

# 1. Kryptologie und RSA-Verfahren

Die Kryptologie ist die Wissenschaft der Ver- und Entschlüsselung von Informationen. Sie lässt sich in die beiden Domänen *Kryptographie* (oder auch Kryptografie) und *Kryptoanalyse* (oder auch Kryptanalyse) aufteilen. Erstere beschäftigt sich mit dem sicheren Verschlüsseln von Informationen, während zweitens die Informationsgewinnung aus verschlüsselter Information darstellt.

## 1.1. Kryptographie

In diesem Kapitel soll erklärt werden, wie die modulare Arithmetik in einem der wichtigsten modernen Verschlüsselungsverfahren, dem *RSA-Verfahren*, angewendet wird. Die zentrale Herausforderung der Kryptographie besteht darin, Nachrichten sicher von einem Sender zu einem Empfänger zu übertragen, sodass unbefugte Dritte keinen Zugriff auf den Inhalt erhalten. Üblicherweise werden hierfür die Namen *Alice* (Sender), *Bob* (Empfänger) und *Charlie* (Dritte, unbefugte Person, die versucht die Nachricht zu entschlüsseln) genutzt. Da Übertragungskanäle abgehört werden könnten, zielt die Kryptographie darauf ab, Nachrichten so zu verschlüsseln, dass nur Bob, aber nicht Charlie, sie entschlüsseln und lesen kann. Nach dem **Kerchhoffschen Prinzip** beruht dabei die Sicherheit eines solchen kryptografischen Verfahrens aber nicht auf der Geheimhaltung des Verfahrens, sondern nur von der Geheimhaltung des entsprechenden Schlüssels zum Ver-/Entschlüsseln.

Die moderne Kryptographie nahm mit der Veröffentlichung eines bahnbrechenden Artikels von W. Diffie und M.E. Hellman im Jahr 1976 ihren Anfang: Sie begründeten die *Public-Key-Kryptographie*. Ihre visionäre Idee bestand darin, ein Verschlüsselungssystem zu entwickeln, bei dem es unmöglich ist, **mit praktikablem Rechenaufwand** aus der Kenntnis des Verschlüsselungsalgorithmus („Wie wird eine Nachricht verschlüsselt?“) **und** des verwendeten Schlüssels („Welche Parameter/Zahlen werden für diesen Algorithmus genutzt?“) die entschlüsselte Nachricht abzuleiten. Falls ein solches System existiert, könnten der Verschlüsselungsalgorithmus und der Schlüssel öffentlich zugänglich gemacht werden, ohne die Sicherheit zu gefährden.

Das Grundprinzip eines Public-Key-Verfahrens lässt sich wie folgt skizzieren: Alice und Bob besitzen ein Schlüsselpaar, bestehend aus einem öffentlichen Schlüssel („public key“) und einem geheimen privaten Schlüssel („secret key“), die unterschiedlich sind. Alice kann mit dem öffentlichen Schlüssel verschlüsselte Nachrichten an Bob senden. Nur Bob selbst ist in der Lage, diese Nachrichten mithilfe seines privaten Schlüssels zu entschlüsseln. Da Alice und Bob verschiedene Schlüssel zum Ver- und Entschlüsseln nutzen, spricht man bei solchen Public-Key-Verfahren daher auch von einem **asymmetrischen** Verfahren; im Gegensatz zu **symmetrischen** Verfahren, bei denen derselbe Schlüssel verwendet wird.

Der entscheidende Vorteil des Public-Key-Verfahrens – im Vergleich zu klassischen Verschlüsselungsmethoden – liegt darin, dass Alice und Bob vorab keinen gemeinsamen geheimen Schlüssel austauschen müssen, der prinzipiell abgehört werden kann. Da hingegen symmetrische Verfahren schneller sind, können auch zwei Verfahren kombiniert werden: Zunächst wird die Nachricht symmetrisch verschlüsselt und anschließend der entsprechende Schlüssel mit Hilfe eines asymmetrischen Verfahrens (**hybride Verschlüsselung**).

## 1.2. Das RSA-Verfahren

Die zentrale Frage, ob ein Public-Key-Verfahren tatsächlich umsetzbar ist, wurde 1978 von R. Rivest, A. Shamir und L. Adleman (**RSA**) in einem wegweisenden Artikel beantwortet. Sie präsentierten ein Verfahren, welches bis heute eine tragende Säule der Kryptographie im Zeitalter des Internets darstellt. Das Verfahren beruht dabei auf grundlegenden Begriffen und Sätzen der Zahlentheorie, die im Kapitel Zahlentheoretischer Hintergrund erläutert sind.

