm{Mat}_n(R_1) \times \mathrm{Mat}_n(R_2)$$
as rings.
6. Let $\mathbb{Q}(\sqrt{2})$ denote the simple algebraic field extension of $\mathbb{Q}$.
   (a) Compute $(2 + \sqrt{2})(5 - 2\sqrt{2})$ in $\mathbb{Q}(\sqrt{2})$.
   (b) Compute $(1 + \sqrt{2})^{-1}$.
7. Let $\mathbb{F}_8$ denote the finite field of $2^3 = 8$ elements.
   (a) Factor the polynomial $x^8 - x$ into a product of irreducible polynomials over $\mathbb{F}_2$.
   (b) Using invented roots, write $\mathbb{F}_8$ as a simple algebraic extension of $\mathbb{F}_2$.
   (c) Using part (b), write each element of $\mathbb{F}_8$ as a sequence of 3 bits.
   (d) Using parts (b) and (c), compute $011 \cdot 101$ in $\mathbb{F}_8$.
8. Let $\mathbb{F}_9$ and $\mathbb{F}_{81}$ be the Galois fields with 9 and 81 elements, respectively. Find an irreducible polynomial $f(x) \in \mathbb{F}_9[x]$ and a root β of $f(x)$ so that $\mathbb{F}_{81} = \mathbb{F}_9(\beta)$.
9. Let $n \geq 1$ be an integer and let $\mathbb{F}_p$ denote the Galois field with p elements. Prove that there exists an irreducible polynomial of degree n over $\mathbb{F}_p$.
10. Prove that $f(x) = x^4 + x^2 + 1$ is not a primitive polynomial over $\mathbb{F}_2$.
11. Determine whether $f(x) = x^3 + 2x^2 + 1$ is a primitive polynomial over $\mathbb{F}_3$.
12. Find the order of the units group $U(\mathrm{Mat}_3(\mathbb{Z}_{10})) = GL_3(\mathbb{Z}_{10})$.
13. Find the order of the units group $U(\mathrm{Mat}_n(\mathbb{F}_p^m)) = GL_n(\mathbb{F}_{p^m})$.

Chapter 8
Symmetric Key Cryptography

In general terms, a cryptosystem is a system of the form

$$\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle$$

where $\mathcal{M}$ is the message space, $\mathcal{C}$ is the space of all possible cryptograms, e is the encryption transformation, d is the decryption transformation, $\mathcal{K}_e$ is the encryption keyspace, and $\mathcal{K}_d$ is the decryption keyspace.

The encryption transformation is a function

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

For message $M \in \mathcal{M}$ and encryption key $k_e \in \mathcal{K}_e$,

$$e(M, k_e) = C \in \mathcal{C}.$$

The decryption transformation is a function

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

For $k_e \in \mathcal{K}_e$ and $M \in \mathcal{M}$, there is a decryption key $k_d \in \mathcal{K}_d$ so that

$$d(e(M, k_e), k_d) = M. \tag{8.1}$$

In other words, the composition $d(e(x, k_e), k_d)$ is the identity function on $\mathcal{M}$.

If $k = k_e = k_d$ is a secret key shared by Alice and Bob, then the cryptosystem is a symmetric (key) cryptosystem. In a symmetric cryptosystem, $\mathcal{K} = \mathcal{K}_e = \mathcal{K}_d$, and we will often write a symmetric cryptosystem as

$$\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle.$$

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_8

This chapter concerns the setup, use, and cryptanalysis of the major symmetric cryptosystems.

8.1 Simple Substitution Cryptosystems

Definition 8.1.1 (Simple Substitution Cryptosystem) Let

$$\Sigma = \{0, 1, 2, 3, \ldots, n-1\}$$

denote the alphabet of n letters. A message $M \in \mathcal{M}$ is a finite sequence of letters:

$$M = M_0 M_1 M_2 \cdots M_{r-1}, \quad M_i \in \Sigma.$$

The encryption transformation e is a permutation σ_k in $\langle S_n, \circ\rangle$, the symmetric group on n letters. The keyspace is the set of integers $\mathcal{K} = \{1, 2, 3, \ldots, n!\}$, which is considered as a set of indices for the $n!$ permutations in S_n. The encryption key $k_e \in \mathcal{K}$ indicates which permutation to use for encryption.

Alice encrypts the plaintext message $M = M_0 M_1 M_2 \cdots M_{r-1}$ by computing

$$C = e(M, k_e) = C_0 C_1 C_2 \cdots C_{r-1},$$

where $C_i = \sigma_{k_e}(M_i)$ for $0 \le i \le r-1$.

Bob then decrypts the ciphertext $C = C_0 C_1 C_2 \cdots C_{r-1}$ by first computing the inverse $\sigma_{k_e}^{-1}$ of the permutation σ_{k_e} in the group S_n. He then computes

$$M = \sigma_{k_e}^{-1}(C_0)\sigma_{k_e}^{-1}(C_1)\sigma_{k_e}^{-1}(C_2)\cdots\sigma_{k_e}^{-1}(C_{r-1}).$$

Note that Bob's decrypting task is different from Alice's encrypting task: given k and a permutation σ_k, Alice computes $C_i = \sigma_k(M_i)$, while Bob first must find the inverse σ_k^{-1}, and then compute $M_i = \sigma_k^{-1}(C_i)$. Both Alice and Bob use the same permutation σ_k (though in different ways), and so the shared key for encryption and decryption is $k = k_e = k_d$. This is analogous to the shared key k in the right shift cipher from Chapter 1: The same key k is used differently, i.e., Alice shifts right k places, while Bob shifts left k places.

The cryptosystem $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}\rangle$ described above is the **simple substitution cryptosystem**. □

We show that the simple substitution cryptosystem "works," i.e., condition (8.1) holds. To this end, let $M = M_0 M_1 M_2 \cdots M_{r-1}$ be a message in $\mathcal{M}$. Then

$$\begin{aligned} d(e(M,k),k) &= d(\sigma_k(M_0)\sigma_k(M_1)\sigma_k(M_2)\cdots\sigma_k(M_{r-1}), k) \\ &= \sigma_k^{-1}(\sigma_k(M_0))\sigma_k^{-1}(\sigma_k(M_1))\sigma_k^{-1}(\sigma_k(M_2)) \\ &\qquad \cdots \sigma_k^{-1}(\sigma_k(M_{r-1})) \end{aligned}$$

$$\begin{aligned} &= M_0M_1M_2\cdots M_{r-1} \\ &= M. \end{aligned}$$

Example 8.1.2 In this case, $\Sigma = \{0, 1, 2, 3, \ldots, 25\}$ is the set of 26 letters. The message space consists of finite sequences of letters in Σ that correspond to plaintext English messages upon encoding the ordinary letters as below:

$$\mathtt{A} \leftrightarrow 0,\ \ \mathtt{B} \leftrightarrow 1,\ \ \mathtt{C} \leftrightarrow 2,\ \ \mathtt{D} \leftrightarrow 3,\ \ldots\ ,\ \mathtt{Z} \leftrightarrow 25.$$

The encryption transformation is a permutation σ_k in the symmetric group S_{26}; k is the shared secret key. For instance, suppose that

$$\sigma_k = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 4 & 14 & 18 & 8 & 13 & 1 & 24 & 19 & 2 & 25 & 15 & 9 & 3 & 20 \end{pmatrix.$$
$$\begin{matrix} 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 \\ 23 & 5 & 11 & 17 & 12 & 6 & 0 & 21 & 16 & 7 & 10 & 22 \end{pmatrix}.$$

Then the encryption of

$$\mathtt{C\,E\,L\,L\,P\,H\,O\,N\,E} \leftrightarrow 2\ 4\ 11\ 11\ 15\ 7\ 14\ 13\ 4$$

is

$$\begin{aligned} C &= e(2\ 4\ 11\ 11\ 15\ 7\ 14\ 13\ 4, k) \\ &= \sigma_k(2)\ \sigma_k(4)\ \sigma_k(11)\ \sigma_k(11)\ \sigma_k(15)\ \sigma_k(7)\ \sigma_k(14)\ \sigma_k(13)\ \sigma_k(4) \\ &= 18\ 13\ 9\ 9\ 5\ 19\ 23\ 20\ 13 \leftrightarrow \mathtt{S\,N\,J\,J\,F\,T\,X\,U\,N}. \end{aligned}$$

□

Example 8.1.3 Suppose $\Sigma = \{0, 1, 2\}$, and let $\mathcal{M} = \mathcal{C} = \Sigma^* = \{0, 1, 2\}^*$, where Σ^* denotes the set of all words of finite length over Σ. The encryption transformation e is a permutation in S_3, the symmetric group on 3 letters. From Section 5.3.1, we have

$$S_3 = \{\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6\}.$$

Let $M = 221010$ be a message in $\mathcal{M}$. To compute $C = e(M, 4)$, we recall that $\sigma_4 = \begin{pmatrix} 0 & 1 & 2 \\ 0 & 2 & 1 \end{pmatrix}$. Thus

$$\begin{aligned} C &= e(221010, 4) \\ &= \sigma_4(2)\sigma_4(2)\sigma_4(1)\sigma_4(0)\sigma_4(1)\sigma_4(0) \\ &= 112020. \end{aligned}$$

Also, $\sigma_4^{-1} = \begin{pmatrix} 0 & 1 & 2 \\ 0 & 2 & 1 \end{pmatrix}$, and so the decryption of $C = 212$ is

$$d(212, 4) = \sigma_4^{-1}(2)\sigma_4^{-1}(1)\sigma_4^{-1}(2) = 121.$$

□

8.1.1 Unicity Distance of the Simple Substitution Cryptosystem

As discussed in Section 3.4, the unicity distance of a symmetric cryptosystem is a lower bound for the smallest positive integer n_0 for which

$$\mathrm{Spur}(n_0) = \sum_{C \in L_{n_0} \cap \mathcal{C}} \Pr(C_{n_0} = C)(|W(C)| - 1) = 0.$$

As such, it is a theoretical measure of the cryptosystem's ability to withstand a brute-force attack by key trial.

From the formula (3.1) given in Section 3.4, the unicity distance of the simple substitution cryptosystem (with $n = 26$) is

$$\begin{aligned} \text{u.d.} &= \frac{\log_2(|\mathcal{K}|)}{\text{redundancy rate of English}} \qquad \frac{\text{bits}}{\text{bits/char}} \\ &= \frac{\log_2(|\mathcal{K}|)}{3.2} \qquad \frac{\text{bits}}{\text{bits/char}} \\ &= \frac{\log_2(26!)}{3.2} \\ &= \frac{88.38}{3.2} \\ &\approx 27.61 \text{ characters.} \end{aligned}$$

Thus, for the simple substitution cryptosystem, $n_0 \geq 28$; though the exact value for n_0 might be significantly larger than 28. If Malice captures 28 characters of ciphertext C and performs a brute-force key trial, there *still may be* spurious keys for C. There is certainly some ciphertext of length < 28 that will have at least one spurious key.

8.2 The Affine Cipher

Definition 8.2.1 (Affine Cipher) Let $\Sigma = \{0, 1, 2, 3, \dots, n-1\}$ denote the set of n letters. A message $M \in \mathcal{M}$ is a finite sequence of letters:

$$M = M_0 M_1 M_2 \cdots M_{r-1}, \quad M_i \in \Sigma.$$

Let a be an integer $1 \le a \le n-1$ that satisfies $\gcd(n, a) = 1$, and let b be any integer in $\{0, 1, 2, \dots, n-1\}$. The message $M = M_0 M_1 M_2 \cdots M_{r-1}$ is encrypted letter by letter to yield the ciphertext $C = C_0 C_1 C_2 \cdots C_{r-1}$ where

$$C_i = e(M_i, (a, b)) = ((aM_i + b) \bmod n).$$

The encryption key is the pair of integers $k = (a, b)$.

By Proposition 6.2.2, the condition $\gcd(n, a) = 1$ implies that a is a unit in $\mathbb{Z}_n$, that is, there exists an element a^{-1} of $\mathbb{Z}_n$ so that

$$aa^{-1} = 1 = a^{-1}a.$$

The ciphertext $C = C_0 C_1 C_2 \cdots C_{r-1}$ is decrypted letter by letter to yield the message $M = M_0 M_1 M_2 \cdots M_{r-1}$ with

$$M_i = d(C_i, (a, b)) = (a^{-1}(C_i - b) \bmod n).$$

The symmetric cryptosystem described above is the **affine cipher**. □

We check to see that the affine cipher works: Let $M = M_0 M_1 M_2 \cdots M_{r-1}$ be a message. Then (8.1) holds since

$$\begin{aligned} d(e(M_i, k), k) &= d(((aM_i + b) \bmod n), k) \\ &= (a^{-1}(((aM_i + b) \bmod n) - b)) \bmod n) \\ &= (a^{-1}(aM_i) \bmod n) \\ &= (M_i \bmod n) \\ &= M_i, \end{aligned}$$

for $i = 0, 1, \dots, r-1$.

Here is an example of an affine cipher.

Example 8.2.2 Let $\Sigma = \{0, 1, 2, 3, \dots, 25\}$ be the set of 26 letters. The message space consists of finite sequences of letters in Σ. Since $\gcd(26, 5) = 1$, we can choose the encryption key to be $k = (a, b) = (5, 18)$. Now

$$e(2\ 0\ 19, (5, 18)) = 2\ 18\ 9,$$

since $C_0 = ((5 \cdot 2 + 18) \bmod 26) = 2$, $C_1 = ((5 \cdot 0 + 18) \bmod 26) = 18$, and $C_2 = ((5 \cdot 19 + 18) \bmod 26) = 9$.

To decrypt the ciphertext $C = 7\ 10\ 22$, we note that decrypting with the key (5, 18) is the same as encrypting with the key (−5, 12) since

$$-5(5x + 18) + 12 = -25x - 78 \equiv x \pmod{26}.$$

Thus $M_0 = e(7, (-5, 12)) = ((-35 + 12) \bmod 26) = 3$, $M_1 = e(10, (-5, 12)) = (-50+12) \bmod 26) = 14$, and $M_2 = e(22, (-5, 12)) = (-110+12) \bmod 26) = 6$. Thus

$$3\ 14\ 6 = d(7\ 10\ 22, (5, 18)).$$

□

In the affine cipher, if we take $a = 1$, $b \in \mathbb{Z}_n$, then we obtain the **shift cipher**, in which encryption is

$$C_i = e(M_i, (1, b)) = ((M_i + b) \bmod n),$$

and decryption is

$$M_i = d(C_i, (1, b)) = ((C_i - b) \bmod n).$$

The following example of a shift cipher should look familiar—it is the right shift cryptosystem given in Section 1.1.

Example 8.2.3 Let $\Sigma = \{0, 1, 2, 3, \ldots, 25\}$ be the set of 26 letters. The message space consists of finite sequences of letters in Σ that correspond to plaintext English messages upon encoding the ordinary letters as below:

$$\mathtt{A} \leftrightarrow 0,\ \ \mathtt{B} \leftrightarrow 1,\ \ \mathtt{C} \leftrightarrow 2,\ \ \mathtt{D} \leftrightarrow 3,\ \ldots\ ,\ \mathtt{Z} \leftrightarrow 25.$$

Now with the key chosen to be $k = 21$, the encryption of

$$M = \mathtt{G\,O\,O\,D\,W\,Y\,N\,H\,A\,L\,L} \leftrightarrow 6\ 14\ 14\ 3\ 22\ 24\ 13\ 7\ 0\ 11\ 11$$

is

$$\begin{aligned} C &= e(6\ 14\ 14\ 3\ 22\ 24\ 13\ 7\ 0\ 11\ 11, 21) \\ &= 1\ 9\ 9\ 24\ 17\ 19\ 8\ 2\ 21\ 6\ 6 \\ &\leftrightarrow \mathtt{B\,J\,J\,Y\,R\,T\,I\,C\,V\,G\,G} \end{aligned}$$

since $C_0 = ((6 + 21) \bmod 26) = 1$, $C_1 = ((14 + 21) \bmod 26) = 9$, $C_2 = ((14 + 21) \bmod 26) = 9$, and so on. □

Both the affine cipher and the shift cipher are special cases of the simple substitution cryptosystem.

Proposition 8.2.4 *The shift cipher and the affine cipher are simple substitution cryptosystems.*

Proof The encryption and decryption transformations for both the shift and affine ciphers are bijective maps $\Sigma \to \Sigma$, and hence are given by permutations in S_n. □

8.2.1 Unicity Distance of the Affine Cipher

We take $n = 26$. We first compute the size of the keyspace for the affine cipher. We have

$$\mathcal{K} = \{(a, b) : \gcd(a, 26) = 1, b \in \mathbb{Z}_{26}\}.$$

Thus

$$|\mathcal{K}| = \phi(26) \cdot 26 = 12 \cdot 26 = 312.$$

Consequently, the unicity distance is

$$\frac{\log_2(312)}{3.2} = 2.589 \text{ char.}$$

8.3 The Hill 2 x 2 Cipher

We generalize the affine cipher to form the **Hill** 2×2 **cipher**.

Definition 8.3.1 (Hill 2×2 **Cipher)** Take Σ to be the alphabet of 2-grams

$$\Sigma = L_2 = \{\mathtt{AA}, \mathtt{AB}, \mathtt{AC}, \dots, \mathtt{ZX}, \mathtt{ZY}, \mathtt{ZZ}\}.$$

These 2-grams are known as **blocks** since they consist of blocks of letters from the standard alphabet $\{\mathtt{A}, \dots, \mathtt{Z}\}$.

We encode each block of Σ as a 2-tuple of integers from 0 to 25, hence,

$$\Sigma = \{0\,0, 0\,1, 0\,2, \dots, 25\,23, 25\,24, 25\,25\}.$$

A message $M \in \mathcal{M}$ is a finite sequence of blocks in Σ:

$$M = M_0 M_1 M_2 \cdots M_{r-1}, \quad M_i \in \Sigma.$$

We consider each block M_i as a 1×2 matrix $(m_{i,1}\ m_{i,2})$, where $m_{i,1}, m_{i,2} \in \mathbb{Z}_{26}$.

Let A be an invertible 2×2 matrix with entries in $\mathbb{Z}_{26}$. These are precisely the 2×2 matrices of the form

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

where $a, b, c, d \in \mathbb{Z}_{26}$, $ad - bc \in U(\mathbb{Z}_{26})$.

Let B^T denote the transpose of any matrix B.

The message $M = M_0 M_1 M_2 \cdots M_{r-1}$ is encrypted block by block to yield the ciphertext $C = C_0 C_1 C_2 \cdots C_{r-1}$ where

$$\begin{aligned} C_i &= e(M_i, A) \\ &= (AM_i^T)^T \\ &= \left(\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} m_{i,1} \\ m_{i,2} \end{pmatrix}\right)^T \\ &= \begin{pmatrix} c_{i,1} \\ c_{i,2} \end{pmatrix}^T \\ &= \begin{pmatrix} c_{i,1} & c_{i,2} \end{pmatrix} \end{aligned}$$

with

$$c_{i,1} = ((am_{i,1} + bm_{i,2}) \bmod 26),$$

$$c_{i,2} = ((cm_{i,1} + dm_{i,2}) \bmod 26).$$

The encryption key is the matrix A.

Since A is an invertible matrix in $\mathrm{Mat}_2(\mathbb{Z}_{26})$, there is a unique matrix A^{-1} with

$$AA^{-1} = I_2 = A^{-1}A,$$

where I_2 is the 2×2 identity matrix $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

The ciphertext $C = C_0 C_1 C_2 \cdots C_{r-1}$ is decrypted block by block to yield the message $M = M_0 M_1 M_2 \cdots M_{r-1}$ with

$$M_i = d(C_i, A) = (A^{-1}C_i^T)^T,$$

modulo 26. □

The symmetric cryptosystem described above is the **Hill** 2×2 **cipher**, named after Lester S. Hill (1891–1961).

We leave it as an exercise to show that the Hill 2 × 2 cipher works.

The Hill 2 × 2 cipher is a generalization of an affine cipher with key $(a, 0)$: An invertible element $a \in \mathbb{Z}_{26}$ can be viewed as an invertible 1×1 matrix in $\mathrm{Mat}_1(\mathbb{Z}_{26})$.

The Hill 2 × 2 cipher is an example of a **block cipher**. Instead of encrypting a message letter by letter, it encrypts blocks of 2 letters at a time. We will discuss block ciphers over the alphabet of bits in Section 8.7.

Here is an example of a Hill 2 × 2 cipher.

Example 8.3.2 Let

$$M = \texttt{Eyes of the World}.$$

We capitalize and consider M as seven blocks of 2-grams, hence

$$M = \texttt{EY ES OF TH EW OR LD}.$$

Encoding as blocks of 2-grams over $\mathbb{Z}_{26}$ gives

$$M = 4\ 24\ \ 4\ 18\ \ 14\ 5\ \ 19\ 7\ \ 4\ 22\ \ 14\ 17\ \ 11\ 3.$$

With key $A = \begin{pmatrix} 10 & 7 \\ 1 & 5 \end{pmatrix} \in \mathrm{GL}_2(\mathbb{Z}_{26})$, encryption is given as

$$C_1 = e(M_1, A) = \left(\begin{pmatrix} 10 & 7 \\ 1 & 5 \end{pmatrix} \begin{pmatrix} 4 \\ 24 \end{pmatrix} \right)^T = \begin{pmatrix} 0 & 20 \end{pmatrix},$$

$$C_2 = e(M_1, A) = \left(\begin{pmatrix} 10 & 7 \\ 1 & 5 \end{pmatrix} \begin{pmatrix} 4 \\ 18 \end{pmatrix} \right)^T = \begin{pmatrix} 10 & 16 \end{pmatrix},$$

$$\vdots$$

$$C_7 = e(M_1, A) = \left(\begin{pmatrix} 10 & 7 \\ 1 & 5 \end{pmatrix} \begin{pmatrix} 11 \\ 3 \end{pmatrix} \right)^T = \begin{pmatrix} 1 & 0 \end{pmatrix}.$$

So that the ciphertext is

$$C = 0\ 20\ \ 10\ 16\ \ 19\ 13\ \ 5\ 2\ \ 12\ 10\ \ 25\ 21\ \ 1\ 0,$$

which decodes as

$$C = \texttt{AU KQ TN FC MK ZV BA}.$$

□

8.3.1 Unicity Distance of the Hill 2 × 2 Cipher

In Section 3.2, we computed that the entropy rate of plaintext English over the ordinary alphabet $L_1 = \{\mathtt{A}, \mathtt{B}, \ldots, \mathtt{Z}\}$. We obtained:

$$H_\infty = \lim_{n\to\infty} \frac{H_n}{n} \approx 1.5 \text{ bits/char},$$

where H_n denotes the entropy of the n-gram relative frequency distribution. The redundancy rate of plaintext is thus 3.2 bits/char.

To compute the unicity distance of the Hill 2×2 cipher, we first need to reconsider the redundancy rate per character, given that plaintext messages are now written as words over the alphabet of 2-grams,

$$\Sigma = L_2 = \{\mathtt{AA}, \mathtt{AB}, \mathtt{AC}, \ldots, \mathtt{ZX}, \mathtt{ZY}, \mathtt{ZZ}\}.$$

Thus, a single character (or letter) is now a 2-gram block.

We need to consider the sequence of entropy rates:

$$\{H_{2n}/n\} = H_2/1, H_4/2, H_6/3, \ldots$$

The limit

$$\lim_{n\to\infty} \frac{H_{2n}}{n}$$

is the entropy rate of plaintext when written in the alphabet L_2.

We compute this limit as follows. The sequence $\{H_{2n}/2n\}$ is a subsequence of the convergent sequence $\{H_n/n\}$, and thus, by Rudin [51, p. 51], $\{\frac{H_{2n}}{2n}\}$ converges to the same limit as $\{\frac{H_n}{n}\}$. So we take

$$\lim_{n\to\infty} \frac{H_{2n}}{2n} = 1.5,$$

thus,

$$\lim_{n\to\infty} \frac{H_{2n}}{n} = 2 \lim_{n\to\infty} \frac{H_{2n}}{2n} = 2(1.5) = 3.0.$$

Hence, the entropy rate is 3.0 bits/char.

Now, the maximum entropy is $\log_2(26^2) = 9.4$, and so, the redundancy rate in the Hill 2×2 cipher is

$$9.4 - 3.0 = 6.4 \text{ bits/char}.$$

The keyspace $\mathcal{K}$ consists of all invertible 2×2 matrices over $\mathbb{Z}_{26}$. From Section 7.4, we find that $|\mathcal{K}| = |\mathrm{GL}_2(\mathbb{Z}_{26})| = 157,248$. Thus, the unicity distance of the Hill 2×2 cipher is

$$\frac{\log_2(157,248)}{6.4} = 2.69 \text{ characters.}$$

Of course, a character is actually a 2-gram block, so we may take the unicity distance of the Hill 2×2 cipher to be 5.38.

The unicity distance of the Hill 2×2 cipher ($u.d. = 5.38$) is larger than that of the affine cipher ($u.d. = 2.59$) but smaller than the unicity distance of the simple substitution cryptosystem ($u.d. = 27.61$).

The Hill 2×2 cipher can be generalized to the Hill $n \times n$ cipher, $n > 2$, which is a block cipher that encrypts blocks of n letters at a time using matrices in $\mathrm{GL}_n(\mathbb{Z}_{26})$. See [47, §8.2].

Using Proposition 7.4.3, one can show that the unicity distance of the Hill $n \times n$ cipher is

$$\frac{\log_2\left(26^{n^2}\prod_{i=1}^{n}\left(1-\frac{1}{2^i}\right)\left(1-\frac{1}{13^i}\right)\right)}{3.2n}$$

characters, where each character is a block of length n.

8.4 Cryptanalysis of the Simple Substitution Cryptosystem

Suppose Alice and Bob are using the simple substitution cryptosystem $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ to communicate securely. They have met previously to establish a shared secret key $k = k_e = k_d$. The shared key k is an integer $1 \le k \le 26!$ and indicates which of the 26! permutations is to be used to encrypt and decrypt messages. Alice will use permutation σ_k to encrypt, and Bob will use its inverse σ_k^{-1} to decrypt.

Malice has eavesdropped on their conversation and has obtained 206 characters of ciphertext:

$C =$

```
EVWB WB X FZZD XFZNE VZG LZYLQREB WY SXEVQSXEWLB XUQ NBQK EZ
KQCQTZR EVQ FXBWL LZSRZYQYEB ZH LUJREZAUXRVJ XE EWSQB EVQ
SXEVQSXEWLXT HZNYKXEWZYB XUQ XB WSRZUEXYE XB EVQ LUJREZAUXRVJ
EVQ EQME XBBNSQB FXBWL DYZGTQKAQ ZH LXTLNTNB XYK TWYQXU
XTAQFUX
```

We can view C as a single word of length 244 over the 27 character alphabet

$$\{\mathtt{A}, \mathtt{B}, \mathtt{C}, \mathtt{D}, \mathtt{E}, \mathtt{F}, \mathtt{G}, \mathtt{H}, \mathtt{I}, \mathtt{J}, \mathtt{K}, \mathtt{L}, \mathtt{M}, \mathtt{N}, \mathtt{O}, \mathtt{P}, \mathtt{Q}, \mathtt{R}, \mathtt{S}, \mathtt{T}, \mathtt{U}, \mathtt{V}, \mathtt{W}, \mathtt{X}, \mathtt{Y}, \mathtt{Z}, \not{b}\},$$

where the additional character $\not{b}$ is the symbol for a word separator (white space). There are 206 ordinary letters plus 38 word separators. The permutation σ_k used by Alice satisfies $\sigma_k(\not{b}) = \not{b}$. Thus there are still only 26! possible keys.

Whether or not we view the white space as a character, Malice's goal is to determine Alice's original plaintext message using a method of cryptanalysis.

Malice knows that the brute-force method of key trial will likely determine the correct key, and thus break the cipher, since the unicity distance of the simple substitution cryptosystem is 28 characters.

Malice wants to avoid the time-consuming method of key trial; key trial is almost impossible since there are $26! \approx 4 \times 10^{26}$ keys, nearly beyond a feasible computation.

Instead, Malice chooses to make an educated guess as to what the key could be. To do this, he will employ a technique of cryptanalysis called **frequency analysis**. Frequency analysis employs the (known) relative frequency distributions of plaintext English n-grams.

We have the 1-gram relative frequency distribution $f_1 : L_1 \rightarrow [0, 1]$ of plaintext English (Figure 3.3), which in table form appears as

Letter	Prob.	Letter	Prob.
A	0.0804	N	0.0709
B	0.0154	O	0.0760
C	0.0306	P	0.0200
D	0.0399	Q	0.0011
E	0.1251	R	0.0612
F	0.0230	S	0.0654
G	0.0196	T	0.0925
H	0.0549	U	0.0271
I	0.0726	V	0.0099
J	0.0016	W	0.0192
K	0.0067	X	0.0019
L	0.0414	Y	0.0173
M	0.0253	Z	0.0009

The eight highest relative frequencies occur for the letters

E, T, A, O, I, N, S, H.

By Bernoulli's theorem (or the Law of Large Numbers), in a sample of plaintext English consisting of 206 letters, we can expect the following frequencies for these letters:

Letter	Expected freq.	Letter	Expected freq.
E	$0.1251 \times 206 \approx 26$	I	$0.0726 \times 206 \approx 15$
T	$0.0925 \times 206 \approx 19$	N	$0.0709 \times 206 \approx 15$
A	$0.0804 \times 206 \approx 17$	S	$0.0654 \times 206 \approx 13$
O	$0.0760 \times 206 \approx 16$	H	$0.0549 \times 235 \approx 11$

In the collection of 206 ciphertext characters that Malice has obtained, the eight highest letter frequencies are

Letter	Frequency	Letter	Frequency
X	24	B	16
E	22	L	11
Q	20	W	11
Z	17	V	10

From these data, together with knowledge of common English words of lengths one, two, and three, Malice can determine the plaintext as follows.

Malice first compares the expected frequencies in the plaintext with the actual frequencies in the ciphertext and guesses that Alice has encrypted the plaintext using a permutation σ_k in which

$$\mathtt{E} \mapsto \mathtt{X}.$$

Notice that the ciphertext contains the 2-gram XE, which is the encryption of a 2-letter word in English. Thus, if Alice had used a permutation with $\mathtt{E} \mapsto \mathtt{X}$, then XE is the encryption of a 2-letter word in English that begins with E. There are no such words in common usage.

So a better guess by Malice is that

$$\mathtt{E} \mapsto \mathtt{E}.$$

(based on a comparison of frequencies.) But then the ciphertext 2-gram EZ is the encryption of a 2-letter word in English that begins with E, which is again not possible.

So Malice assumes that $\mathtt{E} \mapsto \mathtt{Q}$. But now, the 3-gram EVQ in the ciphertext is the encryption of a 3-letter word in English ending in E; it is likely that the plaintext word is THE, which is the most common 3-letter word in English. So Malice concludes further that

$$\mathtt{T} \mapsto \mathtt{E}, \quad \mathtt{H} \mapsto \mathtt{V}.$$

Now, the ciphertext 2-gram EZ is the encryption of a 2-letter word in English beginning with T, which Malice guesses to be TO. Thus

$$\mathtt{O} \mapsto \mathtt{Z}.$$

Finally, Malice guesses that the ciphertext 1-gram X is most likely the encryption of the English 1-gram A. Thus Malice guesses that Alice's permutation satisfies

$$\mathtt{A} \mapsto \mathtt{X}.$$

After this analysis, Malice concludes that Alice's encryption permutation is of the form

$$\left(\begin{array}{ccccccccccccc} \mathtt{A} & \mathtt{B} & \mathtt{C} & \mathtt{D} & \mathtt{E} & \mathtt{F} & \mathtt{G} & \mathtt{H} & \mathtt{I} & \mathtt{J} & \mathtt{K} & \mathtt{L} & \mathtt{M} \\ \mathtt{X} & \ast & \ast & \ast & \mathtt{Q} & \ast & \ast & \mathtt{V} & \ast & \ast & \ast & \ast & \ast \end{array}\right.$$
$$\left.\begin{array}{ccccccccccccc} \mathtt{N} & \mathtt{O} & \mathtt{P} & \mathtt{Q} & \mathtt{R} & \mathtt{S} & \mathtt{T} & \mathtt{U} & \mathtt{V} & \mathtt{W} & \mathtt{X} & \mathtt{Y} & \mathtt{Z} \\ \ast & \mathtt{Z} & \ast & \ast & \ast & \ast & \mathtt{E} & \ast & \ast & \ast & \ast & \ast & \ast \end{array}\right).$$

Decryption of the ciphertext therefore uses an inverse σ_k^{-1} of the form

$$\left(\begin{array}{ccccccccccccc} \mathtt{A} & \mathtt{B} & \mathtt{C} & \mathtt{D} & \mathtt{E} & \mathtt{F} & \mathtt{G} & \mathtt{H} & \mathtt{I} & \mathtt{J} & \mathtt{K} & \mathtt{L} & \mathtt{M} \\ \ast & \ast & \ast & \ast & \mathtt{T} & \ast & \ast & \ast & \ast & \ast & \ast & \ast & \ast \end{array}\right.$$
$$\left.\begin{array}{ccccccccccccc} \mathtt{N} & \mathtt{O} & \mathtt{P} & \mathtt{Q} & \mathtt{R} & \mathtt{S} & \mathtt{T} & \mathtt{U} & \mathtt{V} & \mathtt{W} & \mathtt{X} & \mathtt{Y} & \mathtt{Z} \\ \ast & \ast & \ast & \mathtt{E} & \ast & \ast & \ast & \ast & \mathtt{H} & \ast & \mathtt{A} & \ast & \mathtt{O} \end{array}\right).$$

Malice then computes $M = d(C, k)$ to obtain $M =$

```
TH** ** A *OO* A*O*T HO* *O**E*T* ** *ATHE*AT*** A*E **E* TO
*E*E*O* THE *A*** *O**O*E*T* O* ****TO**A*H* AT T**E* THE
*ATHE*AT**A* *O***AT*O** A*E A* ***O*TA*T A* THE ****TO**A*H*
THE TE*T A****E* *A*** **O**E**E O* *A* * * * * ** A** ***EA*
A**E**A
```

Now, in the partial decryption, we guess that the word A∗E is ARE. Thus

$$\mathtt{ARE} \mapsto \mathtt{XUQ}$$

and so, Malice concludes that Alice's permutation satisfies

$$\mathtt{R} \mapsto \mathtt{U}.$$

Moreover, we guess that ∗ATHE∗AT∗∗∗ is MATHEMATICS, and ∗ATHE∗AT∗∗A∗ is MATHEMATICAL. Thus

$$\mathtt{MATHEMATICS} \mapsto \mathtt{SXEVQSXEWLB}, \quad \mathtt{MATHEMATICAL} \mapsto \mathtt{SXEVQSXEWLXT},$$

thus σ_k satisfies

$$\mathtt{M} \mapsto \mathtt{S}, \quad \mathtt{I} \mapsto \mathtt{W}, \quad \mathtt{C} \mapsto \mathtt{L}, \quad \mathtt{S} \mapsto \mathtt{B}, \quad \mathtt{L} \mapsto \mathtt{T}.$$

Malice now refines his guess for σ_k and deduces that the inverse σ_k^{-1} is of the form

$$\begin{pmatrix} \mathtt{A} & \mathtt{B} & \mathtt{C} & \mathtt{D} & \mathtt{E} & \mathtt{F} & \mathtt{G} & \mathtt{H} & \mathtt{I} & \mathtt{J} & \mathtt{K} & \mathtt{L} & \mathtt{M} \\ * & \mathtt{S} & * & * & \mathtt{T} & * & * & * & * & * & * & \mathtt{C} & * \end{pmatrix.$$

$$\begin{matrix} \mathtt{N} & \mathtt{O} & \mathtt{P} & \mathtt{Q} & \mathtt{R} & \mathtt{S} & \mathtt{T} & \mathtt{U} & \mathtt{V} & \mathtt{W} & \mathtt{X} & \mathtt{Y} & \mathtt{Z} \\ * & * & * & \mathtt{E} & * & \mathtt{M} & \mathtt{L} & \mathtt{R} & \mathtt{H} & \mathtt{I} & \mathtt{A} & * & \mathtt{O} \end{pmatrix}.$$

With this refinement of σ_k^{-1}, Malice computes $M = d(C, k)$ to obtain $M =$

```
THIS IS A *OO* A*O*T HO* CO*CE*TS I* MATHEMATICS ARE *SE* TO
*E*ELO* THE *ASIC COM*O*E*TS O* CR**TO*RA*H* AT TIMES THE
MATHEMATICAL *O***ATIO*S ARE AS IM*ORTA*T AS THE CR**TO*RA*H*
THE TE*T ASS*MES *ASIC **O*LE**E O* CALC*L*S A** LI*EAR
AL*E*RA
```

In this manner, Malice continues to refine σ_k^{-1}, and ultimately, obtains the plaintext,

```
THIS IS A BOOK ABOUT HOW CONCEPTS IN MATHEMATICS ARE USED TO
DEVELOP THE BASIC COMPONENTS OF CRYPTOGRAPHY AT TIMES THE
MATHEMATICAL FOUNDATIONS ARE AS IMPORTANT AS THE CRYPTOGRAPHY
THE TEXT ASSUMES BASIC KNOWLEDGE OF CALCULUS AND LINEAR ALGEBRA
```

from the *Preface*

The technique of frequency analysis works well in this example because the plaintext English message M (the *Preface* selection) exhibits a 1-gram frequency distribution, which is not that different from the accepted 1-gram distribution $f_1 : L_1 \to [0, 1]$ of plaintext English. See Figure 8.1.

More precisely, if $g : L_1 \to [0, 1]$ is the 1-gram relative frequency distribution of M, then the **total variation distance**

$$D_g = \frac{1}{2} \sum_{\alpha \in \Sigma} |g(\alpha) - f_1(\alpha)|$$

will be small.

On the other hand, as a method of cryptanalysis, frequency analysis does not work as well if the plaintext M is not typical English in terms of letter frequencies. For example, the 82 letters of plaintext

$M =$

```
HYDROGEN HELIUM LITHIUM BERYLLIUM BORON CARBON NITROGEN
OXYGEN
FLUORINE NEON SODIUM MAGNESIUM
```

exhibit 1-gram frequencies that vary significantly from the accepted frequencies (see Figure 8.2).

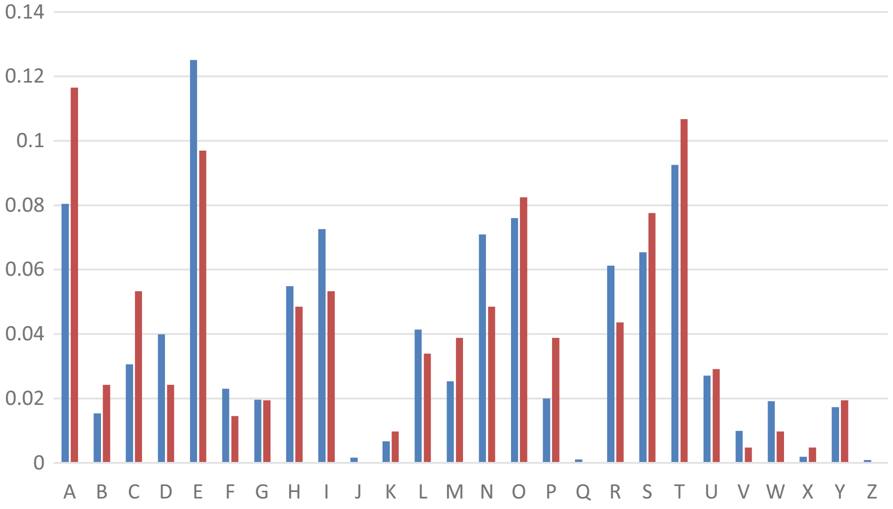


Fig. 8.1 Accepted 1-gram frequencies (blue), compared to 1-gram frequencies of the *Preface* selection (red)

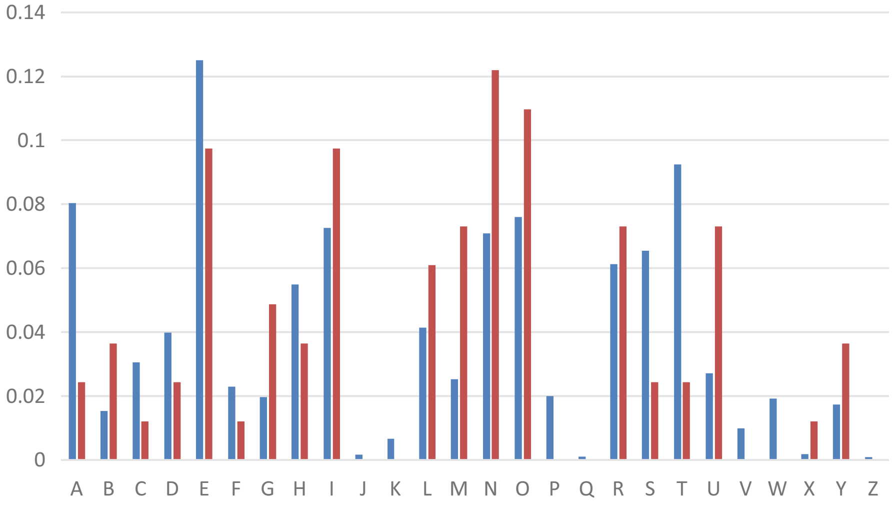


Fig. 8.2 Accepted 1-gram frequencies (blue), compared to 1-gram frequencies atomic elements (red)

Thus it would be relatively difficult to decrypt $C = e(M, k)$ if M was encrypted using a simple substitution cryptosystem.

Block ciphers (such as the Hill 2 × 2 cipher) fare better under an attack by frequency analysis since 2-gram frequencies must be used (Figure 3.4, [35, Table 2.3.4]) rather than 1-gram frequencies. Hence more characters of ciphertext must be captured to make good guesses for the encryption of plaintext 2-grams. Indeed, the unicity distance of the Hill 2×2 cipher is larger than that of the affine cipher.

In the symmetric cryptosystems that we have discussed above: the simple substitution cryptosystem, the affine cipher, and the shift cipher, each letter of the plaintext message is encrypted as a unique letter in the ciphertext. This is also true for the Hill cipher since the characters that make up the plaintext (2-gram blocks of letters AA, AB, and so on) are encrypted as unique 2-gram blocks of letters in the ciphertext.

These ciphers are "monoalphabetic" cryptosystems.

Definition 8.4.1 A **monoalphabetic cryptosystem** is a cryptosystem in which each letter of the plaintext message is encrypted as a unique letter in the ciphertext.

Monoalphabetic cryptosystems have security concerns for the following reasons.

(1) A monoalphabetic cryptosystem is vulnerable to a known-plaintext attack. For instance, let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ be the simple substitution cryptosystem with $\Sigma = \{0, 1, 2, 3, \ldots, 25\}$. If

$$5\ 13\ 2\ 20\ 9\ 8\ 5 = e(0\ 11\ 6\ 4\ 1\ 17\ 0, k),$$

then we can immediately infer that

$$1\ 4\ 0\ 17 = d(9\ 20\ 5\ 8, k).$$

(2) A monoalphabetic cryptosystem is vulnerable to a ciphertext-only attack using frequency analysis. For instance, in plaintext English the letter A occurs with probability ≈ 0.08. If the letter Z occurs with relative frequency ≈ 0.08 in a sample of ciphertext, then we could make a good guess that $e(\texttt{A}, k) = \texttt{Z}$ since Z is the encryption of exactly one plaintext letter.

We shall address these security issues in the next section with the introduction of "polyalphabetic" cryptosystems.

8.5 Polyalphabetic Cryptosystems

Definition 8.5.1 A **polyalphabetic cryptosystem** is a cryptosystem in which a letter in the plaintext message is encrypted as more than one letter in the ciphertext.

We review some important polyalphabetic cryptosystems.

8.5.1 The Vigenère Cipher

Definition 8.5.2 (Vigenère Cipher) Let $\Sigma = \{0, 1, 2, 3, \ldots, n-1\}$ denote the set of n letters. A message $M \in \mathcal{M}$ is a finite sequence of r letters:

$$M = M_0 M_1 M_2 \ldots M_{r-1}, \quad M_i \in \Sigma.$$

The encryption key k_e is a finite sequence of s letters:

$$k = k_0 k_1 k_2 \ldots k_{s-1}, \quad k_i \in \Sigma.$$

The message $M = M_0 M_1 M_2 \cdots M_{r-1}$ is encrypted letter by letter to yield the ciphertext $C = C_0 C_1 C_2 \ldots C_{r-1}$ where

$$C_i = e(M_i, k) = ((M_i + k_{(i \bmod s)}) \bmod n).$$

The ciphertext $C = C_0 C_1 C_2 \cdots C_{r-1}$ is decrypted letter by letter to yield the message $M = M_0 M_1 M_2 \cdots M_{r-1}$ with

$$M_i = d(C_i, k) = ((C_i - k_{(i \bmod s)}) \bmod n).$$

□

We leave it to the reader to show that the Vigenère cipher works, i.e., (8.1) holds. Here is an example.

Example 8.5.3 Let $\Sigma = \{0, 1, 2, 3 \ldots, 25\}$ be the set of 26 letters, and let $\langle \mathcal{M}, \mathcal{C}, e, d, k \rangle$ be the Vigenère cipher with encryption–decryption key

$$k = k_0 k_1 k_2 k_3 = 3\ 7\ 20\ 12.$$

The encryption of $M = 18\ 19\ 0\ 19\ 4\ 14\ 5\ 0\ 11\ 0\ 1\ 0\ 12\ 0$ is

$$\begin{aligned} &e(18\ 19\ 0\ 19\ 4\ 14\ 5\ 0\ 11\ 0\ 1\ 0\ 12\ 0, 3\ 7\ 20\ 12) \\ &\quad = 21\ 0\ 20\ 5\ 7\ 21\ 25\ 12\ 14\ 7\ 21\ 12\ 15\ 7 \end{aligned}$$

since

$$\begin{aligned} C_0 &= ((M_0 + k_0) \bmod 26) = ((18 + 3) \bmod 26) = 21, \\ C_1 &= ((M_1 + k_1) \bmod 26) = ((19 + 7) \bmod 26) = 0, \\ C_2 &= ((M_2 + k_2) \bmod 26) = ((0 + 20) \bmod 26) = 20, \\ C_3 &= ((M_3 + k_3) \bmod 26) = ((19 + 12) \bmod 26) = 5, \\ C_4 &= ((M_4 + k_0) \bmod 26) = ((4 + 3) \bmod 26) = 7, \end{aligned}$$

$$C_5 = ((M_5 + k_1) \bmod 26) = ((14 + 7) \bmod 26) = 21,$$

$$\vdots$$

$$C_{13} = ((M_{13} + k_1) \bmod 26) = ((0 + 7) \bmod 26) = 7.$$

The encryption can be done efficiently using “vertical addition" modulo 26:

$$\begin{array}{cccccccccccccc}
18 & 19 & 0 & 19 & 4 & 14 & 5 & 0 & 11 & 0 & 1 & 0 & 12 & 0 \\
3 & 7 & 20 & 12 & 3 & 7 & 20 & 12 & 3 & 7 & 20 & 12 & 3 & 7 \\
\hline
21 & 0 & 20 & 5 & 7 & 21 & 25 & 12 & 14 & 7 & 21 & 12 & 15 & 7
\end{array}$$

□

The Vigenère cipher is polyalphabetic since the letter 0 in the plaintext message encrypts as the letters 20, 12, and 7. Translating back to the ordinary alphabet, we see that the letter `A` encrypts as the letters `U,M,H`.

8.5.2 Unicity Distance of the Vigenère Cipher

We take $n = 26$, so that we are essentially using the ordinary alphabet of 26 letters.

We first note that the size of the keyspace in the Vigenère cipher is

$$|\mathcal{K}| = 26^s,$$

where s is the key length. Thus (3.1) yields

$$u.d. = \frac{\log_2(26^s)}{3.2} = \frac{s \log_2(26)}{3.2} = \frac{4.7s}{3.2}.$$

In the case of Example 8.5.3, we have $s = 4$, thus the u.d. is 5.87.

8.5.3 Cryptanalysis of the Vigenère Cipher

The Vigenère cipher is less vulnerable than a monoalphabetic cryptosystem to a ciphertext-only attack using frequency analysis. This is the case since a ciphertext letter can be the encryption of more than one plaintext letter, thus the relative frequency of a ciphertext letter reveals less about possible plaintext decryptions.

Still, the Vigenère cipher can be broken using a modification of frequency analysis.

Suppose that message M is encrypted using the Vigenère cipher with key k of length s. The method of cryptanalysis used (to determine M knowing $C = e(M, k)$) depends on whether the key length is known to the attacker.

Key Length Is Known

Suppose Alice and Bob are using the Vigenère cipher $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ with shared key k of length s. Malice knows the value of s and has obtained $m \gg s$ characters of ciphertext:

$$C_0 C_1 C_2 \ldots C_{m-1},$$

$C_i = e(M_i, k)$.