Im Folgenden werden wir den grundlegenden Algorithmus sowie die Prinzipien dieses Systems erläutern.

### 1. Vorbereitungen von Bob

- Bob wählt zwei verschiedene Primzahlen  $p$  und  $q$ . Er berechnet damit  $n := pq$  und  $\varphi(n) = (p - 1)(q - 1)$
- Bob wählt eine beliebige natürliche Zahl  $e$  mit  $\text{ggT}(e, \varphi(n)) = 1$ . Daher existiert in  $\mathbb{Z}_{\varphi(n)}$  ein Element, das zu  $e$  (in  $\mathbb{Z}_{\varphi(n)}$ ) invers ist. Dies bezeichnen wir mit  $d$ :  
$$ed \equiv 1 \pmod{\varphi(n)}$$

Das Element  $d$  kann Bob mithilfe des erweiterten euklidischen Algorithmus bestimmen.

- Bob veröffentlicht das Paar  $(e, n)$  (public key), hält  $d$  geheim (secret key) und vergisst  $p, q$  sowie  $\varphi(n)$ .

### 2. Alice verschlüsselt Nachricht $N$

- Alice verfasst eine Nachricht als  $N \in \mathbb{N}$ .<sup>1</sup>
- Alice holt sich das Paar  $(e, n)$  und überprüft, ob  $N < n$  gilt. Ansonsten muss sie Ihre Nachricht umschreiben, sodass diese Bedingung gilt.
- Alice berechnet die verschlüsselte Nachricht  $V := N^e \pmod{n}$ .
- Alice veröffentlicht die Nachricht  $V$ .

### 3. Bob entschlüsselt $V$

- Bob holt sich  $V$  und berechnet mit seinem geheimen Schlüssel  $d$ :  
$$E := V^d \pmod{n}$$
- $E$  ist dann die entschlüsselte Nachricht, d.h. es gilt  $N = E$ .

#### Bemerkungen:

- Es genügt jede Art von Nachricht  $N$  nur als natürliche Zahlen zu betrachten, weil jede Art von Information – sei es Text, Bilder, Audiodaten oder andere Datenformate – letztlich in digitaler Form durch eine Abfolge von Bits (0 und 1) repräsentiert wird. Diese Bitfolgen können wiederum direkt als natürliche Zahlen interpretiert werden.
- Die Berechnung von  $d$ , die Bob mittels des erweiterten euklidischen Algorithmus durchführt, ist effizient möglich. Des Weiteren müssen lediglich Potenzen mit großen Exponenten (in einem endlichen Restklassenring) berechnet werden; auch hierfür existieren effiziente Algorithmen.
- Das Paar  $(e, n)$  muss nicht geheim halten werden. Mathematisch gesehen lässt sich das Inverse  $d$  zwar mit diesen Informationen eindeutig bestimmen, es existiert aber bisher keine effiziente Methode hierfür. Dies liegt vor allem daran, dass es bisher keine (effizienten) Algorithmen gibt, um **sehr große  $n$  zu faktorisieren** und somit letztlich  $\varphi(n)$  zu berechnen. Somit kennt nur Bob den Wert von  $\varphi(n) = (p - 1)(q - 1)$ .

<sup>1</sup> (warum hier eine Nachricht als natürliche Zahl aufgefasst werden kann, erklären wir gleich)

- Weiterhin existieren keine (effizienten) Algorithmen, um Wurzeln in Restklassenringen zu berechnen. Zwar ist  $N = \sqrt[e]{V}$  und  $e$  sowie  $V$  öffentlich, jedoch ist die konkrete Bestimmung in der Praxis nicht möglich. Daher kann Alice ohne Probleme ihre verschlüsselte Nachricht  $V$  veröffentlichen.
- Bob hingegen ersetzt das problematische Wurzelziehen durch Potenzieren mit  $d$ , was wiederum in Restklassenringen effizient durchführbar ist. Somit ist nur Bob in der Lage die ursprüngliche Nachricht  $N$  zu rekonstruieren.
- Um zu beweisen, dass  $N = E$  gilt, wird der chinesische Restsatz benötigt. Mit diesem lässt sich eine Kongruenz in mehrere Kongruenzen mit paarweise verschiedenen Modulen aufteilen und umgekehrt.

### 1.3. Das RSA-Verfahren – Ein Beispiel

**i Beispiel:** Bob wählt zufällig die beiden verschiedenen Primzahlen  $p = 7$  und  $q = 11$ . Damit ist  $n = pq = 77$  und  $\varphi(n) = \varphi(77) = (7 - 1)(11 - 1) = 60$ . Als dazu teilerfremde, natürliche Zahl nimmt er bspw.  $e = 7$ , da  $\text{ggT}(e, \varphi(n)) = \text{ggT}(7, 60) = 1$  gilt. Nun bestimmt er mittels des erweiterten euklidischen Algorithmus die Zahl  $d$  (In Wahrheit, lässt er diese Zahl natürlich von einem Rechner bestimmen):

- I.  $60 = 8 \cdot 7 + 4$
- II.  $7 = 1 \cdot 4 + 3$
- III.  $4 = 1 \cdot 3 + 1$
- IV.  $3 = 3 \cdot 1 + 0$

$$\text{Es ist } 1 \stackrel{\text{III.}}{=} 4 - 3 \stackrel{\text{II.}}{=} 4 - (7 - 4) \stackrel{\text{I.}}{=} (60 - 8 \cdot 7) - (7 - (60 - 8 \cdot 7)) = 2 \cdot 60 - 17 \cdot 7.$$

Da diese Gleichung modulo  $\varphi(n) = 60$  betrachten werden muss, ist  $d = 43$  wegen  $-17 \equiv 43 \pmod{60}$ .

Nun veröffentlicht Bob das Paar  $(e, n) = (7, 77)$  und vergisst  $p = 7$ ,  $q = 11$  sowie  $\varphi(n) = 60$  und hält  $d = 43$  für sich geheim.

Alice schreibt ihre Nachricht und übersetzt diese in eine natürliche Zahl  $N$ . In unserem Beispiel möchte Sie  $N = 69$  übermitteln. Sie überprüft, ob  $N < n$  gilt, was erfüllt ist ( $69 < 77$ ). Nun berechnet Sie die verschlüsselte Nachricht  $V$  modulo  $n$ :

$$V = N^e = 69^7 \equiv 20 \pmod{77}$$

Diese Nachricht übermittelt sie nun Bob. Er kann die Nachricht mittels seines geheimen Schlüssels  $d$  wieder als entschlüsselte Nachricht  $E$  darstellen:

$$E = V^d = 20^{43} \equiv 69 (= N) \pmod{77}$$

Die geheime Botschaft  $N = 69$  kann also nur von Bob entschlüsselt werden. Selbst wenn Charlie die verschlüsselte Nachricht  $V = 20$  abhören könnte, könnte er mit Hilfe dieser  $N$  nicht entschlüsseln. Bei diesem Beispiel wäre es natürlich sehr leicht, die Zahlen  $p$  und  $q$  zu ermitteln, weshalb in der Realität viel größere Zahlen genutzt werden.

Doch was heißt „große Zahlen“? Das Bundesamt für Sicherheit in der Informationstechnik (BSI) nennt in seinen technischen Leitlinien<sup>2</sup> für die Primzahlen  $p$  und  $q$  beispielsweise:

$$p, q > 2^{1500} \approx 10^{450} \text{ (Stand: 02.02.2024)}$$

Für die Primzahlen wird vom BSI weiter gefordert, dass sie „von vergleichbarer Bitlänge“ sind und „nicht zu nah beieinander liegen“ sollten, da ansonsten Angriffe auf die Verschlüsselung möglich sind. Des Weiteren sollte der frei wählbare und öffentliche Exponent  $e$  die Nebenbedingung

$$2^{16} + 1 \leq e \leq 2^{256} - 1 \text{ erfüllen.}$$

Ein Beweis für die Nicht-Existenz effizienter Algorithmen zur Lösung der oben genannten Probleme steht bis heute aus. Dies bedeutet, dass die Sicherheit des RSA-Verfahrens nicht garantiert werden kann. Dennoch hat sich RSA in den letzten rund 40 Jahren als robust erwiesen: Es wird intensiv beforscht und weltweit angewendet, ohne dass es bisher in seiner allgemeinen Form geknackt werden konnte. Unter der Voraussetzung, dass die Primzahlen  $p$  und  $q$  ausreichend groß gewählt werden, gilt das Verfahren daher nach wie vor als praktisch sicher – zumindest so lange kein Durchbruch in der Entwicklung effizienter Algorithmen oder neuer Angriffsverfahren gelingt. Ein Durchbruch wird aber mit Hilfe von Quantencomputern gelingen, sobald deren Leistung stark genug ist. Diese sind in der Lage, auch extrem große Zahlen effizient zu faktorisieren, was das RSA-Verfahren verwundbar macht. Angesichts intensiver globaler Forschung und bereits erzielter Fortschritte in der Quantencomputer-Technologie scheint das Ende von RSA unausweichlich. Daher hat das National Institute of Standards and Technology einen Wettbewerb zu Post-Quanten-Kryptographie ausgeschrieben – also Verfahren, die gegenüber Quantencomputern sicher sind. Bei diesem stehen inzwischen Gewinner fest, die nun standardisiert werden sollen und von denen auch schon ein Verfahren im Einsatz ist (bspw. im Messengerdienst „Signal“).