To simplify matters we assume that $s \mid m$; let $q = m/s$. Consider the subsets of ciphertext characters:

$$C_0 C_s C_{2s} \ldots C_{(q-1)s}$$

$$C_1 C_{1+s} C_{1+2s} \ldots C_{1+(q-1)s}$$

$$C_2 C_{2+s} C_{2+2s} \ldots C_{2+(q-1)s}$$

$$\vdots$$

$$C_{s-1} C_{2s-1} C_{3s-1} \ldots C_{qs-1}$$

Each subset can be viewed as the encryption of a subset of plaintext letters using the shift cipher with key k_i for some $0 \le i \le s-1$. For example,

$$C_0 C_s C_{2s} \ldots C_{(q-1)s} = e(M_0 M_s M_{2s} \ldots M_{(q-1)s}, k_0)$$

since

$$C_{js} = ((M_{js} + k_{(js \bmod s)}) \bmod n) = ((M_{js} + k_0) \bmod n)$$

for $0 \le j \le q-1$.

Using the method of frequency analysis on each subset of ciphertext letters, we can then determine the most likely values for $k_0, k_1, \ldots, k_{s-1}$.

Example 8.5.4 Suppose Alice and Bob are using the Vigenère cipher with a key length of 2. Malice knows the key length and has obtained 290 characters of ciphertext $C =$

```
KUFEXBGDWRHBLUYUWXDJLMDIVJDDGYQWWXHHHYQJKUZYWSKIZYQTRMDDGYZQY
UGLHHBIOEZBBQWXLCDDGXHMDLHTYUUOVBRMOODJPURKUMDLLDJIHUPUGJRRHL
```

```
HHBTLIWQQJWHDLHBOYQWIHRCRKUQUCVBLAHJZESURFOUZQYYQWDJHQFXRJKUU
YQTLVIUUUQJFYWYHISUUXDFVRHJZUHDWQFEPQDDGIDBHCDDGEXHZQYYQWZQVC
HHHBBQQUFXREIJKULHZQYYQWDSUEVIWXRKVQQTVEICLBHI
```

In this case, $s = 2$, $q = 145$, and there are 2 subsets of ciphertext: $C_0C_2C_4 \ldots C_{288} =$

```
KFXGWHLYWDLDVDGQWHHQKZWKZQRDGZYGHBOZBWLDGHDHYUVRODPRUD
LJHPGRHHBLWQWDHOQIRRUUVLHZSROZYQDHFRKUQLIUQFWHSUDVHZHW
FPDGDHDGXZYQZVHHBQFRIKLZYQDUVWRVQVILH
```

and $C_1C_3C_5 \ldots C_{289} =$

```
UEBDRBUUXJMIJDYWXHYJUYSIYTMDYQULHIEBQXCDXMLTUOBMOJUKML
DIUUJRLHTIQJHLBYWHCKQCBAJEUFUQYWJQXJUYTVUUJYYIUXFRJUDQ
EQDIBCDEHQYWQCHBQUXEJUHQYWSEIXKQTECBI
```

In the subset $C_0C_2 \ldots C_{288}$, the letter H occurs most often (19 times), and so we deduce that the encryption transformation takes E to H, thus $k_0 = 3$. (One could also obtain $k_0 = 3$ using key trial on the subset $C_0C_2 \ldots C_{288}$.)

Likewise, in the subset $C_1C_3 \ldots C_{289}$, the letter U occurs most often, and so we deduce that the encryption transformation takes E to U, thus $k_1 = 16$.

The correct key is indeed $k = (3, 16)$ and we obtain $M = d(C, (3, 16)) =$

```
    He couldn't believe that I was standing there in the
witch's window and I waved very slowly at him and he waved
very slowly at me. Our waving seemed to be very distant
travelling from our arms like two people waving at each other
in different cities, perhaps between Tacoma and Salem, and
our waving was merely an echo of their waving across
thousands of miles.
```

Richard Brautigan, from *1692 Cotton Mather Newsreel*

□

Key Length Is Not Known

Now suppose that message M is encrypted using the Vigenère cipher with key k. Suppose that Malice or an attacker *does not* know the length of the key. The first task for the attacker is to find the length of the key. To this end, we use the **Kasiski method**.

We take advantage of the fact that certain 2-grams appear relatively frequently in plaintext English. For instance the 2-gram TH appears with probability ≈0.03 (see Figure 3.4).

If the ratio of key length to the length of the plaintext message is small enough, then it is likely that some occurrences of common 2-grams (e.g., TH) in the plaintext will coincide with the same letters in the key. When this happens the gcd of the distances between the occurrences is the key length.

With this in mind, we look for 2-grams in the ciphertext that appear relatively often and compute the gcd of the distances between their occurrences. This value is a good guess for the key length.

Example 8.5.5 In the ciphertext of Example 8.5.4 the 2-gram `WX` appears 4 times; the distances between occurrences are 16, 42, 58. We have

$$\gcd(16, 42) = \gcd(42, 58) = 2, \quad \gcd(16, 58) = 8,$$

and so it is likely that the key length is 2, which is correct.

Once the key length has been established the attacker proceeds exactly as in the "Key Length Is Known" case above. □

Our next polyalphabetic symmetric cryptosystem is a special case of the Vigenère cipher.

8.5.4 The Vernam Cipher

Let $\Sigma = \{0, 1\}$ denote the set of 2 letters (bits), and let $\Sigma^* = \{0, 1\}^*$ denote the set of all finite sequences of 0's and 1's. Let $a = a_1a_2 \cdots a_m$, $b = b_1b_2 \cdots b_m \in \{0, 1\}^*$. Then **bit-wise addition** modulo 2 is defined as

$$a \oplus b = c_1c_2 \cdots c_m,$$

where

$$c_i = ((a_i + b_i) \bmod 2).$$

Definition 8.5.6 (Vernam Cipher) Let $\Sigma = \{0, 1\}$ and $\mathcal{M} = \mathcal{C} = \{0, 1\}^*$. A message is a finite sequence of bits

$$M = M_0M_1M_2 \cdots M_{r-1}, \quad M_i \in \{0, 1\}.$$

The key for encryption and decryption is a sequence of r letters

$$k = k_0k_1k_2 \cdots k_{r-1},$$

chosen uniformly at random from the set $\{0, 1\}$ and shared as a secret key by Alice and Bob. It is important to note that the key must be the same length r as the message. The encryption of M is given as

$$C = e(M, k) = M \oplus k.$$

Decryption proceeds as follows:

$$M = d(C, k) = C \oplus k.$$

The Vernam cipher works since

$$\begin{aligned} d(e(M, k), k) &= d(M \oplus k, k) \\ &= (M \oplus k) \oplus k \\ &= M \oplus (k \oplus k) \\ &= M. \end{aligned}$$

Here is an example of a Vernam cipher.

Example 8.5.7 We begin with the familiar correspondence:

$$\texttt{A} \leftrightarrow 0,\ \ \texttt{B} \leftrightarrow 1,\ \ \texttt{C} \leftrightarrow 2,\ \ \texttt{D} \leftrightarrow 3,\ \ldots,\ \texttt{Z} \leftrightarrow 25.$$

We then write the numbers in 5-bit binary, giving the encoding

$$\texttt{A} \leftrightarrow 00000,\ \ \texttt{B} \leftrightarrow 00001,\ \ \texttt{C} \leftrightarrow 00010,\ \ \texttt{D} \leftrightarrow 00011,\ \ldots,\ \texttt{Z} \leftrightarrow 11001.$$

Now, for instance,

$$\texttt{C A T} \leftrightarrow 00010\ 00000\ 10011$$

Alice and Bob have established the shared secret key

$$k = 11100\ 10100\ 10010,$$

where each bit is chosen uniformly at random from {0, 1}, and so the encryption of C A T is

$$\begin{aligned} C &= e(00010\ 00000\ 10011, 11100\ 10100\ 10010) \\ &= 00010\ 00000\ 10011 \oplus 11100\ 10100\ 10010 \\ &= 11110\ 10100\ 00001. \end{aligned}$$

Note: C may not correspond to a sequence of ordinary alphabet letters. □

The Vernam cipher is polyalphabetic; we have

$$\Pr[e(0, k) = 0] = \frac{1}{2}, \quad \Pr[e(0, k) = 1] = \frac{1}{2}.$$

Perfect Secrecy

A cryptosystem $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ provides "perfect secrecy" if knowledge of C gives no new, additional information about the message M. As we shall see, the Vernam cipher has perfect secrecy. Here is a formal definition.

Let $(\Omega, \mathcal{A}, \Pr)$ be an abstract probability space, let $X_{\mathcal{M}} : \Omega \to \mathcal{M}$, $X_{\mathcal{C}} : \Omega \to \mathcal{C}$ denote random variables, where

$$(X_{\mathcal{M}} = M) = \{\omega \in \Omega : \ X_{\mathcal{M}}(\omega) = M\}$$

denotes the event: Plaintext message $M \in \mathcal{M}$ is sent, with probability $\Pr(X_{\mathcal{M}} = M)$, i.e., $\Pr(X_{\mathcal{M}} = M)$ is the probability that the plaintext message is M. Likewise,

$$(X_{\mathcal{C}} = C) = \{\omega \in \Omega : \ X_{\mathcal{C}}(\omega) = C\}$$

denotes the event: Ciphertext $C \in \mathcal{C}$ is received, with probability $\Pr(X_{\mathcal{C}} = C)$, i.e., $\Pr(X_{\mathcal{C}} = C)$ is the probability that the ciphertext is C.

Definition 8.5.8 A cryptosystem has **perfect secrecy** if intercepted ciphertext reveals no *new* information about the corresponding plaintext. More precisely, a cryptosystem has perfect secrecy if

$$\Pr(X_{\mathcal{M}} = M | X_{\mathcal{C}} = C) = \Pr(X_{\mathcal{M}} = M)$$

for all $M \in \mathcal{M}$, $C \in \mathcal{C}$.

Proposition 8.5.9 *The Vernam cipher has perfect secrecy.*

Proof Let $k = k_0k_1 \dots k_{r-1}$ be the random key and let $M = M_0M_1 \dots M_{r-1}$ be a message. Intuitively, the ciphertext $C = e(M, k) = M \oplus k$, which is given by bit-wise addition modulo 2, is just as random as k. Thus knowledge of C gives no new information about the message M, i.e., for any M, C, the events $(X_{\mathcal{M}} = M)$ and $(X_{\mathcal{C}} = C)$ are independent. The result follows. □

8.5.5 Unicity Distance of the Vernam Cipher

In the Vernam cipher, $|\mathcal{K}| = \infty$. To see this, suppose that $|\mathcal{K}| = n < \infty$. Necessarily, $n = 2^m$ for some $m \geq 1$. Then the length of any message must be $\leq m$ since the length of the key must equal the length of the message. But one can always write a message of length $> m$. Thus $|\mathcal{K}| = \infty$.

Now assume that English messages are encoded in ASCII. As shown in Section 3.2.1, the redundancy rate per byte is 6.5. Thus the redundancy rate per bit is $\frac{6.5}{8} = 0.8125$.

Thus the unicity distance of the Vernam cipher is

$$\begin{aligned} \text{u.d.} &= \frac{\log_2(|\mathcal{K}|)}{0.8125} \\ &= \frac{\infty}{0.8125} \\ &= \infty \text{ characters.} \end{aligned}$$

This result is consistent with perfect secrecy: A brute-force attack by key trial cannot be used to uniquely determine the key.

8.6 Stream Ciphers

Definition 8.6.1 (Stream Cipher) Let $\Sigma = \{0, 1\}$ and $\mathcal{M} = \mathcal{C} = \{0, 1\}^*$. A message is a finite sequence of bits

$$M = M_0 M_1 M_2 \cdots M_{m-1}, \quad M_i \in \{0, 1\}$$

of length m. The secret key shared by Alice and Bob is a sequence of l bits

$$k = k_0 k_1 k_2 \cdots k_{l-1},$$

chosen uniformly at random from the set {0, 1}. Alice and Bob use this random “seed" k to generate a longer sequence of $m \geq l$ bits,

$$b = b_0 b_1 b_2 \ldots b_{m-1}$$

using a deterministic process; b is the **key stream**. The encryption of M is given as

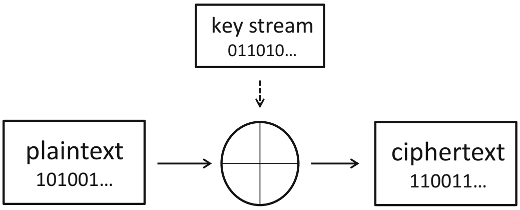
$$C = e(M, b) = M \oplus b,$$

and decryption is

$$M = d(C, b) = C \oplus b.$$

(See Figure 8.3).

Fig. 8.3 A stream cipher



To generate the keystream b from the random seed k, Alice and Bob use a function

$$G : \{0, 1\}^l \to \{0, 1\}^m, \quad k \mapsto b,$$

called a **bit generator**. Hence b is not random. Security of a stream cipher depends on how well the output $b = G(k)$ of the bit generator simulates a truly random stream of bits. We will discuss bit generators in detail in Chapter 11.

Example 8.6.2 Define a bit generator $G : \{0, 1\}^3 \to \{0, 1\}^{12}$ by the rule $G(k) = b = k^4 = kkkk$. If the shared random seed is $k = 101$, then $G(101) = 101101101101$ and the encryption of the message $M = 100101110001$ is

$$C = 100101110001 \oplus 101101101101 = 001000011100.$$

□

Example 8.6.2 shows that the Vigenère cipher with alphabet $\Sigma = \{0, 1\}$ is a special case of the stream cipher.

8.7 Block Ciphers

As we saw in Section 8.3.1, the Hill cipher is a block cipher since it encrypts blocks of 2-grams over the alphabet $\{\mathtt{A}, \mathtt{B}, \ldots, \mathtt{Z}\}$. In this section we discuss block ciphers over the alphabet of bits.

Let $m \geq 1$, $n \geq 1$ be integers. Let $\{0, 1\}^m$ denote the set of all sequences of 0s and 1s of length m, and let $\{0, 1\}^n$ denote the set of all sequences of 0s and 1s of length n.

Definition 8.7.1 A **block cipher** is a symmetric key cryptosystem whose encryption transformation e is a function

$$e : \{0, 1\}^m \times \{0, 1\}^n \to \{0, 1\}^m.$$

Here $\mathcal{M} = \mathcal{C} = \{0, 1\}^m$. Thus a message $M \in \mathcal{M}$ is a sequence of bits of length m. Note that the keyspace $\mathcal{K} = \{0, 1\}^n$; a typical key k is a sequence of n bits.

Decryption is a function d

$$d : \{0, 1\}^m \times \{0, 1\}^n \to \{0, 1\}^m.$$

A block cipher satisfies the fundamental relation:

$$d(e(M, k), k) = M,$$

for $M \in \{0, 1\}^m$, $k \in \{0, 1\}^n$.

8.7.1 Iterated Block Ciphers

An **iterated block cipher** is a block cipher in which encryption is achieved by performing a finite sequence of **rounds**. Each round involves the application of a function called the **round function**.

Feistel Ciphers

A **Feistel cipher** with r rounds is an iterated block cipher in which $\mathcal{M} = \mathcal{C} = \{0, 1\}^{2t}$, where $t \geq 1$. Thus both the message space and the ciphertext space consist of sequences of even length $2t$. The keyspace is of the form $\mathcal{K} = \{0, 1\}^t$. The key $k \in \{0, 1\}^t$ is used in some prescribed way to generate a set of r **round keys** k_i, each of length t: $k_1, k_2, k_3, \ldots, k_r$. The round function has the form

$$f : \{0, 1\}^t \times \{0, 1\}^t \to \{0, 1\}^t.$$

Encryption in a Feistel cipher proceeds as follows.

Step 1. Message $M \in \{0, 1\}^{2t}$ is broken into two blocks of length t:

$$M = (L_0, R_0),$$

where L_0 is the left block and R_0 is the right block.

Step 2. Beginning with the initial blocks (L_0, R_0), each of the r rounds produces a new pair of blocks, in some prescribed manner:

$$(R_0, L_0) \xrightarrow{\text{round } 1} (L_1, R_1) \xrightarrow{\text{round } 2} (L_2, R_2) \cdots \xrightarrow{\text{round } r} (L_r, R_r).$$

Step 3. The final pair of blocks $C = (L_r, R_r)$ is the encryption of message $M = (L_0, R_0)$.

In round i, $1 \leq i \leq r$, new blocks are produced from previous blocks according to formula:

$$\begin{cases} L_i = R_{i-1} \\ R_i = L_{i-1} \oplus f(R_{i-1}, k_i). \end{cases}$$

The ith round of the Feistel cipher is illustrated in Figure 8.4.

Decryption of the ciphertext $C = (L_r, R_r)$ proceeds by reversing the r rounds to obtain $M = (L_0, R_0)$:

$$(R_r, L_r) \xrightarrow{\text{round } 1} (L_{r-1}, R_{r-1}) \cdots \xrightarrow{\text{round } r-1} (L_1, R_1) \xrightarrow{\text{round } r} (L_0, R_0).$$

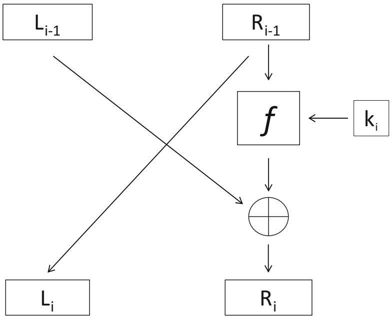


Fig. 8.4 *i*th round of the Feistel cipher

The *i*th round of decryption yields

$$\begin{cases} L_{r-i} = R_{r-i+1} \oplus f(R_{r-i}, k_{r-i+1}) \\ R_{r-i} = L_{r-i+1}. \end{cases}$$

Example 8.7.2 In this 16-bit 2 round Feistel cipher, the encryption transformation is

$$e : \{0, 1\}^{16} \times \{0, 1\}^8 \to \{0, 1\}^{16}.$$

Assume that the key $k \in \{0, 1\}^8$ generates round keys

$$k_1 = 01000111, \qquad k_2 = 11100101,$$

and the round function

$$f : \{0, 1\}^8 \times \{0, 1\}^8 \to \{0, 1\}^8$$

is defined as $(a, b) \mapsto a \oplus b$. Let $M = \texttt{AA}$, which in ASCII appears as

$$0100000101000001.$$

We compute $C = e(0100000101000001, k)$. Note that

$$L_0 = 01000001, \qquad R_0 = 01000001.$$

In round 1,

$$L_1 = R_0 = 01000001,$$

and

$$\begin{aligned} R_1 &= L_0 \oplus f(R_0, k_1) \\ &= 01000001 \oplus f(01000001, 01000111) \\ &= 01000001 \oplus (01000001 \oplus 01000111) \\ &= 01000001 \oplus 00000110 \\ &= 01000111. \end{aligned}$$

In round 2,

$$L_2 = R_1 = 01000111.$$

and

$$\begin{aligned} R_2 &= L_1 \oplus f(R_1, k_2) \\ &= 01000001 \oplus f(01000111, 11100101) \\ &= 01000001 \oplus (01000111 \oplus 11100101) \\ &= 01000001 \oplus 10100010 \\ &= 11100011. \end{aligned}$$

Thus

$$C = e(M, k) = (L_2, R_2) = 0100011111100011.$$

□

The Data Encryption Standard (DES)

First devised in 1977, the **Data Encryption Standard (DES)** was the most widely used and well-known symmetric key cryptosystem of the modern era. The DES is a 16-round iterated block cipher with $\mathcal{M} = \mathcal{C} = \{0, 1\}^{64}$. DES is essentially a Feistel cipher. However in DES the key k is an element of $\{0, 1\}^{64}$, which is used to generate 16 round keys, each of which is an element of $\{0, 1\}^{48}$. Moreover, in DES the round function is of the form

$$f : \{0, 1\}^{32} \times \{0, 1\}^{48} \to \{0, 1\}^{32}.$$

The Advanced Encryption Standard (AES)

Due to security concerns with DES, the **Advanced Encryption Standard (AES)** was proposed by V. Rijmen and J. Daemen in the 1990s. In 2002, AES was accepted by the US Government as the new standard for symmetric key encryption. The AES is a 10-round iterated block cipher with $\mathcal{M} = \mathcal{C} = \{0, 1\}^{128}$. The key k is an element of $\{0, 1\}^{128}$ and is used to generate 10 round keys (as in DES).

8.8 Exercises

1. Let $\{\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{26!}\}$ be the set of permutations of the alphabet

$$A = \{\mathtt{A}, \mathtt{B}, \mathtt{C}, \mathtt{D}, \mathtt{E}, \mathtt{F}, \mathtt{G}, \mathtt{H}, \mathtt{I}, \mathtt{J}, \mathtt{K}, \mathtt{L}, \mathtt{M}, \mathtt{N}, \mathtt{O}, \mathtt{P}, \mathtt{Q}, \mathtt{R}, \mathtt{S}, \mathtt{T}, \mathtt{U}, \mathtt{V}, \mathtt{W}, \mathtt{X}, \mathtt{Y}, \mathtt{Z}\},$$

and consider the simple substitution cryptosystem with

$$\sigma_k = \left(\begin{array}{ccccccccccccc} \mathtt{A} & \mathtt{B} & \mathtt{C} & \mathtt{D} & \mathtt{E} & \mathtt{F} & \mathtt{G} & \mathtt{H} & \mathtt{I} & \mathtt{J} & \mathtt{K} & \mathtt{L} & \mathtt{M} \\ \mathtt{X} & \mathtt{M} & \mathtt{T} & \mathtt{A} & \mathtt{K} & \mathtt{Z} & \mathtt{Q} & \mathtt{B} & \mathtt{N} & \mathtt{O} & \mathtt{L} & \mathtt{E} & \mathtt{S} \end{array}\right.$$
$$\left.\begin{array}{ccccccccccccc} \mathtt{N} & \mathtt{O} & \mathtt{P} & \mathtt{Q} & \mathtt{R} & \mathtt{S} & \mathtt{T} & \mathtt{U} & \mathtt{V} & \mathtt{W} & \mathtt{X} & \mathtt{Y} & \mathtt{Z} \\ \mathtt{I} & \mathtt{F} & \mathtt{G} & \mathtt{R} & \mathtt{V} & \mathtt{J} & \mathtt{H} & \mathtt{C} & \mathtt{Y} & \mathtt{U} & \mathtt{D} & \mathtt{P} & \mathtt{W} \end{array}\right)$$

and key k.

(a) Compute $C = e(\texttt{AUBUR NUNIV ERSIT YATMO NTGOM ERY}, k)$.
(b) Compute $M = d(\texttt{NIZFV SXHNF IHBKF VPXIA KIHVF GP}, k)$.

2. Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ denote the simple substitution cryptosystem with $\Sigma = \{0, 1, 2, 3, 4\}$, $\mathcal{M} = \mathcal{C} = \Sigma^* = \{0, 1, 2, 3, 4\}^*$, and

$$\sigma_k = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 4 & 3 & 0 & 2 & 1 \end{pmatrix}.$$

(a) Compute $C = e(240014, k)$.
(b) Compute $M = d(4130, k)$.

3. Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ denote the affine cipher with

$$\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\},$$

$$\mathcal{M} = \mathcal{C} = \Sigma^* = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\}^*,$$

and key (a, b).

(a) Compute $C = e(1\ 7\ 10, (7, 3))$.

(b) Compute $M = d(12\ 9, (3, 10))$.

4. Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ denote the shift cipher with

$$\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

and $\mathcal{M} = \mathcal{C} = \Sigma^* = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}^*$.

(a) Compute $C = e(7280, 8)$.
(b) Compute $M = d(22109, 5)$.

5. Suppose $\Sigma = \{0, 1, 2, 3, \ldots, 34\}$ is the alphabet for the affine cipher. Find the size of the keyspace.

6. Assume that Alice and Bob are using the Hill cipher with key $A = \begin{pmatrix} 2 & 1 \\ 3 & 1 \end{pmatrix}$.

(a) Verify that A is a valid key.
(b) Compute $M = d(\mathtt{WD}, A)$.
(c) Show that there is at least one spurious key for $C = \mathtt{WD}$.

7. Consider the following generalization of the Hill 2×2 cipher in which $\Sigma = L_3$, the collection of all 3-grams over the ordinary alphabet $\{\mathtt{A}, \mathtt{B}, \ldots, \mathtt{Z}\}$. Messages are finite sequences of 3-grams,

$$M = M_0 M_1 \cdots M_{r-1},$$

where $M_i = \big(m_{i,1}\ m_{i,2}\ m_{i,3}\big)$, $0 \le i \le r-1$, $m_{i,j} \in \mathbb{Z}_{26}$. The encryption and decryption keyspace is $\mathrm{GL}_3(\mathbb{Z}_{26})$; encryption is given as

$$C_i = e(M_i, A) = (AM_i^T)^T.$$

(a) Compute $C = e(\mathtt{CAT}, A)$, where $A = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}$.
(b) Compute the size of the keyspace.
(c) Compute the redundancy rate of plaintext English.
(d) Compute the unicity distance of the cryptosystem.

8. Suppose Malice intercepts the ciphertext

$C =$

```
LW VKRXBG EH REVHUYHG WKDW WKH HTXLYDBHQFHV VKRZQ LQ
FKDSWHU WZR EHWZHHQ WKH YDULRXV CRGHBV RI ILQLWH DXWRCDWD
DQG UHJXBDU HNSUHVVLRQV ZHUH HIIHFWLYH HTXLYDBHQFHV LQ
WKH VHQVH WKDW DBJRULWKCV ZHUH JLYHQ WR WUDQVBDWH IURC
RQH UHSUHVHQWDWLRQ WR DQRWKHU
```

which is known to have been encrypted using a simple substitution cryptosystem with permutation σ_k. Using frequency analysis, determine which of the following values of $\sigma_k(\mathtt{T})$ is most likely to have been used for encryption.

(a) $\sigma_k(\mathtt{T}) = \mathtt{N}$.
(b) $\sigma_k(\mathtt{T}) = \mathtt{W}$.
(c) $\sigma_k(\mathtt{T}) = \mathtt{X}$.

9. Referring to Exercise 8, use frequency analysis to decrypt C.
10. Suppose that Alice and Bob are using the simple substitution cryptosystem on the standard 26 letter alphabet and Malice has obtained 10 characters of ciphertext in a transmission between Alice and Bob. Malice uses the brute-force method of key trial to obtain the key. (Assume that Malice has unlimited computing power.) Prove that there is at least one spurious key for the ciphertext.
11. Assume that the alphabet $\{\mathtt{A}, \mathtt{B}, \dots, \mathtt{Z}\}$ is encoded in ASCII as bytes 01000001, 01000010, ..., 01011010.
12. Compute the redundancy per letter (per byte) of plaintext English with respect to this encoding. How does the redundancy rate compare to the standard value of 3.2 bits/letter?
13. Let M be the *Preface* paragraph from Section 8.4. Let $g : L_1 \to [0, 1]$ be the 1-gram relative frequency distribution of M and let $f_1 : L_1 \to [0, 1]$ be the accepted 1-gram relative frequency distribution of English (Figure 3.3).

    Compute the total variation distance

$$D_g = \frac{1}{2} \sum_{\alpha \in \Sigma} |g(\alpha) - f_1(\alpha)|.$$

Compare D to the total variation distance

$$D_h = \frac{1}{2} \sum_{\alpha \in \Sigma} |h(\alpha) - f_1(\alpha)|$$

where $h : L_1 \to [0, 1]$ is the relative frequency distribution of the atomic elements (see Figure 8.2).

14. Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ denote the Vigenère cipher with $\Sigma = \{0, 1, 2, \dots, 25\}$ and $\mathcal{M} = \mathcal{C} = \Sigma^* = \{0, 1, 2, \dots, 25\}^*$.

(a) Compute $C = e(4\ 17\ 19\ 14\ 22\ 4\ 17, 20\ 18\ 6)$.
(b) Compute $M = d(13\ 20\ 3\ 3\ 2\ 3, 2\ 12)$.

15. Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ denote the Vigenère cipher with $\Sigma = \{0, 1, 2, 3, 4\}$ and $\mathcal{M} = \mathcal{C} = \Sigma^* = \{0, 1, 2, 3, 4\}^*$. It is known that 3124/1200 is a plaintext/ciphertext pairing.

(a) Under the assumption that the key is $k = k_0k_1$, compute $M = d(112, k_0k_1)$.
(b) Prove that the key cannot have length 3.

16. Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ denote the Vernam cipher with $\Sigma = \{0, 1\}$ and $\mathcal{M} = \mathcal{C} = \Sigma^* = \{0, 1\}^*$.

    (a) Compute $C = e(01100\ 00000\ 11000, 10110\ 00110\ 10101)$.
    (b) Compute $M = d(10010\ 00010, 10101\ 01010)$.

17. Consider the 16-bit 2 round Feistel cipher with $\Sigma = \{0, 1\}$ and $\mathcal{M} = \mathcal{C} = \Sigma^{16} = \{0, 1\}^{16}$. Assume that the key k generates the round keys k_1, k_2. The round function

$$f : \{0, 1\}^8 \times \{0, 1\}^8 \to \{0, 1\}^8$$

    is defined as $f(a, b) = a \oplus b$.

    (a) Compute $C = e(01000110\ 01001100, k)$, with $k_1 = 10000000$, $k_2 = 00000001$.
    (b) Compute $M = d(00000000\ 01011010, k)$, with $k_1 = 00000000$, $k_2 = 00000000$.

Chapter 9
Public Key Cryptography

9.1 Introduction to Public Key Cryptography

As we have seen, a cryptosystem is a system of the form

$$\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle,$$

where $\mathcal{M}$ is the message space, $\mathcal{C}$ is the space of all possible cryptograms, e is the encryption transformation, d is the decryption transformation, $\mathcal{K}_e$ is the encryption keyspace, and $\mathcal{K}_d$ is the decryption keyspace.

More formally, the encryption transformation is a function

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

Indeed, for message $M \in \mathcal{M}$ and encryption key $k_e \in \mathcal{K}_e$,

$$e(M, k_e) = C \in \mathcal{C}.$$

The decryption transformation is a function

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

Given the ciphertext $C = e(M, k_e)$, there is a decryption key $k_d \in \mathcal{K}_d$ so that

$$d(e(M, k_e), k_d) = M.$$

If $k = k_e = k_d$ is a shared secret key, then the cryptosystem is a symmetric key cryptosystem. In a symmetric key cryptosystem, once we know the encryption transformation e and the encryption key k, it is "easy" to compute the ciphertext $e(M, k)$, i.e., $e(M, k)$ can be computed using a polynomial time algorithm.

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_9

Likewise, if we know the key k (and hence k_d), then it is "easy" to invert $e(M, k)$, i.e., given the ciphertext $C = e(M, k)$, there is a polynomial time algorithm that computes the unique M for which $e(M, k) = C$. One has $M = d(C, k) = d(e(M, k), k)$.

In public key cryptography, we use a different kind of encryption transformation e. As in a symmetric key cryptosystem, it should be easy to compute $C = e(M, k_e)$ knowing k_e, but unlike a symmetric key cryptosystem, it should be hard to invert *e even with knowledge of k_e*. In fact, the encryption key k_e is made public and is called the **public key**; it is known to everyone (including Malice!). It should be easy however to invert $e(M, k_e)$ if one knows an additional piece of information: the decryption key k_d, which is called the **trapdoor** or **private key**.

Security in a public key cryptosystem depends on the secrecy of k_d. It is essential therefore that $k_e \neq k_d$. For this reason, public key cryptosystems are also called **asymmetric key cryptosystems**.

The type of encryption function that we *want* to use in a public key cryptosystem is called a "one-way trapdoor" function. In order to define such a function, we introduce the idea of a "negligible" function.

9.1.1 Negligible Functions

Let $\mathbb{R}_+ = \{x \in \mathbb{R} : x > 0\}$.

Definition 9.1.1 A function $r : \mathbb{R}_+ \rightarrow \mathbb{R}_+$ is **negligible** if for each positive polynomial $w(x) \in \mathbb{Z}[x]$, there exists an integer $l_0 \geq 1$ for which $r(l) < \frac{1}{w(l)}$ whenever $l \geq l_0$.

For example, $r(x) = \frac{1}{2^x}$ is negligible since for any positive polynomial $w(x)$, there exists l_0 for which $w(l)/2^l < 1$ whenever $l \geq l_0$. We have

$$\lim_{x\to\infty} \frac{\frac{1}{2^x}}{\frac{1}{w(x)}} = \lim_{x\to\infty} \frac{w(x)}{2^x} = 0,$$

by L'Hôpital's rule. A negligible function goes to 0 faster than the reciprocal of any positive polynomial.

The concept of a negligible function is related to the notion that efficient algorithms run in polynomial time. If algorithm A runs in polynomial time as a function of input size l, then repeating A $w(l)$ times, where w is a positive polynomial, results in a new algorithm that also runs in polynomial time and hence is efficient. But when the probability that an algorithm successfully computes a function value is a negligible function of l, then repeating it a polynomial number of times will not change that fact.

Proposition 9.1.2 *Let w be a positive polynomial. Suppose that the probability that algorithm A correctly computes a function value $f(n)$ for some instance n of size*

l is a negligible function of l. Then the probability that A correctly computes $f(n)$ at least once when repeated $w(l)$ times is also negligible, i.e., the probability is a negligible function of l.

Proof Let $P = \Pr(A(n) = f(n))$ for some input n of size l. Let $\Pr(\xi_{w(l)} = i)$ denote the probability that A correctly computes the function value exactly i times when A is repeated $w(l)$ times, cf. Section 2.4, Example 2.4.2.

Then, the probability that A correctly computes the function value at least once when repeated $w(l)$ times is

$$\Pr(\xi_{w(l)} \geq 1) = \sum_{i=1}^{w(l)} \Pr(\xi_{w(l)} = i) = \sum_{i=1}^{w(l)} \binom{w(l)}{i} P^i (1-P)^{w(l)-i}.$$

We claim that $\Pr(\xi_{w(l)} \geq 1)$ is a negligible function of l. To this end, observe that

$$w(l)P \geq \sum_{i=1}^{w(l)} \binom{w(l)}{i} P^i (1-P)^{w(l)-i},$$

thus

$$w(l)P \geq \Pr(\xi_{w(l)} \geq 1). \tag{9.1}$$

By way of contradiction, suppose that $\Pr(\xi_{w(l)} \geq 1)$ is not negligible. Then there exists a positive polynomial $v(x)$ for which

$$\Pr(\xi_{w(l)} \geq 1) \geq \frac{1}{v(l)}$$

for infinite l. Then by (9.1), we have

$$P \geq \frac{1}{w(l)v(l)},$$

for infinite l. Since $w(x)v(x)$ is a positive polynomial, this implies that P is not negligible, a contradiction. □

9.1.2 One-Way Trapdoor Functions

Definition 9.1.3 An encryption function $e(x, k_e) : \mathcal{M} \to \mathcal{C}$ is a **one-way trapdoor** function if

(i) $e(M, k_e)$ can be computed by a polynomial time algorithm.

(ii) Given a positive polynomial w and a probabilistic polynomial time algorithm A, with input $C = e(M, k_e)$, where M is a random element of $\mathcal{M}$ of size l, we have

$$\Pr(A(e(M, k_e)) = M) < \frac{1}{w(l)},$$

for l sufficiently large. Less formally, the probability that A successfully inverts the function $e(x, k_e)$ is a negligible function of l, *even with knowledge of* e *and* k_e.

(iii) However, with an additional piece of information (the trapdoor k_d), $e(M, k_e)$ can be inverted in polynomial time.

□

Unfortunately, it has not been proven that one-way trapdoor functions exist. In fact, the existence of such functions is related to deep questions in complexity theory, see [59, Theorem 6.6].

Nevertheless, there are some very good candidates for one-way trapdoor functions that are used to construct public key cryptosystems. The inputs for these potential one-way functions will be integers; the size of the integers will be measured in bits.

Let $l \geq 2$. An l-**bit prime** is a prime number p with $2^{l-1} + 1 \leq p \leq 2^l - 1$. For such primes, $l = \lfloor \log_2(p) \rfloor + 1$ and so p requires l bits to represent it in binary; the prime p is of size l.

For example, the 5-bit primes are primes that satisfy $17 \leq p \leq 31$, and thus, they are 17, 19, 23, 29, 31. Note that $(17)_2 = 10001$, $(19)_2 = 10011$, $(23)_2 = 10111$, $(29)_2 = 11101$, and $(31)_2 = 11111$.

A prime p is **Mersenne** if $p = 2^l - 1$ for $l \geq 2$. If $p = 2^l - 1$ is Mersenne, then l is prime. The binary representation of a Mersenne prime $2^l - 1$ is 1^l; a Mersenne prime is an l-bit prime.

Since there are an infinite number of primes (see [47, Theorem 3.1]), there are l-bit primes for arbitrarily large $l \geq 2$.

Here is our first candidate for a one-way trapdoor function.

Let $l \geq 1$. Let p and q be distinct primes in which the smaller prime is an l-bit prime. Let s be an integer with the following properties: $1 < s < (p-1)(q-1)$ and $\gcd(s, (p-1)(q-1)) = 1$. Let $n = pq$, and let $\Sigma = \{0, 1, 2, 3, \dots, n-1\}$ denote the set of n letters. Define a function $e(x, (s, n)) : \Sigma \rightarrow \Sigma$ by the rule

$$e(x, (s, n)) = (x^s \bmod n),$$

for $x \in \Sigma$.

Then $e(x, (s, n))$ is a possible one-way trapdoor function. It can be shown that conditions (i) and (iii) of Definition 9.1.3 hold:

(i) $e(x, (s, n))$ can be computed by a polynomial time algorithm.

(iii) $e(x, (s, n))$ can be inverted in polynomial time with the trapdoor.

Moreover, it is *assumed* that condition (ii) of Definition 9.1.3 holds (see Section 9.3):

(ii) Given a positive polynomial w and a probabilistic polynomial time algorithm A with input $e(x, (s, n))$, $x \in_R \Sigma$, and output $y \in \Sigma$, we assume that

$$\Pr(A(e(x, (s, n))) = x) < \frac{1}{w(l)}$$

for l sufficiently large, i.e., we assume the probability that A successfully inverts $e(x, (s, n)) = (x^s \bmod n)$ is negligible, i.e., is a negligible function of l, *even with knowledge of* s, n.

Note: here and elsewhere, the notation $x \in_R S$ means that the element x is chosen uniformly at random from the set S.

The function $e(x, (s, n)) : \Sigma \to \Sigma$ is the basis for our first example of a public key cryptosystem.

9.2 The RSA Public Key Cryptosystem

Definition 9.2.1 (RSA Public Key Cryptosystem) Alice wants to send a secret message to Bob. Let p and q be distinct large prime numbers, and let s be an integer with

$$1 < s < (p-1)(q-1) \quad \text{and} \quad \gcd(s, (p-1)(q-1)) = 1.$$

Let $n = pq$, and let $\Sigma = \{0, 1, 2, 3, \ldots, n-1\}$ denote the set of n letters. We let $\mathcal{M} = \mathcal{C} = \Sigma^*$. A message $M \in \mathcal{M}$ is a finite sequence of letters in Σ. The pair (s, n) is Bob's public key (the encryption key) which he publishes for all to see.

Alice looks up Bob's public key (s, n) and encrypts the message $M = M_0M_1M_2 \cdots M_{r-1}$ letter by letter to form the ciphertext $C = C_0C_1C_2 \cdots C_{r-1}$ using the rule

$$C_i = e(M_i, (s, n)) = (M_i^s \bmod n)$$

for $0 \le i \le r-1$. She then sends the ciphertext C to Bob who will decrypt.

Bob's private key (the trapdoor) is the unique integer t with the following properties:

$$1 < t < (p-1)(q-1) \quad \text{and} \quad st \equiv 1 \pmod{(p-1)(q-1)}.$$

Decryption of the ciphertext $C = C_0C_1 \cdots C_{r-1}$ proceeds letter by letter using the formula

$$M_i = d(C_i, t) = (C_i^t \bmod n)$$

for $0 \le i \le r - 1$.

This cryptosystem is the **RSA public key cryptosystem**. □

RSA is so named since it was developed by R. Rivest, A. Shamir, and L. Adleman in 1977. Today, RSA is by far the most widely used public key cryptosystem in the world.

Proposition 9.2.2 *The RSA cryptosystem works.*

Proof We show that (8.1) holds. Let $M \in \Sigma$. Then

$$\begin{aligned} d(e(M, (s, n)), t) &= d((M^s \bmod n), t) \\ &= ((M^s \bmod n)^t \bmod n) \\ &= ((M^{st} \bmod n) \bmod n) \\ &= (M^{st} \bmod n). \end{aligned}$$

There exists an integer a with $st = 1 + a(p-1)(q-1)$, thus

$$M^{st} \equiv MM^{a(p-1)(q-1)} \equiv M \pmod{p},$$

$$M^{st} \equiv MM^{a(p-1)(q-1)} \equiv M \pmod{q}$$

by Fermat's Little Theorem.

Now both M and M^{st} are solutions to the system of congruences

$$\begin{cases} x \equiv M \pmod{p} \\ x \equiv M \pmod{q}. \end{cases}$$

Thus by the Chinese Remainder Theorem,

$$M^{st} \equiv M \pmod{n},$$

and hence $(M^{st} \bmod n) = (M \bmod n) = M$, as required. □

Example 9.2.3 Bob chooses primes $p = 631$ and $q = 2153$. Then $n = (631)(2153) = 1358543$. Consequently, $\Sigma = \{0, 1, 2, 3, \ldots, 1358542\}$ and $\mathcal{M} = \mathcal{C} = \Sigma^*$. Note that $(p-1)(q-1) = (630)(2152) = 1355760$.

To create his public key, Bob chooses s with the properties: $1 < s < 1355760$ and $\gcd(s, 1355760) = 1$. His choice is $s = 5359$. Bob's public key is now the pair

(5359, 1358543) which he publishes. (For security, Bob also destroys the primes p and q.)

Bob then computes his private key by computing the unique integer t that satisfies $1 < t < 1355760$ and $5359t \equiv 1 \pmod{1355760}$. In fact, $t = 20239$. Thus Bob's private key is 20239.

Alice now encrypts the message $M = 413\ 7000$ letter by letter to yield

$$\begin{aligned} C_0 &= e(M_0, (s, n)) \\ &= e(413, (5359, 1358543)) \\ &= (413^{5359} \bmod 1358543) \\ &= 1311697, \end{aligned}$$

$$\begin{aligned} C_1 &= e(M_1, (s, n)) \\ &= e(7000, (5359, 1358543)) \\ &= (7000^{5359} \bmod 1358543) \\ &= 1262363. \end{aligned}$$

Thus $C = 1311697\ 1262363$.

Alice then sends the ciphertext $C = 1311697\ 1262363$ to Bob who decrypts letter by letter using his private key t:

$$\begin{aligned} M_0 &= d(C_0, t) \\ &= d(1311697, 20239) \\ &= (1311697^{20239} \bmod 1358543) \\ &= 413. \end{aligned}$$

$$\begin{aligned} M_1 &= d(C_1, t) \\ &= d(1262363, 20239) \\ &= (1262363^{20239} \bmod 1358543) \\ &= 7000. \end{aligned}$$

Thus Bob recovers the message $M = 413\ 7000$. □

9.3 Security of RSA

The security of the RSA cryptosystem depends on the secrecy of Bob's private key (the trapdoor). It also depends on the assumption that the RSA encryption function is a one-way trapdoor function: it is hard to invert $e(M, (s, n))$ knowing only e and the public key (s, n). But what evidence do we have that RSA encryption is hard to invert?

RSA encryption is related to another function that is (supposedly) hard to invert. Let $\mathcal{P} = \{2, 3, 5, 7, 11, 13, \dots\}$ denote the set of all prime numbers, let $\mathbb{N} = \{1, 2, 3, \dots\}$, and define

$$\mathrm{PMULT} : \mathcal{P} \times \mathcal{P} \to \mathbb{N},$$

by the rule

$$\mathrm{PMULT}(p, q) = pq = n.$$

The inverse of PMULT is the factorization $n = pq$; factorization is assumed to be hard to compute.

This is formalized in the following assumption.

The Factoring Assumption (FA) *Let $w(x) \in \mathbb{Z}[x]$ be a positive polynomial, let p* and *q be randomly chosen l-bit primes, and let $n = pq$. Let A be a probabilistic polynomial time algorithm with input n and output $A(n) = (p, q) \in \mathcal{P} \times \mathcal{P}$, where $\mathcal{P}$ denotes the set of all primes. Then there exists an integer l_0 for which*

$$\Pr(A(n) = (p, q)) < \frac{1}{w(l)}$$

whenever $l \geq l_0$.

In other words, the probability $\Pr(A(n) = (p, q))$ as a function of l is a negligible function of l.

The FA says that the composite n cannot be factored in polynomial time. This is the basis for the security of RSA.

Proposition 9.3.1 *Assume that the RSA cryptosystem has public key (s, n) and private key t. If n can be factored, then RSA is insecure.*

Proof Suppose that n can be factored into the prime numbers p and q. Then the integer $(p - 1)(q - 1)$ is known. Since $\gcd(s, (p - 1)(q - 1)) = 1$, s is a unit in $\mathbb{Z}_{(p-1)(q-1)}$, and the private key t can be computed as $t = s^{-1}$ in $U(\mathbb{Z}_{(p-1)(q-1)})$. Indeed, t can be found in polynomial time using the Euclidean algorithm. □

The contrapositive of Proposition 9.3.1 holds.

Corollary 9.3.2 *If RSA is secure, then factoring is hard, that is, there is no polynomial time algorithm for inverting* PMULT.

Unfortunately, we do not know if the converse of Corollary 9.3.2 holds; in other words, if factoring is hard, does that guarantee that RSA is secure?

In light of Proposition 9.3.1, attacks on RSA attempt to factor the RSA modulus $n = pq$. In what follows, we review some standard ways to factor the RSA modulus. In view of the FA, the algorithms we present necessarily run in non-polynomial time.

To begin with, we can always use the naive approach of algorithm PRIME (Algorithm 4.3.2) to factor n. From Proposition 4.3.1, we know that n has a prime factor $\le \sqrt{n}$. So we just check each integer j, $2 \le j \le \sqrt{n}$, to see if it is a divisor of n. In this manner, we can find a factor of n in $O(\sqrt{n})$ steps.

J. Pollard has given two methods that improve on this naive approach.

9.3.1 Pollard $p - 1$

Let n be a large composite integer that is a product of primes $n = pq$, $p, q \ge 3$. Suppose that p has the property that the prime factorization of $p - 1$ contains many small primes with small exponents. For instance, suppose $p = 4621$, so that

$$p - 1 = 4620 = 2^2 \cdot 3 \cdot 5 \cdot 7 \cdot 11.$$

In this case, J. M. Pollard [42] has developed a method for factoring n. We note that Pollard's method can be applied to factor an arbitrary integer.

Pollard's method is based on the observation that since $p - 1$ is divisible by only small primes with small exponents, there exists a not-too-large integer m so that $(p - 1) \mid m!$. For example, if $p - 1 = 4620 = 2^2 \cdot 3 \cdot 5 \cdot 7 \cdot 11$, then we can choose $m = 11$, and we see that $p - 1$ divides $11!$. The integer m provides an upper bound on the number of iterations in Pollard's algorithm.

The core idea behind Pollard's $p - 1$ algorithm can be summarized as follows. Since $(p - 1) \mid m!$, there exists an integer k so that $(p - 1)k = m!$. Since $p \ge 3$, $\gcd(2, p) = 1$, and so by Fermat's Little Theorem,

$$2^{m!} \equiv 2^{(p-1)k} \equiv (2^{p-1})^k \equiv 1 \pmod{p}.$$

Thus $p \mid (2^{m!} - 1)$. Since $\gcd(p, n) = p$,

$$p \le \gcd(2^{m!} - 1, n) \le n.$$

If $q \nmid (2^{m!} - 1)$, then

$$p \le \gcd(2^{m!} - 1, n) < n,$$

and thus $p = \gcd(2^{m!} - 1, n)$ is a prime factor of n.

On the other hand, if $q \mid (2^{m!} - 1)$, then $\gcd(2^{m!} - 1, n) = n$, and we do not recover the factor p of n.

We can reduce the chance that $q \mid (2^{m!} - 1)$ to near zero by requiring that $(q-1) \nmid m!$. For then, by the division algorithm, $m! = (q-1)l + r$, $0 < r < q$, hence

$$2^{m!} \equiv 2^{(q-1)l} 2^r \equiv 2^r \pmod{q}.$$

So if $q \mid (2^{m!} - 1)$, then r would have to be a multiple of the order of 2 in $U(\mathbb{Z}_q)$, which is unlikely.

Here is **Pollard's** $p - 1$ **algorithm**.

Algorithm 9.3.3 (POLLARD_$p - 1$**)**

```
Input:     a large integer $n$, which is the product of primes $n = pq$
           with $p, q \geq 3$, a not-too-large integer bound $m$
Output:    a prime factor $p$
Algorithm:
           for $k = 2$ to $m$ do
              $d_k \leftarrow \gcd(2^{k!} - 1, n)$
              if $1 < d_k < n$, then output $p = d_k$
           next $k$
```

□

Example 9.3.4 We use POLLARD_$p - 1$ to factor $n = 21436819$. We set the bound m. After 10 iterations of the for-next loop, we arrive at the computations (the integers $2^{k!} - 1$ are reduced modulo 21436819)

$$\begin{aligned}
\gcd(2^{2!} - 1, 21436819) &= \gcd(3, 21436819) = 1 \\
\gcd(2^{3!} - 1, 21436819) &= \gcd(63, 21436819) = 1 \\
&\vdots \\
\gcd(2^{9!} - 1, 21436819) &= \gcd(10391831, 21436819) = 1 \\
\gcd(2^{10!} - 1, 21436819) &= \gcd(3812898, 21436819) = 1 \\
\gcd(2^{11!} - 1, 21436819) &= \gcd(12162472, 21436819) = 4621.
\end{aligned}$$

The algorithm then outputs the prime factor $p = 4621$. The other prime factor is $q = 21436819/4621 = 4639$. □

The Pollard $p - 1$ algorithm ran efficiently in Example 9.3.4 because $p - 1 = 4620 = 2^2 \cdot 3 \cdot 5 \cdot 7 \cdot 11$ is a product of small primes with small exponents. Moreover, we found a non-trivial factor since $q = 4639 \nmid (2^{11!} - 1)$.

It was highly unlikely that $q = 4639 \mid (2^{11!} - 1)$ since $q - 1 = 4638 \nmid 11!$; $q - 1 = 4638 \nmid 11!$ because 4638 has prime factor decomposition $2 \cdot 3 \cdot 773$ and hence is not a product of primes ≤ 11.

Pollard $p - 1$ will not work efficiently in the case that the primes in the product $n = pq$ are so that both $p - 1$ and $q - 1$ have large prime factors or large prime powers in their decompositions.

Example 9.3.5 We use POLLARD_$p - 1$ to factor $n = 1219$. We set the bound m. After 10 iterations of the for-next loop, we arrive at the computations (the integers $2^{k!} - 1$ are reduced modulo 1219)

$$\begin{aligned}
\gcd(2^{2!} - 1, 1219) &= \gcd(3, 1219) = 1\\
\gcd(2^{3!} - 1, 1219) &= \gcd(63, 1219) = 1\\
&\vdots\\
\gcd(2^{9!} - 1, 1219) &= \gcd(277, 1219) = 1\\
\gcd(2^{10!} - 1, 1219) &= \gcd(1207, 1219) = 1\\
\gcd(2^{11!} - 1, 1219) &= \gcd(575, 1219) = 23.
\end{aligned}$$

The algorithm then outputs the prime factor $p = 23$. The other prime factor is $q = 1219/23 = 53$. □

The computation in Example 9.3.5 is inefficient; it has a large run time relative to the size of the input. In this case the prime factor decompositions are

$$p - 1 = 22 = 2 \cdot 11,$$

$$q - 1 = 52 = 2^2 \cdot 13,$$

which both contain large prime factors relative to the size of p and q.

9.3.2 Pollard ρ

Along with the $p - 1$ method, J. M. Pollard [43] has developed a probabilistic algorithm for factoring $n = pq$; there is a high probability that the algorithm will output a prime factor of n.

Here is Pollard's algorithm.

Algorithm 9.3.6 (POLLARD_ρ)

```
Input:     a large integer $n$, which is the product of primes $n = pq$
           with $p, q \geq 3$, an "average" function $f(x) = x^2 + 1$,
           a not-too-large integer bound $m$
Output:    a prime factor $p$
Algorithm:
           $x_0 \leftarrow 2$
           for $k = 1$ to $m$ do
              $x_k \leftarrow (f(x_{k-1}) \bmod n)$
           next $k$
           for $k = 1$ to $m$ do
              $d_k = \gcd(x_{2k} - x_k, n)$
              if $1 < d_k < n$ then output $p = d_k$
           next $k$
```

□

The algorithm is based on an application of Proposition 2.3.1.

Proposition 9.3.7 *Let n be a large composite integer of the form $n = pq$, where p and q are primes, $p, q \geq 3$. Let $C > 0$ be a real number. Let $m = 1+\lceil\sqrt{2pC}\rceil$. Then randomly choosing a sequence of m terms of $\mathbb{Z}_p$ (with replacement) guarantees that a collision occurs with probability at least $1 - e^{-C}$.*

Proof Take $S = \mathbb{Z}_p$ and $N = p$ in Proposition 2.3.1. Let $y_1, y_2, \ldots, y_m$ be a sequence of terms in $\mathbb{Z}_p$. Then the probability that there is a collision is

$$1 - \prod_{i=1}^{m-1}\left(1 - \frac{i}{p}\right),$$

which by (2.3) is at least $1 - e^{\frac{-m(m-1)}{2p}}$. Now, $\frac{m(m-1)}{2p} > \frac{\sqrt{2pC}\cdot\sqrt{2pC}}{2p} = C$, thus $e^{\frac{-m(m-1)}{2p}} < e^{-C}$, and so

$$1 - e^{\frac{-m(m-1)}{2p}} > 1 - e^{-C}.$$

□

Randomly choosing a sequence of terms in $\mathbb{Z}_p$ is equivalent to choosing an "average" function $f : \mathbb{Z}_p \to \mathbb{Z}_p$ and seed x_0, and defining a sequence of terms iteratively:

$$x_0, x_1 = f(x_0), x_2 = f(f(x_0)), x_3 = f(f(f(x_0))), \ldots \tag{9.2}$$

(In POLLARD_ρ, we choose $f(x) = x^2 + 1$ with seed $x_0 = 2$.)

In the sequence $\{x_i\}_{i\geq 0}$ defined as such, we find the point where the first collision occurs, i.e., we find the smallest j, $j > i$ for which

$$x_i \equiv x_j \pmod{p}. \tag{9.3}$$

Proposition 9.3.7 says that it is very likely that this first collision occurs in the first $m = O(\sqrt{p})$ terms of $\{x_i\}$ modulo p.

Now (9.3) tells us that the period of the sequence $\{x_i\}_{i\geq 0}$ modulo p is $j-i$. Since $j - i \leq j$, there exists a largest integer s for which $s(j-i) \leq j$. We claim that $i \leq s(j-i)$. For if not, then $i > s(j-i)$, or $(s+1)i > sj$, or $(s+1)(j-i) < j$, a contradiction. Thus,

$$i \leq s(j-i) \leq j.$$

Put $k = s(j-i)$. Then

$$x_{2k} \equiv x_k \pmod{p}$$

for some k, whose value is $O(\sqrt{p})$.

POLLARD_ρ finds the first collision modulo p, i.e.,

$$x_{2k} \equiv x_k \pmod{p},$$

which is not a collision modulo n, i.e.,

$$x_{2k} \not\equiv x_k \pmod{n}.$$

Then $1 < \gcd(x_{2k} - x_k, n) < n$, and we have found the factor p.

It follows from Proposition 9.3.7 that Pollard ρ will likely compute a prime factor p of n after $O(\sqrt{p})$ iterations of the for-next loop. Of course, in practice, we will not know the value of p. By Proposition 4.3.1, $p \leq \sqrt{n}$, and thus Pollard ρ will likely compute a prime factor p of n after $O(\sqrt[4]{n})$ iterations of the for-next loop.

Example 9.3.8 We use Pollard_ρ to factor $n = 8131$. With $x_0 = 2$ and $f(x) = x^2 + 1$, the sequence of terms modulo 8131 is

2, 5, 26, 677, 2994, 3675, 35, 1226, 6973, 7481, 7820, 7281, 6973, 7481, 7820, 7281, 6973, ...

We next compute

$$\begin{aligned}
\gcd(x_2 - x_1, 8131) &= \gcd(26 - 5, 8131) = 1\\
\gcd(x_4 - x_2, 8131) &= \gcd(2994 - 26, 8131) = 1\\
\gcd(x_6 - x_3, 8131) &= \gcd(35 - 677, 8131) = 1\\
\gcd(x_8 - x_4, 8131) &= \gcd(6973 - 2994, 8131) = 173.
\end{aligned}$$

The algorithm then outputs the prime factor $p = 173$. The other prime factor is $q = 8131/173 = 47$. □

Remark 9.3.9 In Example 9.3.8, the terms taken modulo 173 and 47, respectively, are

$$2, 5, 26, 158, 53, 42, 35, 15, 53, 42, 35, 15, 53, 42, 35, 15, 53, \dots$$
$$2, 5, 26, 19, 33, 9, 35, 4, 17, 8, 18, 43, 17, 8, 18, 43, 17, \dots$$

For these moduli, the sequences $x_0, x_1, x_2, \dots$ are eventually periodic with period 4 (see Section 11.1). Pollard's method is called Pollard "ρ" because the periodicity of these sequences suggests the shape of the Greek letter ρ when typed. For instance, the non-periodic initial sequence 2, 5, 26, 158 corresponds to the tail of "ρ," while the periodic part

$$42, 35, 15, 53, 42, 35, 15, 53, 42, 35, 15, 53, \dots$$

corresponds to the loop of "ρ." □

Here is a GAP program for finding a prime factor of n using POLLARD_ρ.

```
x:=List([1..m]);
x[1]:=2;
for k in [2..m] do
  x[k]:=(x[k-1]^2+1) mod n;
od;
d:=List([1..m]);
for j in [2..QuoInt(m,2)] do
  d[j-1]:=GcdInt(x[2*j-1]-x[j],n);
  if 1 < d[j-1] and d[j-1] < n then
    Print("a prime factor of"," ", n," ","is"," ",d[j-1]);
    break;
  fi;
od;
```

In Example 9.3.8, we used the function $f : \mathbb{Z}_{8131} \rightarrow \mathbb{Z}_{8131}$ defined as $f(x) = x^2 + 1$ to generate the sequence $x_0, x_1, x_2, \dots$; f is an average "typical" function; it is not a bijection. For instance,

$$f(1) = f(518) = f(7613) = f(8130) = 2.$$

In fact, if we had chosen a bijection $f : \mathbb{Z}_{8131} \rightarrow \mathbb{Z}_{8131}$, it would not be as useful for Pollard ρ.

Proposition 9.3.10 *Suppose S is a finite set with $N = |S|$. Let $f : S \rightarrow S$ be a permutation of S (a bijection from S to itself). Let $x_0 \in S$ and define a sequence $x_i = f(x_{i-1})$ for $i \geq 1$. Let k be the smallest integer, $k \geq 1$, for which $x_0, x_1, \dots, x_{k-1}$ are distinct and $x_k = x_j$ for some $0 \leq j \leq k-1$.*

(i) $\Pr(k = 1) = \Pr(k = 2) = \Pr(k = 3) = \cdots = \Pr(k = N) = \frac{1}{N}$.

(ii) The expected value of k over all permutations $f : S \to S$ is $(N+1)/2 = O(N)$.

Proof See [32, p. 198, 5(a),(b)]. □

Thus if $f : \mathbb{Z}_n \to \mathbb{Z}_n$ is a bijection in Pollard ρ, then the algorithm would likely terminate after $O(\sqrt{n})$ iterations, which is much less efficient than $O(\sqrt[4]{n})$.

9.3.3 Difference of Two Squares

Let $n = pq$ be a product of distinct odd primes p, q, $p > q$. We discuss other algorithms for factoring n. These methods depend on the familiar difference of two squares factorization

$$x^2 - y^2 = (x + y)(x - y).$$

Fermat Factorization

Our first algorithm is called **Fermat factorization**. The idea behind this algorithm is to find an integer x so that $x^2 - n$ is a square of some integer y. For then, $x^2 - n = y^2$, thus, $n = x^2 - y^2 = (x + y)(x - y)$, and so $p = x + y$ and $q = x - y$ are the prime factors of n.

Since we require that $x^2 - n$ is a square of an integer, we can assume that $x^2 - n \geq 0$, thus $x \geq \sqrt{n}$. Thus the algorithm begins the search process with $x = \lceil \sqrt{n} \rceil$. If $x^2 - n$ is a square, then we are done, else we check whether $(x + 1)^2 - n$ is a square, and so on.

The algorithm will always succeed in finding an x so that $x^2 - n$ is a square. Indeed, if $x = (p + q)/2$, then

$$\begin{aligned} x^2 - n &= \left(\frac{p+q}{2}\right)^2 - n \\ &= \frac{1}{4}\left(p^2 + 2pq + q^2\right) - pq \\ &= \frac{1}{4}p^2 - \frac{1}{2}pq + \frac{1}{4}q^2 \\ &= \left(\frac{p-q}{2}\right)^2. \end{aligned}$$

We have $1 < p$ and $q < n$ and so the largest x we have to check is $\leq n$.

Here is the algorithm.

Algorithm 9.3.11 (FERM_FACT)

```
Input:     an integer $n$, which is the product of primes $n = pq$
           with $p, q \geq 3$.
Output:    a prime factor $p$
Algorithm:
           for $i = \lceil\sqrt{n}\rceil$ to $n$ do
              if $i^2 - n$ is a square of an integer $j$, then
              output $p = i + j$
           next $i$
```

□

Example 9.3.12 We use FERM_FACT to factor $n = 1012343$. We start with $i = \lceil\sqrt{1012343}\rceil = 1007$ and check

$$1007^2 - 1012343 = 1706, \text{ (not a square)}$$
$$1008^2 - 1012343 = 3721 = 61^2 \text{ (is a square).}$$

Thus, $p = i + j = 1008 + 61 = 1069$ is a prime factor of 1012343; the other factor is $1008 - 61 = 947$. □

Example 9.3.13 We use FERM_FACT to factor $n = 1967$. We start with $i = \lceil\sqrt{1967}\rceil = 45$ and check

$$45^2 - 1967 = 58, \text{ (not a square)}$$
$$46^2 - 1967 = 149, \text{ (not a square)}$$
$$47^2 - 1967 = 242, \text{ (not a square)}$$
$$\vdots$$
$$143^2 - 1967 = 18482, \text{ (not a square)}$$
$$144^2 - 1967 = 18769 = 137^2 \text{ (is a square).}$$

Thus $p = i + j = 144 + 137 = 281$ is a prime factor of 1967; the other factor is $144 - 137 = 7$. □

Proposition 9.3.14 *The running time of* FERM_FACT *is* $O(n)$.

Proof The algorithm will output a factor in at most

$$n - \lceil\sqrt{n}\rceil + 1 > (p + q)/2 - \lceil\sqrt{n}\rceil + 1$$

iterations, and thus FERM_FACT runs in time $O(n)$. □

If $p \approx q$, then $(p+q)/2 \approx p \approx \sqrt{n}$. So when p and q are close in value, the algorithm finds a factor of n very quickly, as we have seen in Example 9.3.12. On the other hand, if p and q are far apart, then the algorithm is quite inefficient, as shown in Example 9.3.13.

Our next algorithm improves on this factoring method.

Modular Fermat Factorization

The idea is to find integers x and k so that $x^2 - kn$ is a square of some integer y. Then, $x^2 - kn = y^2$ and so

$$x^2 \equiv y^2 \pmod{n}. \tag{9.4}$$

We then have

$$(x+y)(x-y) = kn,$$

and computing $\gcd(n, x+y)$ and $\gcd(n, x-y)$ will yield the prime factors of n.

The algorithm, which we call **modular Fermat factorization**, is a systematic way of solving the congruence (9.4). It uses the notion of a “smooth” integer. Let $B \geq 2$ be a real number. An integer $m \geq 2$ is B**-smooth** if each prime factor of m is less than or equal to B. For example, 16 is 2-smooth and $n!$ is n-smooth for all $n \geq 2$.

For $x \in \mathbb{R}_+$, let $\pi(x)$ denote the number of prime numbers $\leq x$, see [47, p. 72, Definition]. For later use, we state a famous result ([47, Theorem 3.4]), which allows us to approximate the value of $\pi(x)$.

Theorem 9.3.15 (Prime Number Theorem) *For* $x \in \mathbb{R}_+$,

$$\lim_{x\to\infty} \frac{\pi(x)}{x \ln(x)} = 1.$$

Here is the algorithm.

Algorithm 9.3.16 (MOD_FERM_FACT)

Input: an integer n, which is the product of primes $n = pq$
with $p, q \geq 3$.
Output: a prime factor p
Algorithm:

Step 1. Choose a bound $B \geq 0$ and find a sequence of integers $m_1, m_2, m_3, \ldots$ for which $(m_i^2 \bmod n)$ is B-smooth. Once $\pi(B)$ such residues have been found, form a system (re-indexing the m_j if necessary):

$$\begin{cases} m_1^2 \equiv q_1^{e_{1,1}} q_2^{e_{1,2}} \cdots q_k^{e_{1,k}} \pmod{n} \\ m_2^2 \equiv q_1^{e_{2,1}} q_2^{e_{2,2}} \cdots q_k^{e_{2,k}} \pmod{n} \\ \vdots \\ m_r^2 \equiv q_1^{e_{r,1}} q_2^{e_{r,2}} \cdots q_k^{e_{r,k}} \pmod{n} \end{cases} \tag{9.5}$$

for some k, r, with $k \leq \pi(B) \leq r$. Here $q_1, q_2, \cdots q_k$ are B-smooth primes and $e_{a,b} \geq 0$ for $1 \leq a \leq r$, $1 \leq b \leq k$.

Step 2. From (9.5), we take a product of the m_a^2, $1 \leq a \leq r$,

$$(m_1^2)^{d_1}(m_2^2)^{d_2} \cdots (m_r^2)^{d_r},$$

$d_a \in \{0, 1\}$, so that this product satisfies

$$(m_1^2)^{d_1}(m_2^2)^{d_2} \cdots (m_r^2)^{d_r} \equiv y^2 \pmod{n},$$

for some $(y \bmod n)$. Then

$$\begin{aligned} (m_1^2)^{d_1}(m_2^2)^{d_2} \cdots (m_r^2)^{d_r} &\equiv (m_1^{d_1})^2(m_2^{d_2})^2 \cdots (m_r^{d_r})^2 \\ &\equiv (m_1^{d_1} m_2^{d_2} \cdots m_r^{d_r})^2 \\ &\equiv y^2 \pmod{n}. \end{aligned}$$

Thus $x = m_1^{d_1} m_2^{d_2} \cdots m_r^{d_r}$ yields a solution to (9.4).

Step 3. We check $\gcd(n, x + y)$ and $\gcd(n, x - y)$ to obtain the factors of n. □

Example 9.3.17 We use MOD_FERM_FACT to factor $n = 115993$.

We set $B = 7$ and find the following congruences:

$$\begin{aligned} (50948)^2 &\equiv 2 \cdot 3 \cdot 5^2 \cdot 7^2 \pmod{115993} \\ (8596)^2 &\equiv 3 \cdot 5^2 \cdot 7^2 \pmod{115993} \\ (46970)^2 &\equiv 2^7 \cdot 3 \cdot 7^4 \pmod{115993}. \end{aligned}$$

We have

$$\begin{aligned}(50948)^2(46970)^2 &\equiv (50948 \cdot 46970)^2 \pmod{115993}\\ &\equiv (91970)^2 \pmod{115993}\\ &\equiv 2^8 \cdot 3^2 \cdot 5^2 \cdot 7^6 \pmod{115993}\\ &\equiv (2^4 \cdot 3 \cdot 5 \cdot 7^3)^2 \pmod{115993}\\ &\equiv (82320)^2 \pmod{115993}.\end{aligned}$$

And so, the factors are

$$\gcd(115993, 91970 - 82320) = 193 \text{ and } \gcd(115993, 91970 + 82320) = 601.$$

□

How efficient is MOD_FERM_FACT? There is an important theorem of E. R. Canfield et al. [9] that allows us to complete Step 1 in a reasonable (yet non-polynomial) amount of time.

For an integer $n \geq 1$, define

$$L(n) = e^{(\ln(n))^{1/2}(\ln(\ln(n)))^{1/2}}.$$

For $n \geq 2$, let $\Psi(n, B)$ be the number of B-smooth integers j with $2 \leq j \leq n$. For instance, $\Psi(10, 3) = 4$ since 2, 3, 4, and 9 are the only 3-smooth integers with $2 \leq j \leq 10$.

Theorem 9.3.18 (Canfield, Erdős, and Pomerance) *Let c, $0 < c < 1$, and then*

$$\lim_{n\to\infty} \frac{\Psi(n, L(n)^c)}{n} = \frac{1}{L(n)^{\frac{1}{2c}}}.$$

□

Corollary 9.3.19 *Let n be a large integer and let $B = L(n)^{\frac{1}{\sqrt{2}}}$. In a random sequence of $\lceil L(n)^{\sqrt{2}} \rceil$ integers modulo n, we expect to find $\pi(L(n)^{\frac{1}{\sqrt{2}}})$ integers that are $L(n)^{\frac{1}{\sqrt{2}}}$-smooth.*

Proof For any c, $0 < c < 1$, the probability that a random integer modulo n is $L(n)^c$-smooth is

$$\frac{\Psi(n, L(n)^c)}{n} \approx \frac{1}{L(n)^{\frac{1}{2c}}}.$$

And so, we need to check approximately

$$\pi(L(n)^c)L(n)^{\frac{1}{2c}} \tag{9.6}$$

integers to guarantee that $\pi(L(n)^c)$ of them are $L(n)^c$-smooth.

By the Prime Number Theorem,

$$\pi(L(n)^c)L(n)^{\frac{1}{2c}} \approx \frac{L(n)^c}{ln(L(n)^c)} \cdot L(n)^{\frac{1}{2c}} \approx L(n)^{c+\frac{1}{2c}}.$$

So we need to check $L(n)^{c+\frac{1}{2c}}$ integers to find $\pi(L(n)^c)$ integers that are $L(n)^c$-smooth.

Using some elementary calculus, we find that $L(n)^{c+\frac{1}{2c}}$ is minimized when $c = \frac{1}{\sqrt{2}}$, and its minimum value is $L(n)^{\sqrt{2}}$. □

Proposition 9.3.20 *Let* $n = pq$ *be product primes* $p, q \geq 3$. *Then* MOD_FERM_FACT *factors n in subexponential time*

$$O(2^{c(\log_2(n))^{1/2}(\log_2(\log_2(n)))^{1/2}}),$$

where $c > 0$ *is a small constant.*

Proof (Sketch) We compute the amount of time needed to complete Step 1. If an integer $(m \bmod n)$ is chosen at random, then the integer $(m^2 \bmod n)$ is essentially random. By Corollary 9.3.19, in a random sequence of $\lceil L(n)^{\sqrt{2}} \rceil$ residues $(m^2 \bmod n)$, we expect to find $\pi(L(n)^{\frac{1}{\sqrt{2}}})$ residues $(m^2 \bmod n)$ that are $L(n)^{\frac{1}{\sqrt{2}}}$-smooth. Thus Step 1 takes time $O(L(n)^{\sqrt{2}})$. Thus the entire algorithm runs in subexponential time $O(2^{c(\log_2(n))^{1/2}(\log_2(\log_2(n)))^{1/2}})$, where $c > 1$ is a small constant. □

Remark 9.3.21 We can reduce the time required to complete Step 1 down to $O(L(n))$ using the **quadratic sieve** developed by C. Pomerance [44], [25, Section 3.7.2]. Thus the total running time of Algorithm 9.3.16 can be reduced to

$$O(2^{(\log_2(n))^{1/2}(\log_2(\log_2(n)))^{1/2}}).$$

Assuming that a computer can perform at most 2^{40} basic operations in a reasonable amount of time, Algorithm 9.3.16, using the quadratic sieve, can factor numbers $n = pq$ up to size 2^{206}. This follows since

$$2^{40} \approx 2^{\sqrt{\log_2(2^{206})\log_2(\log_2(2^{206}))}}.$$

Thus, to ensure security in RSA, the chosen primes must be at least 206-bit primes. In fact, in RSA-2048 two 1024-bit primes are chosen, and in RSA-4096, two 2048-bit primes are used.

9.4 The ElGamal Public Key Cryptosystem

Our second public key cryptosystem is based on the discrete exponential function.

Definition 9.4.1 (Discrete Exponential Function) Let p be a random l-bit prime, and let g be a primitive root modulo p, i.e., $\langle g\rangle = U(\mathbb{Z}_p)$. Define a function

$$\mathrm{DEXP}_{p,g} : U(\mathbb{Z}_p) \to U(\mathbb{Z}_p)$$

by the rule

$$\mathrm{DEXP}_{p,g}(x) = (g^x \bmod p), \quad x \in U(\mathbb{Z}_p);$$

$\mathrm{DEXP}_{p,g}$ is the **discrete exponential** function.

Since g is a primitive root modulo p, g generates the cyclic group $U(\mathbb{Z}_p)$, which has order $p-1$. Thus $\mathrm{DEXP}_{p,g}$ is a 1-1 correspondence, and so the inverse of $\mathrm{DEXP}_{p,g}$ exists.

Definition 9.4.2 (Discrete Logarithmic Function) The inverse of $\mathrm{DEXP}_{p,g}$ is the function

$$\mathrm{DLOG}_{p,g} : U(\mathbb{Z}_p) \to U(\mathbb{Z}_p),$$

defined as follows: for $y \in U(\mathbb{Z}_p)$, $\mathrm{DLOG}_{p,g}(y)$ is the unique element $x \in U(\mathbb{Z}_p)$ for which $y = (g^x \bmod p)$; $\mathrm{DLOG}_{p,g}$ is the **discrete logarithm** function.

For example, let $p = 31$. Then $g = 3$ is a primitive root modulo 31. We have $\mathrm{DEXP}_{31,3}(17) = 22$ and $\mathrm{DLOG}_{31,3}(22) = 17$; $\mathrm{DEXP}_{31,3}(6) = 16$ and $\mathrm{DLOG}_{31,3}(16) = 6$.

If p is an l-bit prime, then we can consider $\mathrm{DEXP}_{p,g}$ and $\mathrm{DLOG}_{p,g}$ as functions on $[0,1]^l$; passing to base 2 expansions yields

$$\mathrm{DEXP}_{p,g} : [0,1]^l \to [0,1]^l,$$

$$\mathrm{DLOG}_{p,g} : [0,1]^l \to [0,1]^l.$$

For example, for the 5-bit prime 31, we have $\mathrm{DEXP}_{31,3}(10001) = 10110$ and $\mathrm{DLOG}_{31,3}(10000) = 00110$.

Let p be a random l-bit prime, and let g be a primitive root modulo p. We assume that it is "hard" to compute $\mathrm{DLOG}_{p,g}(y)$, where y is a randomly chosen element of $U(\mathbb{Z}_p)$. More formally, we assume the following.

The Discrete Logarithm Assumption (DLA) *Let $w(x) \in \mathbb{Z}[x]$ be a positive polynomial, let p be a randomly chosen l-bit prime, let g be a primitive root modulo p, and let $y \in_R U(\mathbb{Z}_p)$. Let $A_{p,g}$ be a probabilistic polynomial time algorithm dependent on p, g with input y and output $A_{p,g}(y) \in U(\mathbb{Z}_p)$. Then there exists an integer $l_0 \geq 1$ for which*

$$\Pr\left(A_{p,g}(y) = \mathrm{DLOG}_{p,g}(y)\right) < \frac{1}{w(l)},$$

whenever $l \geq l_0$. In other words, as a function of l, the probability

$$\Pr\left(A_{p,g}(y) = \mathrm{DLOG}_{p,g}(y)\right)$$

is a negligible function of l.

Given random $x \in U(\mathbb{Z}_p)$, the DLA says that there is no probabilistic polynomial time algorithm that inverts $\mathrm{DEXP}_{p,g}(x)$, i.e., finds $y \in U(\mathbb{Z}_p)$ so that $\mathrm{DLOG}_{p,g}(y) = x$. There is no polynomial time algorithm that computes $\mathrm{DLOG}_{p,g}$.

The DLA (if true) ensures the security of our next public key cryptosystem.

Definition 9.4.3 (ElGamal Public Key Cryptosystem) Alice wants to send a secret message to Bob. Let p be a prime number, let g be a primitive root modulo p, and let $x \in U(\mathbb{Z}_p)$. Let $\Sigma = \{0, 1, 2, 3, \dots, p-1\}$ denote the set of p letters.

Let $\mathcal{M} = \Sigma^*$. A message is a finite sequence of letters $M = M_0 M_1 \cdots M_{r-1}$. Bob's public key is the triple $(p, g, (g^x \bmod p))$ and his private key is $x \in U(\mathbb{Z}_p)$.

Using Bob's public key, Alice encrypts the message $M = M_0 M_1 \cdots M_{r-1}$ as follows: she chooses an element $y \in U(\mathbb{Z}_p)$ at random and computes

$$h = (g^y \bmod p), \text{ and } N_i = (M_i (g^x)^y \bmod p),$$

for $0 \leq i \leq r-1$. The encryption transformation is

$$C_i = e(M_i, (p, g, (g^x \bmod p))) = (h, N_i),$$

for $0 \leq i \leq r-1$. The resulting ciphertext is

$$C = C_0 C_1 \cdots C_{r-1} = (h, N_0)(h, N_1) \cdots (h, N_{r-1}).$$

Alice then sends the ciphertext C to Bob.

Bob decrypts the message C by computing

$$d(C_i, x) = d((h, N_i), x) = ((h^{-x} N_i) \bmod p),$$

for $0 \le i \le r - 1$.

This is the **ElGamal** public key cryptosystem. □

Here is an example of the ElGamal cryptosystem.

Example 9.4.4 Let $p = 29$. Then $g = 2$ is a primitive root modulo 29. Let $\Sigma = \{0, 1, 2, 3, \ldots, 28\}$ denote the set of 29 letters. Let $x = 10 \in U(\mathbb{Z}_{29})$. Note that $2^{10} \equiv 9 \pmod{29}$. Bob's public key is (29, 2, 9) and his private key is $x = 10$.

With the choice of $y = 5$, Alice encrypts `C A T` ↔ 2 0 19 as follows:

$$C_0 = e(2, (29, 2, 9)) = ((2^5 \bmod 29), ((2 \cdot 9^5) \bmod 29)) = (3, 10),$$

$$C_1 = e(0, (29, 2, 9)) = ((2^5 \bmod 29), ((0 \cdot 9^5) \bmod 29)) = (3, 0),$$

$$C_2 = e(19, (29, 2, 9)) = ((2^5 \bmod 29), ((19 \cdot 9^5) \bmod 29)) = (3, 8).$$

So $C = (3, 10)(3, 0)(3, 8)$. Alice sends C to Bob who decrypts to obtain

$$M_0 = d((3, 10), 10) = ((3^{-10} \cdot 10) \bmod 29) = 2,$$

$$M_1 = d((3, 0), 10) = ((3^{-10} \cdot 0) \bmod 29) = 0,$$

$$M_2 = d((3, 8), 10) = ((3^{-10} \cdot 8) \bmod 29) = 19.$$

Thus Bob recovers the message 2 0 19 ↔ `C A T`.

In another communication, Alice has used $y = 17$ to encrypt a message as $C = (21, 28)(21, 3)$. She then sends C to Bob. To decrypt, Bob computes

$$M_0 = d((21, 28), 10) = ((21^{-10} \cdot 28) \bmod 29) = ((22 \cdot 28) \bmod 29) = 7,$$

$$M_1 = d((21, 3), 10) = ((21^{-10} \cdot 3) \bmod 29) = ((22 \cdot 3) \bmod 29) = 8.$$

And so, $M = 7\ 8$ ↔ `H I`. □

Malice knows Bob's ElGamal public key $(p, g, (g^x \bmod p))$. Yet the DLA (if true) guarantees that he cannot compute Bob's trapdoor x in polynomial time. Malice can still attack ElGamal by finding Bob's trapdoor $x = \mathrm{DLOG}_{p,g}(g^x \bmod p)$ using non-polynomial methods. In Section 12.2, we will discuss several non-polynomial time algorithms that compute $\mathrm{DLOG}_{p,g}$.

9.5 Hybrid Ciphers

The most widespread use of public key cryptosystems (such as RSA) is as a component of a **hybrid cipher**, a cryptosystem in which a public key cryptosystem is used together with a symmetric cryptosystem. In a hybrid cipher, a public key cryptosystem is used to encrypt the symmetric key used in a companion symmetric cryptosystem.

Definition 9.5.1 (Hybrid Cipher) Alice and Bob agree to use the symmetric cryptosystem

$$\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$$

and the public key cryptosystem

$$\langle \mathcal{M}', \mathcal{C}', e', d', \mathcal{K}_{e'}, \mathcal{K}_{d'} \rangle$$

for their hybrid cipher.

Step 1. Alice encrypts message M using the symmetric cryptosystem to yield the ciphertext $C = e(M, k)$.

Step 2. Alice encrypts the symmetric key k using the public key cryptosystem to give the ciphertext $c = e'(k, k_{e'})$, $k_{e'} \in \mathcal{K}_{e'}$. The package (C, c) is then sent to Bob.

Step 3. Bob uses his trapdoor $k_{d'}$ to obtain the symmetric key $k = d'(c, k_{d'})$ and then decrypts the ciphertext: $M = d(C, k)$. □

This is the so-called **(key encapsulation mechanism/data encapsulation mechanism (KEM/DEM)** approach, see [56, Section 16.3].

Example 9.5.2 In this hybrid cipher, we let

$$\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$$

be the Vigenère cipher with $n = 10$ and symmetric key $k = 2619$. We let

$$\langle \mathcal{M}', \mathcal{C}', e', d', \mathcal{K}_{e'}, \mathcal{K}_{d'} \rangle$$

be RSA; Bob's public key is $(s, n) = (5359, 1358543)$ and his trapdoor is $t = 20239$, see Example 9.2.3.

Alice encrypts the message $M = 3343001906$ as

$$C = 5952262822 = e(3343001906, 2619).$$

Alice then encrypts $k = 2619$ as

$$c = e'(2619, (5359, 1358543)) = (2619^{5359} \bmod 1358543) = 455068.$$

Next, Alice sends the package $(C, c) = (5952262822, 455068)$ to Bob.

Bob first decrypts c with his trapdoor:

$$k = d'(455068, 20239) = (455068^{20239} \bmod 1358543) = 2619$$

and then decrypts C to obtain

$$M = d(5952262822, 2619) = 3343001906.$$

□

Remark 9.5.3 As noted in the hybrid cipher, plaintext English messages are generally not encrypted with RSA (or other public key cryptosystems). Public key cryptography is used almost exclusively to encrypt keys. Thus, we do not care as much about the unicity distance of RSA or other public key cryptosystems, though the unicity distance can be computed in theory. □

9.6 Symmetric vs. Public Key Cryptography

One important advantage of public key cryptography over symmetric cryptography lies in their use in communication networks.

Suppose that a communication network consists of $N \geq 2$ principals (or vertices). Each of the N principals wishes to communicate securely with each of the other $N - 1$ principals using a symmetric key cryptosystem.

In order to establish a shared secret key, each principal must make an arrangement to meet each of the other principals to exchange the key. This involves a total of $N(N - 1)/2$ trips (see Section 9.7, Exercise 24).

For example, suppose that the network consists of $N = 6$ principals, listed as A, B, C, D, E, and F. In this case, $(6 \cdot 5)/2 = 15$ trips have to be made to exchange 15 pairs of keys. The network is modeled by the complete graph on 6 vertices (see Figure 9.1).

For large N, however, it is not feasible to make these trips, for instance, if $N = 10^5$, then 4999950000 trips are required.

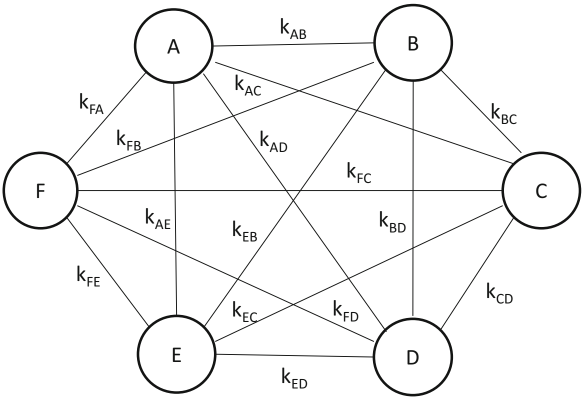


Fig. 9.1 Communication network: 6 principals, using symmetric keys: k_{AB} is the shared key between Alice and Bob, and so on

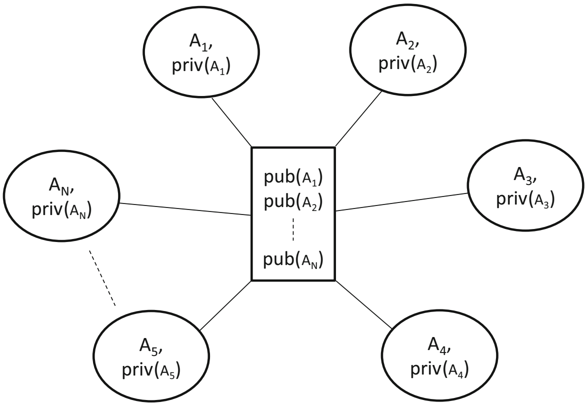


Fig. 9.2 Communication network: N principals, using public keys

On the other hand, if principals $A_1, A_2, \ldots, A_N$ employ a public key cryptosystem, then they need only to establish N public key/private key pairs, denoted as

$$(\text{pub}(A_1), \text{priv}(A_1)),\ (\text{pub}(A_2), \text{priv}(A_2))\ , \ldots,\ (\text{pub}(A_N), \text{priv}(A_N)).$$

The network is now modeled by a hub-and-spoke diagram and is greatly simplified (see Figure 9.2). Note that the hub (or center node) is the listing of all public keys.

To send a secure message M to Bob ($= A_2$), Alice ($= A_1$) first looks up Bob's public key pub(A_2) in the center node. Alice then encrypts to form $C = e(M, \text{pub}(A_2))$ and sends C to Bob. Bob then decrypts with his private key priv(A_2) to yield $M = d(C, \text{priv}(A_2))$.

Here is a general comparison between symmetric key and public key cryptography.

Advantages of Symmetric Cryptosystems

1. The encryption and decryption transformations in symmetric key cryptosystems are simple to implement and allow for very fast encryption/decryption.

2. Symmetric key encryption transformations can be composed or iterated to produce stronger ciphers (as in the Feistel cipher and DES).
3. Symmetric key cryptosystems can provide perfect secrecy as in the Vernam cipher.

Disadvantages of Symmetric Cryptosystems

1. In a symmetric key cryptosystem, the shared key must be kept secret by both the sender and the receiver.
2. In a large communication network, there are many shared keys to be managed; the network graph is complex.

Advantages of Public Key Cryptosystems

1. In a public key cryptosystem, only the private key needs to be kept secret.
2. In a large communication network using a public key cryptosystem, there are fewer keys to be managed; the network graph is simplified.
3. Public key cryptosystems can be employed as digital signature schemes (see Chapter 10).

Disadvantages of Public Key Cryptosystems

1. Encryption and decryption is not as fast or efficient as in a symmetric key cryptosystem.
2. No public key cryptosystem has been proven to achieve perfect secrecy.
3. Public key encryption transformations are modeled on the concept of a one-way trapdoor function. Unfortunately, one-way trapdoor functions have not been proven to exist.

9.7 Exercises

1. Let $f : \mathbb{R}_+ \to \mathbb{R}_+$ be the function defined as $f(x) = \frac{1}{\ln(x)}$. Determine whether f is a negligible function of x.
2. Let $f : \mathbb{R}_+ \to \mathbb{R}_+$ be the function defined as $f(x) = \frac{x}{2^x}$. Prove that f is negligible.
3. Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle$ denote the RSA cryptosystem with public key $(s, n) = (21, 4981)$, private key $t = 445$, and alphabet $\Sigma = \{0, 1, 2, \ldots, 4980\}$ with $\mathcal{M} = \mathcal{C} = \{0, 1, 2, \ldots, 4980\}^*$.

   (a) Compute $C = e(352, (21, 4981))$.
   (b) Verify that $352 = d(e(352, (21, 4981)), 445)$.

4. Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle$ denote the RSA cryptosystem with public key $(s, n) = (631, 7991)$. Suppose that Malice has obtained (somehow) the prime factorization $7991 = 61 \cdot 131$. How can Malice obtain the private key t? Find the value of the private key.

5. Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle$ denote the RSA cryptosystem with public key $(s, n) = (17, 3977)$ and private key $t = 2033$.

   (a) Compute $C = e(65, (17, 3977))$.
   (b) Show that the size of the private keyspace $|\mathcal{K}_d|$ must be smaller than 3977.

6. Create your own RSA public key (s, n) and private key t pair in which the chosen primes satisfy $p, q > 100$.
7. Why cannot (should not) the primes p, q in RSA be identical?
8. Use Pollard's $p - 1$ algorithm to factor 689.
9. Use Pollard's ρ algorithm to factor 36287.
10. Use modular Fermat factorization to factor 198103.
11. Let p be a prime number of the form $p = 2^m + 1$ for some $m > 0$. Prove that $m = 2^k$ for some $k \geq 0$. Primes of the form $2^{2^k} + 1, k \geq 0$, are **Fermat primes**. (It is not known whether there are an infinite number of Fermat primes; in fact, the largest known Fermat prime is $2^{2^4} + 1 = 65537$.)
12. Discuss the Pollard $p - 1$ factorization method in the case that $n = pq$, where p and q are Fermat primes.
13. Suppose that $n = pq$ is a product of a Fermat prime $p = 2^{2^k} + 1, k \geq 0$, and some other prime $q \neq p$. Let x_0 be a primitive root modulo p. Suppose we use Pollard ρ with $f(x) = x^2$ (instead of $f(x) = x^2 + 1$) and seed x_0 to factor n. Show that Pollard ρ factors n in polynomial time.
14. Find the minimal number of terms that need to be selected at random from the residue ring $\mathbb{Z}_{43}$ (with replacement) so that the probability of a collision is at least 0.95.
15. Let $f : \mathbb{Z}_{43} \to \mathbb{Z}_{43}$ be the function defined as $f(x) = ((x^2 + 1) \bmod 43)$. Consider the sequence defined as $x_i = f(x_{i-1})$ with seed $x_0 = 2$. Find the smallest j so that $x_i \equiv x_j \pmod{43}$, $j > i$.
16. In this exercise we introduce the **Cocks–Ellis** public key cryptosystem which actually predates RSA.

    Cocks–Ellis public key cryptosystem
    As always, Alice wants to send a secret message to Bob. Let p and q be large prime numbers for which $p \nmid q - 1$ and $q \nmid p - 1$. Let $n = pq$, and let $\Sigma = \{0, 1, 2, 3, \ldots, n - 1\}$ denote the set of n letters. We let $\mathcal{M} = \mathcal{C} = \Sigma^*$. A message $M \in \mathcal{M}$ is a finite sequence of letters in Σ.

    The integer n is Bob's public key (the encryption key) which he publishes for all to see. Using Bob's public key, Alice encrypts the message $M = M_1 M_2 M_3 \cdots M_r$ letter by letter to form the ciphertext $C = C_1 C_2 C_3 \cdots C_r$ using the rule

    $$C_i = e(M_i, n) = (M_i^n \bmod n).$$

    She then sends the ciphertext C to Bob.

    Bob's private key is a 6-tuple of integers

    $$k_d = (p, q, r, s, u, v),$$

where p and q are primes as above, and r, s, u, and v satisfy the conditions: $pr \equiv 1 \pmod{q-1}$, $qs \equiv 1 \pmod{p-1}$, $pu \equiv 1 \pmod{q}$, and $qv \equiv 1 \pmod{p}$.

Decryption of the ciphertext $C = C_1 C_2 \cdots C_r$ proceeds letter by letter using the decryption function

$$d(x, (p, q, r, s, u, v)) : \mathcal{C} \to \mathcal{M},$$

defined as

$$M_i = d(C_i, (p, q, r, s, u, v)) = ((qvC_i^s + puC_i^r) \bmod n).$$

(a) Suppose Bob's public key is $n = 8557$ and his private key is

$$(p, q, r, s, u, v) = (43, 199, 175, 19, 162, 8).$$

Compute

$$C = e(352\,56, 8557) \text{ and } M = d(7, (43, 199, 175, 19, 162, 8)).$$

(b) Show that the Cocks–Ellis cryptosystem works, i.e., verify formula (8.1).

(c) Prove that the Cocks–Ellis cryptosystem is a special case of RSA.

17. Show that $g = 3$ is a primitive root modulo 31. Compute $\text{DEXP}_{31,3}(20)$ and $\text{DLOG}_{31,3}(25)$.
18. Find all of the primitive roots modulo 43.
19. It is known that $g = 3$ is a primitive root modulo the prime number $p = 257$. Write an algorithm that computes $\text{DLOG}_{257,3}(219)$. How good is your algorithm? When generalized to large primes, is the implementation of your algorithm feasible?
20. Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle$ denote the ElGamal cryptosystem with public key $(p, g, (g^x \bmod p)) = (29, 8, 3)$, private key $x = 11$, $\Sigma = \{0, 1, 2, \ldots, 28\}$, and $\mathcal{M} = \mathcal{C} = \{0, 1, 2, \ldots, 28\}^*$. Verify that $20 = d(e(20, (29, 8, 3)), 11)$.
21. Prove that the ElGamal public key cryptosystem works.
22. Suppose that a communication network contains 1000 principals. How many trips are necessary if each pair of principals wants to establish a shared secret key?
23. In a communication network, suppose that 1225 trips are necessary for each pair of principals to establish a shared secret key. How many principals are in the network?
24. Suppose there are N principals in a communication network. Prove that there are $N(N-1)/2$ trips necessary if each pair of principals wants to establish a shared secret key.

Chapter 10
Digital Signature Schemes

In this chapter we show how public key cryptosystems can be used to create "digital" signatures.

10.1 Introduction to Digital Signature Schemes

Consider the following situations that may occur in a communications network:

1. Malice("Alice") sends Bob the message:
   `Withdraw $1000 from my (Alice's) bank account and give it to Malice.`
   or
2. Alice sends to Bob the message:
   `Transfer $1000 from my (Alice's) bank account and give it to Carol.`
   Malice intercepts the message, replaces `Carol` with `Malice`, and then relays the altered message to Bob.
   or
3. Alice sends to Bob the message:
   `Transfer $1000 from my (Alice's) bank account and give it to Malice.`
   Malice stores this message and resends it to Bob at a later time.

In each case, Bob should verify that the message purportedly from Alice actually came from Alice, and not Malice. That is, Bob should have some method for verifying that the received message reflects the intent of Alice. Any such method is called **message authentication**.

Historically, message authentication has been established by a handwritten signature affixed to the message. A **digital signature** or a **digital signature scheme**

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_10

(DSS) is a method for achieving message authentication through the use of a cryptosystem (usually public key). A DSS is the "digital analog of the handwritten signature."

A digital signature should provide the same guarantees as a traditional signature, namely it should achieve:

1. **Unforgeability**: Only Alice should be able to sign her name to a message.
2. **Undeniability**: Alice should not be able to deny that she signed the message at some later time.
3. **Authentication**: The signature should authenticate the message.

In order for Alice's digital signature to provide unforgeability, it should rely on some secret known only to her (her private key!). In order for Alice's digital signature to provide authentication, it should depend on the message in some way; there should be a cryptographic link between the signature and the message.

Most public key cryptosystems can be employed as digital signature schemes. We require the following conditions on the cryptosystem. First, we have the usual condition that is required of all cryptosystems, recall (8.1): For each message M and each public key k_e, there exists a private key k_d for which

$$d(e(M, k_e), k_d) = M. \tag{10.1}$$

Second, we require an additional condition: For each "message" M and each private key k_d, there exists a public key k_e for which

$$e(d(M, k_d), k_e) = M. \tag{10.2}$$

Definition 10.1.1 (Generic Digital Signature Scheme) Let

$$\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle$$

be a public key cryptosystem in which (10.1) and (10.2) hold.

Step 1. Alice establishes her public key k_e and trapdoor (private key) k_d.

Step 2. Let M be a message that Alice intends to send to Bob. Using her private key k_d, Alice signs the message by computing $S = d(M, k_d)$.

Step 3. Alice sends the message, signature pair (M, S) to Bob.

Step 4. Using Alice's public key k_e, Bob then computes $e(S, k_e)$. If

$$e(S, k_e) = M = e(d(M, k_d), k_e),$$

then Bob is assured that message M originated with Alice; he accepts. If

$$e(S, k_e) \neq M = e(d(M, k_d), k_e),$$

then Bob rejects; there is no message authentication.

□

Authentication is achieved since property (10.2) holds; only those who possess Alice's trapdoor can sign M, thereby guaranteeing that the original message M is recovered with Alice's public key.

10.2 The RSA Digital Signature Scheme

Can the RSA cryptosystem be used to create a digital signature scheme? We check that (10.2) holds.

Proposition 10.2.1 *Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle$ be the RSA public key cryptosystem. For each $M \in \mathcal{M}$ and $k_d \in \mathcal{K}_d$, there exists an element $k_e \in \mathcal{K}_e$ so that*

$$e(d(M, k_d), k_e) = M.$$

Proof Let (s, n) be an RSA public key and let t be the corresponding RSA private key. Then for each M, $0 \le M \le n - 1$,

$$\begin{aligned} e(d(M, t), (s, n)) &= e((M^t \bmod n), (s, n)) \\ &= ((M^t \bmod n)^s \bmod n) \\ &= (M^{ts} \bmod n). \end{aligned}$$

There exists an integer a with

$$st = 1 + a(p - 1)(q - 1),$$

hence

$$(M^{ts} \bmod p) = (M M^{a(p-1)(q-1)} \bmod p) = (M \bmod p)$$

by Fermat's Little Theorem. Similarly,

$$(M^{ts} \bmod q) = (M M^{a(p-1)(q-1)} \bmod q) = (M \bmod q).$$

Thus by the Chinese Remainder Theorem, (10.2) holds. □

Definition 10.2.2 (RSA Digital Signature Scheme)

Step 1. Alice establishes an RSA public key (s, n) and private key t.