## 2. Zahlentheoretischer Hintergrund

### 2.1. Erweiterter euklidischer Algorithmus

Für die Verschlüsselungstechnik spielt die Bestimmung des größten gemeinsamen Teilers ( $ggT$ ) zweier ganzer Zahlen eine wichtige Rolle. Für kleine Zahlen lässt sich dieser mittels der Primfaktorzerlegung (PFZ) beider Zahlen bestimmen. Da das Faktorisieren großer Zahlen ein bis heute rechenaufwendiges Problem darstellt, wird ein anderes Verfahren benötigt: nämlich der euklidische Algorithmus<sup>3</sup>. Er beruht auf folgendem unscheinbarem Satz:

Seien  $a, b \in \mathbb{Z}$ . Für  $q, r \in \mathbb{Z}$  mit  $a = qb + r$  gilt  $ggT(a, b) = ggT(b, r)$ .



**Beispiel:** Für die Zahlen  $a = 308$  und  $b = 182$  gilt:

$$308 = 1 \cdot 182 + 126$$

Es ist damit  $ggT(308, 182) = ggT(182, 126)$

<sup>2</sup> Bundesamt für Sicherheit in der Informationstechnik (2024). BSI TR-02102-01: Kryptographische Verfahren: Empfehlungen und Schlüssellängen. <https://www.bsi.bund.de/TR-02102>

<sup>3</sup> Dieser wurde erstmals von Euklid 300 v. Chr. in seinem Werk *Die Elemente* dargestellt; wurde dort aber – wie für griechische Mathematiker üblich – mittels geometrischer Begriffe beschrieben (über das Maß von Strecken).

Man kann also die  $ggT$ -Bestimmung der (großen) ganzen Zahlen  $b$  und  $a$  auf die  $ggT$ -Bestimmung von  $b$  mit der etwas kleineren Zahl  $r$  zurückführen. Nun kann man das Verfahren rekursiv auf  $b$  und  $r$  anwenden, bis der Rest  $r$  gleich null ist. Der  $ggT$  von  $a$  und  $b$  ist dann der letzte von Null verschiedene Rest.



**Beispiel:**

Bestimmung des  $ggT$  von 308 und 182 mittels euklidischen Algorithmus:

- I.  $308 = 1 \cdot 182 + 126$
- II.  $182 = 1 \cdot 126 + 56$
- III.  $126 = 2 \cdot 56 + 14$
- IV.  $56 = 4 \cdot 14 + 0$

Damit ist 14 der  $ggT$  von 308 und 182.

Dieses Verfahren ist besonders effizient, da sich die Größe der Zahlen in jedem Schritt deutlich verringert. Insbesondere lässt das Verfahren als ein Algorithmus gut in Tabellenkalkulationsprogramme implementieren. Ein wichtiges Resultat bezüglich des  $ggT$  ist das folgende:

**Lemma von Bézout:**

Seien  $a, b \in \mathbb{Z}$ . Dann existieren ganze Zahlen  $s$  und  $t$  mit  $ggT(a, b) = sa + tb$ .

Die Koeffizienten  $s$  und  $t$ , die nach diesem Lemma existieren, lassen sich mittels des sogenannten *erweiterten euklidischen Algorithmus* bestimmen.



**Beispiel:** Nach obigem Beispiel ist  $ggT(308, 182) = 14$ . Um die Koeffizienten der Linearkombination zu bestimmen, die nach dem Lemma von Bézout existieren, „gehen“ wir den euklidischen Algorithmus „rückwärts“ durch und ersetzen stückweise die Faktoren, bis wir bei 308 und 182 angekommen sind:

$$\begin{aligned}
 14 &\stackrel{\text{III.}}{=} 126 - 2 \cdot 56 \stackrel{\text{II.}}{=} 126 - 2 \cdot (182 - 1 \cdot 126) \stackrel{\text{I.}}{=} \\
 &= (308 - 1 \cdot 182) - 2 \cdot (182 - 1 \cdot (308 - 1 \cdot 182)) = 3 \cdot 308 - 5 \cdot 182 (= 928 - 910)
 \end{aligned}$$

Damit ist (in der Notation des Lemmas von Bézout)  $s = 3$  und  $t = -5$ .