Step 2. Let M be a message that Alice intends to send to Bob. Using her private key t, Alice signs the message by computing $S = d(M, t) = (M^t \bmod n)$.

Step 3. Alice sends the message, signature pair (M, S) to Bob.

Step 4. Using Alice's public key (s, n), Bob then computes $e(S, (s, n)) = (S^s \bmod n)$. If $e(S, (s, n)) = M$, then Bob is assured that message M

originated with Alice; he accepts. If $e(S, (s, n)) \neq M$, then Bob rejects; there is no message authentication.

□

The RSA digital signature scheme is effective because $e(x, (s, n))$ has the qualities of a one-way trapdoor function: It is essentially impossible for Malice to find x so that

$$M = e(x, (s, n)) = (x^s \bmod n)$$

without knowledge of Alice's trapdoor t.

Example 10.2.3 Suppose Alice's RSA public key is chosen to be $(s, n) = (1003, 85651)$ and her private key is computed to be $t = 13507$. (Note that $85651 = pq = (97)(883)$.)

She signs the message $M = M_1 M_2 = 2\ 100$ by computing $S = S_1 S_2$ with

$$S_1 = d(2, 13507) = 2^{13507} \bmod 85651 = 37736.$$

$$S_2 = d(100, 13507) = 100^{13507} \bmod 85651 = 58586.$$

Alice then sends the message, signature pair

$$(M, S) = (2\ 100, 37736\ 58586)$$

to Bob. Bob can verify that M came from Alice by computing

$$e(S, (1003, 85651)) = M,$$

thus:

$$\begin{aligned} M_1 &= e(S_1, (s, n)) \\ &= e(37736, (1003, 85651)) \\ &= 37736^{1003} \bmod 85651 \\ &= 2, \end{aligned}$$

$$\begin{aligned} M_2 &= e(S_2, (s, n)) \\ &= e(58586, (1003, 85651)) \\ &= 58586^{1003} \bmod 85651 \\ &= 100. \end{aligned}$$

□

Example 10.2.4 Assume that Alice's public key is $(1003, 85651)$ and her private key is 13507 as in Example 10.2.3. Suppose that Bob receives the message, signature pair

$$(M, S) = (345, 7219)$$

presumably from Alice.

Should Bob conclude that Alice wrote the message? Bob computes

$$e(7219, (1003, 85651)) = 7219^{1003} \bmod 85651 = 83169.$$

Since $83169 \neq 345$ Bob rejects!

□

10.3 Signature with Privacy

In the RSA DSS, Alice sends the message, signature pair (M, S) with M in plaintext. Thus anyone intercepting the transmission can read Alice's message. If Alice wants to sign an encrypted message, she should use a "signature with privacy scheme."

Definition 10.3.1 (RSA Signature with Privacy)

Step 1. Alice establishes a public key and private key pair (s_A, n_A), t_A. Bob establishes a public key and private key pair (s_B, n_B), t_B.

Step 2. Alice signs the message M as

$$S = d(M, t_A) = (M^{t_A} \bmod n_A).$$

Step 3. Alice then encrypts the message M using Bob's public key:

$$C_1 = e(M, (s_B, n_B)) = (M^{s_B} \bmod n_B).$$

She also encrypts the signature S using Bob's public key:

$$C_2 = e(S, (s_B, n_B)) = (S^{s_B} \bmod n_B).$$

She then sends the pair (C_1, C_2) to Bob.

Step 4. Bob uses his private key t_B to recover

$$M = d(C_1, t_B) = (C_1^{t_B} \bmod n_B).$$

He also recovers

$$S = d(C_2, t_B) = (C_2^{t_B} \bmod n_B).$$

Step 5. Finally, using Alice's public key (s_A, n_A) he authenticates the message by verifying that

$$e(S, (s_A, n_A)) = e(d(M, t_A), (s_A, n_A)) = M.$$

□

10.4 Security of Digital Signature Schemes

Suppose Alice and Bob are using a digital signature scheme for message authentication. Malice can attack (or break) the digital signature scheme by producing forgeries. Specifically, a **forgery** is a message, signature pair (M, S) for which S is Alice's signature of the message M.

Essentially, there are two types of forgeries. An **existential forgery** is a forgery of the form (M, S) for some $M \in \mathcal{M}$. A **selective forgery** is a forgery of the form (M, S) in which M is chosen by Malice.

Malice will produce forgeries by engaging in several types of attacks. A **direct attack** is an attack in which Malice only knows Alice's public key. A **known-signature attack** is an attack in which Malice knows Alice's public key together with a set of message, signature pairs

$$(M_1, S_1), (M_2, S_2), \dots, (M_r, S_r)$$

signed by Alice. A **chosen-message attack** is an attack in which Malice knows Alice's public key and has (somehow) convinced her to sign a set of messages $M_1, M_2, \dots, M_r$ that Malice has chosen.

How secure is the RSA Digital Signature Scheme?

Proposition 10.4.1 *The RSA DSS is existentially forgeable under a direct attack.*

Proof We show that knowing only Alice's public key, Malice can forge at least one message M. Suppose that Alice's RSA public key is (s, n) and her private key is t. Malice chooses any integer N, $0 \le N \le n - 1$, and computes

$$M = e(N, (s, n)) = (N^s \bmod n).$$

Malice then sends the message, signature pair (M, N) to Bob. Bob will conclude (incorrectly) that Alice signed the message since

$$e(N, (s, n)) = (N^s \bmod n) = M.$$

□

Example 10.4.2 Suppose Alice's RSA public key is $(s, n) = (1003, 85651)$ and her private key is $t = 13507$. Let $N = 600$ and compute

$$M = (600^{1003} \bmod 85651) = 34811.$$

Then the message, signature pair $(M, N) = (34811, 600)$ is correctly signed by Alice: We have

$$(34811^{13507} \bmod 85651) = 600.$$

□

Proposition 10.4.3 *The RSA DSS is selectively forgeable under a chosen-message attack.*

Proof Let M, $0 \le M \le n-1$ be a message chosen by Malice, and let R be an integer, $1 \le R \le n-1$, with $\gcd(R, n) = 1$. Suppose that Malice gets Alice to sign the messages $M \cdot R$ and R^{-1}, yielding the message, signature pairs $(M \cdot R, T)$ and (R^{-1}, U). Malice computes $V = ((T \cdot U) \bmod n)$ and sends the message, signature pair (M, V) to Bob. Upon receipt, Bob accepts the message M since it has a valid signature. Indeed,

$$\begin{aligned} d(M, t) &= (M^t \bmod n) \\ &= ((M \cdot (R \cdot R^{-1}))^t \bmod n) \\ &= (((M \cdot R) \cdot R^{-1})^t \bmod n) \\ &= ((M \cdot R)^t (R^{-1})^t \bmod n) \\ &= ((T \cdot U) \bmod n) \\ &= V. \end{aligned}$$

Bob concludes (incorrectly) that Alice signed the message M. □

10.5 Hash Functions and DSS

In this section we introduce cryptographic hash functions as a way to improve the efficiency and security of the RSA DSS.

Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K}_e, \mathcal{K}_d \rangle$ be a public key cryptosystem to be used as a DSS.

Definition 10.5.1 A **hash function** is a function

$$h : \mathcal{M} \to \mathcal{M}$$

in which the image $h(M)$, called the **digest** of message M, is significantly smaller than M.

Definition 10.5.2 A hash function $h : \mathcal{M} \to \mathcal{M}$ is **regular** if each digest $y \in h(\mathcal{M}) = \{h(M) : M \in \mathcal{M}\}$ has the same number of preimages in $\mathcal{M}$.

Definition 10.5.3 A hash function $h : \mathcal{M} \to \mathcal{M}$ is **cryptographic** if h (ideally) satisfies the following conditions:

(i) h is a "one-way" function, i.e., given $M \in \mathcal{M}$, the digest $h(M)$ is easy to compute, and given $h(M)$, it is hard to find a preimage M' for which $h(M') = h(M)$,

(ii) h destroys any algebraic relationship between messages M_1, M_2 and their digests $h(M_1)$, $h(M_2)$, i.e., h is "non-homomorphic."

(iii) h is "collision resistant," i.e., it is infeasible to find $M_1, M_2 \in \mathcal{M}$ with $h(M_1) = h(M_2)$.

In what follows, we give examples of hash functions and show that they are cryptographic.

10.5.1 The Discrete Log Family

A **discrete log hash function** is a hash function of the form

$$h : \mathbb{Z}_n \to U(\mathbb{Z}_p),$$

where $n = pq$ is an RSA modulus.

The hash function is constructed as follows. Let (s, n) be Alice's RSA public key and let t be her private key with $n = pq$ for primes p, q, with p a **safe prime**, i.e., $p = 2r + 1$ where r is prime. Suppose that $r^2 > n = pq$. Let g_1, g_2 be primitive roots modulo p and let

$$f : \mathbb{Z}_r \times \mathbb{Z}_r \to U(\mathbb{Z}_p)$$

be the function given as

$$f(x, y) = (g_1^x g_2^y \bmod p)$$

for $x, y \in \mathbb{Z}_r$ [10], [59, p. 180].

Then we can produce a digest of any message M, $0 \le M \le n - 1$ as follows: List the elements of $\mathbb{Z}_r \times \mathbb{Z}_r$ in genealogical order

$$(0, 0), (0, 1), (0, 2), \ldots, (0, r - 1), (1, 0), \ldots, (r - 1, r - 1). \qquad (10.3)$$

Every M in $\mathbb{Z}_r$ has the form $M = ri + j$ where $0 \le j < r$ and $i < r$ (since $n < r^2$). So we have an embedding

$$\lambda : \mathbb{Z}_n \to \mathbb{Z}_r \times \mathbb{Z}_r$$

by $\lambda(M) = \lambda(ri + j) = (i, j)$, the $(M + 1)$st element of the genealogical list (10.3). Then the hash function is the composite function

$$h = (f \circ \lambda) : \mathbb{Z}_n \to U(\mathbb{Z}_p),$$

where $h(M) = h(ri + j) = (g_1^i g_2^j \bmod p)$.

Example 10.5.4 Choose primes $p = 47$, $q = 11$ and RSA modulus $n = 47 \cdot 11 = 517$. Let $(s, n) = (149, 517)$ be Alice's public key and let $t = 389$ be her private key. Choose $g_1 = 5$ and $g_2 = 31$, which are primitive roots modulo 47. Note that $47 = 2 \cdot 23 + 1$ is a safe prime with $r = 23$, $23^2 > 517$.

Thus there is an embedding $\mathbb{Z}_{517} \to \mathbb{Z}_{23} \times \mathbb{Z}_{23}$ and a corresponding hash function

$$h : \mathbb{Z}_{517} \to U(\mathbb{Z}_{47}).$$

For instance, to compute the digest $h(100)$, we first embed $100 \in \mathbb{Z}_{517}$ into $\mathbb{Z}_{23} \times \mathbb{Z}_{23}$ to obtain $(4, 8) \in \mathbb{Z}_{23} \times \mathbb{Z}_{23}$. Thus

$$h(100) = f(4, 8) = ((5^4 \cdot 31^8) \bmod 47) = 24.$$

Likewise, we compute

$$h(258) = f(11, 5) = ((5^{11} \cdot 31^5) \bmod 47) = 16.$$

□

Proposition 10.5.5 *Let $h : \mathbb{Z}_n \to U(\mathbb{Z}_p)$ be a discrete log hash function. Assume that h is regular and assume that p is at least a* 160*-bit prime. Then h is a cryptographic hash function.*

Proof The number of possible digests is $|U(\mathbb{Z}_p)| = p - 1$, which is significantly smaller that $n = pq$. So it remains to show that the conditions of Definition 10.5.3 are satisfied.

For (i): Assuming that the Discrete Logarithm Assumption holds, h is a one-way function.

For (ii): The discrete log function h is not a multiplicative homomorphism. For instance,

$$h(100 \cdot 258) = h(44) = f(1, 21) = ((5^1 \cdot 31^{21}) \bmod 47) = 2,$$

while

$$h(100) \cdot h(258) = ((24 \cdot 16) \bmod 47) = 8.$$

Finally, we check that h is collision resistant. Because h is regular, if we choose m messages at random from $\mathcal{M}$, then we are essentially choosing m digests at random from $h(\mathcal{M})$.

By the first Collision Theorem (Proposition 2.3.1), in order to ensure the likelihood of at least one collision among the randomly selected digests, we need to choose at least $m = 1 + \lceil \sqrt{1.4p} \rceil$ messages in $\mathcal{M}$, and then compute their digests in $h(\mathcal{M})$. But since p is at least a 160-bit prime, m is at least $\sqrt{p} \approx 2^{80}$.

So, in order to make a collision feasible (i.e., $\Pr > 1/2$), our computer must perform at least 2^{80} operations (message selections), which is beyond the number of operations that a computer can perform in a reasonable amount of time. Thus h is collision resistant.

□

10.5.2 The MD-4 Family

The **MD-4** family is a large collection of hash functions of the form

$$h : \{0, 1\}^m \to \{0, 1\}^t,$$

$m > t$, computed using multiple rounds and steps. These include MD-4, MD-5, and SHA-1.

For instance, MD-4 is of the form

$$h : \{0, 1\}^{512} \to \{0, 1\}^{128},$$

and contains 3 rounds of 16 steps each; SHA-1 consists of 4 rounds of 20 steps and has the form

$$h : \{0, 1\}^{512} \to \{0, 1\}^{160}.$$

We illustrate how the MD-4 family works by giving an example of a simplified single round MD-4-type hash function.

Our MD-4 hash function

$$h : \{0, 1\}^{16} \to \{0, 1\}^8$$

involves parameters (s_1, s_2, f, σ) as follows: s_1, s_2 are two fixed half-bytes (i.e., elements of $\{0, 1\}^4$), the round function is $f(x, y) = x \oplus y$, and σ is a fixed permutation in S_4,

$$\sigma = \begin{pmatrix} 0 & 1 & 2 & 3 \\ 1 & 2 & 3 & 0 \end{pmatrix}.$$

Let $M \in \{0, 1\}^{16}$ be a message. The digest $h(M)$ is computed using the algorithm:

Algorithm 10.5.6 (MD-4-Round)

```
Input:     a message $M \in \{0, 1\}^{16}$, divided into 4 half-bytes,
           $M_0, M_1, M_2, M_3$.
Output:    the digest $h(M) \in \{0, 1\}^8$.
Algorithm:
           $(a, b) \leftarrow (s_1, s_2)$
           for $i = 0$ to 3 do
              $t \leftarrow f(a, b) \oplus M_i$
              $(a, b) \leftarrow (b, \sigma(t))$
           next $i$
           output $ab$
```

Example 10.5.7 Let $s_1 = 1111$, $s_2 = 0000$. Let

$$h : \{0, 1\}^{16} \to \{0, 1\}^8$$

be the single round MD-4 hash function. We compute $h(M)$ where

$$M = 1001101111000010.$$

In this case, $M_0 = 1001$, $M_1 = 1011$, $M_2 = 1100$, $M_3 = 0010$.

For $i = 0$,

$$t = f(1111, 0000) \oplus 1001 = 0110$$

$$(a, b) = (0000, \sigma(0110)) = (0000, 1100).$$

For $i = 1$,

$$t = f(0000, 1100) \oplus 1011 = 0111$$

$$(a, b) = (1100, \sigma(0111)) = (1100, 1110).$$

For $i = 2$,

$$t = f(1100, 1110) \oplus 1100 = 1110$$

$$(a, b) = (1110, \sigma(1110)) = (1110, 1101).$$

For $i = 3$,

$$t = f(1110, 1101) \oplus 0010 = 0001$$

$$(a, b) = (1101, \sigma(0001)) = (1101, 0010).$$

Thus

$$h(1001101111000010) = 11010010.$$

□

Proposition 10.5.8 *Let $h : \{0, 1\}^m \to \{0, 1\}^t$, $m \gg t$, be an MD-4 hash function with $t \geq 160$. Assume that h is regular. Then h is a cryptographic hash function.*

Proof The number of possible digests is 2^t, which is significantly smaller than the number of messages, which is 2^m.

We show that the conditions of Definition 10.5.3 are satisfied.

The important condition is collision resistance. Because h is regular, if we choose n messages at random from $\mathcal{M} = \{0, 1\}^m$, then we are essentially choosing n digests at random from $h(\{0, 1\}^m)$.

By the first Collision Theorem (Proposition 2.3.1), a collision is likely if we choose at least $n = 1 + \lceil\sqrt{1.4 \cdot 2^t}\rceil$ messages in $\{0, 1\}^m$. But since $t \geq 160$, n is at least $\sqrt{2^{160}} = 2^{80}$.

So, in order to make a collision feasible (i.e., $\Pr > 1/2$), our computer must perform at least 2^{80} operations (message selections), which is beyond the number of operations that a computer can perform in a reasonable amount of time. Thus h is collision resistant.

□

10.5.3 Hash-Then-Sign DSS

Hash functions improve the efficiency and security of RSA DSS, provided that the hash function is cryptographic, i.e., has some (or all) of the properties of Definition 10.5.3.

In a "hash-then-sign DSS" the digest of a message is signed instead of the message.

Definition 10.5.9 (Hash-Then-Sign RSA DSS)

Step 1. Alice and Bob agree to use a discrete log hash function $h : \mathbb{Z}_n \to U(\mathbb{Z}_p)$, where $n = pq$ is the RSA modulus.

Step 2. Alice establishes a public key and private key pair (s_A, n_A), t_A.

Step 3. Alice signs the message $M \in \mathbb{Z}_n$ by first computing its digest $h(M)$ and then signing the digest as $S = d(h(M), t_A) = (h(M)^{t_A} \bmod n_A)$. She then sends to Bob the pair (M, S) to Bob.

Step 4. Bob computes $h(M)$ and using Alice's public key, he authenticates the message by verifying that

$$e(S, (s_A, n_A)) = e(d(h(M), t_A), (s_A, n_A)) = h(M).$$

□

The hash-then-sign RSA DSS described above is more efficient than RSA DSS since Alice need only sign a much shorter digest $h(M)$.

Moreover, the collision resistance of h ensures that the DSS is more secure than ordinary RSA DSS. Suppose that S is Alice's signature of the digest $h(M)$. Since it is infeasible for Malice to obtain a collision $h(N) = h(M)$, it is not possible for Malice to obtain a forgery (S, N), where N is a message that is signed by Alice.

Finally, the non-homomorphism property of h makes it unlikely that the chosen-message attack of Proposition 10.4.3 will succeed in producing a forgery.

10.6 Exercises

1. Suppose that Alice is using the RSA Digital Signature Scheme with public key $(s, n) = (631, 7991)$ and private key $t = 1471$ to sign messages.

   (a) Bob receives the message, signature pair

   $$(M, S) = (7\ 4\ 11\ 11\ 14,\ 7374\ 2810\ 175\ 175\ 612)$$

   presumably from Alice. Should Bob accept the message?

   (b) Acting as Malice, produce a message, signature pair (M, S) that is a forgery of Alice's signature.

2. Most digital signature schemes are obtained from public key cryptosystems. Is it possible to use a symmetric key cryptosystem to create a digital signature scheme (in an analogous manner)?
3. Consider the Hash-Then-sign RSA DSS with hash function

   $$h : \mathbb{Z}_{517} \to U(\mathbb{Z}_{47})$$

   of Example 10.5.4.

   (a) Compute the digest $h(312)$.

   (b) Suppose that Bob receives the message, signature pair (312, 175) presumably from Alice. Should he accept?

4. Given the discrete log hash function $h : \mathbb{Z}_n \to U(\mathbb{Z}_p)$, p, q prime, $n = pq$, find the minimum number of values $h(M)$ required so that the probability of a collision is $> \frac{3}{4}$.
5. Let $h : \{0, 1\}^{16} \to \{0, 1\}^8$ be the MD-4-type hash function of Example 10.5.7.

   (a) Compute $h(1^{16})$. (Here 1^{16} denotes $\underbrace{111 \cdots 1}_{16}$.)

   (b) Suppose that 10 values $h(M)$ are computed. What is the probability that there is at least one collision?

6. Show that the hash function of Example 10.5.7 is not a homomorphism with respect to bit-wise addition modulo 2, i.e., show that

$$h(M \oplus N) \neq h(M) \oplus h(N)$$

for some $M, N \in \{0, 1\}^{16}$.

7. Suppose that message M is a 25 letter sequence of English words. Thus, M can be encoded over the alphabet $\{0, 1\}$ as a string of $25 \cdot \log_2(26) = 118$ bits.

Let

$$h : \{0, 1\}^{118} \to \{0, 1\}^{59}$$

be a hash function that is regular, i.e., for each $y \in \{0, 1\}^{59}$, there are exactly $\{0, 1\}^{59}$ strings $x \in \{0, 1\}^{118}$ for which $h(x) = y$.

(a) Assuming h regular, estimate the number of strings $N \in \{0, 1\}^{118}$ for which N is the encoding of a meaningful 25 letter sequence of English words.

(b) Estimate the number of strings N satisfying part (a), and for which $h(N) = h(M)$.

Hint: From Section 3.4, we have

$$\lim_{n\to\infty} H_n/n = 1.5,$$

thus, we can assume that $H_{25}/25 = 1.5$, hence $H_{25} = 37.5$. Use this fact to estimate the probability that a string in $\{0, 1\}^{118}$ is the encoding of a meaningful sequence in English.

Chapter 11
Key Generation

The Vernam cipher of Section 8.5 is the only cryptosystem that has perfect secrecy (Definition 8.5.8), because the key is a random sequence of bits of the same length as the message (Proposition 8.5.9).

The problem studied in this chapter is to describe generators for “pseudorandom” sequences of bits and try to determine how close they come to being truly random sequences.

The pseudorandom sequences are produced by bit generators of the form

$$G : \{0, 1\}^l \rightarrow \{0, 1\}^m,$$

$m \gg l$, where the random seed $k \in \{0, 1\}^l$ is used to generate longer bit strings of length m.

We can then use a pseudorandom sequence as a key stream in a stream cipher; the stream cipher together with the pseudorandom key stream then exhibits “almost perfect secrecy,” approximating the perfect secrecy of the Vernam cipher.

11.1 Linearly Recursive Sequences

In this section we show how a sequence of bits can be generated as a linearly recursive sequence. Under certain conditions, these sequences are periodic with period s. The first s bits of the sequence can be viewed as a key of length s in the Vigenère cipher over the alphabet $\Sigma = \{0, 1\}$.

We begin with a review of the general theory of linearly recursive sequences over a field.

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_11

Definition 11.1.1 Let K be a field, and let $l > 0$ be a positive integer. An l**th-order linearly recursive sequence in** K is a sequence $\{s_n\}_{n\geq 0}$ for which

$$s_{n+l} = a_{l-1}s_{n+l-1} + a_{l-2}s_{n+l-2} + \cdots + a_0 s_n + a \tag{11.1}$$

for some elements $a, a_0, a_1, a_2, \ldots, a_{l-1} \in K$ and all $n \geq 0$. The relation (11.1) is the **recurrence relation** of the sequence.

For $n \geq 0$, the vector $\mathbf{s}_n = (s_n, s_{n+1}, s_{n+2}, \ldots, s_{n+l-1})$ is the n**th state vector** of $\{s_n\}$; $\mathbf{s}_0 = (s_0, s_1, s_2, \ldots, s_{l-1})$ is the **initial state vector**. A linearly recursive sequence is completely determined by specifying the recurrence relation (11.1) and initial state vector.

Example 11.1.2 Let $K = \mathbb{Q}$, and let $l = 3$. Then the recurrence relation

$$s_{n+3} = 2s_{n+1} + s_n, \quad n \geq 0,$$

and initial state vector $\mathbf{s}_0 = (2, 0, 1)$ define the 3rd-order linearly recursive sequence

$$\{s_n\} = 2, 0, 1, 2, 2, 5, 6, 12, \ldots$$

The linearly recursive sequence $\{s_n\}$ is **homogeneous** if $a = 0$. The sequence $\{s_n\}$ is **eventually periodic** if there exist integers $N \geq 0, t > 0$ for which $s_{n+t} = s_n$, for all $n \geq N$. The sequence $\{s_n\}$ is **periodic** if $\{s_n\}$ is eventually periodic with $N = 0$; that is, $\{s_n\}$ is periodic if there exists an integer $t > 0$ so that $s_{n+t} = s_n$ for all $n \geq 0$.

Suppose $\{s_n\}$ is eventually periodic. Then the smallest positive integer r for which $s_{n+r} = s_n$ for all $n \geq N$ is the **period** of $\{s_n\}$.

We assume that all linearly recursive sequences are homogeneous.

Homogeneous linearly recursive sequences can be described in terms of matrices. Let $\{s_n\}$ be an lth-order linearly recursive sequence with recurrence relation (11.1) ($a = 0$). Let A be the $l \times l$ matrix defined as

$$A = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ a_0 & a_1 & a_2 & \cdots & a_{l-2} & a_{l-1} \end{pmatrix}.$$

Let M^T denote the transpose of a matrix M.

Proposition 11.1.3 *With A defined as above,*

$$\mathbf{s}_n^T = A^n \mathbf{s}_0^T$$

for all $n \geq 0$.

Proof Use induction on n. The trivial case is $n = 0$: $\mathbf{s}_0^T = I_l\mathbf{s}_0^T$, where I_l denotes the $l \times l$ identity matrix. For the induction hypothesis, assume that $\mathbf{s}_{n-1}^T = A^{n-1}\mathbf{s}_0^T$. Then $A\mathbf{s}_{n-1}^T = AA^{n-1}\mathbf{s}_0^T$, hence $\mathbf{s}_n^T = A^n\mathbf{s}_0^T$.

□

The matrix A is the **matrix** of the homogeneous linearly recursive sequence. Let $\{s_n\}$ be an lth-order linearly recursive sequence with matrix A. The **characteristic polynomial** of $\{s_n\}$ is the characteristic polynomial of A in the usual sense; that is, the characteristic polynomial of $\{s_n\}$ is

$$f(x) = \det(xI_l - A).$$

Proposition 11.1.4 *Let $\{s_n\}$ be an lth-order linearly recursive sequence defined by (11.1). Let $f(x)$ be the characteristic polynomial of $\{s_n\}$. Then*

$$f(x) = x^l - a_{l-1}x^{l-1} - a_{l-2}x^{l-2} - \cdots - a_0 \in K[x].$$

Proof We proceed by induction on the order l of the linearly recursive sequence. For a detailed proof, see [61, Proposition 5.13].

□

Let K be a field, let

$$g(x) = b_tx^t + b_{t-1}x^{t-1} + \cdots + b_1x + b_0$$

be a polynomial in $K[x]$, and let A be a matrix in $\mathrm{Mat}_l(K)$. Then by the evaluation $g(A)$ we mean the linear combination of matrices

$$g(A) = b_tA^t + b_{t-1}A^{t-1} + \cdots + b_1A + b_0I_l.$$

The polynomial $g(x)$ **annihilates** A if $g(A) = 0$, where 0 denotes the $l \times l$ zero matrix. The set of all polynomials in $K[x]$ that annihilate A is a non-zero ideal J of $K[x]$ (Section 11.5, Exercise 6). By Proposition 7.1.1, J is a principal ideal, i.e., $J = (m(x))$, for some monic polynomial $m(x)$. The polynomial $m(x)$ is the **minimal polynomial** of A.

Proposition 11.1.5 *Let A be a matrix in $Mat_l(K)$ of the form*

$$A = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ a_0 & a_1 & a_2 & \cdots & a_{l-2} & a_{l-1} \end{pmatrix}.$$

Then the characteristic polynomial of A is the minimal polynomial of A.

Proof The proof of this well-known result uses the Cayley–Hamilton theorem; see [24, Section 7.1, Corollary].

□

An application of Proposition 11.1.5 says that the characteristic polynomial of a linearly recursive sequence is the minimal polynomial of its matrix.

For the remainder of this section, we are going to consider lth-order linearly recursive sequences $\{s_n\}$ over the field $K = \mathbb{F}_{p^m}$, the Galois field of p^m elements.

Recall that a polynomial $f(x) \in \mathbb{F}_{p^m}$ of degree l is primitive if $f(x)$ is irreducible and there exists a root α of $f(x)$ that generates the cyclic group of units $U(\mathbb{F}_{p^m}(\alpha)) = \mathbb{F}_{p^m}(\alpha)^\times$.

We will prove the fundamental result on lth-order linearly recursive sequences in $\mathbb{F}_{p^m}$: if the characteristic polynomial of the sequence is primitive, then the sequence is periodic with maximal period $r = p^{ml} - 1$. We begin by showing that every linearly recursive sequence in a finite field is periodic.

Proposition 11.1.6 *Let $\{s_n\}$ be an lth-order linearly recursive sequence in $\mathbb{F}_{p^m}$. Then $\{s_n\}$ is eventually periodic with period $1 \leq r \leq p^{ml} - 1$.*

Proof If $\mathbf{s}_i = \underbrace{(0, 0, \ldots, 0)}_{l}$ for any $i \geq 0$, then $\{s_n\}$ is eventually periodic with $N = i$ and $r = 1 \leq p^{ml} - 1$. So we assume that $\mathbf{s}_n$ is not the zero vector for all $n \geq 0$.

Observe that there are precisely $p^{ml} - 1$ distinct l-tuples of elements in $\mathbb{F}_{p^m}$ other than the zero vector. Thus there exist integers i, j, $0 \leq i < j \leq p^{ml} - 1$, so that $\mathbf{s}_j = \mathbf{s}_i$. We claim that $\mathbf{s}_{n+j-i} = \mathbf{s}_n$ for all $n \geq i$. To prove this claim, we proceed by induction on $n \geq i$, with the trivial case $n = i$ already established. For the induction hypothesis, we assume that $\mathbf{s}_{n+j-i} = \mathbf{s}_n$ holds for $n = i + \omega$, where $\omega \geq 0$ is a fixed integer. Thus $\mathbf{s}_{\omega+j} = \mathbf{s}_{i+\omega}$, and hence

$$(s_{\omega+j}, s_{\omega+j+1}, \ldots, s_{\omega+j+l-1}) = (s_{i+\omega}, s_{i+\omega+1}, \ldots, s_{i+\omega+l-1}). \tag{11.2}$$

Consequently

$$s_{\omega+1+j+\eta} = s_{i+\omega+1+\eta},$$

for $\eta = 0, 1, 2, \ldots, l-2$. Moreover, from (11.2) and the recurrence relation (11.1) one obtains

$$s_{\omega+1+j+\eta} = s_{i+\omega+1+\eta},$$

for $\eta = l - 1$. Thus

$$\begin{aligned}&(s_{(\omega+1)+j}, s_{(\omega+1)+j+1}, \ldots, s_{(\omega+1)+j+l-1})\\&\quad = (s_{i+(\omega+1)}, s_{i+(\omega+1)+1}, \ldots, s_{i+(\omega+1)+l-1}),\end{aligned}$$

which yields $\mathbf{s}_{n+j-i} = \mathbf{s}_n$ for $n = i + (\omega + 1)$. The proof by induction is complete and hence $s_{n+(j-i)} = s_n$ for all $n \geq i$. Thus $\{s_n\}$ is eventually periodic with $N = i$ and period r satisfying $1 \leq r \leq j - i \leq p^{ml} - 1$.

□

Proposition 11.1.7 *Let $\{s_n\}$ be an lth-order linearly recursive sequence in $\mathbb{F}_{p^m}$ with recurrence relation (11.1). If $a_0 \neq 0$, then $\{s_n\}$ is periodic with period $1 \leq r \leq p^{ml} - 1$.*

Proof By Proposition 11.1.6, $\{s_n\}$ is eventually periodic with period $1 \leq r \leq p^{ml} - 1$. Suppose N_0 is the smallest integer for which $s_{n+r} = s_n$ for all $n \geq N_0$. If $N_0 = 0$, then $\{s_n\}$ is periodic. So we assume that $N_0 \geq 1$. But now from (11.1),

$$\begin{aligned} s_{N_0-1+r} &= a_0^{-1}\left(s_{(N_0-1+r)+l} - a_{l-1}s_{(N_0-1+r)+l-1} - \cdots - a_1 s_{(N_0-1+r)+1}\right) \\ &= a_0^{-1}\left(s_{N_0+l-1+r} - a_{l-1}s_{N_0+l-2+r} - \cdots - a_1 s_{N_0+r}\right) \\ &= a_0^{-1}\left(s_{N_0+l-1} - a_{l-1}s_{N_0+l-2} - \cdots - a_1 s_{N_0}\right) \\ &= a_0^{-1}\left(s_{(N_0-1)+l} - a_{l-1}s_{(N_0-1)+l-1} - \cdots - a_1 s_{(N_0-1)+1}\right) \\ &= s_{N_0-1}. \end{aligned}$$

Thus $s_{n+r} = s_n$ for all $n \geq N_0 - 1$, which contradicts the minimality of N_0.

□

Let $\{s_n\}$ be an lth-order linearly recursive sequence with recurrence relation (11.1). In what follows we assume that $a_0 \neq 0$ so that $\{s_n\}$ is periodic with period r, $1 \leq r \leq p^{ml} - 1$. Let

$$A = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ a_0 & a_1 & a_2 & \cdots & a_{l-2} & a_{l-1} \end{pmatrix}$$

be the matrix of $\{s_n\}$. Observe that $\det(A) = (-1)^{l+1}a_0$ and, consequently, A is an invertible element of $GL_l(\mathbb{F}_{p^m})$, a finite group. Hence A has finite order in $GL_l(\mathbb{F}_{p^m})$.

Proposition 11.1.8 *Let $\{s_n\}$ be an lth-order linearly recursive sequence in $\mathbb{F}_{p^m}$. Then the period r of $\{s_n\}$ divides the order of A in $GL_l(\mathbb{F}_{p^m})$.*

Proof Let w be the order of A. Then by Proposition 11.1.3

$$\mathbf{s}_{n+w}^T = A^{n+w}\mathbf{s}_0^T = A^n\mathbf{s}_0^T = \mathbf{s}_n^T,$$

for all $n \geq 0$. Thus $r \leq w$. There exist integers t, u so that $w = rt + u$ with $0 \leq u < r, t > 0$. Now, for all $n \geq 0$,

$$\mathbf{s}_n^T = \mathbf{s}_{n+w}^T = \mathbf{s}_{n+rt+u}^T = \mathbf{s}_{n+r+r(t-1)+u}^T = \mathbf{s}_{n+r(t-1)+u}^T,$$

$$\mathbf{s}_{n+r(t-1)+u}^T = \mathbf{s}_{n+r+r(t-2)+u}^T = \mathbf{s}_{n+r(t-2)+u}^T,$$

$$\mathbf{s}_{n+r(t-2)+u}^T = \mathbf{s}_{n+r+r(t-3)+u}^T = \mathbf{s}_{n+r(t-3)+u}^T,$$

$$\vdots$$

$$\mathbf{s}_{n+r+u}^T = \mathbf{s}_{n+u}^T.$$

Consequently, $\mathbf{s}_n^T = \mathbf{s}_{n+u}^T$. Since r is the period of $\{s_n\}$, $u = 0$, and so $r \mid w$.

□

Proposition 11.1.9 *Let $\{s_n\}$ be an lth-order linearly recursive sequence in $\mathbb{F}_{p^m}$. Let A be the matrix of $\{s_n\}$, and let $f(x)$ be the characteristic polynomial of A. Assume that $a_0 \neq 0$. Then* order$(f(x))$ *equals the order of the matrix A in the finite group $GL_l(\mathbb{F}_{p^m})$.*

Proof Recall that order$(f(x))$ is the smallest positive integer e for which $f(x) \mid x^e - 1$, cf. Proposition 7.3.8.

The order of A is the smallest positive integer w so that $A^w - I_l = 0$. That is, w is the smallest positive integer so that $x^w - 1$ is in the annihilator ideal of A. Consequently, the minimal polynomial $m(x) \mid x^w - 1$. By Proposition 11.1.5, $f(x) = m(x)$, and so $f(x) \mid x^w - 1$. Thus order$(f(x)) \leq w$. If order$(f(x)) < w$, then there exists an integer q, $0 < q < w$, with $f(x) \mid x^q - 1$. Consequently, $A^q - I_l = 0$, which contradicts our assumption that w is the order of A. It follows that order$(f(x)) = w$.

□

Proposition 11.1.10 *Let $\{s_n\}$ be an lth-order linearly recursive sequence in $\mathbb{F}_{p^m}$. Let A be the matrix of $\{s_n\}$, and let $f(x)$ be the characteristic polynomial of A. Assume that $a_0 \neq 0$. Let r be the period of $\{s_n\}$. Then $r \mid order(f(x))$.*

Proof Let w be the order of A in $GL_l(\mathbb{F}_{p^m})$. By Proposition 11.1.8, $r \mid w$. By Proposition 11.1.9, $w =$ order$(f(x))$. Thus $r \mid$ order$(f(x))$.

□

Proposition 11.1.11 *Let $\{s_n\}$ be an lth-order linearly recursive sequence in $\mathbb{F}_{p^m}$. Let A be the matrix of $\{s_n\}$, and let $f(x)$ be the characteristic polynomial of A. Assume that $a_0 \neq 0$. Let r be the period of $\{s_n\}$. If $f(x)$ is irreducible over $\mathbb{F}_{p^m}$, then $r = order(f(x))$.*

Proof By Proposition 11.1.10, $r \mid \text{order}(f(x))$. Thus $r \leq \text{order}(f(x))$. By [61, Proposition 5.21],

$$f(x)v(x) = (1 - x^r)w(x)$$

for some $v(x), w(x) \in \mathbb{F}_{p^m}[x]$, $w(x) \neq 0$, and $\deg(w(x)) \leq k - 1$. Thus $f(x) \mid (1 - x^r)w(x)$. Since $f(x)$ is irreducible, either $f(x) \mid 1 - x^r$ or $f(x) \mid w(x)$. Since $\deg(w(x)) < \deg(f(x))$, one has $f(x) \mid 1 - x^r$ and so, $\text{order}(f(x)) \leq r$.

□

Here is the key result regarding linearly recursive sequences.

Theorem 11.1.12 *Let $\{s_n\}$ be an lth-order linearly recursive sequence in $\mathbb{F}_{p^m}$ with characteristic polynomial $f(x)$. Assume that $a_0 \neq 0$ and let r be the period of $\{s_n\}$. If $f(x)$ is primitive over $\mathbb{F}_{p^m}$, then $r = p^{ml} - 1$.*

Proof By Proposition 11.1.11, $r = \text{order}(f(x))$. By Proposition 7.3.12, $\text{order}(f(x))$ is the order of any root α of $f(x)$ in $\mathbb{F}_{p^m}(\alpha)^\times$. Since $f(x)$ is primitive of degree l, the order of α is $p^{ml} - 1$.

□

Example 11.1.13 By Example 7.3.16, $f(x) = x^4 + x + 1$ is a primitive polynomial over $\mathbb{F}_2$. Thus we can apply Theorem 11.1.12 to produce a 4th-order linearly recursive sequence $\{s_n\}$ in $\mathbb{F}_2$ that has maximal period $r = 2^4 - 1 = 15$. From $f(x) = x^4 + x + 1$, we obtain the recurrence relation

$$s_{n+4} = s_{n+1} + s_n, \quad n \geq 0.$$

Choosing the initial state vector $\mathbf{s}_0 = 0110$, we obtain the sequence

$$011010111100010011010111100010011\ldots$$

of maximal period 15. As another example, the initial state vector $\mathbf{s}_0 = 0001$ yields the sequence

$$000100110101111000100110101111000\ldots$$

of period 15. We have the bit generator

$$G : \{0, 1\}^4 \to \{0, 1\}^m,$$

$m \geq 4$, defined as

$$G(\mathbf{s}_0) = s_0 s_1 s_2 s_3 \ldots$$

For instance,

$$G(0110) = 011010111100010011010111100010011\ldots$$

□

Let $\langle \mathcal{M}, \mathcal{C}, e, d, \mathcal{K} \rangle$ denote the Vigenère cipher over the alphabet $\Sigma = \{0, 1\}$. We have $\mathcal{M} = \mathcal{C} = \{0, 1\}^r$ and $\mathcal{K} = \{0, 1\}^s$, where $r, s \geq 1$ are integers. The integer r is the message length, and s is the key length. We take $s = 15$.

Let

$$k = 011010111100010$$

be the first 15 terms of the linearly recursive sequence $G(0110)$ of Example 11.1.13. Then k is a key for the Vigenère cipher.

The message $M = 10110011000101001110$ is encrypted by vertical addition modulo 2:

$$\begin{array}{cccccccccccccccccccc}
1&0&1&1&0&0&1&1&0&0&0&1&0&1&0&0&1&1&1&0\\
0&1&1&0&1&0&1&1&1&1&0&0&0&1&0&0&1&1&0&1\\
\hline
1&1&0&1&1&0&0&0&1&1&0&1&0&0&0&0&0&0&1&1
\end{array}$$

Theorem 11.1.12 suggests that we can construct a bit generator with a very large period by finding a primitive polynomial over $\mathbb{F}_2$

$$f(x) = x^l - a_{l-1}x^{l-1} - a_{l-2}x^{l-2} - \cdots - a_0$$

of large degree l and setting

$$s_{n+l} = a_{l-1}s_{n+l-1} + a_{l-2}s_{n+l-2} + \cdots + a_0 s_n$$

for $n \geq 0$. Let $\mathbf{s}_0 = (s_0, s_1, \ldots, s_{l-1})$, $s_i \in \mathbb{F}_2$, be an initial state vector. Then the recurrence relation defines a bit stream $\{s_n\}$ with period $2^l - 1$.

Example 11.1.14 It is known that

$$f(x) = x^{20} + x^{19} + x^4 + x^3 + 1$$

is a primitive polynomial in $\mathbb{F}_2[x]$. The polynomial f defines a 20th-order linearly recursive sequence $\{s_n\}$ with recurrence relation

$$s_{n+20} = s_{n+19} + s_{n+4} + s_{n+3} + s_n$$

for $n \geq 0$. Since f is primitive, $\{s_n\}$ has period

$$2^{20} - 1 = 1,048,575.$$

We let

$$\mathbf{s}_0 = 01100010100010001110$$

be the randomly chosen initial state vector.

The first 500 terms of $\{s_n\}$ are computed using the simple GAP program:

```
s:=List([1..500]);

s[1]:=0; s[2]:=1; s[3]:=1; s[4]:=0;
s[5]:=0; s[6]:=0; s[7]:=1 s[8]:=0;
s[9]:=1; s[10]:=0; s[11]:=0; s[12]:=0;
s[13]:=1; s[14]:=0; s[15]:=0; s[16]:=0;
s[17]:=1; s[18]:=1; s[19]:=1; s[20]:=0; #initial state vector

for j in [1..500] do;
  s[j+20]:=(s[j+19]+s[j+4]+s[j+3]+s[j]) mod 2;
  Print(s[j]);
od;
```

The sequence is

```
011000101000100011100110101110000001001011110101000100110011
111101110010111000100010100011010110000101000010000001010101
101000111010110000110000011111110100110110000101111010101110
101001111110001000010100010010011101010111001110110000111010
011001110100011101011100111000001111101001101011000011110101
000000101111111101100010110010100011100100000010111110111110
001011001110110010010000011001111110000110011100010010100111
0010110011010100100011...
```

The bit generator is

$$G : \{0, 1\}^{20} \to \{0, 1\}^m,$$

$m \geq 20$.

□

Despite having large periods, bit streams as in Example 11.1.14 cannot be considered cryptographically secure.

Proposition 11.1.15 *Let $\{s_n\}$ be an lth-order linearly recursive sequence defined by the degree l primitive polynomial $f(x)$ over $\mathbb{F}_2$. Suppose Malice has obtained $2l$ consecutive terms of $\{s_n\}$. Then Malice can compute all of the remaining terms of $\{s_n\}$ in $O(l^3)$ steps.*

Proof We can assume without loss of generality that Malice has obtained the first $2l$ terms of $\{s_n\}$. Consequently, there is a system of equations

$$\begin{cases} a_0s_0 + a_1s_1 + a_2s_2 + \cdots + a_{l-1}s_{l-1} & = s_l \\ a_0s_1 + a_1s_2 + a_2s_3 + \cdots + a_{l-1}s_l & = s_{l+1} \\ a_0s_2 + a_1s_3 + a_2s_4 + \cdots + a_{l-1}s_{l+1} & = s_{l+2} \\ \qquad\vdots \qquad\qquad\qquad\qquad \vdots & \;\;\vdots \\ a_0s_{l-1} + a_1s_l + a_2s_{l+1} + \cdots + a_{l-1}s_{2l-2} & = s_{2l-1} \end{cases}$$

in the variables $a_0, a_1, \ldots, a_{l-1}$. This system has a unique solution, which can be obtained using Gaussian elimination in $O(l^3)$ bit operations.

□

Thus, the computation of a primitive polynomial of degree 20, and hence a sequence of period $2^{20} - 1$, would require 40 consecutive terms of the sequence and $\approx 20^3 = 8000$ bit operations, easily within the range of a modern digital computer. Malice's attack on the key stream would succeed.

11.2 The Shrinking Generator Sequence

In order to use linearly recursive sequences to generate key streams for stream ciphers with a reasonable level of security, one must combine several sequences together in a way that eliminates the linearity that leads to Malice's successful attack given in Proposition 11.1.15.

One way to do this is to use two linearly recursive sequences in the following way. Let $\{s_n\}$, $\{t_n\}$ be linearly recursive sequences in $\mathbb{F}_2$ of order k, l, respectively. In the sequence $\{s_n\}$, suppose that the 1s occur in the terms

$$s_{n_0}, s_{n_1}, s_{n_2}, s_{n_3}, \ldots$$

Define a new sequence $\{v_m\}$ by the rule:

$$v_m = t_{n_m},$$

for $m \geq 0$. The sequence $\{v_m\}$, $m \geq 0$, is the **shrinking generator sequence**. The sequence s is the **selector sequence**.

The shrinking generator sequence was introduced by D. Coppersmith, H. Krawczyk, and Y. Mansour in the paper [18].

Example 11.2.1 (Shrinking Generator Sequence) Let $\{s_n\}$ be the 4th-order linearly recursive sequence in $\mathbb{F}_2$ with recurrence relation $s_{n+4} = s_{n+1} + s_n$ and initial state $\mathbf{s}_0 = 0100$. The sequence $\{s_n\}$ has period 15 and begins

$$010011010111100\ldots$$

Let $\{t_n\}$ be the 5th-order linearly recursive sequence in $\mathbb{F}_2$ with recurrence relation $t_{n+5} = t_{n+2} + t_n$ and initial state $\mathbf{t}_0 = 10110$. The sequence has period 31 and begins

$$1011001111100011011101010000100\ldots$$

The shrinking generator sequence $\{v_m\}$, $m \geq 0$, is

$$000111000100000\ldots$$

□

The shrinking generator sequence is periodic, though its period is significantly larger than those of the linearly recursive sequences used to compute it.

To compute the period of the shrinking generator sequence, we begin with some lemmas.

Lemma 11.2.2 *Let $\{s_n\}$ be a kth-order linearly recursive sequence in $\mathbb{F}_2$ with primitive characteristic polynomial $f(x)$. Then there are 2^{k-1} occurrences of 1 in the first $2^k - 1$ terms of $\{s_n\}$.*

Proof Note that the period of $\{s_n\}$ is $2^k - 1$. The state vectors

$$\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_{2^k-2}$$

constitute all possible non-zero vectors of length k over $\mathbb{F}_2$ (see Section 11.5, Exercise 11). Now, there are exactly

$$\sum_{i=1}^{k} \binom{k}{i} i$$

total occurrences of 1 in the listing of the state vectors $\mathbf{s}_0, \ldots, \mathbf{s}_{2^k-2}$.

An occurrence of 1 in the first $2^k - 1$ terms of the sequence $\{s_n\}$ determines an occurrence of 1 in k consecutive state vectors. And so there are precisely

$$\left(\sum_{i=1}^{k} \binom{k}{i} i\right) / k = \sum_{i=0}^{k-1} \binom{k-1}{i} = 2^{k-1}$$

occurrences of 1 in the first $2^k - 1$ terms of $\{s_n\}$. □

Lemma 11.2.3 *Let $\{s_n\}$ be a kth-order linearly recursive sequence in $\mathbb{F}_2$ with primitive characteristic polynomial $f(x)$. For convenience of notation, we write $s(n) = s_n$, $n \geq 0$. Let $r_s = 2^k - 1$ denote the (maximal) period of s. Fix $b, c \geq 0$ with $\gcd(c, r_s) = 1$, and let $s(b + ac)$, $a \geq 0$, denote the subsequence of s. Then the period of $s(b + ac)$ is r_s.*

Proof Let A be the matrix of s. Then $s(b + ac)$ is determined by the matrix A^c. Now, the characteristic polynomial of A is $f(x)$, and since f is primitive, there is a root α of $f(x)$ for which $\langle\alpha\rangle = \mathbb{F}_2(\alpha)^\times = \mathbb{F}_{2^k}^\times$. Note $|\mathbb{F}_{2^k}^\times| = r_s$.