Eine besondere Eigenschaft tritt auf, wenn für zwei ganze Zahlen  $a$  und  $b$  gilt, dass  $ggT(a, b) = 1$  ist. In diesem Fall nennt man  $a$  und  $b$  *teilerfremd*.

## 2.2. Modulare Arithmetik

Der Kerngedanke der modularen Arithmetik besteht darin, die ganzen Zahlen in sogenannte Restklassen zu unterteilen, um bei Problemen lediglich auf Grundlage von Resten argumentieren zu können. Diese können als Zahlssysteme mit endlich vielen Elementen betrachtet werden.

Teilbarkeit: Seien  $a$  und  $b$  ganze Zahlen:  $a$  *teilt*  $b$  (in Zeichen  $a \mid b$ ) genau dann, wenn  $b = a \cdot k$  für ein  $k \in \mathbb{Z}$  gilt. Ist diese Darstellung für kein  $k \in \mathbb{Z}$  möglich, sagt man  $a$  *teilt nicht*  $b$  (in Zeichen  $a \nmid b$ ).<sup>4</sup> Beachten Sie hierbei, dass insbesondere  $k = 0$  zugelassen ist!

<sup>4</sup> Diese Zeichen können dabei in Microsoft Word recht einfach über den „Unicode“ eingegeben werden – egal ob im Formeleditor (Alt + Shift + 0) oder im Standardtext. Hierzu werden vier Ziffern/Symbole geschrieben und anschließend die Tastenkombination Alt + C gedrückt (ohne Leerzeichen). Dabei gilt,  $\mid \triangleq 2223$  und  $\nmid \triangleq 2224$ . Im Formeleditor ist  $\mathbb{N} \triangleq \backslash double N$  etc.



**Beispiel:** Es gilt  $a \mid 0$  für alle  $a \in \mathbb{Z}$ , da stets  $a \cdot 0 = 0$  gilt.

Zusätzlich gilt für alle  $a \in \mathbb{Z} \setminus \{0\}$  auch  $0 \nmid a$ . Denn für kein  $k \in \mathbb{Z}$  gilt  $0 \cdot k = a$

Zusammengefasst: „Alles teilt die Null, aber die Null teilt nur die Null.“

### 2.2.1. Restklassen und Kongruenzen

Sei  $n \in \mathbb{N}$  und  $a, b \in \mathbb{Z}$ . Man nennt  $a$  *kongruent zu  $b$  modulo  $n$*  genau dann, wenn  $n$  die Differenz  $a - b$  teilt, d.h.  $n \mid a - b$ . Man schreibt dann  $a \equiv b \pmod{n}$ . In diesem Kontext nennt man  $n$  den zugrunde liegenden *Modul*.



**Beispiel:** Sei  $n = 3$ . Dann gibt es genau drei Möglichkeiten:

$a \equiv 0 \pmod{3}$	$a \equiv 1 \pmod{3}$	$a \equiv 2 \pmod{3}$
$a \in \{\dots, -6, -3, 0, 3, 6, \dots\}$	$a \in \{\dots, -5, -2, 1, 4, 7, \dots\}$	$a \in \{\dots, -7, -4, -1, 2, 5, \dots\}$
$a$ ist durch 3 teilbar	$a$ lässt bei der Division durch 3 den Rest 1	$a$ lässt bei der Division durch 3 den Rest 2

Die Kongruenz  $a \equiv b \pmod{n}$  bedeutet also nichts anderes, als dass  $a$  und  $b$  bei Division mit Rest durch  $n$  denselben Rest lassen. All jene ganze Zahlen mit dieser Eigenschaft lassen sich dann zusammenfassen:

Seien  $n \in \mathbb{N}$  und  $a \in \mathbb{Z}$ . Dann bezeichnet man mit

$$[a]_n := \{b \in \mathbb{Z} \mid b \equiv a \pmod{n}\}$$

die *Restklasse von  $a$  modulo  $n$* . Die Menge aller Restklassen modulo  $n$  bezeichnet man mit

$$\mathbb{Z}_n = \{[0]_n, [1]_n, \dots, [n-1]_n\}.$$

### 2.2.2. Grundrechenarten mit Kongruenzen

Wir möchten die Menge  $\mathbb{Z}_n$  der Restklassen modulo  $n$  zu einem Zahlbereich erweitern, indem wir je eine geeignete Addition und Multiplikation definieren. Die grundsätzliche Idee dabei ist, die Addition und Multiplikation aus den ganzen Zahlen zu benutzen, aber das Ergebnis jeweils modulo  $n$  zu „reduzieren“.

Die Addition und Multiplikation werden daher formal wie folgt definiert:

$$+ : \mathbb{Z}_n \times \mathbb{Z}_n \rightarrow \mathbb{Z}_n, [a]_n + [b]_n = [a + b]_n$$

$$\cdot : \mathbb{Z}_n \times \mathbb{Z}_n \rightarrow \mathbb{Z}_n, [a]_n \cdot [b]_n = [a \cdot b]_n$$





**Beispiel:** Mit Hilfe einer Additions- und Multiplikationstafel lassen sich bspw. für  $\mathbb{Z}_4$  alle Möglichkeiten wie folgt darstellen:

+	$[0]_4$	$[1]_4$	$[2]_4$	$[3]_4$
$[0]_4$	$[0]_4$	$[1]_4$	$[2]_4$	$[3]_4$
$[1]_4$	$[1]_4$	$[2]_4$	$[3]_4$	$[0]_4$
$[2]_4$	$[2]_4$	$[3]_4$	$[0]_4$	$[1]_4$
$[3]_4$	$[3]_4$	$[0]_4$	$[1]_4$	$[2]_4$

$\cdot$	$[0]_4$	$[1]_4$	$[2]_4$	$[3]_4$
$[0]_4$	$[0]_4$	$[0]_4$	$[0]_4$	$[0]_4$
$[1]_4$	$[0]_4$	$[1]_4$	$[2]_4$	$[3]_4$
$[2]_4$	$[0]_4$	$[2]_4$	$[0]_4$	$[2]_4$
$[3]_4$	$[0]_4$	$[3]_4$	$[2]_4$	$[1]_4$

Zum Beispiel ist  $[2]_4 + [3]_4 = [5]_4 = [1]_4$ .

Dabei überträgt sich die Kommutativität der Addition und Subtraktion aus den dem Bereich der ganzen Zahlen, weshalb die Tabellen symmetrisch sind. In der Addition fällt auf, dass es für jede Restklasse eine weitere Restklasse existiert, sodass deren Summe das neutrale Element  $[0]_4$  ergibt:  $[1]_4 + [3]_4 = [0]_4 = [2]_4 + [2]_4$ .

Diese und weitere Erkenntnisse aus dem Beispiel lassen sich verallgemeinern und zu Rechenregeln formulieren:

Sei  $n \in \mathbb{N}$  beliebig,  $\mathbb{Z}_n$  die Menge der Restklassen modulo  $n$  und  $[a]_n, [b]_n, [c]_n \in \mathbb{Z}_n$ . Dann gilt:

(R1)  $[a]_n + [b]_n = [b]_n + [a]_n$  (Kommutativgesetz der Addition)

(R2)  $[a]_n + ([b]_n + [c]_n) = ([a]_n + [b]_n) + [c]_n$  (Assoziativgesetz der Addition)

(R3) Für  $[0]_n \in \mathbb{Z}_n$  gilt  $[0]_n + [a]_n = [a]_n$  ( $[0]_n$  ist neutrales Element der Addition)

(R4) Für jedes  $[a]_n \in \mathbb{Z}_n$  existiert eine Restklasse  $-[a]_n$  mit  $[a]_n + (-[a]_n) = [0]_n$   
( $-[a]_n = [n - a]_n$  ist inverses Element der Addition)

(R5)  $[a]_n \cdot [b]_n = [b]_n \cdot [a]_n$  (Kommutativgesetz der Multiplikation)

(R6)  $[a]_n \cdot ([b]_n \cdot [c]_n) = ([a]_n \cdot [b]_n) \cdot [c]_n$  (Assoziativgesetz der Multiplikation)

(R7) Für  $[1]_n \in \mathbb{Z}_n$  gilt  $[1]_n \cdot [a]_n = [a]_n$  ( $[1]_n$  ist neutrales Element der Multiplikation)

(R8)  $[a]_n \cdot ([b]_n + [c]_n) = [a]_n \cdot [b]_n + [a]_n \cdot [c]_n$   
( $[a]_n + [b]_n) \cdot [c]_n = [a]_n \cdot [c]_n + [b]_n \cdot [c]_n$  (Distributivgesetze))

Die Rechenregeln R3 und R7 gelten aufgrund der Kommutativität der Addition und Multiplikation auch umgekehrt. Diese Rechenregeln erlauben nun bei der Addition und Multiplikation in  $\mathbb{Z}_n$  die Summanden und Faktoren separat modulo  $n$  zu reduzieren.