Let $g(x)$ be the characteristic polynomial of A^c. We have $\deg(g(x)) = k$. Since $\gcd(c, r_s) = 1$, α^c is a root of $g(x)$, which satisfies $\langle\alpha^c\rangle = \mathbb{F}_{2^k}^\times$, hence $g(x)$ is a primitive polynomial over $\mathbb{F}_2$ (we could have $f(x) = g(x)$, but this is not necessary).

It follows that $s(b + ac)$ is a k-order linearly recursive sequence with maximal period $2^k - 1 = r_s$.

□

Proposition 11.2.4 *Let $\{s_n\}$ be a kth-order linearly recursive sequence in $\mathbb{F}_2$ with characteristic polynomial $f(x)$ (the selector sequence), and let $\{t_n\}$ be an lth-order linearly recursive sequence in $\mathbb{F}_2$ with characteristic polynomial $g(x)$. Assume both $f(x)$ and $g(x)$ are primitive polynomials over $\mathbb{F}_2$ and that $\gcd(k, l) = 1$ with $k \le l$. Let $\{v_m\}$ be the shrinking generator sequence constructed from $\{s_n\}$ and $\{t_n\}$. Then $\{v_m\}$ has period $2^{k-1}(2^l - 1)$.*

Proof Note that the period of $\{s_n\}$ is $2^k - 1$ and the period of $\{t_n\}$ is $2^l - 1$. Let $r_s = 2^k - 1$, $r_t = 2^l - 1$, and $w = 2^{k-1}$. We first note that $\gcd(k, l)$ implies that $\gcd(r_s, r_t) = 1$ (see Section 11.5, Exercise 12). Note also that $k \le l$, and so, $k \le r_t$.

From Lemma 11.2.2, there are w occurrences of 1 in the first r_s terms of $\{s_n\}$.

For the purposes of this proof, we let $s(n) = s_n$, $t(n) = t_n$, and $v(m) = v_m$ for $m, n \ge 0$.

By the definition of v, for each $m \ge 0$,

$$v(m + aw) = t(n_m + ar_s),$$

for $a \ge 0$. With $a = r_t$,

$$v(m + r_t w) = t(n_m + r_t r_s) = t(n_m) = v(m),$$

for all $m \ge 0$ and so, v is periodic with period $r_v \le wr_t$.

By the Division Algorithm,

$$wr_t = r_v q + r,$$

for integers $q, r, 0 \le r < r_v$. Assume that $r > 0$. Then

$$v(m) = v(m + wr_t) = v(m + r_v q + r) = v(m + r), \forall m,$$

which contradicts that r_v is the period of v. Thus, $r_v \mid wr_t$.

We claim that $wr_t \mid r_v$, which will show that $r_v = wr_t = 2^{k-1}2^l - 1$, and the proof will be completed.

To this end, we have

$$v(m + aw) = v(m + r_v + aw), \forall a$$

thus

$$t(n_m + ar_s) = t(n_{m+r_v} + ar_s),$$

for all $m, a \geq 0$. Thus,

$$t((n_{m+r_v} - n_m) + n_m + ar_s) = t(n_m + ar_s),$$

for all $m, a \geq 0$.

Now applying Lemma 11.2.3 to the sequence t with $b = n_m$ and $c = r_s$, yields that the period of the subsequence $t(n_m + ar_s)_{a \geq 0}$ is r_t. Thus

$$r_t \mid (n_{m+r_v} - n_m), \forall m. \tag{11.3}$$

We claim that (11.3) implies that $w \mid r_v$.

From (11.3), we have that for each $m \geq 0$, there exists an integer d_m for which

$$n_{m+r_v} = n_m + d_m r_t. \tag{11.4}$$

Thus

$$n_{m+1+r_v} = n_{m+1} + d_{m+1} r_t. \tag{11.5}$$

Subtracting (11.4) from (11.5) yields

$$n_{m+1+r_v} - n_{m+r_v} = n_{m+1} - n_m + (d_{m+1} - d_m) r_t. \tag{11.6}$$

If

$$n_{m+1+r_v} - n_{m+r_v} > n_{m+1} - n_m,$$

then

$$(d_{m+1} - d_m) r_t > 0,$$

and so

$$n_{m+1+r_v} - n_{m+r_v} > r_t.$$

Consequently, the sequence s contains more than r_t consecutive 0s and since $k \leq r_t$, this implies that s is the sequence of 0s, a contradiction.

If

$$n_{m+1+r_v} - n_{m+r_v} < n_{m+1} - n_m,$$

then

$$(d_{m+1} - d_m)r_t < 0,$$

and so

$$n_{m+1} - n_m > r_t,$$

thus, the sequence s contains more than r_t consecutive 0s, again a contradiction. We conclude that

$$n_{m+r_v+1} - n_{m+r_v} = n_{m+1} - n_m.$$

This says that the subsequence $s(i)$, $i \geq n_m$, is identical to the subsequence $s(j)$, $j \geq n_{m+r_v}$. Thus $r_s \mid (n_{m+r_v} - n_m)$ and so, the number of terms in s from $s(n_m)$ to $s(n_{m+r_v} - 1)$ is a multiple of r_s. It follows that the number of 1's in s from $s(n_m)$ to $s(n_{m+r_v} - 1)$ is a multiple of w. But the number of 1s is exactly r_v, so

$$wc = r_v, \tag{11.7}$$

for some c. Now,

$$t(n_0) = v(0) = v(r_v) = v(cw) = t(n_0 + cr_s). \tag{11.8}$$

Next, let $t^{(i)}(n)$ denote the sequence

$$t^{(i)}(n) = t(n + i)$$

for $0 \leq i \leq r_t$, $n \geq 0$. The sequence $t^{(i)}$ is an lth-order linearly recursive sequence with exactly the same primitive characteristic polynomial $g(x)$ as t. Using the formula (11.8), we obtain

$$t(n_0 + i) = t^{(i)}(n_0) = v(0) = t^{(i)}(n_0 + cr_s) = t((n_0 + i) + cr_s).$$

It follows that r_t divides cr_s. Since $\gcd(r_s, r_t) = 1$, this implies that r_t divides c; we have $r_t d = c$. Now by (11.7),

$$r_v = wr_t d,$$

hence wr_t divides r_v. We conclude that the period of $\{v_m\}$ is $2^{k-1}(2^l - 1)$.

□

As suggested by Proposition 11.2.4, a stream cipher in which the key stream is given by a shrinking generator sequence is more secure than one in which the key stream is generated by a linearly recursive sequence. See Section 11.3 in which we compute the "linear complexity" of the shrinking generator sequence.

For a survey of other ways to combine linearly recursive sequences to obtain cryptographically secure bit streams, see [56, Section 12.3].

11.3 Linear Complexity

Let $\{s_n\}_{n\geq 0}$ be an arbitrary sequence over $\mathbb{F}_2$, i.e., $\{s_n\}$ is a sequence of bits. For $N \geq 1$, consider the first N terms of $\{s_n\}$:

$$\{s_n\}_{n=0}^{N-1} = s_0, s_1, s_2, \ldots, s_{N-1}.$$

We ask: can $\{s_n\}_{n=0}^{N-1}$ be generated as the first N terms of a linearly recursive sequence? The answer is "yes." Let $\{s_n\}$ be an Nth-order (homogeneous) linearly recursive sequence of bits with recurrence relation

$$s_{n+N} = a_{N-1}s_{n+N-1} + a_{N-2}s_{n+N-2} + \cdots + a_1 s_{n+1} + a_0 s_n, \quad n \geq 0,$$

and initial state vector $\mathbf{s}_0 = (s_0, s_1, s_2, \ldots, s_{N-1})$. Then certainly, the first N terms of $\{s_n\}$ are $s_0, s_1, \ldots, s_{N-1}$.

So the question is now: what is the smallest integer $L > 0$ so that the first N terms $s_0, s_1, \ldots, s_{N-1}$ can be generated as the first N terms of an Lth-order linearly recursive sequence? That is, what is the smallest integer $L > 0$ for which the terms can be generated by a recurrence relation

$$s_{n+L} = a_{L-1}s_{n+L-1} + a_{L-2}s_{n+L-2} + \cdots + a_1 s_{n+1} + a_0 s_n,$$

for $0 \leq n \leq N - L - 1$?

Definition 11.3.1 For $N \geq 1$, the N**th linear complexity** of the sequence $s = \{s_n\}$ is the length L of a shortest recurrence relation

$$s_{n+L} = a_{L-1}s_{n+L-1} + a_{L-2}s_{n+L-2} + \cdots + a_1 s_{n+1} + a_0 s_n,$$

$0 \leq n \leq N - L - 1$, that is satisfied by the first N terms of $\{s_n\}$. We denote the Nth linear complexity of s as $L_{s,N}$.

Example 11.3.2 Let $N \geq 1$ and let $s = \{s_n\}$ be so that $s_n = 0$ for $0 \leq n \leq N - 1$. Then (by convention) we have $L_{s,N} = 0$.

□

Example 11.3.3 Let $N \geq 2$ and let $s = \{s_n\}$ be so that $s_n = 0$ for $0 \leq n \leq N-2$ and $s_{N-1} = 1$. Then we have $L_{s,N} = N$.

□

Example 11.3.4 Let $\{s_n\}$ be the 4th order linearly recursive sequence over $\mathbb{F}_2$ with recurrence relation

$$s_{n+4} = s_{n+1} + s_n, \quad n \geq 0$$

and initial state vector $\mathbf{s}_0 = 0001$. We have

$$\{s_n\} = 0001001101011110001001101011111000\ldots$$

Thus $L_{s,1} = L_{s,2} = L_{s,3} = 0$ and $L_{s,4} = 4$. Moreover, $L_{s,N} \leq 4$ for $N \geq 5$ since $\{s_n\}$ is a 4th-order linearly recursive sequence.

In fact, $L_{s,N} = 4$ for $N \geq 5$. If $L_{s,N} < 4$ for some $N \geq 5$, then s would begin with N 0s.

□

We need an algorithm that will enable us to compute $L_{s,N}$ precisely.

Lemma 11.3.5 *Let $\{s_n\}$ be a sequence of bits. Then the first N terms of $\{s_n\}$ are generated by a recurrence relation of length L if the system of equations*

$$\begin{cases} a_0 s_0 + a_1 s_1 + a_2 s_2 + \cdots + a_{L-1} s_{L-1} & = s_L \\ a_0 s_1 + a_1 s_2 + a_2 s_3 + \cdots + a_{L-1} s_L & = s_{L+1} \\ a_0 s_2 + a_1 s_3 + a_2 s_4 + \cdots + a_{L-1} s_{L+1} & = s_{L+2} \\ \quad\vdots \qquad\qquad\qquad \vdots & \quad \vdots \\ a_0 s_{N-L-1} + a_1 s_{N-L} + a_2 s_{N-L+1} + \cdots + a_{L-1} s_{N-2} & = s_{N-1} \end{cases}$$

has a solution $a_0, a_1, a_2, \ldots, a_{L-1}$.

Proof This is clear.

□

Algorithm 11.3.6 (N_LIN_COMP)

```
Input:      a sequence of N bits, $s = s_0, s_1, \ldots, s_{N-1}$
Output:     $L_{s,N}$
Algorithm:
            lower $\leftarrow$ 0, upper $\leftarrow$ $N$.
            while (upper − lower) $\geq$ 2 do
               Step 1. Compute $L = \lfloor(\text{lower} + \text{upper})/2\rfloor$.
               Use Lemma 11.3.5 to determine whether $s$ is generated
               by a recurrence relation of length $L$.
               Step 2. If “yes” in Step 1, then set upper = $L$.
               If “no” if Step 1, then set lower = $L$.
```

```
end-while
output upper = $L_{s,N}$
```

□

Example 11.3.7 Let $s = 00010011$ be the first 8 terms of the sequence of Example 11.3.4. We employ Algorithm 11.3.6 with initial values lower $= 0$, upper $= 8$: after the first iteration, we have $L = 4$, lower $= 0$, upper $= 4$; after the second iteration, we have $L = 2$, lower $= 2$, upper $= 4$; after the third iteration, we have $L = 3$, lower $= 3$, upper $= 4$; thus, $L_{s,8} = 4$.

□

Example 11.3.8 Let $s = 00000011$. We employ Algorithm 11.3.6 with initial values lower $= 0$, upper $= 8$: after the first iteration, we have $L = 4$, lower $= 4$, upper $= 8$; after the second iteration, we have $L = 6$, lower $= 6$, upper $= 8$; after the third iteration, we have $L = 7$, lower $= 6$, upper $= 7$; thus, $L_{s,8} = 7$.

□

Definition 11.3.9 Let $s = \{s_n\}_{n\geq 0}$ be a sequence of bits. For $N \geq 1$, let $L_{s,N}$ be the Nth linear complexity of s. Then

$$L_s = \sup_{N\geq 1}(\{L_{s,N}\})$$

is the **linear complexity** of s. Here $\sup_{N\geq 1}(\{L_{s,N}\})$ denotes the least upper bound of the set $\{L_{s,N} : N \geq 1\}$, cf. [51, Definition 1.8].

For example, if s is the sequence of Example 11.3.4, then $L_s = 4$.

Theorem 11.3.10 *$L_s < \infty$ if and only if $\{s_n\}_{n\geq 0}$ is a linearly recursive sequence over $\mathbb{F}_2$.*

Proof Let $s = \{s_n\}$ be a sequence of bits. If $\{s_n\}$ is an lth-order linearly recursive sequence, then $L_{s,N} \leq l$ for all $N \geq 1$. Thus $L_s \leq l < \infty$.

Conversely, if $\{s_n\}$ is not linearly recursive, then there is no finite set of bits $a_0, a_1, \ldots, a_{l-1}$ with

$$s_{n+l} = a_{l-1}s_{n+l-1} + a_{l-2}s_{n+l-2} + \cdots + a_0 s_n$$

for all $n \geq 0$. Thus for any integer $l \geq 1$, and any set of bits $a_0, a_1 \ldots, a_{l-1}$, there exists an integer $N \geq 0$ so that

$$s_{N+l} \neq a_{l-1}s_{N+l-1} + a_{l-2}s_{N+l-2} + \cdots + a_0 s_N.$$

Hence $L_{s,N+l+1} > l$. Thus L_s is not finite.

□

Theorem 11.3.11 *$L_s < \infty$ if and only if $\{s_n\}_{n\geq 0}$ is eventually periodic.*

Proof Let $s = \{s_n\}$ be a sequence of bits. We show that $\{s_n\}$ is eventually periodic if and only if $\{s_n\}$ is linearly recursive. To this end, suppose $\{s_n\}$ is linearly recursive. Then $\{s_n\}$ is eventually periodic by Proposition 11.1.6.

For the converse, suppose that $\{s_n\}$ is eventually periodic. Then there exist integers $N \geq 0$, $t > 0$ for which $s_{n+t} = s_n$, for all $n \geq N$. It follows that $\{s_n\}$ is an $(N + t)$th order linearly recursive sequence with recurrence relation

$$s_{n+N+t} = s_{n+N},$$

for $n \geq 0$. □

We recall the shrinking generator sequence $\{v_m\}$ of Section 11.2, constructed from the linearly recursive sequences $\{s_n\}$ and $\{t_n\}$. The sequence $\{s_n\}$ is the selector sequence and is kth-order with degree k primitive polynomial $f(x)$; $\{t_n\}$ is lth-order with degree l primitive polynomial $g(x)$.

We assume $\gcd(k, l) = 1$ and $k \leq l$. The period of s is $r_s = 2^k - 1$, the period of t is $r_t = 2^l - 1$ and $w = 2^{k-1}$, which is the number of 1s in the first $2^k - 1$ terms of s.

We write $s_n = s(n)$, $t_n = t(n)$, and $v_m = v(m)$.

We obtain a lower bound on the linear complexity of $\{v(m)\}$.

Proposition 11.3.12 *Let $v = \{v(m)\}$ be a shrinking generator sequence defined by sequences $\{s(n)\}$, $\{t(n)\}$. Then*

$$L_v > 2^{k-2}l.$$

Proof By Proposition 11.2.4, $v = \{v(m)\}$ has period wr_t. Thus v is a linearly recursive sequence and, as such, has matrix A with minimal polynomial $m(x) \in \mathbb{F}_2[x]$.

As in the proof of Proposition 11.2.4, we have

$$v(aw) = t(n_0 + ar_s), \quad a \geq 0, \tag{11.9}$$

where n_0 is the position of the first 1 in s.

Let B be the matrix of t. Since $\gcd(r_s, r_t) = 1$, the characteristic polynomial of the matrix B^{r_s} is a primitive polynomial $h(x)$ over $\mathbb{F}_2$ of degree l; we have $h(B^{r_s}) = 0$.

From (11.9), we conclude that the matrix A^w also satisfies the polynomial $h(x)$. Hence, $m(x)$ divides $h(x^w)$. Since we are working in characteristic 2, we conclude that $m(x)$ divides $(h(x))^w$.

Since h is irreducible,

$$m(x) = (h(x))^c$$

for some $1 \leq c \leq w = 2^{k-1}$. We claim that $c > 2^{k-2}$. If $c \leq 2^{k-2}$, then $m(x)$ divides $(h(x))^{2^{k-2}}$. By Proposition 7.3.9, $h(x)$ divides $1 + x^{r_t}$. Thus, $m(x)$ divides

$$(1 + x^{r_t})^{2^{k-2}} = 1 + x^{r_t 2^{k-2}}.$$

But this says that the period of t is at most $2^{k-2} r_t$, which contradicts Proposition 11.2.4.

So the degree of $m(x)$ is greater than $2^{k-2}l$, hence $L_t > 2^{k-2}l$, as claimed.

□

In the special case that $\{v_m\}$ is constructed from the 1st order selector sequence $\{s_n\}$, $s_n = 1$, $n \geq 0$, and the lth-order sequence $\{t_n\}$, we obtain

$$v_m = t_m,$$

$m \geq 0$, as one can easily check. In this case, Proposition 11.3.12 yields

$$L_v = L_t > l/2.$$

As a consequence of Proposition 11.3.12, Malice needs more than $2^{k-1}l$ consecutive bits in the key stream to determine the recurrence relation for $\{t_m\}$. To see this, suppose that Malice needs $\leq 2^{k-1}l = 2 \cdot 2^{k-2}l$ bits to determine a linear recurrence for $\{t_m\}$. Then the linear recurrence is of length $\leq 2^{k-2}l$. Thus, $L_t \leq 2^{k-2}l$, which contradicts Proposition 11.3.12.

The number of bits that need to be captured to determine $\{t_m\}$ is an exponential function of the order k of $\{r_n\}$ and a linear function of the order l of $\{s_n\}$. This is a significant improvement compared to the case where the key stream is generated by a single linearly recursive sequence, in that case, the number of bits required is a linear function of k.

11.4 Pseudorandom Bit Generators

In Section 11.1 we constructed a bit generator with a very long period using the theory of lth order linearly recursive sequences. However, we showed that this bit generator is cryptographically insecure since the sequence can be recovered knowing a subsequence of length $2l$. We improved this situation in Section 11.2 by introducing the shrinking generator sequence.

In this section we develop the tools to construct other cryptographically secure bit generators.

Let l, m be integers with $m \gg l \geq 1$, and let

$$G : \{0, 1\}^l \to \{0, 1\}^m$$

be a bit generator with seed $x \in \{0, 1\}^l$. We want to decide whether a bit generator G produces strings that are pseudorandom or "as good as random." To do this, we introduce a test.

A bit generator is pseudorandom if it is not possible to distinguish its output from a truly random bit stream with reasonable computing power (or by using an algorithm that is practical and efficient).

Let $y_0 y_1 \dots y_i$ be a truly random sequence of bits. Given $y_0 y_1 \dots y_{i-1}$, if we just guessed the value of the next bit y_i, then we would be correct with probability $\frac{1}{2}$.

On the other hand, suppose that $y_0 y_1 \dots y_i$ is generated by G. Then G is pseudorandom if given $y_0 y_1 \dots y_{i-1}$, there is no practical algorithm that will give us a non-negligible advantage over merely guessing the value of the next bit y_i.

More formally, we have the following test.

Definition 11.4.1 (Next-Bit Test) Let $G : \{0, 1\}^l \to \{0, 1\}^m$ be a bit generator. Let x be a randomly chosen bit string in $\{0, 1\}^l$, and let

$$G(x) = y_0 y_1 y_2 \dots y_{m-2} y_{m-1} \in \{0, 1\}^m.$$

Let i be an integer, $1 \le i \le m - 1$. Let A be a probabilistic polynomial time algorithm with input $y_0 y_1 \dots y_{i-1}$ and output $A(y_0 y_1 \dots y_{i-1}) \in \{0, 1\}$. Then G **passes the next-bit test** if there exists a negligible function $r : \mathbb{R}_+ \to \mathbb{R}_+$ and an integer l_0 so that

$$\Pr(A(y_0 y_1 \dots y_{i-1}) = y_i) \le \frac{1}{2} + r(l)$$

for $l \ge l_0$.

Definition 11.4.2 A bit generator G is a **pseudorandom bit generator** if G passes the next-bit test.

Remark 11.4.3 The algorithm in the next-bit test is a probabilistic polynomial time algorithm in the sense of Section 4.4. The algorithm attempts to solve the decision problem:

$\mathcal{D}$ = given the bit stream $y_0 y_1 \dots y_i$, decide whether the next bit y_{i+1} is equal to 1 (YES) or equal to 0 (NO)

At issue here is whether $\mathcal{D} \in$ PP. If the bit stream is pseudorandom, then $\mathcal{D} \notin$ PP. □

Let $G : \{0, 1\}^l \to \{0, 1\}^m$ be a bit generator. We define $G_R : \{0, 1\}^l \to \{0, 1\}^m$ to be the bit generator that reverses the bits of G; that is, $G_R(x) = y_R = y_{m-1} y_{m-2} \dots y_1 y_0$, where $G(x) = y = y_0 y_1 \dots y_{m-1}$.

Proposition 11.4.4 *If G is a pseudorandom generator, then G_R is a pseudorandom generator.*

Proof Suppose $G_R : \{0,1\}^l \to \{0,1\}^m$ is not pseudorandom. Then there exists a positive polynomial $w(x)$, an integer i, $1 \leq i \leq m-1$, and a probabilistic polynomial time algorithm A so that

$$\Pr(A(y_{m-1}y_{m-2}\ldots y_i) = y_{i-1}) \geq \frac{1}{2} + \frac{1}{w(l)}$$

for infinite l. Define an algorithm A' as follows. Given $y_0y_1\ldots y_{i-2}$,

$$A'(y_0y_1\ldots y_{i-2}) = A(y_{m-1}y_{m-2}\ldots y_i).$$

Then A' is a probabilistic polynomial time algorithm with

$$\Pr(A'(y_0y_1\ldots y_{i-2}) = y_{i-1}) \geq \frac{1}{2} + \frac{1}{w(l)}$$

for infinite l. Thus G is not pseudorandom, a contradiction.

□

11.4.1 Hard-Core Predicates

Let $\{0,1\}^*$ denote the set of all sequences of 0's and 1's of finite length. A **predicate** is a function

$$B : \{0,1\}^* \to \{0,1\}.$$

For example, for $x \in \{0,1\}^*$, $B(x) = (|x| \bmod 2)$ is a predicate. Here $|x|$ is the length of x, example: $B(10011) = 1$. A predicate is a type of decision problem (1 = YES, 0 = NO).

Definition 11.4.5 Let $f : \{0,1\}^* \to \{0,1\}^*$ be a function. A predicate $B : \{0,1\}^* \to \{0,1\}$ is a **hard-core predicate** for the function f if:

(i) There is a probabilistic polynomial time algorithm that computes B.
(ii) For any probabilistic polynomial time algorithm A with input $f(x)$ and output $A(f(x)) \in \{0,1\}$, $x \in_R \{0,1\}^l$, there exists an integer l_0 so that for $l \geq l_0$,

$$\Pr(A(f(x)) = B(x)) \leq \frac{1}{2} + r(l),$$

where r is a negligible function.

Hard-core predicates are essentially **unbiased**, i.e., there is a negligible difference between $\Pr(B(x) = 0)$ and $\Pr(B(x) = 1)$.

Proposition 11.4.6 *Let* $B : \{0, 1\}^* \to \{0, 1\}$ *be a hard-core predicate for the function* $f : \{0, 1\}^* \to \{0, 1\}^*$. *Let* $x \in_R \{0, 1\}^l$. *Then there exist an integer* l_0 *and a negligible function* $r(x)$ *so that*

$$|\Pr(B(x) = 0) - \Pr(B(x) = 1)| \leq r(l)$$

whenever $l \geq l_0$.

Proof Suppose the condition of the proposition does not hold. Then there exists a positive polynomial $w(x)$ for which

$$|\Pr(B(x) = 0) - \Pr(B(x) = 1)| \geq \frac{1}{w(l)}$$

for infinite l. We can assume without loss of generality that $\Pr(B(x) = 0) \geq \Pr(B(x) = 1)$, thus

$$\Pr(B(x) = 0) - \Pr(B(x) = 1) \geq \frac{1}{w(l)}$$

for infinite l. Since $\Pr(B(x) = 1) = 1 - \Pr(B(x) = 0)$,

$$\Pr(B(x) = 0) \geq \frac{1}{2} + \frac{1}{2w(l)}.$$

Let A be the polynomial time algorithm with $A(f(x)) = 0$ for all $x \in \{0, 1\}^l$. Then

$$\Pr(A(f(x)) = B(x)) \geq \frac{1}{2} + \frac{1}{2w(l)},$$

for infinite l, and so B is not a hard-core predicate, a contradiction.

□

11.4.2 Hard-Core Predicates and the DLA

In this section we will apply the Discrete Logarithm Assumption (DLA), which was stated in Section 9.4.

Let p be a random l-bit prime. We define two predicates on $x \in \{0, 1\}^l$. The predicate LEAST : $\{0, 1\}^l \to \{0, 1\}$ is defined as

$$\mathrm{LEAST}(x) = \begin{cases} 0 \text{ if } x \text{ is even (as a decimal integer)} \\ 1 \text{ otherwise.} \end{cases}$$

The predicate MOST : $\{0, 1\}^l \to \{0, 1\}$ is defined as

$$\mathrm{MOST}(x) = \begin{cases} 0 \text{ if } x < (p-1)/2 \\ 1 \text{ otherwise.} \end{cases}$$

Lemma 11.4.7 *Let p be an odd prime, let g be a primitive root modulo p, let $x \in U(\mathbb{Z}_p)$, and let*

$$y = \mathrm{DEXP}_{p,g}(x) = g^x$$

denote the discrete exponential function. Then $\mathrm{LEAST}(x) = 0$ *if and only if y is a quadratic residue modulo p.*

Proof Suppose $\mathrm{LEAST}(x) = 0$. Then $x = 2m$, hence $y = g^x = (g^m)^2$, and thus y is a quadratic residue modulo p. Conversely, if $y = g^x$ is a quadratic residue modulo p, then $y = g^x = (g^m)^2$ for some $m \in U(\mathbb{Z}_p)$. Hence $x = 2m$, and so $\mathrm{LEAST}(x) = 0$. □

Are either of these predicates hard-core for some function, say $\mathrm{DEXP}_{p,g}$? There is certainly a polynomial time algorithm for computing LEAST, but LEAST is not a hard-core predicate for $f(x) = \mathrm{DEXP}_{p,g}(x)$.

Proposition 11.4.8 *There is a polynomial time algorithm for computing* $\mathrm{LEAST}(x)$ *given* $\mathrm{DEXP}_{p,g}(x)$.

Proof Let p be an odd prime, and let $y = \mathrm{DEXP}_{p,g}(x) = g^x$ for $x \in U(\mathbb{Z}_p)$. By Lemma 11.4.7, $\mathrm{LEAST}(x) = 0$ if and only if y is a quadratic residue modulo p, hence by Proposition 6.4.5, $\mathrm{LEAST}(x) = 0$ if and only if $y^{(p-1)/2} \equiv 1 \pmod{p}$.

This is the basis for the following algorithm that runs in polynomial time:

```
Algorithm

Input:     $y = \mathrm{DEXP}_{p,g}(x)$
Output:    $\mathrm{LEAST}(x)$
Algorithm:
           $r \leftarrow (y^{(p-1)/2} \bmod p)$
           if $r = 1$, then $\mathrm{LEAST}(x) = 0$
           else $\mathrm{LEAST}(x) = 1$
```

□

On the other hand, $\mathrm{MOST}(x)$ is a hard-core predicate of $\mathrm{DEXP}_{p,g}(x)$. To see this, we first prove a lemma.

Lemma 11.4.9 *Let p be a prime, and let b be a quadratic residue modulo p. Then there exists a probabilistic polynomial time algorithm for computing the two square roots of b modulo p.*

Proof In the case that $p \equiv 3 \pmod 4$ (p is Blum), this follows from Proposition 6.4.5. If $p \equiv 1 \pmod 4$, use the result of E. Berlekamp [5].

□

Proposition 11.4.10 *Under the Discrete Logarithm Assumption,* $\mathrm{MOST}(x)$ *is a hard-core predicate of* $\mathrm{DEXP}_{p,g}(x)$.

Proof Suppose $\mathrm{MOST}(x)$ is not a hard-core predicate of $\mathrm{DEXP}_{p,g}(x)$. Then we show that the DLA cannot hold.

If $\mathrm{MOST}(x)$ is not hard-core, then, since Definition 11.4.5 (i) clearly holds, we have that Definition 11.4.5 (ii) fails; that is, there exists a probabilistic polynomial time algorithm A with input $\mathrm{DEXP}_{p,g}(x)$ and output $A(\mathrm{DEXP}_{p,g}(x)) \in \{0, 1\}$, $x \in_R \{0, 1\}^l$, and a positive polynomial $w(x)$ for which

$$\Pr(A(\mathrm{DEXP}_{p,g}(x)) = \mathrm{MOST}(x)) \geq \frac{1}{2} + \frac{1}{w(l)}$$

for infinitely many l.

To simplify the proof, we assume that A is a polynomial time algorithm that always computes MOST

$$A(\mathrm{DEXP}_{p,g}(x)) = \mathrm{MOST}(x)$$

for all $x \in_R \{0, 1\}^l$ and all l. This A will be used to devise a probabilistic polynomial time algorithm A', which will compute $\mathrm{DLOG}(g^x) = x$, thus contradicting the DLA.

Here is the algorithm A'.

Algorithm A':

Input: $y = g^x = \mathrm{DEXP}_{p,g}(x)$
Output: $x = \mathrm{DLOG}_{p,g}(y)$

Round 1

Let $y_0 = y = (g^x \bmod p)$, use the algorithm of Proposition 11.4.8 to compute $\mathrm{LEAST}(x)$; let $b_0 = \mathrm{LEAST}(x)$.

If $b_0 = 0$ (x is even) let $y_1 = g^{x/2}$, end of Round 1.

Else, if $b_0 = 1$ (x is odd), compute $(g^x)/g = g^{x-1}$, with $x - 1$ even. Thus g^{x-1} is a quadratic residue modulo p and using the algorithm of Proposition 11.4.9, compute the two square roots, $r_1 = g^{(x-1)/2}$ and $r_2 = g^{(x-1)/2+(p-1)/2}$. Now using A we obtain

$$A(r_1) = \mathrm{MOST}((x-1)/2) = 0,$$

$$A(r_2) = \mathrm{MOST}((x-1)/2 + (p-1)/2) = 1.$$

Let $y_1 = r_1 = g^{(x-1)/2}$, end of Round 1.

So, after Round 1, either $y_1 = g^{x/2}$ or $y_1 = g^{(x-1)/2}$.

Round 2

If $y_1 = (g^{x/2} \bmod p)$ $(b_0 = 0)$, use the algorithm of Proposition 11.4.8 to compute $b_1 = \text{LEAST}(x/2)$. If $b_1 = 0$ ($x/2$ is even) let $y_2 = g^{x/4}$, end of Round 2.

Else, if $b_1 = 1$ ($x/2$ is odd), compute $(g^{x/2})/g = g^{x/2-1} = g^{(x-2)/2}$, with $(x-2)/2$ even. Thus $g^{(x-2)/2}$ is a quadratic residue modulo p, and using the algorithm of Proposition 11.4.9, compute the two square roots, $r_1 = g^{(x-2)/4}$ and $r_2 = g^{(x-2)/4+(p-1)/2}$. Now use A:

$$A(r_1) = \text{MOST}((x-2)/4) = 0,$$

$$A(r_2) = \text{MOST}((x-2)/4 + (p-1)/2) = 1.$$

Let $y_2 = r_1 = g^{(x-2)/4}$, end of Round 2.

If $y_1 = g^{(x-1)/2}$ $(b_0 = 1)$, use the algorithm of Proposition 11.4.8 to compute $b_1 = \text{LEAST}((x-1)/2)$. If $b_1 = 0$ ($(x-1)/2$ is even) let $y_2 = g^{(x-1)/4}$, end of Round 2.

Else, if $b_1 = 1$ ($(x-1)/2$ is odd), compute $(g^{(x-1)/2})/g = g^{(x-1)/2-1} = g^{(x-3)/2}$, with $(x-3)/2$ even. Thus $g^{(x-3)/2}$ is a quadratic residue modulo p and using the algorithm of Proposition 11.4.9, compute the two square roots, $r_1 = g^{(x-3)/4}$ and $r_2 = g^{(x-3)/4+(p-1)/2}$. Now use A:

$$A(r_1) = \text{MOST}((x-3)/4) = 0,$$

$$A(r_2) = \text{MOST}((x-3)/4 + (p-1)/2) = 1.$$

Let $y_2 = r_1 = g^{(x-3)/4}$, end of Round 2.

So, after Round 2, we have one of the following:

$$y_2 = g^{x/4}, \quad y_2 = g^{(x-2)/4}, \quad y_2 = g^{(x-1)/4}, \quad y_2 = g^{(x-3)/4}.$$

This terminates after m rounds with $y_m = 1$ and with $b_{m-1}b_{m-2}\dots b_2b_1b_0$ the binary expansion of $x = \text{DLOG}_{p,g}(y)$.

□

Here is a numerical example of algorithm A'.

Example 11.4.11 Let $p = 7$, a Mersenne prime, with primitive root $g = 3$. We compute $x = \text{DLOG}_{7,3}(6)$ using A'.

Round 1

$y_0 = 6$. By Proposition 11.4.8, $\text{LEAST}(\text{DLOG}_{7,3}(6)) = 1$ since $\left(\frac{6}{7}\right) = -1$, thus $b_0 = 1$.

Since $b_0 = 1$, we compute $6/3 = 2$, which is a quadratic residue modulo 7; the two square roots of 2 are $r_1 = 3$ and $r_2 = 4$.

Now, we use A to obtain

$$A(3) = \text{MOST}(\text{DLOG}_{7,3}(3)) = \text{MOST}(1) = 0,$$

$$A(4) = \text{MOST}(\text{DLOG}_{7,3}(4)) = \text{MOST}(4) = 1.$$

Let $y_1 = 3$, end of Round 1.

At the end of Round 1, $y_1 = 3$ and $b_0 = 1$.

Round 2
Since $b_0 = 1$, use the algorithm of Proposition 11.4.8 to compute $b_1 = \text{LEAST}(\text{DLOG}_{7,3}(3)) = \text{LEAST}(1) = 1$. We compute $3/3 = 1$, which is a quadratic residue modulo 7; the two square roots of 1 are $r_1 = 1$ and $r_2 = 6$.

Now, we use A to obtain

$$A(1) = \text{MOST}(\text{DLOG}_{7,3}(1)) = \text{MOST}(0) = 0,$$

$$A(6) = \text{MOST}(\text{DLOG}_{7,3}(6)) = \text{MOST}(3) = 1.$$

Let $y_2 = 1$, end of Round 2.

A' terminates with $b_0 = 1$, $b_1 = 1$. Thus, $\text{DLOG}_{7,3}(6) = 3$.

□

11.4.3 The Blum–Micali Bit Generator

Definition 11.4.12 Let p be a random l-bit prime, and let g be a primitive root modulo p. Let x be a randomly chosen element of $U(\mathbb{Z}_p)$. Let $x_0 = x$ and set

$$x_i = (g^{x_{i-1}} \bmod p),$$

for $i \geq 1$. Let $b_i = \text{MOST}(x_i)$, $i \geq 0$. Then the sequence $\{b_i\}_{i\geq 0}$ is the **Blum–Micali sequence** with seed x.

Example 11.4.13 Take $p = 31$, a 5-bit prime, with $g = 3$. Let $x = 11$ be the seed. Then

$$\{x_i\} = 11, 13, 24, 2, 9, 29, 21, 15, 30, 1, 3, 27, 23, 11, 13, \ldots,$$

and

$$\{b_i\} = 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, \ldots$$

□

Example 11.4.14 Take $p = 19$, a 5-bit prime, with $g = 2$, $(19)_2 = 10011$. Let $x = 5$, $(5)_2 = 00101$ be the seed. Then

$$\{x_i\} = 5, 13, 3, 8, 9, 18, 1, 2, 4, 16, 5, 13, 3, \ldots$$

and

$$\{b_i\} = 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, \ldots$$

□

Definition 11.4.15 Let $x \in_R U(\mathbb{Z}_p)$. The bit generator $G : \{0, 1\}^l \to \{0, 1\}^m$ defined as

$$G(x) = b_0 b_1 b_2 \ldots b_{m-1},$$

where $b_i = \text{MOST}((g^{x_{i-1}} \bmod p))$, $0 \le i \le m - 1$, is the **Blum–Micali bit generator.**

In Example 11.4.13, $l = 5$, $x = 11$, $(11)_2 = 01011$. We have $G : \{0, 1\}^5 \to \{0, 1\}^m$, with

$$G(01011) = 001001111001100\ldots$$

Proposition 11.4.16 *Under the DLA, the Blum–Micali bit generator is pseudorandom.*

Proof Let G be the Blum–Micali generator with seed $x_0 = x$. We show that G_R is pseudorandom, thus by Proposition 11.4.4, so is G.

By way of contradiction, suppose that G_R is not pseudorandom; G_R fails the next-bit test. Then there exist a positive polynomial w, an integer i, $1 \le i \le m - 1$, and a probabilistic polynomial time algorithm A so that

$$\Pr(A(b_{m-1} b_{m-2} \ldots b_i) = b_{i-1}) \ge \frac{1}{2} + \frac{1}{w(l)}$$

for infinite l. Now,

$$\begin{aligned} b_i &= \text{MOST}(\text{DEXP}^i_{p,g}(x_0)) \\ &= \text{MOST}(\text{DEXP}_{p,g}(\text{DEXP}^{i-1}_{p,g}(x_0))) \\ &= \text{MOST}(\text{DEXP}_{p,g}(z)), \end{aligned}$$

where $z = \mathrm{DEXP}_{p,g}^{i-1}(x_0)$, $\mathrm{MOST}(z) = b_{i-1}$.

We define a new algorithm A' as follows. Since $\mathrm{DEXP}_{p,g}$ is a permutation, we may consider z as a randomly chosen element of $U(\mathbb{Z}_p)$. On input $\mathrm{DEXP}_{p,g}(z)$, the output of A' is $A(b_{m-1}b_{m-2}\dots b_i)$. Thus,

$$\begin{aligned}\Pr(A'(\mathrm{DEXP}_{p,g}(z)) = \mathrm{MOST}(z)) &= \Pr(A(b_{m-1}b_{m-2}\dots b_i) = b_{i-1})\\ &\geq \frac{1}{2} + \frac{1}{w(l)},\end{aligned}$$

for infinite l. This says that $\mathrm{MOST}(x)$ is not a hard-core predicate for $\mathrm{DEXP}_{p,g}(x)$, which contradicts Proposition 11.4.10.

□

Proposition 11.4.17 *The Blum–Micali sequence is periodic.*

Proof The function $\mathrm{DEXP}_{p,g}$ has a finite codomain $U(\mathbb{Z}_p)$. Thus the sequence x_0

$$x_i = (g^{x_{i-1}} \bmod p)$$

for $i \geq 1$ is periodic. Therefore so is the Blum–Micali sequence $b_i = \mathrm{MOST}(x_i)$.

□

Example 11.4.18 Let $\{b_i\} = 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, \dots$ be the Blum–Micali sequence of Example 11.4.14; $\mathrm{DEXP}_{19,2}(x)$ is a permutation of $U(\mathbb{Z}_{19})$ and can be written in standard notation as $\mathrm{DEXP}_{19,2}$

$$= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 \\ 2 & 4 & 8 & 16 & 13 & 7 & 14 & 9 & 18 & 17 & 15 & 11 & 3 & 6 & 12 & 5 & 10 & 1 \end{pmatrix}.$$

The cyclic decomposition of $\mathrm{DEXP}_{19,2}$ is

$$(1, 2, 4, 16, 5, 13, 3, 8, 9, 18)(6, 7, 14)(10, 17)(11, 15, 12).$$

Since the seed 5 is in the cycle (1, 2, 4, 16, 5, 13, 3, 8, 9, 18) of length 10, the Blum–Micali sequence has period 10.

□

So here we have a pseudorandom sequence that is periodic! This is possible since the period of a Blum–Micali sequence is "non-polynomial."

Proposition 11.4.19 *Let p be a random l-bit prime, let g be a primitive root modulo p, and let $x \in_R U(\mathbb{Z}_p)$. Let w be any positive polynomial. Under the DLA, there exists an integer l_0 so that the Blum–Micali sequence $\{b_i\}_{i\geq 0}$ has period greater than $w(l)$ for $l \geq l_0$.*

Proof Suppose there exists a polynomial w so that $\{b_i\}$ has period $\leq w(l)$ for infinite l. Then there exist a polynomial time algorithm A and an integer $0 \leq i \leq w(l) - 2$ for which

$$\Pr(A(b_0 b_1 \ldots b_i) = b_{i+1}) = 1.$$

In fact, A runs in time $O(w(l))$. Thus the Blum–Micali generator fails the next-bit test, and so, the Blum–Micali bit generator is not pseudorandom. But then Proposition 11.4.16 implies that the DLA cannot hold. □

11.4.4 The Quadratic Residue Assumption

Let p, q be distinct primes, and let $n = pq$. Define a function

$$\mathrm{DSQR}_n : U(\mathbb{Z}_n) \to U(\mathbb{Z}_n)$$

by the rule

$$\mathrm{DSQR}_n(x) = (x^2 \bmod n).$$

As in Section 6.4.1, we let QR_n denote the set of quadratic residues modulo n, i.e., those elements $x \in U(\mathbb{Z}_n)$ for which there exists $y \in U(\mathbb{Z}_n)$ with $x \equiv y^2 \pmod{n}$.

There are $\phi(n) = (p-1)(q-1)$ elements in $U(\mathbb{Z}_n)$. Let

$$J_n^{(1)} = \left\{ x \in U(\mathbb{Z}_n) : \left(\frac{x}{n}\right) = 1 \right\},$$

$$J_n^{(-1)} = \left\{ x \in U(\mathbb{Z}_n) : \left(\frac{x}{n}\right) = -1 \right\}.$$

Then $|J_n^{(1)}| = |J_n^{(-1)}| = (p-1)(q-1)/2$. We have $QR_n \subseteq J_n^{(1)}$; exactly half of the elements of $J_n^{(1)}$ are quadratic residues modulo n : $|QR_n| = (p-1)(q-1)/4$.

Let $f : J_n^{(1)} \to \{0, 1\}$ be the function defined as

$$f(x) = \begin{cases} 0 \text{ if } x \text{ is a quadratic residue modulo } n \\ 1 \text{ otherwise.} \end{cases}$$

The function f is unbiased in the sense that

$$|\Pr(f(x) = 0) - \Pr(f(x) = 1)| = \left| \frac{(p-1)(q-1)/4}{(p-1)(q-1)/2} - \frac{(p-1)(q-1)/4}{(p-1)(q-1)/2} \right| = 0.$$

Given $x \in J_n^{(1)}$, it seems difficult to predict whether x is in QR_n or not. If we use a coin flip to guess whether x is a quadratic residue (i.e., we guess $x \in QR_n$ if "heads" and $x \notin QR_n$ if "tails"), then the probability of guessing correctly is $\frac{1}{2}$. If we always guess that x is a quadratic residue, then the probability that we will be correct is $\frac{1}{2}$. So the issue is the following: is there some efficient, practical algorithm for guessing that will yield the correct result *significantly more* than half of the time (i.e., with probability $\geq \frac{1}{2} + \epsilon, \epsilon > 0$)?

We assume no such algorithm exists.

The Quadratic Residue Assumption (QRA) *Let $w(x) \in \mathbb{Z}[x]$ be a positive polynomial, let p, q be randomly chosen odd l-bit primes, and let $n = pq$. Let $x \in_R J_n^{(1)}$. Let A be a probabilistic polynomial time algorithm with input x and output $A(x) \in \{0, 1\}$. Then there exists an integer l_0 for which*

$$\Pr(A(x) = f(x)) < \frac{1}{2} + \frac{1}{w(l)}$$

whenever $l \geq l_0$.

The QRA says that $f(x)$ is essentially unbiased (but we already knew that).

Proposition 11.4.20 *For $x \in_R J_n^{(1)}$, there exist a negligible function r and an integer l_0 so that*

$$|\Pr(f(x) = 0) - \Pr(f(x) = 1)| \leq r(l)$$

whenever $l \geq l_0$.

Proof Suppose no such r exists. Then as a function of l, $|\Pr(f(x) = 0) - \Pr(f(x) = 1)|$ is not a negligible function. Hence, there exists a positive polynomial $w(x)$ with

$$|\Pr(f(x) = 0) - \Pr(f(x) = 1)| \geq \frac{1}{w(l)}$$

for infinite l. Without loss of generality, we can assume $\Pr(f(x) = 0) \geq \Pr(f(x) = 1)$, hence $\Pr(f(x) = 0) - \Pr(f(x) = 1) \geq \frac{1}{w(l)}$. Since $\Pr(f(x) = 1) = 1 - \Pr(f(x) = 0)$,

$$\Pr(f(x) = 0) \geq \frac{1}{2} + \frac{1}{2w(l)},$$

for infinite l. Note that $2q(x)$ is a positive polynomial. Now let A be the polynomial time algorithm that satisfies $A(x) = 0$ for all $x \in J_n^{(1)}$. Then

$$\Pr(A(x) = f(x)) \geq \frac{1}{2} + \frac{1}{2w(l)},$$

for infinite l, which contradicts the QRA.

□

The QRA is equivalent to the Factoring Assumption (FA) of Section 9.3.

Proposition 11.4.21 *The QRA holds if and only if the FA holds.*

Proof See [8, Note added in proof] and [1]. □

11.4.5 The Blum–Blum–Shub Bit Generator

Clearly, DSQR_n restricts to an function

$$\text{DSQR}_n : QR_n \to QR_n.$$

Lemma 11.4.22 *Suppose p, q are distinct Blum primes, i.e., $p, q \equiv 3 \pmod 4$. Then the function $\text{DSQR}_n : QR_n \to QR_n$ is a 1–1 correspondence.*

Proof Suppose a is a quadratic residue modulo n. By Proposition 6.4.9, a has exactly four square roots modulo n: x, $-x$, y, $-y$. By Proposition 6.4.10, exactly one of them, say x, is in QR_n. Thus DSQR_n is onto. It follows that DSQR_n is 1–1.

□

Proposition 11.4.23 *Under the Quadratic Residue Assumption, $\text{LEAST}(x)$ is a hard-core predicate of $\text{DSQR}_n(x)$.*

Proof Suppose $\text{LEAST}(x)$ is not a hard-core predicate of $\text{DSQR}_n(x)$. Then we show that the QRA cannot hold.

If $\text{LEAST}(x)$ is not hard-core, then, since Definition 11.4.5(i) clearly holds, we have that Definition 11.4.5(ii) fails, i.e., there exists a probabilistic polynomial time algorithm A with input $\text{DSQR}_n(x)$, $x \in QR_n$, and output $A(\text{DSQR}_n(x)) \in \{0, 1\}$ and a positive polynomial $w(x)$ for which

$$\Pr(A(\text{DSQR}_n(x)) = \text{LEAST}(x)) \geq \frac{1}{2} + \frac{1}{w(l)}$$

for infinitely many l. This A will be used to devise a probabilistic polynomial time algorithm A' that exhibits an "ϵ-advantage" in guessing whether an element of $J_n^{(1)}$ is a quadratic residue or not. This will contradict the QRA.

Here is the algorithm A'.

Let $x \in J_n^{(1)}$ and put $z = (x^2 \bmod n)$. Then $z \in QR_n$ and hence, by Proposition 6.4.9, z has exactly four square roots modulo n: x, $-x$, y, $-y$. We have $x, -x \in J_n^{(1)}$ with $\text{LEAST}(x) \neq \text{LEAST}(-x)$.

Next, we use A to (attempt to) compute $\text{LEAST}(r)$, where r is the unique square root (in QR_n) of z (Proposition 6.4.10). If $\text{LEAST}(r) = \text{LEAST}(x)$, then $x \in QR_n$;

set $A'(x) = 0$. If $\text{LEAST}(r) = \text{LEAST}(-x)$, then $x \notin QR_n$; set $A'(x) = 1$. We have

$$\Pr(A(\text{DSQR}_n(r)) = \text{LEAST}(r)) = \Pr(A'(x) = f(x)),$$

and so,

$$\Pr(A'(x) = f(x)) \geq \frac{1}{2} + \frac{1}{w(l)}$$

for infinitely many l, contradicting the QRA.