**i Beispiel:** Was ist die letzte Ziffer der Zahl  $7^{2024}$ ?

Diese Frage lässt sich in die Frage überführen, in welcher Restklasse  $7^{2024}$  modulo 10 liegt, da man jede Dezimalzahl mit den Ziffern  $z_i \in \{0, \dots, 9\}$  schreiben kann als  $z_n \dots z_2 z_1 z_0 = z_n \cdot 10^n + \dots + z_1 \cdot 10^1 + z_0 \equiv z_0 \pmod{10}$ . In unserem Beispiel ist

$$7^{2024} = 7^{2 \cdot 2 \cdot 506} = ((7^2)^2)^{506} = ((49)^2)^{506} \stackrel{49 \equiv -1 \pmod{10}}{\equiv} ((-1)^2)^{506} = (1)^{506} = 1 \pmod{10}$$

und daher ist die letzte Ziffer der Zahl  $7^{2024}$  eine 1.

Ein wesentlicher Vorteil des Zahlbereichs  $\mathbb{Z}_n$  ist also, dass in  $\mathbb{Z}$  gearbeitet wird, jedoch trotzdem nur endlich viele Zahlen/Restklassen zu betrachten sind, da jederzeit die ganze Zahl  $a$  durch eine andere ganze Zahl  $b$  mit  $a \equiv b \pmod{n}$  ersetzt werden kann.

Eine Besonderheit im Vergleich zu  $\mathbb{Z}$  ist die Eigenschaft der Invertierbarkeit bzgl. der Multiplikation. So existiert beispielweise für keine ganze Zahl  $a \in \mathbb{Z} \setminus \{\pm 1\}$  eine ganze Zahl  $b \in \mathbb{Z}$  mit  $ab = 1$ . (So ein  $b$  hieße dann das *Inverse* zu  $a$ .) Die ganzen Zahlen (ausgenommen  $\pm 1$ ) sind also nicht invertierbar. Für Restklassen muss dies nicht immer der Fall sein, obwohl diese auf den ganzen Zahlen aufbauen:

**i Beispiel:** In  $\mathbb{Z}_4$  sind die Elemente  $[1]_4$  und  $[3]_4 = [-1]_4$  invertierbar, da  $[1]_4 \cdot [1]_4 = [1]_4$  sowie  $[3]_4 \cdot [3]_4 = [1]_4$  gilt. Das Element  $[2]_4$  ist nicht invertierbar, da es mit keiner anderen Restklasse multipliziert  $[1]_4$  ergibt, wie in der Multiplikationstabelle zu sehen ist.

In  $\mathbb{Z}_5$  sind die Elemente  $[1]_5, [2]_5, [3]_5, [4]_5$  invertierbar, also alle Restklassen außer  $[0]_5$ , da gilt

$$[1]_5 = [1]_5 \cdot [1]_5 = [2]_5 \cdot [3]_5 = [4]_5 \cdot [4]_5.$$

Diese Erkenntnis aus dem Beispiel lässt sich verallgemeinern: Sei  $n \in \mathbb{N}$  und  $[a]_n \in \mathbb{Z}_n$ . Dann ist  $[a]_n$  invertierbar genau dann, wenn  $\text{ggT}(a, n) = 1$  gilt (also, wenn  $a$  zu  $n$  teilerfremd ist).

**Beweis:** Da der Satz eine Äquivalenz darstellt, sind zwei Richtungen zu zeigen.

Sei zunächst  $[a]_n$  invertierbar. Es existiert also eine Restklasse  $[b]_n \in \mathbb{Z}_n$  mit  $[a]_n \cdot [b]_n = [1]_n$ . Anders formuliert heißt das, dass  $ab \equiv 1 \pmod{n}$  gilt und somit per Definition  $n \mid ab - 1$ . Wegen der Definition der Teilbarkeit existiert ein  $r \in \mathbb{Z}$  mit  $rn = ab - 1$ . Sei nun weiter  $d \in \mathbb{Z}$  ein gemeinsamer Teiler von  $a$  und  $n$ . Weil dieser Teiler nun sowohl  $rn$  als auch den Minuenden  $ab$  teilt, muss er auch den Subtrahenden  $-1$  teilen. Damit sind  $d = \pm 1$  die einzig möglichen gemeinsamen Teiler von  $a$  und  $n$ . Da der  $\text{ggT}$  eine natürliche Zahl ist, ist somit  $\text{ggT}(a, n) = 1$ .