□

Definition 11.4.24 Let p, q be random l-bit Blum primes, let $n = pq$, let $x \in_R U(\mathbb{Z}_n)$, and let $x_0 = x^2 \bmod n$. For $i \geq 1$, let $x_i = x_{i-1}^2 \bmod n$, and let $b_i = \text{LEAST}(x_i)$. Then the sequence $\{b_i\}_{i \geq 0}$ is the **Blum–Blum–Shub (BBS)** sequence with seed x.

Example 11.4.25 Suppose $p = 19$, $q = 23$, which are 5-bit Blum primes. Then $n = pq = 437$. Choose $x = 2 \in U(\mathbb{Z}_{437})$. Then

$\{x_i\} = 4, 16, 256, 423, 196, 397, 289, 54, 294, 347, 234, 131, 118, 377, 104, 328, 82, 169, 156, 301, 142, 62, 348, 55, 403, 282, 427, 100, 386, 416, 4, 16, 256, 423, 196, 397, 289, 54, 294, 347, 234, 131, 118, 377, 104, 328, 82, 169, 156, 301, 142, 62, 348, 55, 403, 282, 427, 100, 386, 416, 4, 16, 256, 423, 196, 397, \ldots$

And so, the BBS sequence is

$\{b_i\} = 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, \ldots$

□

Here is a very simple GAP program to compute the first 100 terms of the BBS of Example 11.4.25.

```
p:=2;
for i in [1..100] do;
  p:=PowerMod(Integers,p,2,437);
  Print(p mod 2,",");
od;
```

Example 11.4.26 Suppose $p = 71$, $q = 127$, which are 7-bit Blum primes. Then $n = pq = 9017$. Choose $x = 2019 \in U(\mathbb{Z}_{9017})$. Then

$\{x_i\} = 677, 7479, 2990, 4253, 8924, 8649, 169, 1510, 7816, 8698, 2574, 6998, 677, 7479, 2990, 4253, 8924, 8649, 169, 1510, 7816, 8698, 2574, 6998, 677, 7479, 2990, 4253, 8924, 8649, 169, 1510, 7816, 8698, 2574, 6998, 677, 7479, 2990, \ldots$

And so the BBS sequence is

$$\{b_i\} = 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1,$$
$$1, 0, 0, 0, 0, 0, \ldots$$

□

Definition 11.4.27 Let p, q be random l-bit Blum primes, let $n = pq$, and let $x \in_R U(\mathbb{Z}_n)$. Let l' be the length of x. The bit generator

$$G : \{0, 1\}^{l'} \to \{0, 1\}^m,$$

$m > l'$, defined as

$$G(x) = b_0 b_1 b_2 \ldots b_{m-1},$$

where $b_i = \text{LEAST}(x_i)$ is the **Blum–Blum–Shub bit generator** with seed x.

In Example 11.4.26, $x = 2019$, $(2019)_2 = 11111100011$. We have $G : \{0, 1\}^{11} \to \{0, 1\}^{20}$, with $G(11111100011) = 11010110000011010110$.

Proposition 11.4.28 *Under the QRA, the Blum–Blum–Shub bit generator is pseudorandom.*

Proof Let G be the Blum–Blum–Shub generator with seed x. We show that G_R is pseudorandom, thus by Proposition 11.4.4, so is G.

By way of contradiction, suppose that G_R is not pseudorandom; G_R fails the next-bit test. Then there exist a positive polynomial w, an integer i, $0 \le i \le m - 2$, and a probabilistic polynomial time algorithm A so that

$$\Pr(A(b_{m-1} b_{m-2} \ldots b_{i+1}) = b_i) \ge \frac{1}{2} + \frac{1}{w(l)}$$

for infinite l. Now,

$$\begin{aligned} b_{i+1} &= \text{LEAST}(\text{DSQR}_n^{i+1}(x_0)) \\ &= \text{LEAST}(\text{DSQR}_n(\text{DSQR}_n^i(x_0))) \\ &= \text{LEAST}(\text{DSQR}_n(z)), \end{aligned}$$

where $z = \text{DSQR}_n^i(x_0)$, $\text{LEAST}(z) = b_i$.

We define a new algorithm A' as follows. Since DSQR_n is a permutation of QR_n, we may consider z as a randomly chosen element of QR_n. On input $\text{DSQR}_n(z)$, the output of A' is $A(b_{m-1} b_{m-2} \ldots b_{i+1})$. Thus,

$$\begin{aligned} \Pr(A'(\text{DSQR}_n(z)) = \text{LEAST}(z)) &= \Pr(A(b_{m-1} b_{m-2} \ldots b_{i+1}) = b_i) \\ &\ge \frac{1}{2} + \frac{1}{w(l)}, \end{aligned}$$

for infinite l. This says that LEAST is not a hard-core predicate for DSQR_n, a contradiction.

□

Since QR_n is finite, all BBS sequences are periodic. What is the period of the BBS sequence? By inspection, we find that the period of the BBS sequence in Example 11.4.25 is 12, and the period of the BBS sequence in Example 11.4.26 is 30.

Proposition 11.4.29 *Let p, q be random l-bit Blum primes, let $n = pq$, and let $x \in_R U(\mathbb{Z}_n)$. Let d be the order of x_0 in $U(\mathbb{Z}_n)$, and write $d = 2^e m$, where m is odd. Then*

(i) The period of the sequence $\{x_i\}_{i\geq 0}$ is the order r of 2 in $U(\mathbb{Z}_m)$.
(ii) The period of the BBS sequence $\{b_i\}_{i\geq 0}$ is less than or equal to r.

Proof

For (i): The sequence $\{x_i\}$ appears as $x_0, x_0^2, x_0^{2^2}, x_0^{2^3}, \ldots$ modulo n. So the period of $\{x_i\}$ is the smallest $r > 0$ for which $2^{s+r} \equiv 2^s \pmod{d}$ for some $s \geq 1$. Write $s = e + l$ for some integer l. Then $2^{e+l}2^r \equiv 2^{e+l} \pmod{d}$, or $2^{e+l}2^r = 2^{e+l} + (2^e m)t$, $t \in \mathbb{Z}$, hence $2^l 2^r \equiv 2^l \pmod{m}$. Since m is odd, $2^l \in U(\mathbb{Z}_m)$, and so, $2^r \equiv 1 \pmod{m}$. Thus $r \geq r'$, where r' is the order of 2 in m. Suppose $r > r'$. Then working the argument above in reverse, we obtain $2^{s+r'} \equiv 2^s \pmod{d}$ for some $s \geq 1$, which contradicts the minimality of r. Thus $r = r'$.

For (ii): If the period of $\{b_i\}$ is r as in (i), then the period of $\{b_i\}$ is $\leq r$.

□

Example 11.4.30 In Example 11.4.25, the order of 4 in $U(\mathbb{Z}_{19})$ is 18 and the order of 4 in $U(\mathbb{Z}_{23})$ is 22. By Proposition 6.3.3, the order of 4 in $U(\mathbb{Z}_{437})$ is $d = \mathrm{lcm}(18, 22) = (18 \cdot 22)/\gcd(18, 22) = 198$. Now, $198 = 2 \cdot 99$. The order of 2 in $U(\mathbb{Z}_9)$ is $\phi(9) = 6$. The order of 2 in $U(\mathbb{Z}_{11})$ is 10, and so, the order of 2 in $U(\mathbb{Z}_{99})$ is $\mathrm{lcm}(6, 10) = 30$, which is the period of both $\{x_i\}$ and $\{b_i\}$.

□

How can we guarantee that the BBS sequence has a long period?

A prime p is a **safe prime** if $p = 2p' + 1$ for some prime p'. A **2-safe prime** is a safe prime $p = 2p' + 1$ in which the prime p' is safe prime, i.e., $p' = 2p'' + 1$ for some prime p''. For example, $p = 11 = 2 \cdot 5 + 1$ is a safe prime and $p = 23 = 2 \cdot 11 + 1$ is a 2-safe prime. Every 2-safe prime (and safe prime) is a Blum prime.

Proposition 11.4.31 *Let p, q be random l-bit safe-2 primes, $p = 2p' + 1$, $p' = 2p'' + 1$, $q = 2q' + 1$, $q' = 2q'' + 1$, for primes p', p'', q', q''. Let $n = pq$, let x_0 be a seed in $\mathbb{Z}_n$, with $\gcd(x_0, p) = \gcd(x_0, q) = 1$, $x_0 \not\equiv \pm 1 \bmod p$, $x_0 \not\equiv \pm 1 \bmod q$. Let $\{x_n\}_{n\geq 0}$ be the BBS sequence given as $x_i = \mathrm{DSQR}_n(x_{i-1})$ for $i \geq 1$. Then $\{x_n\}$ has period at least $p''q''$.*

Proof Since $\gcd(x_0, p) = 1$, then the order of x_0 in $U(\mathbb{Z}_p)$ divides $|U(\mathbb{Z}_p)| = p - 1 = 2p'$. Since $x_0 \not\equiv \pm 1 \bmod p$, then the order of x_0 in $U(\mathbb{Z}_p)$ is either p' or

$2p'$. Likewise, the order of x_0 in $U(\mathbb{Z}_q)$ is either q' or $2q'$. Thus by Proposition 6.3.3, the order of x_0 in $U(\mathbb{Z}_n)$ is either $p'q'$ or $2p'q'$.

By Proposition 11.4.29, the period of $\{x_0\}$ is the order of 2 in $U(\mathbb{Z}_{p'q'})$. The order of 2 in $U(\mathbb{Z}_{q'})$ divides $q' - 1 = 2q''$, thus is either q'' or $2q''$. Likewise, the order of 2 in $U(\mathbb{Z}_{q'})$ is either q'' or $2q''$. Thus the order of 2 in $U(\mathbb{Z}_{p'q'})$ is at least $p''q''$. □

Remark 11.4.32 The function $\mathrm{DSQR}_n : QR_n \rightarrow QR_n$ that is iterated to create a BBS sequence is a permutation, a bijection. Also, $|QR_n| = \frac{(p-1)(q-1)}{4}$ as we have seen in Proposition 6.4.8(ii). Now,

$$O(|QR_n|) = O\left(\frac{(p-1)(q-1)}{4}\right) = O(pq) = O(n).$$

In view of Proposition 11.4.31, there is a BBS sequence whose period is

$$O(p''q'') = O(p'q') = O(pq) = O(n).$$

This BBS sequence has period on the order of the size of QR_n. This is expected in view of Proposition 9.3.10.

□

11.5 Exercises

1. Let $\{s_n\}$ be the sequence in $\mathbb{Q}$ given as

$$1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, \ldots.$$

Show that $\{s_n\}$ is a linearly recursive sequence.

2. Let $\{s_n\}$ be the sequence in $\mathbb{Q}$ given as

$$1, 2, 3, 4, 5, 6, 7, 8, \ldots.$$

Show that $\{s_n\}$ is a 3rd-order linearly recursive sequence.

3. Let $\{s_n\}$ be the sequence in $\mathbb{Q}$ given as:

$$3, 2, 5, 1, 7, 0, 0, 0, 0, 0 \ldots.$$

Show that $\{s_n\}$ is a linearly recursive sequence.

4. Let $\{s_n\}$ be the sequence in $\mathbb{Q}$ given as:

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

Show that $\{s_n\}$ is not a linearly recursive sequence in $\mathbb{Q}$.

5. Let $\{s_n\}$ be the 2nd-order linearly recursive sequence in $\mathbb{R}$ with recurrence relation

$$s_{n+2} = 2s_{n+1} + s_n$$

and initial state vector $\mathbf{s}_0 = (-1, 2)$.

(a) Compute the matrix and the characteristic polynomial for $\{s_n\}$.
(b) Find a formula for the nth term of the sequence.

6. Let K be a field, and let A be a matrix in $\mathrm{Mat}_l(K)$. Prove that the set of all polynomials that annihilate A is a non-zero ideal of $K[x]$.
7. Let $\mathbb{F}_5$ be the Galois field with 5 elements. Write out the first five terms of the linearly recursive sequence $\{s_n\}$ with recurrence relation

$$s_{n+2} = 4s_{n+1} + s_n,$$

and initial state vector $\mathbf{s}_0 = (2, 3)$. What is the period of $\{s_n\}$?
8. Let $\mathbb{F}_9 = \mathbb{F}_3(\alpha)$ be the Galois field with 9 elements given in Example 7.3.14. Write out the first five terms of the linearly recursive sequence $\{s_n\}$ with recurrence relation

$$s_{n+5} = s_{n+3} + s_{n+1},$$

and initial state vector $\mathbf{s}_0 = (1, \alpha, 2, 1, \alpha)$. What is the period of $\{s_n\}$?
9. Suppose that 01001101 is the first byte of a key stream generated by a 4th order linearly recursive sequence $\{s_n\}$ over $\mathbb{F}_2$.

(a) Find the recurrence relation and the characteristic polynomial of $\{s_n\}$. What is the period of $\{s_n\}$?
(b) What can be said if an attacker obtains only the first six bits 010011 of the sequence?

10. Let $\{s_n\}$ be a linearly recursive sequence over $\mathbb{F}_2$ with primitive characteristic polynomial of degree l. Prove that $L_s = l$.
11. Let $\{s_n\}$ be a linearly recursive sequence over $\mathbb{F}_2$ with primitive characteristic polynomial of degree l. Prove that the state vectors

$$\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_{2^l-2}$$

constitute all possible non-zero vectors of length l over $\mathbb{F}_2$.
12. Prove that $\gcd(k, l) = 1$ if and only if $\gcd(2^k - 1, 2^l - 1) = 1$.
13. Write a `GAP` program to compute the first 1000 terms of the shrinking generator sequence where the selector sequence s has characteristic polynomial $f(x) = x^3 + x + 1$ and initial state $\mathbf{s}_0 = 110$, and the sequence t has characteristic polynomial $g(x) = x^2 + x + 1$ and initial state $\mathbf{t}_0 = 10$.

14. The **Thue–Morse sequence** over $\mathbb{F}_2$ is the sequence $s = \{s_n\}$, $n \geq 0$, defined as: $s_n = 0$ if the number of 1s in the canonical base-2 representation of n is even, and $s_n = 1$ otherwise. Thus,

$$\{s_n\} = 0110100110010110100101100, \ldots$$

Let $t = \{t_n\}$ be a 4th-order linearly recursive sequence with recurrence relation $s_{n+4} = s_{n+1} + s_n$ and initial state $\mathbf{s}_0 = 1110$.

(a) Using s as the selector sequence, find the first 10 terms of the shrinking generator sequence $v = \{v_m\}$ constructed from s and t.

(b) It is known that s is not periodic. Is v periodic?

15. Let $p(x)$ be a positive polynomial in $\mathbb{Z}[x]$ of degree $d \geq 2$. Let $s = \{s_n\}$ be the Thue–Morse sequence (as defined in Exercise 14). The subsequence $t_p = \{t_n\}_{n\geq 0}$ defined as

$$t_n = s_{p(n)},$$

$n \geq 0$, is the **Thue–Morse sequence along** $p(x)$.

Let $L_{t_p,N}$ denote the Nth linear complexity of t_p. Then P. Popoli [45, Theorem 3] has shown that there exist a constant $c > 0$ (dependent of $p(x)$) and an integer N_0 so that

$$L_{t_p,N} \geq cN^{1/d},$$

whenever $N \geq N_0$. Thus the linear complexity of t_p is ∞.
The evidence is that t_p has strong cryptographic properties.

(a) Use GAP to compute the first 1000 terms of t_p where $p(x) = x^2$.

(b) Use GAP to compute the first 1000 terms of t_p where $p(x) = x^4 + 3x^2 + x + 5$.

16. Let $G : \{0, 1\}^l \to \{0, 1\}^m$ be a bit generator defined by an lth order linearly recursive sequence $\{s_n\}$. Is G pseudorandom?

17. Let $p = 977$, a 10-bit prime, $(977)_2 = 1111010001$. Let $g = 3$, and let $x = (2^9)_2 = 10^9$ be the seed.

(a) Compute the sequence $\{x_i\}$ and the Blum–Micali bit sequence $\{b_i\}$.

(b) Consider the associated stream cipher with $\{b_i\}$ as its key stream. Encrypt $M = 10101010101010101010101$.

18. Prove that every safe prime is a Blum prime.

19. Let $p = 19$, $q = 23$.

(a) Show that 19, 23 are 5-bit Blum primes.

(b) Compute the Blum–Blum–Shub sequence $\{b_i\}$ with seed $x = 200 \in U(\mathbb{Z}_{437})$.

(c) Compute the period of $\{b_i\}$.

Chapter 12
Key Distribution

AES and other symmetric key cryptosystems are in wide use because they transmit data much faster than public key cryptosystems (e.g., RSA), but they need a shared secret key. Moreover, the Blum–Micali and Blum–Blum–Shub bit generators require both Alice and Bob to share an initial key consisting of a finite string of random bits.

In this chapter we introduce the Diffie–Hellman key exchange protocol, which is a method to distribute keys (or other information) securely among the principals in a network.

12.1 The Diffie–Hellman Key Exchange Protocol

Protocol 12.1.1 (Diffie–Hellman Key Exchange Protocol)

Premise: Alice and Bob share a large prime p and a primitive root g modulo p.
Goal: Alice and Bob share a secret random element of $\mathbb{Z}_p^\times$.

1. Alice randomly chooses an integer x, $0 \le x \le p-2$. She computes $h = (g^x \bmod p)$ and sends h to Bob.
2. Bob randomly chooses an integer y, $0 \le y \le p-2$. He computes $k = (g^y \bmod p)$ and sends k to Alice.
3. Bob computes

$$(h^y \bmod p) = ((g^x)^y \bmod p) = (g^{xy} \bmod p).$$

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_12

4. Alice computes

$$(k^x \bmod p) = ((g^y)^x \bmod p) = (g^{yx} \bmod p) = (g^{xy} \bmod p).$$

5. Alice and Bob share the secret random integer $(g^{xy} \bmod p)$.

□

Example 12.1.2 Suppose Alice and Bob share prime $p = 7057$ and primitive root 5 modulo 7057.

Alice randomly chooses integer $x = 365$, computes

$$h = 3448 = (5^{365} \bmod 7057),$$

and sends 3448 to Bob.

Bob chooses $y = 61$, computes

$$k = 1822 = (5^{61} \bmod 7057),$$

and sends 1822 to Alice.

Alice computes

$$(1822^{365} \bmod 7057) = 715.$$

Bob computes

$$(3448^{61} \bmod 7057) = 715.$$

In binary, $(715)_2 = 1011001011$, and so Alice and Bob share the secret random string of bits 1011001011.

□

12.2 The Discrete Logarithm Problem

The standard attack on the Diffie–Hellman key exchange protocol (DHKEP) is to find x given $(g^x \bmod p)$ and to find y given $(g^y \bmod p)$, i.e., the attack attempts to compute the base-g logarithms of g^x modulo p, and g^y modulo p, respectively. Once the attacker obtains x, y, he can then easily compute the shared secret information $(g^{xy} \bmod p)$.

Finding the value of x knowing $(g^x \bmod p)$ is the Diffie–Hellman discrete logarithm problem.

Definition 12.2.1 The **Diffie–Hellman discrete logarithm problem (DHDLP)** is stated as follows: let $g \in \mathbb{Z}_p^\times$ be a primitive root modulo p. Given an element $h \in \mathbb{Z}_p^\times$, find the unique value of x, $0 \leq x \leq p-2$, for which

$$h = (g^x \bmod p).$$

The Discrete Logarithm Assumption (DLA) (Section 9.4) says for any probabilistic polynomial time algorithm A, the probability that A gives a solution to the DHDLP

$$h = (g^x \bmod p)$$

is negligible, i.e., it is a negligible function of l.

The DLA (if true) says that the DHDLP cannot be solved in polynomial time, and this ensures the security of the DHKEP.

The DHDLP is a special case of the general discrete logarithm problem.

We recall some basic notions of group theory from Chapter 5.

Let G be a finite group, let $g \in G$, and let $H = \langle g \rangle$ be the cyclic subgroup generated by g. The order of H equals the order of the element g; it is the smallest positive integer n so that $g^n = 1$ (1 denotes the identity element in G). Let $h \in H$. Then

$$g^x = h$$

for a unique x with $0 \leq x \leq n-1$.

Definition 12.2.2 The **discrete logarithm problem (DLP)** is stated as follows: given $g \in G$, $h \in H = \langle g \rangle$, $n = |H|$, find the unique value of x, $0 \leq x \leq n-1$, so that the equation

$$g^x = h$$

is satisfied.

Example 12.2.3 Take $G = U(\mathbb{Z}_{18})$, and let $g = 5 \in U(\mathbb{Z}_{18})$. Then $\langle 5 \rangle = H = U(\mathbb{Z}_{18})$. (In fact, $U(\mathbb{Z}_{18})$ is cyclic of order $\phi(18) = 6$.) With $h = 13 \in U(\mathbb{Z}_{18})$, the DLP is stated as follows: find the unique $0 \leq x \leq 5$ so that

$$5^x = 13.$$

As one can check, the solution to the DLP is $x = 4$.

□

Example 12.2.4 In the group S_4, the solution to the DLP

$$\begin{pmatrix} 0\,1\,2\,3 \\ 2\,0\,3\,1 \end{pmatrix}^x = \begin{pmatrix} 0\,1\,2\,3 \\ 1\,3\,0\,2 \end{pmatrix}$$

is $x = 3$.

□

If the group G is additive, then the DLP seeks x so that

$$xg = \underbrace{g + g + \cdots + g}_{x} = h.$$

For example, given the cyclic additive group $\mathbb{Z}_{10}$ generated by $g = 3$, then with $h = 4$, the DLP is

$$x \cdot 3 = 3x = 4$$

in $\mathbb{Z}_{10}$, equivalently, $3x \equiv 4 \pmod{10}$. Since $3^{-1} = 7$ in $U(\mathbb{Z}_{10})$, then $x = ((7 \cdot 4) \bmod 10) = 8$ solves the DLP.

For certain groups, the DLP is very easy to solve. For instance, suppose G is the cyclic additive group $\mathbb{Z}_n$, generated by g with $\gcd(g, n) = 1$, and let $y \in \mathbb{Z}_n$. The corresponding DLP $gx = y$ has solution $x = g^{-1}y$ which can be found in polynomial time $O(m^3)$, where $m = \log_2(n)$.

For other groups, the DLP is very hard to solve, for instance, the case $G = \mathbb{Z}_p^\times$, for a large prime p (this is the DHDLP).

In what follows, we review the most efficient (non-polynomial) algorithms for solving the DHDLP. We begin with several methods for solving the general DLP, i.e., the DLP for an arbitrary finite group G. These methods can then be applied to the case $G = \mathbb{Z}_p^\times$.

12.2.1 The General DLP

Algorithm 12.2.5 (NAIVE_DLP)

```
Input:     $G$, a cyclic group of order $N$, generated by $g$,
           and an element $h$ randomly chosen from $G$
Output:    integer $i$, $0 \le i \le N-1$, with $g^i = h$
Algorithm:
           for $i = 0$ to $N-1$ do
              $s \leftarrow g^i$
              if $s = h$, then output $i$
           next $i$
```

Clearly, Algorithm 12.2.5 solves the DLP in $O(N)$ steps. However, the running time is non-polynomial as a function of input as measured in bits. We can improve on the efficiency of our solution to the DLP. We first prove a lemma.

Suppose $G = \langle g \rangle$, $N = |G|$, and $h \in_R G$. Thus, there exists an integer x, $0 \le x \le N-1$, with $g^x = h$; a solution to the DLP exists.

Lemma 12.2.6 *Let* $n = 1 + \lfloor\sqrt{N}\rfloor$ *and let* x *be an integer with* $0 \le x \le N-1$. *Then there exist integers* q *and* r *with* $x = qn + r$ *and* $0 \le r < n$, $0 \le q < n$.

Proof Using the Division Algorithm, write

$$x = nq + r, \quad 0 \le r < n,$$

for integers q and r. We show that $q < n$. Suppose that $q \ge n$. Then

$$nq \ge n^2 = (1 + \lfloor\sqrt{N}\rfloor)^2 > (\sqrt{N})^2 = N > x,$$

and thus $r < 0$, which is impossible. Thus $q < n$.

We claim that $q \ge 0$. For if $q < 0$, then $-q > 0$, and hence $n(-q) > n$. But then $r + nq \ge 0$ implies $r \ge -nq = n(-q) > n$, a contradiction.

□

Algorithm 12.2.7 (Baby-Step/Giant-Step (BSGS))

Input: G, a cyclic group of order N, generated by g,
and an element h randomly chosen from G
Output: integer i, $0 \le i \le N-1$, with $g^i = h$
Algorithm:

Step 1. Let $n = 1 + \lfloor\sqrt{N}\rfloor$ and construct two sets of group elements:

$$S_1 = \{e = g^0, g = g^1, g^2, g^3, \ldots, g^{n-1}\},$$

$$S_2 = \{h, hg^{-n}, hg^{-2n}, hg^{-3n}, \ldots, hg^{-(n-1)n}\}.$$

Step 2. Find an element hg^{-jn} in S_2 that matches an element g^k in S_1. Now, $hg^{-jn} = g^k$ implies that $g^{k+jn} = h$. Hence a solution is $i = k + jn$.

□

Lemma 12.2.6 guarantees that Step 2 of BSGS results in a match. The BSGS algorithm is attributed to Shanks [53].

Proposition 12.2.8 *The Baby-Step/Giant-Step algorithm solves the DLP in* $O(\sqrt{N}\log_2(N))$ *steps.*

We know that $h = g^x$ for some x, $0 \le x \le N - 1$. Let $n = 1 + \lfloor \sqrt{N} \rfloor$. By Lemma 12.2.6, there exist $0 \le q$ and $r < n$ with $x = qn + r$. Thus $h = g^x = g^{qn+r} = g^{qn} g^r$, and so $g^r = hg^{-qn}$. Since $0 \le q, r < n$, $g^r \in S_1$ and $hg^{-qn} \in S_2$. So there is a match; $qn + r$ is a solution to the DLP.

Regarding the time complexity of this solution, Step 1 can be completed in $O(n)$ steps. To find a match in Step 2, we compare each element of S_2 to all elements of S_1. We do this by sorting all $2n$ elements in S_1, S_2 using a standard sorting algorithm (like MERGE_SORT). Thus Step 2 is completed in $O(n \log_2(n))$ steps. So the total number of steps required to implement BSGS is

$$O(n) + O(n \log(n)) = O(n \log(n)).$$

Since $n \approx \sqrt{N}$, the time complexity of BSGS is therefore

$$O(\sqrt{N} \log_2(\sqrt{N})) = O(\sqrt{N} \log_2(N)).$$

□

Of course, the BSGS algorithm has running time that is non-polynomial as a function of input as measured in bits, but it is clearly more efficient than Algorithm 12.2.5 (NAIVE_DLP).

Example 12.2.9 Let $G = U(\mathbb{Z}_{17})$; 3 is a primitive root modulo 17. We solve the DLP

$$3^x = 15$$

using the BSGS algorithm. In this case, $N = 16$, and so $n = 1 + \lfloor \sqrt{16} \rfloor = 5$. Thus,

k	S_1	S_2
0	$3^0 \equiv 1$	$15 \cdot 3^0 \equiv 15$
1	$3^1 \equiv 3$	$15 \cdot 3^{-5} \equiv 3$
2	$3^2 \equiv 9$	$15 \cdot 3^{-10} \equiv 4$
3	$3^3 \equiv 10$	$15 \cdot 3^{-15} \equiv 11$
4	$3^4 \equiv 13$	$15 \cdot 3^{-20} \equiv 9$

Thus a match occurs with the pairing 3^1 and $15 \cdot 3^{-5}$. Thus $(3^6 \bmod 17) = 15$ and so the DLP has solution $i = 6$.

□

Remark 12.2.10 In the BSGS algorithm, if there is exactly one match between elements of S_1 and S_2, it cannot occur for certain pairings. For instance, we cannot have $g^{n-1} = hg^{-(n-1)n}$ as the only match between elements of S_1 and S_2. For then

$$g^{(n-1)+(n-1)n} = g^{n^2-1} = h.$$

The condition $n^2 > N$ then implies $n^2 - 1 \geq N$, which violates Lemma 12.2.6.

□

There is a probabilistic version of the BSGS algorithm based on the collision theorem of Proposition 2.3.4.

Algorithm 12.2.11 (Probabilistic BSGS)

Input: G, a cyclic group of order N, generated by g,
and an element h randomly chosen from G
Output: integer i, $0 \leq i \leq N-1$, with $g^i = h$
Algorithm:

Step 1. Choose an integer n, $1 \leq n \leq N$, and compute the group elements

$$S_1 = \{e = g^0, g = g^1, g^2, g^3, \ldots, g^{n-1}\}.$$

Step 2. Choose an integer $m \geq 1$, and let $k_1, k_2, \cdots, k_m$ be a randomly chosen sequence of integers, $0 \leq k_i < N$ (with replacement). Then $g^{k_1}, g^{k_2}, \ldots, g^{k_m}$ can be considered as a sequence of terms chosen at random from G (with replacement). Multiply each term g^{k_i} by h to form the sequence of group elements

$$S_2 = hg^{k_1}, hg^{k_2}, hg^{k_3}, \ldots, hg^{k_m}.$$

Step 3. If a match between S_1 and S_2 occurs, i.e., $hg^{k_j} = g^i$ for some i, j, then $g^{i-k_j} = h$ and $x = ((i - k_j) \bmod N)$ is a solution to the DLP $g^x = h$.

□

The success of Algorithm 12.2.11 depends on whether a match occurs on Step 3. We can calculate the probability of such a match.

Proposition 12.2.12 *In Algorithm 12.2.11, the probability that at least one term of sequence S_2 matches some element of S_1 is*

$$\Pr(hg^{k_j} = g^i \text{ for some } i, j) = 1 - \left(1 - \frac{n}{N}\right)^m.$$

Proof Since g generates G, the powers of g in S_1 are distinct. Since the terms $g^{k_1}, g^{k_2}, \ldots, g^{k_m}$ are chosen at random from G, multiplying each term by h results in a random sequence of terms because the map $\psi_h : G \to G$ defined as $\psi_h(g) = hg$ is a permutation of G. The result now follows from Proposition 2.3.4.

□

Example 12.2.13 We take $G = \mathbb{Z}_{7057}^\times$, with primitive root $g = 5$. Thus $N = |\mathbb{Z}_{7057}^\times| = 7056$. Let $h = 1000 \in G$. We seek to solve the DHDLP

$$5^x = 1000.$$

In Step 1 of Algorithm 12.2.11, we take $n = 60$, so that

$$S_1 = \{1, 5, 5^2, 5^3, \dots, 5^{59}\}.$$

In Step 2, we seek the minimal $m \geq 1$ so that it is likely that a match occurs between S_1 and

$$S_2 = 1000(5^{k_1}), 1000(5^{k_2}), \dots, 1000(5^{k_m}).$$

Applying Proposition 12.2.12, we take $m = 82$, since $m = 82$ is minimal with

$$\Pr(1000(5^{k_j}) = 5^i \text{ for some } i, j) = 1 - \left(1 - \frac{60}{7056}\right)^{82} > 1/2.$$

Thus with $n = 60$ and $m = 82$, it is likely that Algorithm 12.2.11 solves the DHDLP.

□

12.2.2 Index Calculus

The index calculus algorithm is a method for solving the DLP in the case $G = U(\mathbb{Z}_p)$ that is more efficient than BSGS. We assume that p is a large random prime and g is a primitive root modulo p. Let $h \in U(\mathbb{Z}_p)$. We seek x, $0 \leq x \leq p-2$, for which $g^x = h$ in $U(\mathbb{Z}_p)$.

We recall the notion of a smooth integer from Section 9.3.3. Let $B \geq 2$ be a real number. An integer $m \geq 2$ is **B-smooth** if each prime factor of m is less than or equal to B.

For $n \geq 2$, let $\Psi(n, B)$ be the number of B-smooth integers j with $2 \leq j \leq n$. For instance, $\Psi(10, 3) = 4$ since $2, 3, 4, 9$ are the only 3-smooth integers with $2 \leq j \leq 10$.

Algorithm 12.2.14 (Index Calculus)

Input: A large prime p, a primitive root g modulo p,
and an element h randomly chosen from $U(\mathbb{Z}_p)$
Output: integer i, $0 \leq i \leq p-2$, with $g^i = h$
Algorithm:

Step 1. The first step in the index calculus is to choose a relatively small bound B and then find more than $\pi(B)$ residues $(g^i \bmod p)$ for which $(g^i \bmod p)$ is B-smooth. These residues are randomly generated by choosing a random sequence of integers $m_1, m_2, m_3, \dots$ and checking to see whether each $2 \leq (g^{m_j} \bmod p) \leq p-1$ is B-smooth.

Step 2. Once $\pi(B)$ such residues have been found, they form a system (re-indexing the m_j if necessary):

$$\left\{\begin{array}{l} g^{m_1} \equiv q_1^{e_{1,1}} q_2^{e_{1,2}} \cdots q_k^{e_{1,k}} \pmod{p} \\ \\ g^{m_2} \equiv q_1^{e_{2,1}} q_2^{e_{2,2}} \cdots q_k^{e_{2,k}} \pmod{p} \\ \\ \qquad\qquad \vdots \\ \\ g^{m_r} \equiv q_1^{e_{r,1}} q_2^{e_{r,2}} \cdots q_k^{e_{r,k}} \pmod{p} \end{array}\right. \tag{12.1}$$

for some k, r, with $k \leq \pi(B) \leq r$. Also, the primes satisfy $2 \leq q_b \leq B$ and $e_{a,b} \geq 0$ for all $1 \leq a \leq r$, $1 \leq b \leq k$.

Let DLOG $=$ DLOG$_{p,g}$. For each $1 \leq i \leq r$,

$$g^{m_i} g^{-\mathrm{DLOG}(q_1^{e_{i,1}} q_2^{e_{i,2}} \cdots q_k^{e_{i,k}})} \equiv g^{m_i - \mathrm{DLOG}(q_1^{e_{i,1}} q_2^{e_{i,2}} \cdots q_k^{e_{i,k}})} \equiv 1 \pmod{p}.$$

By Proposition 5.7.3,

$$m_i - \mathrm{DLOG}(q_1^{e_{i,1}} q_2^{e_{i,2}} \cdots q_k^{e_{i,k}})$$

is a multiple of the order of g in $U(\mathbb{Z}_p)$, which is $p - 1$. Using familiar laws of logarithms,

$$\begin{aligned} &\mathrm{DLOG}(q_1^{e_{i,1}} q_2^{e_{i,2}} \cdots q_k^{e_{i,k}}) \\ &\qquad = e_{1,1}\mathrm{DLOG}(q_1) + e_{1,2}\mathrm{DLOG}(q_2) + \cdots + e_{1,k}\mathrm{DLOG}(q_k), \end{aligned}$$

thus the system (12.1) yields the $r \times k$ linear system, taken modulo $p - 1$:

$$\left\{\begin{array}{l} e_{1,1}\mathrm{DLOG}(q_1) + e_{1,2}\mathrm{DLOG}(q_2) + \cdots + e_{1,k}\mathrm{DLOG}(q_k) \equiv m_1 \\ \\ e_{2,1}\mathrm{DLOG}(q_1) + e_{2,2}\mathrm{DLOG}(q_2) + \cdots + e_{2,k}\mathrm{DLOG}(q_k) \equiv m_2 \\ \\ \qquad\qquad\qquad \vdots \\ \\ e_{r,1}\mathrm{DLOG}(q_1) + e_{r,2}\mathrm{DLOG}(q_2) + \cdots + e_{r,k}\mathrm{DLOG}(q_k) \equiv m_r. \end{array}\right.$$

We think of DLOG(q_t), $1 \leq t \leq k$, as variables. Since $r \geq k$, there are at least as many equations as variables and so the system has a solution in DLOG(q_t), $1 \leq t \leq k \leq \pi(B)$.

Step 3. We find an integer k for which $(hg^{-k} \mod p)$ is B-smooth. For such k,

$$h \equiv g^k q_1^{e_1} q_2^{e_2} \cdots q_k^{e_k} \pmod{p}$$

for $q_t \leq B$, $e_t \geq 0$. Thus

$$\mathrm{DLOG}(h) \equiv k + e_1\mathrm{DLOG}(q_1) + e_2\mathrm{DLOG}(q_2) + \cdots + e_k\mathrm{DLOG}(q_k)$$

modulo $p-1$. The discrete log DLOG(h) is then computed by substituting in the values of DLOG(q_t) from Step 2.

□

Example 12.2.15 We take $p = 997$ and $g = 7$ and use Algorithm 12.2.14 to compute $\mathrm{DLOG}_{997,7}(831)$, i.e., we find the unique x, $0 \leq x \leq 995$ so that

$$7^x = 831$$

in $\mathbb{Z}_{997}^{\times}$.

Step 1. We choose $B = 3$ so that $\pi(3) = 2$. We select random i, $0 \leq i \leq 995$, until we obtain three residues $(7^i \bmod 997)$ that are 3-smooth. The residues are

$$\begin{cases} (7^{615} \bmod 997) = 2^3 \cdot 3^2 \\ (7^{231} \bmod 997) = 2 \cdot 3^5 \\ (7^{15} \bmod 997) \ = 2^5 \cdot 3. \end{cases}$$

Step 2. Applying DLOG yields

$$\begin{cases} 3 \cdot \mathrm{DLOG}(2) + 2 \cdot \mathrm{DLOG}(3) \equiv 615 \pmod{996} \\ \mathrm{DLOG}(2) + 5 \cdot \mathrm{DLOG}(3) \quad\ \equiv 231 \pmod{996} \\ 5 \cdot \mathrm{DLOG}(2) + \mathrm{DLOG}(3) \quad\ \equiv 15 \pmod{996}. \end{cases}$$

The system has solution DLOG(2) = 201, DLOG(3) = 6.

Step 3. We find that

$$831 \cdot 7^{-162} \equiv 2^3 \cdot 3^4 \pmod{997},$$

and thus

$$\begin{aligned}\text{DLOG}(831) &= 162 + 3 \cdot \text{DLOG}(2) + 4 \cdot \text{DLOG}(3)\\ &= 162 + 3 \cdot 201 + 4 \cdot 6\\ &= 789.\end{aligned}$$

□

12.2.3 *Efficiency of Index Calculus*

How efficient is the Index Calculus algorithm?

Due to the theorem of E. R. Canfield, P. Erdős and C. Pomerance (Theorem 9.3.18), and its corollary (Corollary 9.3.19), we can complete Step 1 in a reasonable (yet non-polynomial) amount of time.

Proposition 12.2.16 *The Index Calculus algorithm solves the DLP in subexponential time*

$$O(2^{c(\log_2(p))^{1/3}(\log_2(\log_2(p)))^{2/3}}),$$

where c is a small constant.

Proof (Sketch) Let p be a large prime. The most time-consuming part of the algorithm is Step 1. By Corollary 9.3.19, in a random sequence of $\lceil L(p)^{\sqrt{2}} \rceil$ integers modulo p, we expect to find $\pi(L(p)^{\frac{1}{\sqrt{2}}})$ integers that are $L(p)^{\frac{1}{\sqrt{2}}}$-smooth.

If an integer $(i \bmod p)$ is chosen at random, then the integer $(g^i \bmod p)$ is also (essentially) chosen at random since the map

$$\text{DEXP}_{p,g} : U(\mathbb{Z}_p) \to U(\mathbb{Z}_p)$$

is a permutation. Therefore in a random sequence of $\lceil L(p)^{\sqrt{2}} \rceil$ residues $(g^i \bmod p)$, we expect to find $\pi(L(p)^{\frac{1}{\sqrt{2}}})$ residues $(g^i \bmod p)$ that are $L(p)^{\frac{1}{\sqrt{2}}}$-smooth. Thus Step 1 takes time $L(p)^{\sqrt{2}} = e^{\sqrt{2}(\ln(p))^{1/2}(\ln(\ln(p)))^{1/2}}$. Ultimately this can be reduced to an overall running time of $e^{a(\ln(p))^{1/3}(\ln(\ln(p)))^{2/3}}$ where a is a small constant. Changing base yields run time

$$e^{a(\ln(p))^{1/3}(\ln(\ln(p)))^{2/3}} = O(2^{c(\log_2(p))^{1/3}(\log_2(\log_2(p)))^{2/3}}),$$

where c is a small constant.

□

In Example 12.2.15, $p = 997$. Thus

$$L(p) = L(997) \approx 38.57451136,$$

and so, $\lceil L(997)^{\sqrt{2}} \rceil = 176$, $L(997)^{\frac{1}{\sqrt{2}}} \approx 13.23377643$, and $\pi(13) = 6$. In view of Corollary 9.3.19, we need to choose 176 random residues ($7^i \bmod 997$) to expect 6 integers to be 13-smooth. Of course, $p = 997$ is not a large prime. In Example 12.2.15 with some trial and error, we found 3 residues ($7^i \bmod 997$) that were 3-smooth.

Remark 12.2.17 Godušová [20] has extended the Index Calculus method to $U(\mathbb{F}_{p^m}) = \mathbb{F}_{p^m}^{\times}$.

We ask: Can we extend the Index Calculus method to arbitrary groups?

Suppose G is any finite cyclic group of order n generated by h, $\langle h \rangle = G$. By a theorem of Dirichlet, there exists a prime p of the form $p = nk + 1$ for some k [47, Theorem 3.3]. Now,

$$U(\mathbb{Z}_p) = U(\mathbb{Z}_{nk+1}),$$

which is a cyclic group of order nk.

Let g be a primitive root modulo p, so that $\langle g \rangle = U(\mathbb{Z}_p)$. Then $U(\mathbb{Z}_p)$ has a cyclic subgroup H of order n generated by g^k; $\langle g^k \rangle = H$. Moreover, there is a group isomorphism

$$\psi : G \to H$$

defined as $h \mapsto g^k$. Thus G can be embedding into $U(\mathbb{Z}_p)$; G is essentially a cyclic subgroup of $U(\mathbb{Z}_p)$ of order n.

So, given the DLP

$$h^x = a$$

in G, we can apply the map ψ to yield the DLP in $U(\mathbb{Z}_p)$:

$$\psi(h)^x = \psi(a),$$

or

$$g^{xk} = b, \tag{12.2}$$

with $\psi(a) = b$. The DLP (12.2) could then be solved using Index Calculus, obtaining xk; dividing by k would then give x.

Unfortunately, there seems to be no easy way to find the required integer k so that $nk + 1$ is prime, i.e., the embedding ψ is difficult to compute. Even if we were able to find xk to solve the DLP (12.2), we need k to find x.

Moreover, for an arbitrary finite cyclic group G, there seems to be no obvious way to compute the image $\psi(a) = b$ *without already knowing* the discrete logarithm of a in G.

So there seems to be no practical way to extend the Index Calculus method to arbitrary groups using the notion of an embedding $\psi : G \to H \leq U(\mathbb{Z}_p)$.

For certain groups however, i.e., cyclic subgroups of the elliptic curve group $E_{ns}(K)$, one can compute the embedding ψ and thus solve the DLP, see Section 14.2.

□

12.2.4 The Man-in-the-Middle Attack

We close this chapter with another type of attack on the DHKEP; the "Man-in-the-Middle" attack is a "network" attack on the DHKEP.

In the attack, the notation Malice ("Alice") indicates that Malice is posing as Alice, and the notation Malice ("Bob") indicates that Malice is posing as Bob.

Attack 12.2.18 (Man-in-the-Middle Attack on DHKEP)

Premise: Alice and Bob share a large prime p and g, a primitive root modulo p.

Result of Attack: Alice and Bob think they are sharing a new element $\mathbb{Z}_p^\times$, but actually, Alice is sharing a new element with Malice and Bob and sharing a new element with Malice.

1. Alice randomly chooses an integer x, $0 \leq x \leq p-2$. She computes $h = (g^x \bmod p)$ and sends h to Malice("Bob").

1′. Malice("Alice") randomly chooses an integer m, $0 \leq m \leq p-2$. He computes $l = (g^m \bmod p)$ and sends l to Bob.

2. Bob randomly chooses an integer y, $0 \leq y \leq p-2$. He computes $k = (g^y \bmod p)$ and sends k to Malice("Alice").

2′. Malice("Bob") sends l to Alice.

3. Alice computes $(l^x \bmod p) = ((g^m)^x \bmod p) = (g^{mx} \bmod p)$.
4. Bob computes $(l^y \bmod p) = ((g^m)^y \bmod p) = (g^{my} \bmod p)$.

□

After Step 3 in the attack, Alice and Malice share the integer $(g^{mx} \bmod p)$, and after Step 4, Bob and Malice share $(g^{my} \bmod p)$ (Figure 12.1).

As a result of the Man-in-the-Middle attack, Malice will possess the residues $(g^x \bmod p)$ and $(g^y \bmod p)$. His goal is to determine the residue $(g^{xy} \bmod p)$ possessed by Alice and Bob. Can he use his knowledge of $(g^x \bmod p)$ and $(g^y \bmod p)$ to determine $(g^{xy} \bmod p)$?

Malice wants to solve the "computational DHDLP."

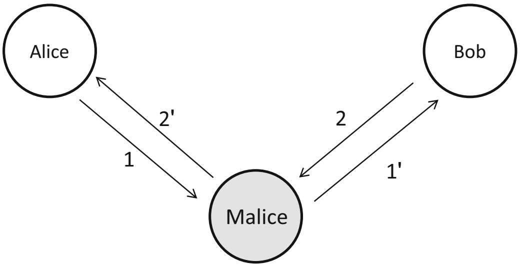


Fig. 12.1 Man-in-the-Middle attack on DHKEP

Definition 12.2.19 The **Computational Diffie–Hellman discrete logarithm problem (CDHDLP)** is stated as follows: let g be a primitive root modulo p. Given the residues $(g^x \bmod p)$, $g^y \bmod p)$, find the residue $(g^{xy} \bmod p)$.

It is clear that the CDHDLP is no harder than the DHDLP: if the discrete logarithms x, y can be found, then $(g^{xy} \bmod p)$ can be easily computed. On the other hand, if an efficient solution to the CDHDLP exists, then it is not known whether this would imply an efficient solution to the DHDLP.

12.3 Exercises

1. Alice and Bob are using the DHKEP to exchange keys using the group $G = U(\mathbb{Z}_{2861})$ and primitive root $g = 2$. Alice randomly chooses $x = 623$ and Bob randomly chooses $y = 14$.
   (a) Compute the key k shared by Alice and Bob. Write k as a 12-bit binary number.
   (b) Suppose Alice and Bob are using the Vernam cipher with message length 12 to communicate securely. Compute
   $$C = e(100101111000, k).$$
2. Use the Baby-Step/Giant-Step algorithm to solve the DLP
   $$2^x = 887$$
   in $U(\mathbb{Z}_{2861})$.
3. Find the minimal size of a prime p (in bits) to avert an attack on the DHKEP using the BSGS algorithm.
4. Use Index Calculus to solve the DLP
   $$2^x = 100$$

in $U(\mathbb{Z}_{2861})$. Hint: $2^{1117} = 110$, $2^{243} = 1485$, and $2^{1416} = 2711$ are 3-smooth residues in $\mathbb{Z}_{2861}$.

5. Find the minimal size of a prime p (in bits) to safeguard against an attack on the DHKEP using Index Calculus.
6. For an integer $n \geq 1$, let $L(n) = e^{(\ln(n))^{1/2}(\ln(\ln(n)))^{1/2}}$. Prove that $L(n) = O(2^{(\log_2(n))^{1/2}(\log_2(\log_2(n)))^{1/2}})$.
7. Let $p = 2^{31} - 1 = 2147483647$ be the Mersenne prime.

   (a) Use Exercise 6 to approximate $B = L(2^{31} - 1)^{\frac{1}{\sqrt{2}}}$.
   (b) Estimate the number of random integers modulo $2^{31} - 1$ that need to be selected in order to find $\pi(B)$ integers that are B-smooth.

8. Let $U(\mathbb{Z}_{100})$ be the units group of the residue ring $\mathbb{Z}_{100}$.

   (a) Show that 3 has order 20 in $U(\mathbb{Z}_{100})$.
   (b) Prove that $\langle 3 \rangle \leq U(\mathbb{Z}_{100})$ can be embedded into the group of units $U(\mathbb{Z}_p)$ for some prime p.
   (c) Use (a) and (b) to write the DLP $3^x = 23$ in $\langle 3 \rangle$ as a DLP in $U(\mathbb{Z}_p)$.

Chapter 13
Elliptic Curves in Cryptography

The Diffie–Hellman protocol uses the group $U(\mathbb{Z}_p)$ to exchange keys. Other groups can be employed in a Diffie–Hellman-type protocol. For instance, we could use an elliptic curve group.

In this chapter we introduce the elliptic curve group and show how it can be used to strengthen the Diffie–Hellman key exchange protocol.

13.1 The Equation $y^2 = x^3 + ax + b$

Let K be a field. Let

$$x^3 + a_2x^2 + a_1x + a_0 \tag{13.1}$$

be a monic cubic polynomial over K. Assuming $\mathrm{char}(K) \neq 3$, the transformation $x' = x - \frac{a_2}{3}$ yields a somewhat simpler equation:

$$\begin{aligned}
&\left(x - \frac{a_2}{3}\right)^3 + a_2\left(x - \frac{a_2}{3}\right)^2 + a_1\left(x - \frac{a_2}{3}\right) + a_0 \\
&\quad = x^3 - 3x^2\left(\frac{a_2}{3}\right) + 3x\left(\frac{a_2^2}{9}\right) - \frac{a_2^3}{27} + a_2\left(x^2 - 2x\left(\frac{a_2}{3}\right) + \frac{a_2^2}{9}\right) \\
&\qquad\quad + \ a_1x - \frac{a_1a_2}{3} + a_0 \\
&\quad = x^3 + \left(-\frac{a_2^2}{3} + a_1\right)x + \frac{2a_2^3}{27} - \frac{a_1a_2}{3} + a_0.
\end{aligned}$$

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_13

So with $a = -\frac{a_2^2}{3} + a_1$, $b = \frac{2a_2^3}{27} - \frac{a_1 a_2}{3} + a_0$, the cubic (13.1) can be written as

$$x^3 + ax + b.$$

Consider the equation

$$y^2 = x^3 + ax + b \tag{13.2}$$

over K. The **graph** of equation (13.2) is defined as

$$\{(x, y) \in K \times K : \ y^2 = x^3 + ax + b\}.$$

The graph can be viewed as the collection of zeros in $K \times K$ of the function

$$f(x, y) = y^2 - (x^3 + ax + b).$$

Let $\overline{K}$ be an algebraic closure of K (for instance, if $K = \mathbb{R}$, then $\overline{\mathbb{R}} = \mathbb{C}$). Let (s, t) be a zero of the polynomial $f(x, y) = y^2 - (x^3 + ax + b)$ in $\overline{K} \times \overline{K}$. The Taylor series expansion of $f(x, y)$ about the point (s, t) is

$$\begin{aligned}
f(x, y) = {} & \frac{\partial f}{\partial x}(s, t)(x - s) + \frac{\partial f}{\partial y}(s, t)(y - t) \\
& + \frac{1}{2!}\left(\frac{\partial^2 f}{\partial x^2}(s, t)(x - s)^2 + \frac{\partial^2 f}{\partial y^2}(s, t)(y - t)^2\right) \\
& + \frac{1}{3!}\frac{\partial^3 f}{\partial x^3}(s, t)(x - s)^3 \\
= {} & (3s^2 + a)(x - s) + 2t(y - t) \\
& + 3s(x - s)^2 + (y - t)^2 + (x - s)^3.
\end{aligned}$$

The **linear form** L of the Taylor series expansion is the first two terms of the expansion, thus

$$\begin{aligned}
L &= \frac{\partial f}{\partial x}(s, t)(x - s) + \frac{\partial f}{\partial y}(s, t)(y - t) \\
&= (3s^2 + a)(x - s) + 2t(y - t).
\end{aligned}$$

The **tangent space** to the graph of equation (13.2) at the point (s, t) is defined as

$$\Theta_{(s,t)} = \{(x, y) \in \overline{K} \times \overline{K} : \ (3s^2 + a)(x - s) + 2t(y - t) = 0\};$$

$\Theta_{(s,t)}$ is the graph of the equation $L = 0$ in $\overline{K} \times \overline{K}$, cf. [52, Chapter II, Sections 1.2–1.5].

The graph of (13.2) is **smooth** if the graph contains no points in $\overline{K} \times \overline{K}$ for which both partial derivatives $\dfrac{\partial f}{\partial x}$ and $\dfrac{\partial f}{\partial y}$ vanish simultaneously. In other words, the graph is smooth if there is no point $(s, t) \in \overline{K} \times \overline{K}$ on the graph for which

$$\begin{cases} \frac{\partial f}{\partial x}(s,t) = 0 \\ \\ \frac{\partial f}{\partial y}(s,t) = 0. \end{cases}$$

A graph $y^2 = x^3 + ax + b$ is **singular** if it is not smooth. If a graph is singular, then there exists a point $(s, t) \in \overline{K} \times \overline{K}$ on the graph where both partials vanish, and this is a **singular point** on the graph.

Proposition 13.1.1 *The graph of $y^2 = x^3 + ax + b$ is smooth if and only if the dimension of the tangent space at every point of the graph is* 1.

Proof Suppose the graph $y^2 = x^3 + ax + b$ is not smooth. Then there exists a singular point (s, t) on the graph. At this singular point (s, t), the linear part of the Taylor series expansion of $f(x, y) = y^2 - (x^2 + ax + b)$ is identically zero. Thus

$$\Theta_{(s,t)} = \{(x, y) \in \overline{K} \times \overline{K} : \ (3s^2 + a)(x - s) + 2t(y - t) = 0\} = \overline{K}^2,$$

and so $\dim(\Theta_{(s,t)}) = 2 \neq 1$.

For the converse, suppose that $\dim(\Theta_{(s,t)}) \neq 1$ at some point (s, t) on the graph. Now, $L = 0$ is a linear equation in x and y and so its solutions have dimension 1 in $\overline{K}^2$ unless the coefficients of x and y are both 0. It follows that both partials vanish at (s, t), and so the graph is not smooth.

□

Proposition 13.1.2 *If K is a field with* $\mathrm{char}(K) = 2$, *then every curve of the form $y^2 = x^3 + ax + b$ is singular.*

Proof Given $y^2 = x^3 + ax + b$ over K, let $\alpha, \beta \in \overline{K}$ with $\alpha^2 = -a$, $\beta^2 = b$. Then

$$(\beta)^2 = (\alpha)^3 + a(\alpha) + b,$$

so that (α, β) is a point of $y^2 = x^3 + ax + b$ in $\overline{K} \times \overline{K}$ with $\frac{\partial f}{\partial x}(\alpha, \beta) = 3(\alpha)^2 + a = -3a + a = -2a = 0$ and $\frac{\partial f}{\partial y}(\alpha, \beta) = 2\beta = 0$.

□

Whether or not a graph $y^2 = x^3 + ax + b$ is smooth depends entirely on the value of its "elliptic" discriminant (not to be confused with the polynomial discriminant). The **elliptic discriminant** of equation $y^2 = x^3 + ax + b$ is given as

$$\mathcal{D}_E = -16(4a^3 + 27b^2).$$

Proposition 13.1.3 *The graph of $y^2 = x^3 + ax + b$ is smooth if and only if its elliptic discriminant $\mathcal{D}_E$ is non-zero.*

Proof Suppose $y^2 = x^3 + ax + b$ is not smooth. If $\mathrm{char}(K) = 2$, then $\mathcal{D}_E = 0$. So we assume that $\mathrm{char}(K) \neq 2$. There exists a point $(\alpha, \beta) \in \overline{K} \times \overline{K}$ with $\beta^2 = \alpha^3 + a\alpha + b$, $2\beta = 0$, and $3\alpha^2 + a = 0$. Thus

$$0 = 2\beta \cdot \beta = 2\beta^2 = 2\alpha^3 + 2a\alpha + 2b,$$

and so α is a root of the polynomial $2x^3 + 2xa + 2b$ in $\overline{K}$. Moreover, $6\alpha^2 + 2a = 0$ and so α is a root of multiplicity > 1. Consequently, α is a root of $x^3 + ax + b$ of multiplicity > 1. Thus by Ireland and Rosen [28, Chapter 19, Section 1, Lemma 1], $-4a^3 - 27b^2 = 0$, and so, $\mathcal{D}_E = 0$. Note also that the elliptic discriminant is the ordinary polynomial discriminant of $2x^3 + 2ax + 2b$, see [36, page 211].

Conversely, suppose $\mathcal{D}_E = 0$. If $\mathrm{char}(K) = 2$, then $y^2 = x^3 + ax + b$ is not smooth by Proposition 13.1.2. So we assume that $\mathrm{char}(K) \neq 2$. From $\mathcal{D}_E = 0$, we obtain $-4a^3 - 27b^2 = 0$, and so by Ireland and Rosen [28, Chapter 19, Section 1, Lemma 1], $x^3 + ax + b$ has a zero α in $\overline{K}$ of multiplicity > 1. Thus $3\alpha^2 + a = 0$. The point $(\alpha, 0) \in \overline{K} \times \overline{K}$ is a singular point of $y^2 = x^3 + ax + b$ since $(0)^2 = 0 = \alpha^3 + a\alpha + b$ with $2 \cdot 0 = 0$ and $3\alpha^2 + a = 0$. Thus $y^2 = x^3 + ax + b$ is not smooth.

□

Example 13.1.4 Let $K = \mathbb{R}$; an algebraic closure of $\mathbb{R}$ is $\mathbb{C}$. Let $y^2 = x^3 - x + 6$. The elliptic discriminant is $\mathcal{D}_E = -16(4(-1)^3 + 27(6)^2) = -15488 \neq 0$, and thus the graph is smooth. The point $(-2, 0)$ is on the graph. We compute the tangent space

$$\Theta_{(-2,0)} = \{(x, y) :\ (3(-2)^2 - 1)(x + 2) + 2(0)(y - 0) = 11(x + 2) = 0\},$$

and thus $\Theta_{(-2,0)}$ is the graph of $x+2 = 0$. The curve and the tangent space restricted to $\mathbb{R} \times \mathbb{R}$ are given in Figure 13.1.

□

Example 13.1.5 Let $K = \mathbb{R}$, $\overline{\mathbb{R}} = \mathbb{C}$, and $y^2 = x^3 - 3x + 2$. The elliptic discriminant is $\mathcal{D}_E = -16(4(-3)^3 + 27(2)^2) = 0$. Thus the graph is singular. The point $(1, 0)$ is a singular point of the graph. We have $\Theta_{(1,0)} = \mathbb{C}^2$. The tangent space at $(1, 0)$ contains two lines tangent to the curve at $(1, 0)$. They can be found by solving the equation $Q = 0$, where

$$Q = \frac{1}{2!}\left(\frac{\partial^2 f}{\partial x^2}(s, t)(x - s)^2 + \frac{\partial^2 f}{\partial y^2}(s, t)(y - t)^2\right)$$

is the **quadratic form** of the expansion of $f(x, y)$. We have

$$\begin{aligned} Q &= -3(x - 1)^2 + y^2 \\ &= (y - \sqrt{3}(x - 1))(y + \sqrt{3}(x - 1)). \end{aligned}$$

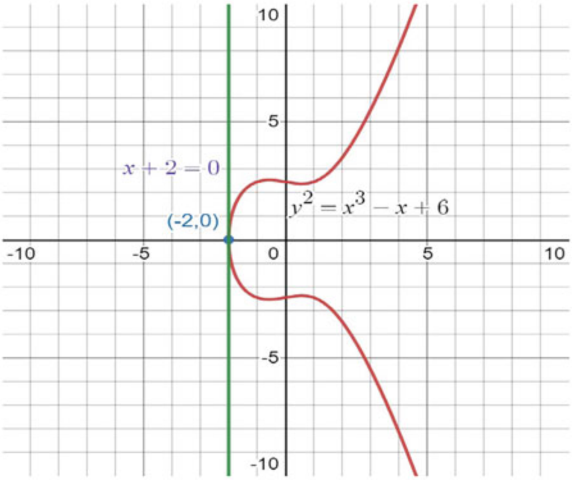


Fig. 13.1 Graph of $y^2 = x^3 - x + 6$, with 1-dimensional tangent space $\Theta_{(-2,0)}$

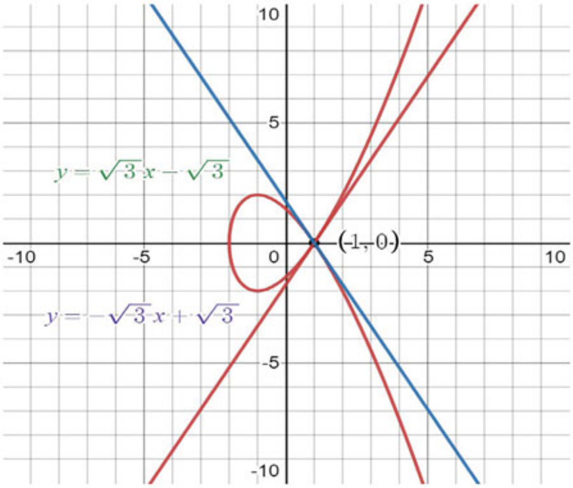


Fig. 13.2 Graph of $y^2 = x^3 - 3x + 2$, with singular point $(1, 0)$ and tangent cone $T_{(1,0)}$

Thus the tangent lines are $y = \sqrt{3}x - \sqrt{3}$ and $y = -\sqrt{3}x + \sqrt{3}$. The **tangent cone** at the singular point $(1, 0)$ is the collection of the tangent lines at $(1, 0)$,

$$T_{(1,0)} = \{y = \sqrt{3}x - \sqrt{3},\ y = -\sqrt{3}x + \sqrt{3}\},$$

[52, Chapter II, Section 1.5]; see Figure 13.2.

□

13.2 Elliptic Curves

Definition 13.2.1 Let K be a field. An **elliptic curve** over K is the graph of an equation

$$y^2 = x^3 + ax + b \tag{13.3}$$

over K which is smooth, together with a "point at infinity" $O = (\infty, \infty)$. We denote an elliptic curve over K by $E(K)$. Equivalently, an elliptic curve over K is defined as

$$E(K) = \{(x, y) \in K \times K : \ y^2 = x^3 + ax + b\} \cup \{O\},$$

where $\mathcal{D}_E = -16(4a^3 + 27b^2) \neq 0$; (13.3) is the **Weierstrass equation**.

According to our definition, there are no elliptic curves over fields of characteristic 2, cf. Proposition 13.1.2. An elliptic curve in characteristic 2 can be defined however as the graph of a **generalized Weierstrass equation**

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6.$$

For example, the **Koblitz curve**

$$y^2 + xy = x^3 + 1$$

is smooth over $\mathbb{F}_2$ and defines an elliptic curve in characteristic 2. We will not consider the characteristic 2 case in this book. The interested reader should see [63, Chapter 2, Section 2.7], [25, Chapter 5, Section 5.7], and [32, Chapter VI] for treatments of the characteristic 2 case.

Proposition 13.2.2 *Let $y^2 = x^3 + ax + b$ be an elliptic curve over K. Then the cubic $x^3 + ax + b$ has distinct zeros in an algebraic closure $\overline{K}$ of K.*

Proof Since $y^2 = x^3 + ax + b$ is an elliptic curve, $\mathcal{D}_E = -16(4a^3 + 27b^2) \neq 0$, and so $\mathcal{D} = -4a^3 - 27b^2 \neq 0$, where $\mathcal{D}$ is the ordinary polynomial discriminant of the cubic.

Let r_1, r_2, and r_2 be the zeros of $x^3 + ax + b$ in $\overline{K}$. By Ireland and Rosen [28, Chapter 19, Section 1, Lemma 1],

$$\mathcal{D} = (r_1 - r_2)^2(r_1 - r_3)^2(r_2 - r_3)^2,$$

and so the zeros of the cubic are distinct.

□

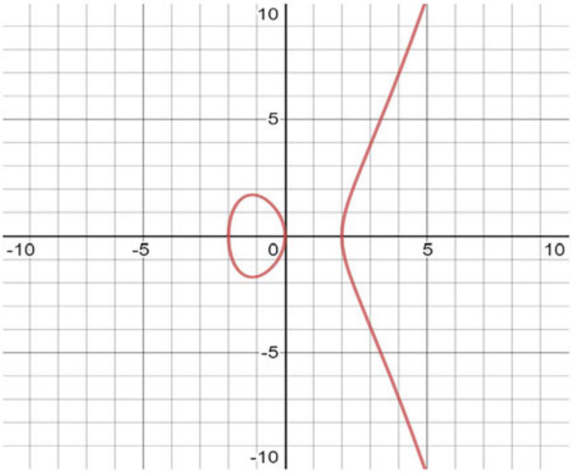


Fig. 13.3 Elliptic curve over $\mathbb{R}$ defined by $y^2 = x^3 - 4x$; cubic has three real roots $r_1 = -2$, $r_2 = 0$, and $r_3 = 2$

Elliptic curves over $\mathbb{R}$ are the easiest to visualize. The equation $y^2 = x^3 - x + 6$ defines an elliptic curve $E(\mathbb{R})$ over $\mathbb{R}$ whose graph is given in Figure 13.1. On the other hand, $y^2 = x^3 - 3x + 2$ does not define an elliptic curve.

Given an elliptic curve $y^2 = x^3 + ax + b$ over $\mathbb{R}$, the cubic $x^3 + ax + b$ has three distinct real zeros if $\mathcal{D} > 0$. If $\mathcal{D} < 0$, then the cubic has one real zero and two complex zeros [48, Proposition 4.60].

For example, the elliptic curve $y^2 = x^3 - x + 6$ has cubic $x^3 - x + 6$ with exactly one real root $r = -2$, and the complex roots are $1 \pm i\sqrt{2}$. On the other hand, the elliptic curve $y^2 = x^3 - 4x$ over $\mathbb{R}$ with cubic $x^3 - 4x$ has three real roots $r_1 = -2$, $r_2 = 0$, and $r_3 = 2$; the graph of $E(\mathbb{R})$ is given in Figure 13.3.

Elliptic curves can be defined over finite fields of characteristic not 2.

Example 13.2.3 Let $K = \mathbb{F}_5$. The graph of

$$y^2 = x^3 + 4x + 1$$

over $\mathbb{F}_5$ is an elliptic curve since the elliptic discriminant

$$\mathcal{D}_E = -16(4(4^3) + 27(1)^2) = 2 \neq 0$$

in $\mathbb{F}_5$. We have

$$E(\mathbb{F}_5) = \{(x, y) \in \mathbb{F}_5 \times \mathbb{F}_5 : \ y^2 = x^3 + 4x + 1\} \cup \{\mathcal{O}\}.$$

In this case, $E(\mathbb{F}_5)$ contains a finite number of points, which can be easily found using a table:

x	x^3+4x+1	y	points
0	1	± 1	(0, 1), (0, 4)
1	1	± 1	(1, 1), (1, 4)
2	2	none	none
3	0	0	(3, 0)
4	1	± 1	(4, 1), (4, 4)

There is no point in $E(\mathbb{F}_5)$ with x-coordinate 2 since $(2^3 + 4 \cdot 2 + 1 \bmod 5) = 2$ is not a quadratic residue mod 5, that is, $\left(\frac{2}{5}\right) = -1$. We have

$$E(\mathbb{F}_5) = \{(0, 1), (0, 4), (1, 1), (1, 4), (3, 0), (4, 1), (4, 4), \mathcal{O}\}.$$

□

13.3 Singular Curves

Let K be a field and let $y^2 = x^3 + ax + b$ with $\mathcal{D}_E = 0$. By Proposition 13.1.3, the graph of the equation is not smooth and hence does not define an elliptic curve over K; the graph contains at least one singular point. Such curves are still of interest however and will be discussed in detail in Chapter 14.

Assuming $\mathrm{char}(K) \neq 2$, $\mathcal{D}_E = 0$ implies $\mathcal{D} = 0$. Thus the cubic $x^3 + ax + b$ has either a double root or triple root. The root r with multiplicity > 1 gives rise to a singular point $(r, 0)$ on the graph of $y^2 = x^3 + ax + b$.

Example 13.3.1 Let $K = \mathbb{R}$ and $y^2 = x^3 - 3x - 2$. Then $\mathcal{D}_E = \mathcal{D} = 0$ and $x^3 - 3x - 2$ has a double root at $x = -1$. The only singular point on the graph is $(-1, 0)$, see Figure 13.4.

□

We let $E_{ns}(K)$ denote the set of all non-singular points on the curve $y^2 = x^3 + ax + b$ over K, together with $\mathcal{O}$. For example, for the curve $y^2 = x^3 - 3x - 2$,

$$E_{ns}(\mathbb{R}) = \{(x, y) : \; y^2 = x^3 - 3x - 2, (x, y) \neq (-1, 0)\} \cup \{\mathcal{O}\}.$$

Here is an example over a finite field.

Example 13.3.2 Let $K = \mathbb{F}_7$ and $y^2 = x^3 + 4x + 5$. Then (6, 0) is the only singular point and

$$E_{ns}(\mathbb{F}_7) = \{(2, 0), (3, 3), (3, 4), (4, 1), (4, 6), \mathcal{O}\}.$$

□

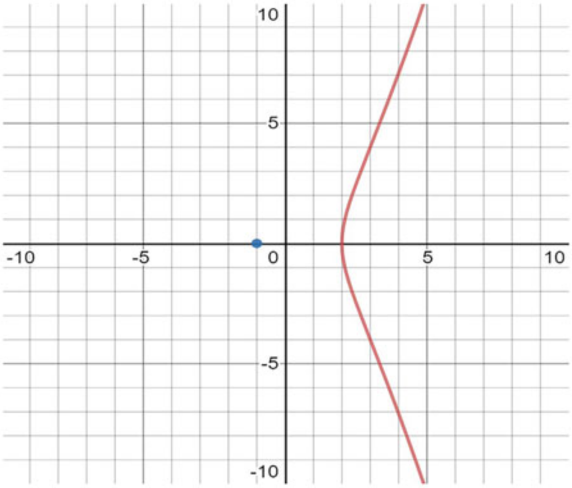


Fig. 13.4 Graph of $y^2 = x^3 - 3x - 2$, singular point is $(-1, 0)$

13.4 The Elliptic Curve Group

Let K be a field and let $E(K)$ be an elliptic curve over K. We can put a group structure on the points of $E(K)$; the special element O will serve as the identity element for the group. This group will be the "elliptic curve" group $E(K)$. Since elliptic curves over $\mathbb{R}$ are easier to visualize, we first show how the elliptic curve group is constructed for the case $K = \mathbb{R}$.

Let $E(\mathbb{R})$ be an elliptic curve over $\mathbb{R}$,

$$E(\mathbb{R}) = \{(x, y) \in \mathbb{R} \times \mathbb{R} : \ y^2 = x^3 + ax + b\} \cup \{O\}.$$

We begin by defining a binary operation

$$+ : \ E(\mathbb{R}) \times E(\mathbb{R}) \to E(\mathbb{R})$$

on $E(\mathbb{R})$. (As we shall see, the elliptic curve group is abelian, and thus we denote the binary operation by "+"; the binary operation is *not* the component-wise addition of the coordinates of the points.)

Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be points of $E(\mathbb{R})$.

Case 1. $P_1 \neq P_2$, $x_1 \neq x_2$. The line $\overleftrightarrow{P_1P_2}$ intersects $E(\mathbb{R})$ at a point P'. Let P_3 be the reflection of P' through the x-axis. We define

$$P_1 + P_2 = P_3.$$

Case 2. $P_1 \neq P_2$, $x_1 = x_2$, $y_1 \neq y_2$. In this case, the line $\overleftrightarrow{P_1P_2}$ is vertical and intersects $E(\mathbb{R})$ at the point at infinity $\mathcal{O}$. The reflection of $\mathcal{O}$ through the x-axis is again $\mathcal{O}$. So we define

$$P_1 + P_2 = \mathcal{O}.$$

Case 3. $P_1 = P_2$, $y_1 = y_2 \neq 0$. Since P_1 is a non-singular point of $y^2 = x^3 + ax + b$, the dimension of the tangent space Θ_{P_1} is 1. Hence there is a unique tangent line to the curve at the point P_1. The tangent line intersects the curve at a point P'; reflecting P_1 through the x-axis yields a point P_3 and we define

$$P_1 + P_2 = P_1 + P_1 = 2P_1 = P_3.$$

Case 4. $P_1 = P_2$, $y_1 = y_2 = 0$. As in Case 3, there is a unique tangent line to the curve at the point P_1, and in this case the tangent line is vertical and intersects the curve at $\mathcal{O}$; reflecting $\mathcal{O}$ through the x-axis yields $\mathcal{O}$ and we define

$$P_1 + P_2 = P_1 + P_1 = 2P_1 = \mathcal{O}.$$

Case 5. $P_2 = \mathcal{O}$. As in Case 2, the line $\overleftrightarrow{P_1\mathcal{O}}$ is vertical and intersects $E(\mathbb{R})$ at a point P' which is the reflection of P_1 through the x-axis. Reflecting P' back through the x-axis yields P_1, so we define

$$P_1 + \mathcal{O} = P_1.$$

In the manner described above, we define a binary operation on $E(\mathbb{R})$. Case 1 is illustrated in Figure 13.5.

We can derive explicit formulas for the addition of points in $E(\mathbb{R})$.

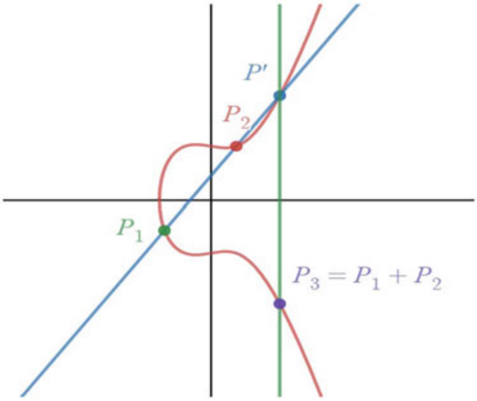


Fig. 13.5 Case 1: addition of points, $P_1 + P_2 = P_3$

Proposition 13.4.1 (Binary Operation on $E(\mathbb{R})$) *Let $E(\mathbb{R})$ be an elliptic curve and let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be points of $E(\mathbb{R})$. There exists a binary operation on $E(\mathbb{R})$ defined as follows:*

(i) If $P_1 \neq P_2$ and $x_1 \neq x_2$, then

$$P_1 + P_2 = P_3 = (x_3, y_3),$$

where

$$x_3 = m^2 - x_1 - x_2 \ \textit{and} \ y_3 = m(x_1 - x_3) - y_1.$$

with $m = \dfrac{y_2 - y_1}{x_2 - x_1}$.

(ii) If $P_1 \neq P_2$ and $x_1 = x_2$, $y_1 \neq y_2$, then

$$P_1 + P_2 = O.$$

(iii) If $P_1 = P_2$ and $y_1 = y_2 \neq 0$, then

$$P_1 + P_2 = P_1 + P_1 = 2P_1 = P_3 = (x_3, y_3),$$

where

$$x_3 = m^2 - 2x_1 \ \textit{and} \ y_3 = m(x_1 - x_3) - y_1$$

with $m = \dfrac{3x_1^2 + a}{2y_1}$.

(iv) If $P_1 = P_2$ and $y_1 = y_2 = 0$, then

$$P_1 + P_2 = P_1 + P_1 = 2P_1 = O.$$

(v) If $P_2 = O$, then

$$P_1 + O = P_1.$$

Proof

(i) $P_1 \neq P_2, x_1 \neq x_2$. In this case, the line $\overleftrightarrow{P_1P_2}$ has equation

$$y - y_1 = m(x - x_1), \quad m = \frac{y_2 - y_1}{x_2 - x_1}.$$

We seek the intersection of this line with $E(\mathbb{R})$. We find x so that

$$(m(x - x_1) + y_1)^2 = x^3 + ax + b. \tag{13.4}$$

By construction, x_1 and x_2 are solutions to (13.4). The third solution is obtained by rewriting Equation (13.4) as

$$x^3 - x^2(m^2) + x(2m^2 - 2my_1 + a) - m^2x_1^2 + 2mx_1y_1 - y_1^2 + b = 0.$$

Thus $x_3 = m^2 - x_1 - x_2$ is the x-coordinate of the third point P' of intersection of $\overleftrightarrow{P_1P_2}$ with $E(\mathbb{R})$. The y-coordinate of P' is therefore $y' = m(x_3 - x_1) + y_1$. The reflection of P' through the x-axis is the point $P_3 = (x_3, y_3)$, where $y_3 = m(x_1 - x_3) - y_1$. Thus, in Case 1,

$$P_1 + P_2 = P_3 = (x_3, y_3),$$

where

$$x_3 = m^2 - x_1 - x_2 \text{ and } y_3 = m(x_1 - x_3) - y_1$$

with $m = \dfrac{y_2 - y_1}{x_2 - x_1}$.

(ii) $P_1 \neq P_2, x_1 = x_2, y_1 \neq y_2$. In this case, the line $\overleftrightarrow{P_1P_2}$ is vertical and intersects $E(\mathbb{R})$ at the point at infinity O. The reflection of O through the x-axis is again O. So we define

$$P_1 + P_2 = O.$$

(iii) $P_1 = P_2, y_1 = y_2 \neq 0$. Since P_1 is a non-singular point of $y^2 = x^3 + ax + b$, the dimension of the tangent space Θ_{P_1} is 1. Hence there is a unique tangent line to the curve at the point P_1. By implicit differentiation, the slope of the tangent line is $m = \frac{3x_1^2+a}{2y_1}$, and thus the equation for the tangent line is

$$y - y_1 = m(x - x_1).$$

We seek the intersection of this line with $E(\mathbb{R})$. We find x so that

$$(m(x - x_1) + y_1)^2 = x^3 + ax + b. \tag{13.5}$$

Clearly, x_1 is a solution. Taking derivatives yields

$$2m\,(m(x - x_1) + y_1) = 3x^2 + a,$$

and so x_1 is a zero of (13.5) of multiplicity ≥ 2.

The coefficient of $-x^2$ in the expansion of (13.5) is m^2. And so, $x_3 = m^2 - 2x_1$ is the x-coordinate of the second point P' of intersection of the tangent line with $E(\mathbb{R})$. The y-coordinate of P' is therefore $y' = m(x_3 - x_1) + y_1$.

The reflection of P' through the x-axis is the point $P_3 = (x_3, y_3)$ where $y_3 = m(x_1 - x_3) - y_1$. Thus,

$$P_1 + P_2 = P_1 + P_1 = 2P_1 = P_3 = (x_3, y_3),$$

where

$$x_3 = m^2 - 2x_1 \text{ and } y_3 = m(x_1 - x_3) - y_1,$$

with $m = \dfrac{3x_1^2 + a}{2y_1}$.

(iv) $P_1 = P_2$, $y_1 = y_2 = 0$. As in (iii), there is a unique tangent line to the curve at the point P_1, and in this case the tangent line is vertical and intersects the curve at O; reflecting O through the x-axis yields O and we define

$$P_1 + P_2 = P_1 + P_1 = 2P_1 = O.$$

(v) $P_2 = O$. As in (ii), the line $\overleftrightarrow{P_1 O}$ is vertical and intersects $E(\mathbb{R})$ at a point P' which is the reflection of P_1 through the x-axis. Reflecting P' back through the x-axis yields P_1, so we define

$$P_1 + O = P_1.$$

□

Example 13.4.2 Let $y^2 = x^3 - x + 6$ be the elliptic curve defined over $\mathbb{R}$. Then $P_1 = (-2, 0)$ and $P_2 = (0, \sqrt{6})$ are points of $E(\mathbb{R})$. We have $m = \frac{\sqrt{6}}{2}$ and so

$$(-2, 0) + (0, \sqrt{6}) = \left(\frac{3}{2} + 2, \frac{\sqrt{6}}{2}\left(-2 - \frac{7}{2}\right)\right) = \left(\frac{7}{2}, \frac{-11\sqrt{6}}{4}\right).$$

Moreover,

$$2P_1 = 2(-2, 0) = O,$$

and

$$2P_2 = 2(0, \sqrt{6}) = \left(\frac{1}{24}, \frac{-287\sqrt{6}}{288}\right).$$

□

The good news is that the formulas of Proposition 13.4.1 extend to any field K to define a binary operation on an elliptic curve $E(K)$.

Example 13.4.3 Let $K = \mathbb{Q}$. Then $y^2 = x^3 - 4x = x(x + 2)(x - 2)$ defines an elliptic curve $E(\mathbb{Q})$. It is easy to see that $(0, 0)$, $(2, 0)$, $(-2, 0)$, and O are points of $E(\mathbb{Q})$. In fact, by Washington [63, Chapter 8, Example 8.5], these are the only points of $E(\mathbb{Q})$. We have

$$(0, 0) + (-2, 0) = (2, 0),$$

$$2(2, 0) = \mathcal{O},$$

and

$$(0, 0) + (2, 0) = (-2, 0).$$

□

Example 13.4.4 Let $K = \mathbb{F}_5$. Then $y^2 = x^3 + 4x + 1$ defines an elliptic curve over $\mathbb{F}_5$. As we have seen in Example 13.2.3,

$$E(\mathbb{F}_5) = \{(0, 1), (0, 4), (1, 1), (1, 4), (3, 0), (4, 1), (4, 4), \mathcal{O}\}.$$

We have

$$(3, 0) + (4, 1) = (4, 4),$$

and

$$2(1, 1) = (4, 1).$$

□

Proposition 13.4.5 (Elliptic Curve Group) *Let K be a field and let $E(K)$ be an elliptic curve over K. Then $E(K)$ is an abelian group under the binary operation defined by the formulas of Proposition 13.4.1.*

Proof To show that $E(K)$ is a group, we check that the conditions of Definition 5.1.1 are satisfied. To this end, we claim that the binary operation given in Proposition 13.4.1 is associative. This can be verified directly, but the calculation is lengthy. L. C. Washington proves the associative property using projective space, see [63, Chapter 2, Section 2.4].

For an identity element, we take $\mathcal{O}$. By Proposition 13.4.1(v),

$$\mathcal{O} + P = P = P + \mathcal{O}$$

for all $P \in E(K)$, and so $\mathcal{O}$ serves as a left and right identity element in $E(K)$.

Let $P = (x, y) \in E(K)$. Then the point $P' = (x, -y)$ is on the curve $E(K)$. By Proposition 13.4.1(ii),

$$P' + P = \mathcal{O} = P + P'$$

and so there exists a left and right inverse element P' for P, which we denote by $-P$.

Thus $E(K)$ is a group. It is straightforward to check that the binary operation is commutative, and thus $E(K)$ is an abelian group.

□

13.4.1 Structure of $E(K)$

The structure of the group $E(K)$ depends on the field K. We consider the case where $K = \mathbb{Q}$ or where K is a finite field.

Theorem 13.4.6 (Mordell–Weil Theorem) *Let $E(\mathbb{Q})$ be an elliptic curve group over $\mathbb{Q}$. Then $E(\mathbb{Q})$ is a finitely generated abelian group. Thus*

$$E(\mathbb{Q}) \cong \mathbb{Z}_{p_1^{e_1}} \times \mathbb{Z}_{p_2^{e_2}} \times \cdots \times \mathbb{Z}_{p_r^{er}} \times \mathbb{Z}^t,$$

where p^i are primes not necessarily distinct, $e_i \geq 1$, and $t \geq 0$.

Proof The proof is beyond the scope of this book. The interested reader is referred to [63, Chapter 8, Section 8.3, Theorem 8.17]. □

Example 13.4.7 Let $E(\mathbb{Q})$ be the elliptic curve group defined by $y^2 = x^3 - 4x = x(x+2)(x-2)$. As we have seen in Example 13.4.3,

$$E(\mathbb{Q}) = \{(0,0), (2,0), (-2,0), \mathcal{O}\},$$

and thus, $E(\mathbb{Q})$ is a finitely generated abelian group. In fact,

$$E(\mathbb{Q}) \cong \mathbb{Z}_2 \times \mathbb{Z}_2.$$

□

Example 13.4.8 Let $E(\mathbb{Q})$ be the elliptic curve group defined by $y^2 = x^3 - 25x = x(x+5)(x-5)$. By the Mordell–Weil theorem, $E(\mathbb{Q})$ is a finitely generated abelian group. In fact, as shown in [63, Chapter 8, Section 8.4],

$$E(\mathbb{Q}) \cong \mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}.$$

□

In cryptography, we are mainly interested in the case where K is a finite field $\mathbb{F}_q$ with $q = p^n$ elements for p prime and $n \geq 1$. If $K = \mathbb{F}_q$, then $E(\mathbb{F}_q)$ is a finite abelian group. We review (without proofs) two fundamental results on the structure of $E(\mathbb{F}_q)$.

Theorem 13.4.9 *Let $E(\mathbb{F}_q)$ be an elliptic curve group over $\mathbb{F}_q$. Then*

$$E(\mathbb{F}_q) \cong \mathbb{Z}_n$$

for some integer $n \geq 1$, or

$$E(\mathbb{F}_q) \cong \mathbb{Z}_m \times \mathbb{Z}_n,$$

for some integers $m, n \geq 1$ with $m \mid n$.

Proof For a proof, see [63, Chapter 4, Section 4.1, Theorem 4.1]. □

A second fundamental result due to Hasse gives an upper and lower bound on the number of elements in $E(\mathbb{F}_q)$.

Theorem 13.4.10 (Hasse) *Let $E(\mathbb{F}_q)$ be an elliptic curve group over $\mathbb{F}_q$. Then*

$$q + 1 - 2\sqrt{q} \leq |E(\mathbb{F}_q)| \leq q + 1 + 2\sqrt{q}.$$

Example 13.4.11 Let $K = \mathbb{F}_5$. Then $y^2 = x^3 + 4x + 1$ defines an elliptic curve over $\mathbb{F}_5$,

$$E(\mathbb{F}_5) = \{(0, 1), (0, 4), (1, 1), (1, 4), (3, 0), (4, 1), (4, 4), O\}.$$

According to Hasse's theorem,

$$6 - 2\sqrt{5} \leq |E(\mathbb{F}_5)| \leq 6 + 2\sqrt{5},$$

which holds since $|E(\mathbb{F}_5)| = 8$.

By Theorem 13.4.9, either $E(\mathbb{F}_5) \cong \mathbb{Z}_8$ or $E(\mathbb{F}_5) \cong \mathbb{Z}_2 \times \mathbb{Z}_4$. Since $(0, 1)$ has order 8, we conclude that

$$E(\mathbb{F}_5) \cong \langle (0, 1) \rangle \cong \mathbb{Z}_8.$$

□

Example 13.4.12 Let $K = \mathbb{F}_7$. Then $y^2 = x^3 + 2$ defines an elliptic curve over $\mathbb{F}_7$ with group

$$E(\mathbb{F}_7) = \{(0, 3), (0, 4), (3, 1), (3, 6), (5, 1), (5, 6), (6, 1), (6, 6), O\}.$$

According to Hasse's theorem,

$$8 - 2\sqrt{7} \leq |E(\mathbb{F}_7)| \leq 8 + 2\sqrt{7},$$

which holds since $|E(\mathbb{F}_7)| = 9$.

By Theorem 13.4.9, either $E(\mathbb{Z}_7) \cong \mathbb{Z}_9$ or $E(\mathbb{F}_7) \cong Z_3 \times \mathbb{Z}_3$. But since $3P = O$ for all $P \in E(\mathbb{F}_7)$, we have

$$E(\mathbb{F}_7) \cong \mathbb{Z}_3 \times \mathbb{Z}_3.$$

□

13.5 The Elliptic Curve Key Exchange Protocol

The Elliptic Curve Key Exchange Protocol (ECKEP) is essentially the DHKEP with an elliptic curve group in place of the group $U(\mathbb{Z}_p)$.

Protocol 13.5.1 (Elliptic Curve Key Exchange Protocol)

Premise: Alice and Bob choose an elliptic curve group $E(\mathbb{F}_p)$ over a finite field $\mathbb{F}_p$, where p is a large prime. Alice and Bob share a point $P \in E(\mathbb{F}_p)$ with a large order r.
Goal: Alice and Bob share a secret random point of $E(\mathbb{F}_p)$.

1. Alice randomly chooses an integer m, $0 \le m \le r-1$. She computes
$$Q = mP$$
and sends Q to Bob.
2. Bob randomly chooses an integer n, $0 \le n \le r-1$. He computes
$$R = nP$$
and sends R to Alice.
3. Bob computes
$$S = nQ = n(mP) = nmP.$$
4. Alice computes
$$T = mR = m(nP) = mnP = nmP.$$
5. Alice and Bob share the secret random point $S = mnP$.

□

Example 13.5.2 Alice and Bob choose the elliptic curve group $E(\mathbb{F}_5)$ defined by $y^2 = x^3 + x + 1$. In fact, $E(\mathbb{F}_5) \cong \mathbb{Z}_9$ and is generated by the point $P = (0, 1)$ (Section 13.6, Exercise 9).

Alice and Bob decide to share the point $P = (0, 1)$, which has order $r = 9$.

Alice randomly chooses integer $m = 3$ and computes
$$Q = 3P = 3(0, 1) = (2, 1)$$
and sends Q to Bob.

Bob randomly chooses integer $n = 5$, computes
$$R = 5P = 5(0, 1) = (3, 1),$$

and sends R to Alice.

Bob computes

$$5Q = 5(2, 1) = 15(0, 1) = 6(0, 1) = (2, 4),$$

and Alice computes

$$3R = 3(3, 1) = 15(0, 1) = 6(0, 1) = (2, 4).$$

□

We show that point multiplication nP for $n \geq 1$ can be computed efficiently. This will show that the ECKEP is a practical method of key exchange.

Algorithm 13.5.3 (POINT_MULT)

```
Input:      an elliptic curve group $E(\mathbb{F}_p)$ and a point $P \in E(\mathbb{F}_p)$,
            $n$ encoded in binary as $b_m b_{m-1} \cdots b_1$
Output:     $Q = nP$
Algorithm:
            $Q \leftarrow O$
            for $i = 1$ to $m$
               if $b_i = 1$ then $Q \leftarrow Q + P$
               $P \leftarrow 2P$
            next $i$
            output $Q$
```

□

To see how POINT_MULT works, we compute

$$6P = 6(0, 1)$$

in the elliptic curve group $E(\mathbb{F}_5)$ of Example 13.4.11. In this case, $P = (0, 1)$, $n = 6$, which in binary is 110, and so $m = 3$.

On the first iteration of the loop, Q remains O, and $P = (0, 1)$ doubles to become

$$P = 2(0, 1) = (4, 1).$$

On the second iteration, Q becomes

$$Q + P = O + P = (4, 1),$$

and P doubles again as

$$P = 2(4, 1) = (3, 0).$$

On the third (last) iteration, Q becomes

$$Q + P = (4, 1) + (3, 0) = (4, 4),$$

which is then output as the correct result

$$Q = 6P = (4, 4).$$

Proposition 13.5.4 POINT_MULT *runs in time* $O(m^4)$, *where* $m = \lfloor \log_2(p) \rfloor + 1$.

Proof The algorithm performs m iterations of the for-next loop. On each iteration, the algorithm performs at most 2 steps; each step is an addition of two points of $E(\mathbb{F}_p)$. An examination of the formulas for point addition (Proposition 13.4.1) reveals that addition of points in $E(\mathbb{F}_p)$ requires at most 20 additions, subtractions, multiplications, and divisions in $\mathbb{F}_p$. Thus each iteration costs $O(m^3)$ steps. Consequently, the running time of POINT_MULT is $O(m^4)$.

□

Remark 13.5.5 Similar to Algorithms 6.4.1, Algorithm 13.5.3 is analogous to Algorithm 4.2.6 with $2P$ playing the role of $2a$.

□

There is an analog for the DHDLP:

Definition 13.5.6 The **Elliptic Curve Discrete Logarithm Problem** (ECDLP) is stated as follows: given $P \in E(\mathbb{F}_p)$, $Q \in H = \langle P \rangle \leq E(\mathbb{F}_p)$, $n = |H|$, find the unique value of m, $0 \leq m \leq n - 1$ for which

$$Q = mP.$$

Like the DHKEP, the security of the ECKEP depends on the fact that the ECDLP is difficult to solve.

13.5.1 Comparing ECKEP and DHKEP

So far we have two protocols for the distribution of keys: the Diffie–Hellman key exchange which uses the group $U(\mathbb{Z}_p) = \mathbb{F}_p^\times$ and Elliptic Curve key exchange which employs the group $E(\mathbb{F}_p)$.

The elliptic curve group $E(\mathbb{F}_p)$ does seem more complicated than the group $\mathbb{F}_p^\times$. The elements of $E(\mathbb{F}_p)$ require two elements of $\mathbb{F}_p$ for their description, and the group operation in $E(\mathbb{F}_p)$ involves more steps than the simple modulo p multiplication in $\mathbb{F}_p^\times$. What is the major advantage in choosing the ECDEK over the DHDEK?

To attack the ECKEP by solving the ECDLP, one is limited to the algorithms that solve the general DLP, namely, NAIVE_DLP and BSGS, which run in time $O(\sqrt{p}\log_2(p))$ at best.

The Index Calculus method, with its faster subexponential running time, is effective in attacking the DHKEP but can only be applied to the DHDLP; currently there is no analog of the Index Calculus method for the ECDLP.

So, the fastest known algorithm for solving the ECDLP runs in time $O(\sqrt{p}\log_2(p))$; faster methods cannot be applied to the ECDLP.

For more discussion of this matter, see [33, Chapter 6, Section 4.5] and [7, Chapter V, Section V.6].

Despite the advantage of ECKEP over DHKEP, care must be taken when selecting an elliptic curve group.

13.5.2 *What Elliptic Curves to Avoid*

Let $E(\mathbb{F}_q)$ be an elliptic curve over $\mathbb{F}_q$, $q = p^m$, p prime, $m \geq 1$. By Hasse's theorem (Theorem 13.4.10),

$$q + 1 - 2\sqrt{q} \leq |E(\mathbb{F}_q)| \leq q + 1 + 2\sqrt{q},$$

thus

$$|E(\mathbb{F}_q)| \leq q + 1.$$

The integer t for which

$$|E(\mathbb{F}_q)| + t = q + 1$$

is the **trace** of $E(\mathbb{F}_q)$ at q.

Let P be a point of $E(\mathbb{F}_q)$ of order $n \geq 1$. Assume that $\gcd(n, q) = 1$ and let $\langle P \rangle$ denote the cyclic subgroup of $E(\mathbb{F}_q)$ generated by P; we have $n = |\langle P \rangle|$. Let $Q \in \langle P \rangle$. We want to solve the ECDLP

$$xP = Q. \tag{13.6}$$

The ECDLP (13.6) can be solved using MOV attack, which is named for its authors Menezes et al. [38]. We give a brief outline of the MOV attack below. For complete details, see [63, Chapter 5, Section 5.3] and [7, Chapter V, Section V.2].

The MOV Attack

Let $\overline{\mathbb{F}}_q$ denote an algebraic closure of $\mathbb{F}_q$. Let $n \geq 1$, and let

$$E(n) = \{P \in E(\overline{\mathbb{F}}_q) : \text{ the order of } P \text{ in } E(\overline{\mathbb{F}}_q) \text{ divides } n\}.$$

Using the formula

$$\overline{\mathbb{F}}_q = \bigcup_{i=1}^{\infty} \mathbb{F}_{q^i},$$

one sees that there exists a smallest positive integer m for which

$$E(n) \subseteq E(\mathbb{F}_{q^m}).$$

Thus, by Washington [63, Corollary 3.11], $\mathbb{F}_{q^m}$ contains the group μ_n of the nth roots of unity, i.e., $\mathbb{F}_{q^m}$ contains all of the roots of the equation $x^n - 1$. Since μ_n is a subgroup of $\mathbb{F}_{q^m}^{\times}$,

$$q^m \equiv 1 \pmod{n},$$

and thus m is a multiple of the order of $(q \bmod n)$ in $U(\mathbb{Z}_n)$.

Once the value of m has been determined, the MOV attack solves the ECDLP by solving the DLP in $\mathbb{F}_{q^m}^{\times}$. As we have indicated, there is an Index Calculus algorithm for solving the DLP in $\mathbb{F}_{p^m}^{\times}$ [20].

If m is relatively small, then solving the DLP in $\mathbb{F}_{p^m}^{\times}$ is a much easier problem than solving (13.6).

One case where m is small occurs when the elliptic curve is supersingular.

Supersingular Curves

An elliptic curve $E(\mathbb{F}_q)$ is **supersingular** if $p \mid t$, where t is the trace of the curve. If $q = p$ and $p \geq 5$, then $E(\mathbb{F}_p)$ is supersingular if and only if $t = 0$ (use Hasse's theorem).

For example, the elliptic curve $E(\mathbb{F}_{151})$ defined by

$$y^2 = x^3 + 2x$$

is supersingular, $|E(\mathbb{F}_{151})| = 152$ and $t = 0$.

Let $E(\mathbb{F}_p)$, $p \geq 5$, be a supersingular elliptic curve, and suppose that P is a point of $E(\mathbb{F}_p)$ of order n. Then n divides $p + 1 = |E(\mathbb{F}_p)|$ and so, $\gcd(n, p) = 1$.

Now, by Washington [63, Proposition 5.3], $E(n) \subseteq E(\mathbb{F}_{p^2})$ and so $m = 1$ or 2.

So, under the MOV attack, solving the ECDLP in $\langle P \rangle$ is no harder than solving the DLP in $\mathbb{F}_{p^2}^{\times}$.

Thus for use in ECKEP, an elliptic curve $E(\mathbb{F}_p)$, $p \geq 5$, should not be supersingular, else we lose the main advantage of ECKEP over DHKEP: there are no Index Calculus methods for ECDLP.

Anomalous Curves

To assure that the MOV attack cannot be employed, one strategy is to choose curves that are **anomalous**, i.e., satisfy

$$|E(\mathbb{F}_q)| = q,$$

where $q = p^m$ (that is, the trace $t = 1$).

If $E(\mathbb{F}_q)$ is anomalous and $P \in E(\mathbb{F}_q)$ is a point of order n, then $n \mid q$. Hence there is no $m \geq 1$ for which

$$q^m \equiv 1 \bmod n.$$

Consequently, there is no embedding $E(n) \subseteq E(\mathbb{F}_{q^m})$, and thus the MOV attack cannot be used to solve the ECDLP.

However, in the case that $E(\mathbb{F}_p)$ is an anomalous curve for p prime, an algorithm has been found that solves the ECDLP in polynomial time. The algorithm uses the field of p-adic rationals, $\mathbb{Q}_p$; see [63, Chapter 5, Section 5.4] and [7, Chapter V, Section V.3]. Thus anomalous curves must also be avoided when choosing an elliptic curve for ECKEP.

Example 13.5.7 Let $E(\mathbb{F}_{43})$ be the elliptic curve defined by

$$y^2 = x^3 + 10x + 36$$

over $\mathbb{F}_{43}$. Then $E(\mathbb{F}_{43})$ has exactly 43 points, and hence $E(\mathbb{F}_{43})$ is anomalous.

□

Here is a GAP program that will compute the 42 non-trivial points of $E(\mathbb{F}_{43})$.

```
q:=List([0..42],i->(i^3+10*i+36) mod 43);
for j in [1..43] do
  if Legendre(q[j],43)=1 then
  Print("(",j-1,",",RootMod(q[j],43),")",",");
  Print("(",j-1,",",-1*RootMod(q[j],43) mod 43,")",",");
  fi;
od;
```

13.5.3 Examples of Good Curves

As we have seen, in selecting a curve to use in the ECKEP, we need to avoid supersingular and anomalous curves.

In this section we give examples of elliptic curves over $\mathbb{F}_p$ that are appropriate for use in the ECKEP. These curves are given in the form

$$y^2 = x^3 + ax + b$$

and can be found in [7, Appendix A]. Our computations are done using `GAP`.

Example 13.5.8 We take

$$p = 2^{130} + 169 = 1361129467683753853853498429727072845993,$$

which is a 131-bit prime. We let

$$\begin{aligned} a &= 3, \\ b &= 1043498151013573141076033119958062900890, \end{aligned}$$

which defines the elliptic curve

$$y^2 = x^3 + 3x + 1043498151013573141076033119958062900890,$$

with group $E(\mathbb{F}_p)$.

Now,

$$|E(\mathbb{F}_p)| = 1361129467683753853808807784495688874237,$$

which is a 130-bit prime.

The trace of $E(\mathbb{F}_p)$ is

$$t = p + 1 - |E(\mathbb{F}_p)| = 44690645231383971757,$$

and thus $E(\mathbb{F}_p)$ is not supersingular.

If P is any non-trivial point in $E(\mathbb{F}_p)$, then it has order

$$n = |E(\mathbb{F}_p)| = 1361129467683753853808807784495688874237,$$

thus

$$\langle P \rangle = E(\mathbb{F}_p).$$

Now, $\gcd(n, p) = 1$ and the order of $(p \bmod n)$ in $U(\mathbb{Z}_n)$ is

$$l = 680564733841876926904403892247844437118.$$

Thus, when applying the MOV attack, the smallest integer m for which

$$E(n) \subseteq E(\mathbb{F}_{p^m})$$

is at least

$$680564733841876926904403892247844437118.$$

Thus, it is infeasible to use the MOV attack to solve the ECDLP in $E(\mathbb{F}_p)$

The curve $E(\mathbb{F}_p)$ is not anomalous (since $t \neq 1$). So the approach in the anomalous case will not work, either.

The best we can do to solve this ECDLP is to use the Baby-Step/Giant-Step (BSGS) algorithm, which will solve the DLP in time $O(\sqrt{n}\log_2(n))$.

To use BSGS to solve the DLP, our computer would have to perform $\approx 2^{65} \cdot 130 = 2^{72}$ basic operations, which is beyond the number of operations that a computer can do in a reasonable amount of time.

We can use GAP to compute an explicit example of a hard DLP in $E(\mathbb{F}_p)$. For instance, one can use GAP to show that the point

$$P = (0, 1314511337629110386987830837960619486151)$$

is in $E(\mathbb{F}_p)$ and thus generates $E(\mathbb{F}_p)$. Moreover,

$$Q = (2, 466980356246987135141837142580539177759) \in E(\mathbb{F}_p).$$

Thus the ECDLP

$$xP = Q$$

is hard.

The relevant GAP code is

```
  p:=2^130+169; #a prime
  a:=3;
  b:=1043498151013573141076033119958062900890; #curve parameters
  n:=1361129467683753853808807784495688874237; #the order of the
elliptic curve group (a prime)
  t:=p+1-n; #the trace
  l:=OrderMod(p,n); #the order of p modulo n
  x1:=0;
  s1:=(x1^3+a*x1+b) mod p;
  y1:=RootMod(s1,p); #P=(x1,y1) generates the elliptic curve
group
  x2:=2;
  s2:=(x2^3+a*x2+b) mod p;
  y2:=RootMod(s2,p); #Q=(x2,y2) is the second point
```

Example 13.5.9 We take

$$p = 2^{160} + 7 = 1461501637330902918203684832716283019655932542983,$$

which is a 161-bit prime.

We let

$$a = 10,$$
$$b = 1343632762150092499701637438970764818528075565078,$$

which defines the elliptic curve

$$y^2 = x^3 + 10x + 1343632762150092499701637438970764818528075565078,$$

with group $E(\mathbb{F}_p)$.

Now,

$$|E(\mathbb{F}_p)| = 1461501637330902918203683518218126812711137002561,$$

which is a 160-bit prime.

The trace of $E(\mathbb{F}_p)$ is

$$t = p + 1 - |E(\mathbb{F}_p)| = 1314498156206944795540423,$$

and thus $E(\mathbb{F}_p)$ is not supersingular.

If P is any non-trivial point in $E(\mathbb{F}_p)$, then it has order

$$n = |E(\mathbb{F}_p)| = 1461501637330902918203683518218126812711137002561,$$

thus

$$\langle P \rangle = E(\mathbb{F}_p).$$

Now, $\gcd(n, p) = 1$ and the order of $(p \bmod n)$ in $U(\mathbb{Z}_n)$ is

$$l = 730750818665451459101841759109063406355568501280.$$

Thus, when applying the MOV attack, the smallest integer m for which

$$E(n) \subseteq E(\mathbb{F}_{p^m})$$

is at least

$$730750818665451459101841759109063406355568501280,$$

and so, it is infeasible to use the MOV attack to solve the ECDLP in $E(\mathbb{F}_p)$.

The curve $E(\mathbb{F}_p)$ is not anomalous (since $t \neq 1$). So the approach in the anomalous case will not work, either.

The best we can do to solve this ECDLP is to use the BSGS algorithm, which will solve the DLP in time $O(\sqrt{n}\log_2(n))$.

Fortunately (from a security standpoint), to use BSGS to solve the DLP, our computer would have to perform $\approx 2^{80} \cdot 160 \approx 2^{72}$ basic operations, which is beyond the number of operations that a computer can do in a reasonable amount of time.

We conclude that $E(\mathbb{F}_p)$ is a good curve for an ECKEP application.

13.6 Exercises

1. Show that let $y^2 = x^3 - 3x - 1$ define an elliptic curve over $\mathbb{Q}$. Does this equation define an elliptic curve over $\mathbb{F}_3$?
2. Determine whether $y^2 = x^2 - \frac{1}{3}x + \frac{2}{27}$ defines an elliptic curve over $\mathbb{Q}$.
3. Let $E(\mathbb{Q})$ be the elliptic curve group given by the equation $y^2 = x^3 + 17$.
   (a) Show that $P = (-1, 4)$ and $Q = (4, 9)$ are elements of $E(\mathbb{Q})$.
   (b) Compute $P + Q$ and $2P$ in $E(\mathbb{Q})$.
4. Let $E(\mathbb{Q})$ be the elliptic curve group defined by the equation $y^2 = x^3 - 43x + 166$. Show that $P = (3, 8)$ generates a subgroup of order 7 in $E(\mathbb{Q})$.
5. Let $E(\mathbb{F}_5)$ be the elliptic curve group defined by the equation $y^2 = x^3 + 2$.
   (a) Show that $E(\mathbb{F}_5)$ is an abelian group with $2 \leq |E(\mathbb{F}_5)| \leq 11$.
   (b) Construct all of the elements of $E(\mathbb{F}_5)$.
   (c) Find a group isomorphic to $E(\mathbb{F}_5)$ of the form $\mathbb{Z}_m \times \mathbb{Z}_n$, $m \mid n$.
6. Find the smallest prime p so that the elliptic curve group $E(\mathbb{F}_p)$ has at least 100 elements. Note: once such a prime p is found, you should show that there exists an elliptic curve over $\mathbb{F}_p$.
7. Alice and Bob are using the ECKEP with the group $E(\mathbb{F}_5)$ of Example 13.4.11. Alice and Bob share the point $P = (0, 1)$ of order 8. Alice randomly chooses $x = 3$ and Bob randomly chooses $y = 7$. Compute the point shared by Alice and Bob.
8. Let $E(\mathbb{F}_5)$ be the group of Example 13.4.11. Let $P = (0, 1)$, $Q = (3, 0)$. Solve the DLP $mP = Q$.
9. Let $E(\mathbb{F}_5)$ be the elliptic curve group of Example 13.5.2.
   (a) Compute all of the elements of $E(\mathbb{F}_5)$.
   (b) Prove that $E(\mathbb{F}_5) \cong \mathbb{Z}_9$ and that $E(\mathbb{F}_5)$ is generated by $(0, 1)$.
10. Let $y^2 = x^3 + 547x + 21$ be the elliptic curve over $\mathbb{F}_{557}$. Write a `GAP` program to compute all of the non-trivial points of $E(\mathbb{F}_{557})$.
11. Consider the ElGamal public key cryptosystem of Section 9.4. Devise an elliptic curve ElGamal public key cryptosystem with a cyclic subgroup of $E(\mathbb{F}_p)$ in place of $U(\mathbb{Z}_l)$, l prime.

    As a first step, choose an elliptic curve group $E(\mathbb{F}_p)$ and a point $P \in E(\mathbb{F}_p)$ of order r. The subgroup $\langle P \rangle$ plays the role of $U(\mathbb{F}_l)$; P plays the role of a primitive root modulo l.

Chapter 14
Singular Curves

Let K be a field not of characteristic 2. Let $K^\times$ denote the multiplicative group of non-zero elements of K. Let

$$y^2 = x^3 + ax + b$$

be a singular curve and assume that the cubic $x^3 + ax + b$ has a double root. Then the curve does not define an elliptic curve group. The set of non-singular points, $E_{ns}(K)$, however, is still a group under the point addition of Proposition 13.4.1.

In this chapter, we study the structure of the group $E_{ns}(K)$. The group $E_{ns}(K)$ is certainly of interest mathematically and may yet yield applications to cryptography, for instance, see [63, Section 2.9].

14.1 The Group $E_{ns}(K)$

To define a group structure on $E_{ns}(K)$, we begin by rewriting the singular curve that defines $E_{ns}(K)$.

Proposition 14.1.1 *Let K be a field of characteristic not 2 or 3. Let $y^2 = x^3 + ax + b$ be a singular curve in which $x^3 + ax + b$ has a double root in $\overline{K}$. Then the curve can be written as*

$$y^2 = x^2(x + c)$$

for some $c \in K^\times$.

Proof Since $y^2 = x^3 + ax + b$ is singular, it does not define an elliptic curve over K. Since $x^3 + ax + b$ has a double root in $\overline{K}$, the cubic is reducible over K. Thus $x^3 + ax + b = (x - r)q(x)$, $r \in K$, where $q(x)$ is a quadratic over K. Either r is the

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7_14

double root, so that $x^3 + ax + b = (x - r)^2(x - s)$ for some $s \in K$, $s \neq r$, or the quadratic factor has the double root as its roots. But then $q(x)$ must be reducible, hence $q(x) = (x - s)^2$, $s \in K$, $s \neq r$. In any case, if $x^3 + ax + b$ has a double root, then

$$x^3 + ax + b = (x - r)^2(x - s)$$

for $r, s \in K$, $r \neq s$.

Let $x' = x - r$. Then we obtain the curve

$$y^2 = (x')^2(x' + r - s),$$

or

$$y^2 = x^2(x + c),$$

where $c = r - s \neq 0$.

□

The only singular point of the curve $y^2 = x^2(x + c)$ is (0, 0).

Let $E_{ns}(K)$ be the collection of non-singular points of the curve $y^2 = x^2(x+c)$, $c \neq 0$.

Proposition 14.1.2 (Binary Operation on $E_{ns}(K)$) *Let K be a field not of characteristic 2 or 3, and let $E_{ns}(K)$ denote the set of non-singular points of the curve $y^2 = x^2(x+c)$, $c \neq 0$. Let $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$ be points of $E_{ns}(K)$. There exists a binary operation on $E_{ns}(K)$ defined as follows. (Note: these formulas are not the same as those given in Proposition 13.4.1.)*

(i) If $P_1 \neq P_2$, $x_1 \neq x_2$, then

$$P_1 + P_2 = P_3 = (x_3, y_3),$$

where

$$x_3 = m^2 - c - x_1 - x_2 \ \ \textit{and} \ \ y_3 = m(x_1 - x_3) - y_1.$$

with $m = \dfrac{y_2 - y_1}{x_2 - x_1}$.

(ii) If $P_1 \neq P_2$, $x_1 = x_2$, $y_1 \neq y_2$, then

$$P_1 + P_2 = O.$$

(iii) If $P_1 = P_2$, $y_1 = y_2 \neq 0$, then

$$P_1 + P_2 = P_1 + P_1 = 2P_1 = P_3 = (x_3, y_3),$$

where

$$x_3 = m^2 - c - 2x_1 \ \textit{and} \ y_3 = m(x_1 - x_3) - y_1.$$

with $m = \dfrac{3x_1^2 + 2cx_1}{2y_1}$.

(iv) If $P_1 = P_2$, $y_1 = y_2 = 0$, *then*

$$P_1 + P_2 = P_1 + P_1 = 2P_1 = O.$$

(v) If $P_2 = O$, *then*

$$P_1 + O = P_1.$$

Proof See Section 14.5, Exercise 1. □

Now, under the binary operation of Proposition 14.1.2, $E_{ns}(K)$ is an additive abelian group.

Let $E_{ns}(K)$ be the group of non-singular points on the curve $y^2 = x^2(x + c)$, $c \neq 0$. Let $\beta^2 = c$ for some $\beta \in \overline{K}$. Let $K(\beta)$ be the simple algebraic extension field of K.

Proposition 14.1.3 *If* $\beta \notin K$, *then the subset*

$$J_c = \{u + \beta v : \ u, v \in K, u^2 - cv^2 = 1\} \subseteq K(\beta)$$

is a subgroup of $K(\beta)^\times$ *under the multiplication of* $K(\beta)$.

Proof See Section 14.5, Exercise 4.

□

We have the following characterization of $E_{ns}(K)$ due to L. C. Washington [63, Theorem 2.30].

Theorem 14.1.4 *Let* K *be a field not of characteristic* 2 *or* 3, *and let* $E_{ns}(K)$ *be the group of non-singular points of* $y^2 = x^2(x + c)$, $c \neq 0$. *Let* $\beta^2 = c$ *for* $\beta \in \overline{K}$.

(i) If $\beta \in K$, *then* $E_{ns}(K) \cong K^\times$.
(ii) If $\beta \notin K$, *then* $E_{ns}(K) \cong J_c$.

Proof

For (i): there is a group isomorphism

$$\psi : E_{ns}(K) \to K^\times$$

defined by $\psi(O) = 1$, and

$$\psi(x, y) = \frac{y + \beta x}{y - \beta x}.$$

The inverse of ψ is given as

$$\psi^{-1} : K^{\times} \to E_{ns}(K),$$

with $\psi^{-1}(1) = O$, and $\psi^{-1}(r) = (x, y)$, where

$$x = \frac{4cr}{(r-1)^2}, \quad y = \frac{4\beta cr(r+1)}{(r-1)^3},$$

for $r \neq 1$. For details, see [63, Theorem 2.30(i)].

For (ii): in this case, there is a group isomorphism

$$\psi : E_{ns}(K) \to J_c \leq K(\beta)^{\times}$$

defined by $\psi(O) = 1$, and $\psi(x, y) = u + \beta v$, where

$$u = \frac{y^2 + cx^2}{y^2 - cx^2}, \quad y = \frac{2xy}{y^2 - cx^2}.$$

The inverse of ψ is given as

$$\psi^{-1} : J_c \to E_{ns}(K),$$

with $\psi^{-1}(1) = O$, $\psi^{-1}(-1) = (-c, 0)$, and $\psi^{-1}(u + \beta v) = (x, y)$, where

$$x = \left(\frac{u+1}{v}\right)^2 - c, \quad y = \left(\frac{u+1}{v}\right)x,$$

for $u + \beta v \neq \pm 1$. For details, see [63, Theorem 2.30(ii)].

□

Example 14.1.5 Let $K = \mathbb{F}_7$, and let $E_{ns}(\mathbb{F}_7)$ be the group defined by $y^2 = x^2(x+2)$,

$$E_{ns}(\mathbb{F}_7) = \{O, (2, 4), (2, 3), (5, 0), (6, 1), (6, 6)\}.$$

We have $3^2 \equiv 2 \pmod 7$, thus $\beta = 3 \in \mathbb{F}_7$. Thus Theorem 14.1.4(i) applies to give an isomorphism

$$\psi : E_{ns}(\mathbb{F}_7) \to \mathbb{F}_7^{\times}$$

defined by $\psi(O) = 1$, and

$$\psi(x, y) = \frac{y + 3x}{y - 3x}.$$

We have

$$\psi(O)=1, \quad \psi(2,4)=2, \quad \psi(2,3)=4,$$

$$\psi(5,0)=6, \quad \psi(6,1)=3, \quad \psi(6,6)=5.$$

14.2 The DLP in $E_{ns}(K)$

Theorem 14.1.4(i) shows that there is no great advantage in using $E_{ns}(K)$ over $K^\times$ in a cryptographic application: if $\beta \in K$, then solving the DLP in $E_{ns}(K)$ is no harder than solving the DLP in $K^\times$.

Let $E_{ns}(K)$ be defined by $y^2 = x^2(x+c)$, where $c \neq 0$. Suppose that $\beta^2 = c$ for some $\beta \in K$. Let $P \in E_{ns}(K)$, and let $\langle P \rangle$ be the cyclic subgroup of $E_{ns}(K)$ generated by P. Let $Q \in \langle P \rangle$. We seek to solve the DLP

$$mP = Q. \tag{14.1}$$

But this is easy if we apply the isomorphism ψ of Theorem 14.1.4(i) to both sides of (14.1) to obtain

$$\psi(P)^m = \psi(Q),$$

which is a DLP in $K^\times$ and is usually quite simple to solve.

Example 14.2.1 Let $K = \mathbb{R}$, and let $E_{ns}(\mathbb{R})$ be defined by $y^2 = x^2(x+4)$. Then $c = 4$ and $\beta = 2$. By Theorem 14.1.4(i), there is a group isomorphism

$$\psi : E_{ns}(\mathbb{R}) \to \mathbb{R}^\times,$$

given as

$$(x, y) \mapsto \frac{y+2x}{y-2x}.$$

It is known that $P = (32, 192) \in E_{ns}(\mathbb{R})$ and that $Q = (\frac{512}{961}, \frac{33792}{29791}) \in \langle P \rangle$. We want to solve the DLP

$$m(32, 192) = \left(\frac{512}{961}, \frac{33792}{29791}\right). \tag{14.2}$$

Now,

$$\psi(32, 192) = \frac{(192 + (2 \cdot 32))}{(192 - (2 \cdot 32))} = 2,$$

and

$$\psi\left(\frac{512}{961}, \frac{33792}{29791}\right) = \frac{\left(\frac{33792}{29791} + \left(2 \cdot \frac{512}{961}\right)\right)}{\left(\frac{33792}{29791} - \left(2 \cdot \frac{512}{961}\right)\right)} = 32.$$

Thus the DLP (14.2) becomes the DLP

$$2^m = 32,$$

which is easily solved by $m = 5$. Consequently,

$$5P = Q.$$

□

Here is an example involving finite fields.

Example 14.2.2 Let $K = \mathbb{F}_7$, and let $E_{ns}(\mathbb{F}_7)$ be defined by $y^2 = x^2(x + 2)$. As in Example 14.1.5,

$$E_{ns}(\mathbb{F}_7) = \{O, (2, 4), (2, 3), (5, 0), (6, 1), (6, 6)\},$$

$\beta = 3 \in \mathbb{F}_7$. Thus Theorem 14.1.4(i) applies to give an isomorphism

$$\psi : E_{ns}(K) \to \mathbb{F}_7^\times$$

defined by $\psi(O) = 1$, and

$$\psi(x, y) = \frac{y + 3x}{y - 3x}.$$

We want to solve the DLP

$$m(6, 1) = (5, 0). \tag{14.3}$$

Now,

$$\psi(6, 1) = 3, \quad \psi(5, 0) = 6.$$

Thus the DLP (14.3) becomes the DLP

$$3^m = 6,$$

which is solved by $m = 3$. Consequently,

$$3(6, 1) = (5, 0).$$

□

14.3 The Group $G_c(K)$

Let K be a field of characteristic not 2, and let $c \in K^\times$. In this section we show that the points on the curve

$$x^2 - cy^2 = 1$$

form an additive abelian group.

Theorem 14.3.1 *Let K be a field of characteristic not* 2*, and let $c \in K^\times$. Let*

$$G_c(K) = \{(x, y) \in K \times K : \ x^2 - cy^2 = 1\}.$$

Then $G_c(K)$ is an abelian group under the binary operation

$$+ : G_c(K) \times G_c(K) \to G_c(K)$$

defined as follows. For $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2) \in G_c(K)$,

$$P_1 + P_2 = P_3 = (x_3, y_3),$$

where $x_3 = x_1x_2 + cy_1y_2$ and $y_3 = x_1y_2 + y_1x_2$.

Proof See Section 14.5, Exercise 6.

□

Example 14.3.2 Let $K = \mathbb{R}$, $c = -1$. Then

$$G_{-1}(\mathbb{R}) = \{(x, y) \in \mathbb{R} \times \mathbb{R} : \ x^2 + y^2 = 1\}$$

with binary operation

$$P_1 + P_2 = P_3 = (x_3, y_3),$$

where $x_3 = x_1x_2 - y_1y_2$ and $y_3 = x_1y_2 + y_1x_2$. For instance, let $P_1 = (\frac{4}{5}, \frac{3}{5})$, $P_2 = (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})$. Then

$$x_3 = \frac{4}{5} \cdot \frac{\sqrt{2}}{2} - \frac{3}{5} \cdot \frac{\sqrt{2}}{2} = \frac{\sqrt{2}}{10},$$

$$y_3 = \frac{4}{5} \cdot \frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2} \cdot \frac{3}{5} = \frac{7\sqrt{2}}{10}.$$

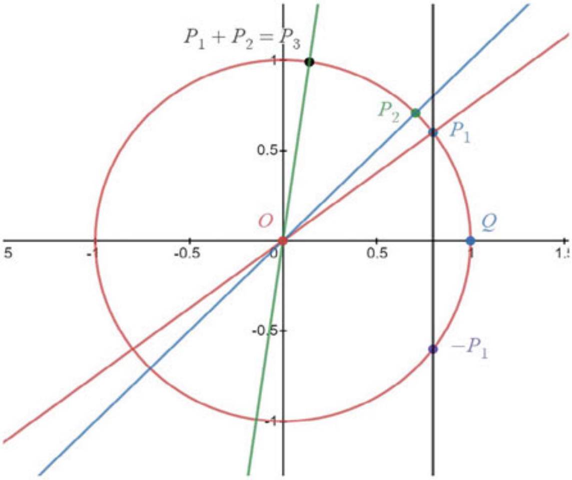


Fig. 14.1 Addition of points $P_1 + P_2 = P_3$ and computation of $-P_1$ in the circle group

Thus

$$\left(\frac{4}{5}, \frac{3}{5}\right) + \left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right) = \left(\frac{\sqrt{2}}{10}, \frac{7\sqrt{2}}{10}\right).$$

Also, $-(\frac{4}{5}, \frac{3}{5}) = (\frac{4}{5}, -\frac{3}{5})$. The group $G_{-1}(\mathbb{R})$ is the **circle group**. See Figure 14.1. □

A point $P \in G_{-1}(\mathbb{R})$ in the circle group can be given as

$$P = (\cos(\alpha), \sin(\alpha)),$$

for some α, $0 \leq \alpha < 2\pi$.

Now, the sum of points

$$P_1 + P_2 = (\cos(\alpha), \sin(\alpha)) + (\cos(\beta), \sin(\beta)) = P_3$$

is given by the formulas

$$x_3 = \cos(\alpha)(\cos(\beta) - \sin(\alpha)\sin(\beta),$$

$$y_3 = \cos(\alpha)\sin(\beta) + \sin(\alpha)\cos(\beta),$$

for some α, β, $0 \leq \alpha, \beta < 2\pi$.

In Figure 14.1, if $m(\angle P_1 O Q) = \alpha$ and $m(\angle P_2 O Q) = \beta$, then $m(\angle P_3 O Q) = \alpha + \beta$. Consequently,

$$\cos(\alpha + \beta) = \cos(\alpha)\cos(\beta) - \sin(\alpha)\sin(\beta),$$

$$\sin(\alpha + \beta) = \cos(\alpha)\sin(\beta) + \sin(\alpha)\cos(\beta),$$

which are familiar trigonometric identities.

Here are some examples of the group $G_c(K)$ over finite fields.

Example 14.3.3 Let $K = \mathbb{F}_{11}$, $c = 3$, so that

$$G_3(\mathbb{F}_{11}) = \{(x, y) \in \mathbb{F}_{11} \times \mathbb{F}_{11} : x^2 - 3y^2 = 1\}$$

with binary operation

$$P_1 + P_2 = P_3 = (x_3, y_3),$$

where $x_3 = x_1x_2 + 3y_1y_2$ and $y_3 = x_1y_2 + y_1x_2$. We can compute all of the points of $G_3(\mathbb{F}_{11})$ using the following table.

y	$3y^2 + 1$	x	Points
0	1	± 1	$(1, 0), (10, 0)$
1	4	± 2	$(2, 1), (9, 1)$
2	2	None	None
3	6	None	None
4	5	± 4	$(4, 4), (7, 4)$
5	10	None	None
6	10	None	None
7	5	± 4	$(4, 7), (7, 7)$
8	6	None	None
9	2	None	None
10	4	± 2	$(2, 10), (9, 10)$

Thus, $G_3(\mathbb{F}_{11})$

$$= \{(1, 0), (10, 0), (2, 1), (9, 1), (4, 4), (7, 4), (4, 7), (7, 7), (2, 10), (9, 10)\}.$$

We have $(2, 1) + (4, 7) = (7, 7)$ and $2(4, 4) = (9, 10)$. Note that $\left(\frac{3}{11}\right) = 1$, thus 3 is a quadratic residue modulo 11. In fact, $G_3(\mathbb{F}_{11}) \cong \mathbb{F}_{11}^{\times} \cong \mathbb{Z}_{10}$, as we shall soon see.

□

Example 14.3.4 Let $K = \mathbb{F}_7$, $c = 3$, so that

$$G_3(\mathbb{F}_7) = \{(x, y) \in \mathbb{F}_7 \times \mathbb{F}_7 : x^2 - 3y^2 = 1\}$$

with binary operation

$$P_1 + P_2 = P_3 = (x_3, y_3),$$

where $x_3 = x_1x_2 + 3y_1y_2$ and $y_3 = x_1y_2 + y_1x_2$. The points of $G_3(\mathbb{F}_7)$ are given by the following table.

y	$3y^2 + 1$	x	Points
0	1	± 1	$(1, 0), (6, 0)$
1	4	± 2	$(2, 1), (5, 1)$
2	6	None	None
3	0	0	$(0, 3)$
4	0	0	$(0, 4)$
5	6	None	None
6	4	± 2	$(2, 6), (5, 6)$

Thus

$$G_3(\mathbb{F}_7) = \{(1, 0), (6, 0), (2, 1), (5, 1), (0, 3), (0, 4), (2, 6), (5, 6)\}.$$

Note that $\left(\frac{3}{7}\right) = -1$, thus 3 is a not a quadratic residue modulo 7. In fact, $G_3(\mathbb{F}_7)$ is isomorphic to the subgroup $\mathbb{Z}_8$ of $\mathbb{F}_{49}^{\times} \cong \mathbb{Z}_{48}$.

□

Remark 14.3.5 Let $(x^2 - cy^2 - 1)$ denote the principal ideal of $K[x, y]$ generated by $x^2 - cy^2 - 1$. Then the quotient ring

$$H = K[x, y]/(x^2 - cy^2 - 1)$$

is a **Hopf algebra** over K. The group structure of $G_c(K)$ is a consequence of the Hopf algebra structure of H.

For details regarding Hopf algebras, the reader is referred to this author's text [60].

□

14.4 $E_{ns}(K) \cong G_c(K)$

We now show that the group $G_c(K)$ is essentially the same as the group $E_{ns}(K)$ of non-singular points.

Proposition 14.4.1 *Let K be a field of characteristic not 2 or 3. Let $E_{ns}(K)$ be the collection of non-singular points of the curve $y^2 = x^2(x + c)$, $c \neq 0$. Let $G_c(K)$ be the group of points*

$$G_c(K) = \{(u, v) \in K \times K : u^2 - cv^2 = 1\}.$$

Then there is an isomorphism of groups

$$\theta : E_{ns}(K) \to G_c(K)$$

defined as

$$\theta(x, y) = \left(\frac{y^2 + cx^2}{y^2 - cx^2}, \frac{2xy}{y^2 - cx^2}\right),\ \theta(O) = (1, 0),\ \theta(-c, 0) = (-1, 0).$$

Proof Let $\beta^2 = c$ for some $\beta \in \overline{K}$. Assume that $\beta \in K$. Then by Theorem 14.1.4(i), there is an isomorphism of groups $\psi : E_{ns}(K) \to K^\times$ defined by:

$$\psi(x, y) = \frac{y + \beta x}{y - \beta x}, \quad \rho(O) = 1.$$

By Pareigis [41, Section 1, p. 77], there is an isomorphism $\rho : K^\times \to G_c(K)$ defined by:

$$\rho(t) = \left(\frac{t + t^{-1}}{2}, \frac{t - t^{-1}}{2\beta}\right).$$

The composition

$$\theta = \rho \circ \psi$$

is an isomorphism $\theta : E_{ns}(K) \to G_c(K)$ with

$$\begin{aligned}
\theta(x, y) &= (\rho \circ \psi)(x, y) \\
&= \rho(\psi(x, y)) \\
&= \rho\left(\frac{y + \beta x}{y - \beta x}\right) \\
&= \left(\frac{1}{2}\left(\frac{y + \beta x}{y - \beta x} + \frac{y - \beta x}{y + \beta x}\right), \frac{1}{2\beta}\left(\frac{y + \beta x}{y - \beta x} - \frac{y - \beta x}{y + \beta x}\right)\right) \\
&= \left(\frac{y^2 + cx^2}{y^2 - cx^2}, \frac{2xy}{y^2 - cx^2}\right),
\end{aligned}$$

and

$$\theta(O) = (\rho \circ \psi)(O) = \rho(\psi(O)) = \rho(1) = (1, 0).$$

We next assume that $\beta \notin K$. Let $\psi : E_{ns}(K) \to J_c$,

$$(x, y) \mapsto \frac{y + \beta x}{y - \beta x} = \left(\frac{y^2 + cx^2}{y^2 - cx^2}\right) + \beta\left(\frac{2xy}{y^2 - cx^2}\right),$$

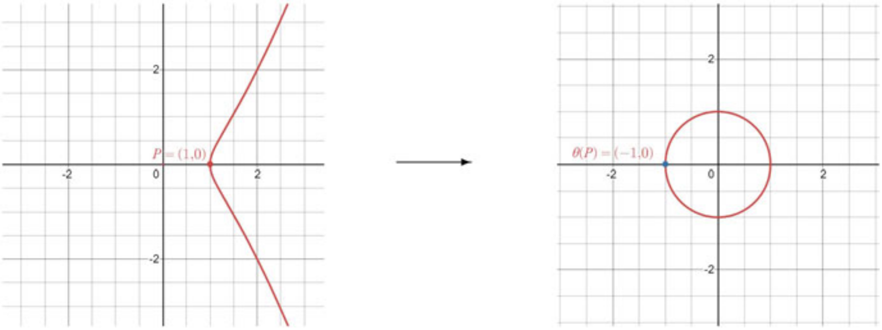


Fig. 14.2 The isomorphism $\theta : E_{ns}(\mathbb{R}) \to G_{-1}(\mathbb{R})$, $(x, y) \mapsto \left(\frac{y^2-x^2}{y^2+x^2}, \frac{2xy}{y^2+x^2}\right)$

be the isomorphism of Theorem 14.1.4(ii). There is an isomorphism $\eta : J_c \to G_c(K)$, given as $\eta(u + \beta v) = (u, v)$. Thus the composition $\theta = \eta \circ \psi$ is the isomorphism $E_{ns}(K) \to G_c(K)$, as claimed.

□

Example 14.4.2 Let $K = \mathbb{R}$ and $c = -1$, so that $E_{ns}(\mathbb{R})$ is the group of non-singular points on the curve $y^2 = x^2(x - 1)$ and $G_{-1}(\mathbb{R})$ is the circle group. The group isomorphism

$$\theta : E_{ns}(\mathbb{R}) \to G_{-1}(\mathbb{R})$$

is defined as

$$\theta(x, y) = \left(\frac{y^2 - x^2}{y^2 + x^2}, \frac{2xy}{y^2 + x^2}\right), \qquad \theta(O) = (1, 0).$$

For instance, $\theta(1, 0) = (-1, 0)$; see Figure 14.2.

□

Example 14.4.3 We take $K = \mathbb{F}_7$, $c = 3$, so that $E_{ns}(\mathbb{F}_7)$ is the group of non-singular points on the curve $y^2 = x^2(x + 3)$ and $G_3(\mathbb{F}_7)$ is the group of Example 14.3.4. We have

$$E_{ns}(\mathbb{F}_7) = \{(1, 2), (1, 5), (4, 0), (5, 2), (5, 5), (6, 3), (6, 4), O\},$$

and

$$G_3(\mathbb{F}_7) = \{(1, 0), (6, 0), (2, 1), (5, 1), (0, 3), (0, 4), (2, 6), (5, 6)\}.$$

The group isomorphism

$$\theta : E_{ns}(\mathbb{F}_7) \to G_3(\mathbb{F}_7)$$

is defined as

$$\theta(x, y) = \left(\frac{y^2 + 3x^2}{y^2 - 3x^2}, \frac{2xy}{y^2 - 3x^2}\right), \quad \theta(O) = (1, 0).$$

We have

$$\theta(O) = (1, 0), \quad \theta(1, 2) = (0, 4), \quad \theta(1, 5) = (0, 3), \quad \theta(4, 0) = (6, 0),$$

$$\theta(5, 2) = (5, 1), \quad \theta(5, 5) = (5, 6), \quad \theta(6, 3) = (0, 4), \quad \theta(6, 4) = (0, 3).$$

□

14.5 Exercises

1. Verify that the formulas given in Proposition 14.1.2 define a binary operation on $E_{ns}(K)$.
2. Compute the points of $E_{ns}(\mathbb{F}_7)$ defined by $y^2 = x^2(x + 2)$.
3. Show that $E_{ns}(\mathbb{F}_7)$ is a group under the binary operation of Proposition 14.1.2.
4. Prove Proposition 14.1.3.
5. Let p be an odd prime, and let $E_{ns}(\mathbb{F}_p)$ be the group of non-singular points defined by $y^2 = x^2(x + c)$, $c \neq 0$. Suppose that c is a quadratic residue modulo p, i.e., $\beta^2 = c$ for some $\beta \in \mathbb{F}_p^\times$.

   (a) Show that the map $\psi : E_{ns}(\mathbb{F}_p) \to \mathbb{F}_p^\times$ defined by $f(x, y) = \frac{y+\beta x}{y-\beta x}$, $f(O) = 1$, is an isomorphism of groups.

   (b) Let $E_{ns}(\mathbb{F}_7)$ be the group given in Exercise 3. Use part (a) to solve the DLP

   $$m(6, 1) = (5, 0)$$

   in $E_{ns}(\mathbb{F}_7)$ by taking images under ψ, and then solving the DLP

   $$\psi(6, 1)^m = \psi(5, 0)$$

   in $\mathbb{F}_7^\times$.
6. Prove Theorem 14.3.1.
7. Show that $G_{-1}(\mathbb{Q})$ (circle group over $\mathbb{Q}$) has an infinite number of points.

8. Let $G_{-1}(\mathbb{R})$ be the circle group.

   (a) Let $P = \left(\frac{5}{13}, \frac{12}{13}\right)$. Compute $2P$ and $-P$.

   (b) Compute $(-1, 0) + \left(\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}\right)$.

9. Compute the points of $G_5(\mathbb{F}_7)$.
10. Compute the points of $G_1(K)$, where K is the field $\mathbb{F}_9$ from Example 7.3.14.
11. Let $E_{ns}(\mathbb{Q})$ be the group of non-singular points on the curve $y^2 = x^2(x - 4)$, and let $\theta : E_{ns}(\mathbb{Q}) \to G_{-4}(\mathbb{Q})$ be the isomorphism defined as

$$\theta(x, y) = \left(\frac{y^2 - 4x^2}{y^2 + 4x^2}, \frac{2xy}{y^2 + 4x^2}\right), \quad \theta(O) = (1, 0).$$

   (a) Compute $\theta(5, -5)$.

   (b) Find the preimage of $\left(0, \frac{1}{2}\right) \in G_{-4}(\mathbb{Q})$ under θ.

12. Let $\theta : E_{ns}(\mathbb{R}) \to G_{-1}(\mathbb{R})$ be the isomorphism of groups of Example 14.4.2.

   (a) Prove that $J = \{O, (\frac{4}{3}, \frac{4}{3\sqrt{3}}), ((\frac{4}{3}, -\frac{4}{3\sqrt{3}})\}$ is a subgroup of $E_{ns}(\mathbb{R})$.

   (b) Find the image of J under θ.

13. Is $G_c(\mathbb{F}_p)$ a good choice for the group in a Diffie–Hellman key exchange protocol? Why or why not?
14. Let $E(\mathbb{R})$ be the elliptic curve group given by $y^2 = (x)(x + \epsilon)(x + c)$ for $\epsilon > 0$, $c > 0$. Let $E_{ns}(\mathbb{R})$ be the group of non-singular points on the curve $y^2 = x^2(x + c)$. Let $mP = Q$ be a DLP in $E(\mathbb{R})$. Show that a solution to this DLP can be approximated by a solution to some other DLP in $E_{ns}(\mathbb{R})$ and hence in $\mathbb{R}^\times$.

References

1. W. Alexi, B. Chor, O. Goldreich, C.P. Schorr, RSA/Rabin bits are $\frac{1}{2} + (1/\text{poly}(\log N))$ secure, in *IEEE 25th Symposium on Foundations of Computer Science* (1984), pp. 449–457
2. J.-P. Allouche, J.O. Shallit, *Automatic Sequences* (Cambridge University Press, Cambridge, 2003)
3. S. Baase, A. Van Gelder, *Computer Algorithms: Introduction to Design and Analysis* (Addison-Wesley, Reading, 2000)
4. G. Baumslag, B. Fine, M. Kreuzer, G. Rosenberger, *A Course in Mathematical Cryptography* (De Gruyter, Berlin, 2015)
5. E.R. Berlekamp, Factoring polynomials over large finite fields. Math. Comput. **24**, 713–735 (1970)
6. J. Berstel, C. Reutenauer, *Noncommutative Rational Series with Applications* (Cambridge University Press, Cambridge, 2011)
7. I. Blake, G. Seroussi, N. Smart, *Elliptic Curves in Cryptography* (Cambridge University Press, Cambridge, 1999)
8. L. Blum, M. Blum, M. Shub, A simple unpredictable pseudo-random number generator. Siam. J. Comput. **15**(2), 364–383 (1886)
9. E.R. Canfield, P. Erdős, C. Pomerance, On a problem of Oppenheim concerning "factorisatio numerorum. J. Num. Theory **17**(1), 1–28 (1983)
10. D. Chaum, E. van Heijst, B. Pfitzmann, Cryptographically strong undeniable signatures, unconditionally secure for the signer, in *Advances in Cryptology CRYPTO'91*. Lecture Notes in Computer Science, vol. 576 (Springer, Berlin, 1992), pp. 470–484
11. L.N. Childs, *Cryptology and Error Correction: An Algebraic Introduction and Real-World Applications* Springer Undergraduate Texts in Mathematics and Technology (Springer, Cham, 2019)
12. L.N. Childs, *Taming Wild Extensions: Hopf Algebras and Local Galois Module Theory*. AMS: Mathematical Surveys and Monographs, vol. 80 (American Mathematical Society, Providence, 2000)
13. G. Christol, Ensembles presque périodiques k-reconnaissables. Theor. Comput. Sci. **9**(1), 141–145 (1979)
14. G. Christol, T. Kamae, M.M. France, G. Rauzy, Suites algébriques, automates et substitutions. Bull. Soc. Math. France **108**, 401–419 (1980)
15. A. Cobham, On the base-dependence of sets of numbers recognizable by finite automata. Math. Syst. Theory **3**, 186–192 (1969)
16. A. Cobham, Uniform tag sequences. Math. Syst. Theory **6**, 164–192 (1972)

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7

17. M. Coons, P. Vrbik, An irrationality measure of regular paperfolding numbers. J. Int. Seq. **15**, 1–10 (2012)
18. D. Coppersmith, H. Krawczyk, Y. Mansour, *The Shrinking Generator* (IBM T. J. Watson Research Center, New York, 1998)
19. M.M. Eisen, C.A. Eisen, *Probability and its Applications* (Quantum, New York, 1975)
20. A. Godušová, Number field sieve for discrete logarithm, Masters Thesis, Charles University, Prague (2015)
21. C. Greither, B. Pareigis, Hopf Galois theory for separable field extensions. J. Algebra **106**, 239–258 (1987)
22. R. Haggenmüller, B. Pareigis, Hopf algebra forms on the multiplicative group and other groups. Manuscripta Math. **55**, 121–135 (1986)
23. D.W. Hardy, C.L. Walker, *Applied Algebra: Codes, Ciphers, and Discrete Algorithms* (Pearson, New Jersey, 2003)
24. K. Hoffman, R. Kunze, *Linear Algebra*, 2e (Prentice-Hall, New Jersey, 1971)
25. J. Hoffstein, J. Pipher, J.H. Silverman, *An Introduction to Mathematical Cryptography*. Undergraduate Texts in Mathematics Book Series (Springer, New York, 2008)
26. J.E. Hopcroft, J.D. Ulman, *Introduction to Automata Theory, Languages, and Computation* (Addison-Wesley, Reading, 1979)
27. J.E. Hopcroft, R. Motwani, J.D. Ulman, *Introduction to Automata Theory, Languages, and Computation*, 3e (Addison-Wesley, Reading, 2007)
28. K. Ireland, M. Rosen, *A Classical Introduction to Modern Number Theory*, Graduate Text in Mathematics, vol. 84, 2nd edn. (Springer, New York, 1990)
29. L. Işik, A. Winterhof, Maximum-order complexity and correlation measures (2017). arXiv:1703.09151
30. C.J.A. Jansen, The maximum order complexity of sequence ensembles, in *Advances in Cryptology-EUROCRYPT '91*, ed. by D.W. Davies. Lecture Notes in Computer Science, vol. 547 (Springer, Berlin, 1991), pp. 153–159
31. C.J.A. Jansen, D.E. Boekee, The shortest feedback shift register that can generate a given sequence, in *Advances in Cryptology-CRYPTO'89*, ed. by G. Brassard, Lecture Notes in Computer. Science (Springer, Berlin, 1990), pp. 435, 90–99
32. N. Koblitz, *A Course in Number Theory and Cryptography*. Graduate Text in Mathematics, vol. 114 (Springer, New York, 1987)
33. N. Koblitz, *Algebraic Aspects of Cryptography* (Springer, Berlin, 1998)
34. A. Koch, T. Kohl, P. Truman, R. Underwood, The structure of hopf algebras acting on dihedral extensions, in *Advances in Algebra. SRAC 2017*, ed. by J. Feldvoss, L. Grimley, D. Lewis, A. Pavelescu, C. Pillen. Springer Proceedings in Mathematics & Statistics, vol 277 (Springer, Cham, 2019)
35. A.G. Konheim, *Cryptography: A Primer* (Wiley, New York, 1981)
36. S. Lang, *Algebra*, 2nd edn. (Addison-Wesley, Reading, 1984)
37. W. Mao, *Modern Cryptography* (Prentice-Hall, New Jersey, 2004)
38. A.J. Menezes, T. Okamoto, S.A. Vanstone, Reducing elliptic curve logarithms to logarithms in a finite field. IEEE Trans. Inf. Theory **39**(5), 1639–1646 (1993)
39. A.J. Menezes, P.C. van Oorschot, S.A.Vanstone, *Handbook of Applied Cryptography* (CRC Press, Boca Raton, 1997)
40. L. Mérai, A. Winterhof, On the Nth linear complexity of automatic sequences. J. Num. Theory **187**, 415–429 (2018)
41. B. Pareigis, *Forms of Hopf Algebras and Galois Theory*. Topics in Algebra, Banach Center Publications, vol. 26, Part 1 (PWN Polish Scientific Publishers, Warsaw, 1990)
42. J.M. Pollard, Theorems on factorizations and primality testing. Proc. Cambridge. Phil. Soc. **76**, 521–528 (1974)
43. J.M. Pollard, A Monte Carlo method for factorization. Nor. Tid. Inform **15**, 331–334 (1975)
44. C. Pomerance, A tale of two sieves. Not. Am. Math. Soc. **43**(12), 1473–1485 (1996)
45. P. Popoli, On the maximum order complexity of the Thue-Morse and Rudin-Shapiro sequences along polynomial values (2020). arXiv: 2011.03457

46. M. Rigo, *Formal Languages, Automata and Numeration Systems 1* (Wiley, New Jersey, 2014)
47. K. Rosen, *Elementary Number Theory and Its Applications*, 6th edn. (Addison-Wesley, Boston, 2011)
48. J.J. Rotman, *Advanced Modern Algebra* (Prentice-Hall, New Jersey, 2002)
49. E. Rowland, What is… an automatic sequence? Not. Am. Math. Soc. **62**, 274–276 (2015)
50. S. Rubinstein-Salzedo, *Cryptography*. Springer Undergraduate Mathematics Series (Springer, Cham, 2018)
51. W. Rudin, *Principles of Mathematical Analysis*, 3rd edn. (McGraw-Hill, New York, 1976)
52. I.R. Shafarevich, *Basic Algebraic Geometry* (Springer, New York, 1974)
53. D. Shanks, Class number, a theory of factorization and genera, in *Proceedings of Symposium of Pure Mathematics*, vol. 20 (American Mathematical Society, Providence, 1971), pp. 415–440
54. C.E. Shannon, A mathematical theory of communication. Bell Syst. Tech. J. **27**(3), 379–423 (1948)
55. C.E. Shannon, A mathematical theory of communication. Bell Syst. Tech. J. **27**, 623–656 (1948)
56. N.P. Smart, *Cryptography Made Simple* (Springer, Cham, 2016)
57. Z. Sun, A. Winterhof, On the maximum order complexity of the Thue-Morse and Rudin-Shapiro sequence (2019). arXiv:1910.13723
58. Z. Sun, A. Winterhof, On the maximum order complexity of subsequences of the Thue-Morse and Rudin-Shapiro sequence along squares. Int. J. Comput. Math.: Comput. Syst. Theory **4**(1), 30–36 (2019)
59. J.T. Talbot, D. Welsh, *Complexity and Cryptography: An Introduction* (Cambridge University Press, Cambridge, 2006)
60. R.G. Underwood, *Fundamentals of Hopf Algebras*. Universitext (Springer, Cham, 2015)
61. R.G. Underwood, *Fundamentals of Modern Algebra: A Global Perspective* (World Scientific, New Jersey, 2016)
62. R. Underwood, Hopf forms and Hopf-Galois theory. www.scm.keele.ac.uk/staff/p_truman/ConferenceArchive/2020Omaha/index.html
63. L.C. Washington, *Elliptic Curves, Number Theory and Cryptography* (Chapman/Hall/CRC, Boca Raton, 2003)
64. J. Winter, Erratum to various proofs of Christol's theorem. www.mimuw.edu.pl/~jwinter/articles/christol.pdf
65. S. Woltmann, www.happycoders.eu/algorithms/merge-sort/#Merge_Sort_Time_Complexity

Index

R. G. Underwood, *Cryptography for Secure Encryption*, Universitext,
https://doi.org/10.1007/978-3-030-97902-7