Sei nun  $\text{ggT}(a, n) = 1$  gegeben. Nach dem Lemma von Bézout existieren  $s, t \in \mathbb{Z}$  mit  $1 = \text{ggT}(a, n) = sa + tn$ . Betrachten wir diese Gleichung modulo  $n$  erhalten wir  $1 = sa + tn \equiv sa \pmod{n}$  bzw. als Restklassen:  $[1]_n = [sa]_n = [s]_n \cdot [a]_n$ . Damit ist  $[a]_n$  invertierbar und die Restklasse  $[s]_n$  das Inverse zu  $[a]_n$ .

Der Beweis dieser Aussage liefert uns auch direkt ein Verfahren, um das Inverse einer Restklasse zu bestimmen. Wir müssen „lediglich“ den Bézout-Koeffizienten  $s$  bestimmen, welchen wir mittels des erweiterten euklidischen Algorithmus ermitteln können.



**i Beispiel:** Gegeben sei die Restklasse  $[9]_{14}$  (in  $\mathbb{Z}_{14}$ ). Wegen  $ggT(9,14) = 1$  existiert das Inverse zu  $[9]_{14}$ . Dieses wollen wir mittels des erweiterten euklidischen Algorithmus und dem Lemma von Bézout ermitteln:

- I.  $14 = 1 \cdot 9 + 5$
- II.  $9 = 1 \cdot 5 + 4$
- III.  $5 = 1 \cdot 4 + 1$
- IV.  $4 = 4 \cdot 1 + 0$

Also ist  $1 \stackrel{\text{III.}}{\equiv} 5 - 1 \cdot 4 \stackrel{\text{II.}}{\equiv} 5 - 1 \cdot (9 - 1 \cdot 5) \stackrel{\text{I.}}{\equiv} (14 - 1 \cdot 9) - 1 \cdot (9 - 1 \cdot (14 - 1 \cdot 9)) = 2 \cdot 14 - 3 \cdot 9$

Somit ist  $[-3]_{14} = [11]_{14}$  die inverse Restklasse zu  $[9]_{14}$ . Die Probe zeigt auch  $[9]_{14} \cdot [11]_{14} = [99]_{14} = [1]_{14}$ .

Des Weiteren kann im Falle der Existenz eines Inversen (einer Restklasse), diese auch mittels der sogenannten *Eulersche Phi-Funktion*  $\varphi$  bestimmt werden. Diese gibt die Anzahl an Zahlen zu einer natürlichen Zahl  $n$  an, die kleiner gleich  $n$  **und** teilerfremd zu  $n$  sind.

Formal ist  $\varphi: \mathbb{N} \rightarrow \mathbb{N}, n \mapsto |\{a \in \mathbb{N} | 1 \leq a \leq n \wedge ggT(a, n) = 1\}|$ .

**i Beispiel:** Die Zahl 6 besitzt genau zwei Zahlen, die kleiner gleich 6 sind und teilerfremd zu 6 sind, nämlich 1 und 5. Daher ist  $\varphi(6) = 2$ .

Die Zahl 7 ist eine Primzahl und daher zu allen Zahlen kleiner als 7 teilerfremd (zu sich selbst natürlich nicht). Daher ist  $\varphi(7) = 6$ .

### Satz von Euler-Fermat:

Sei  $n \in \mathbb{N}$ . Für alle  $a \in \mathbb{Z}$  mit  $ggT(a, n) = 1$  gilt

$$a^{\varphi(n)} \equiv 1 \pmod{n}.$$

Wir können damit also relativ einfach das Inverse zur Zahl  $a$  modulo  $n$  bestimmen. Es handelt sich um  $a^{\varphi(n)-1}$ , da  $a^{\varphi(n)-1} \cdot a = a^{\varphi(n)} \equiv 1 \pmod{n}$  gilt. Um mit Hilfe dieses Satzes aber gezielt Inverse bestimmen zu können, ist es noch notwendig  $\varphi(n)$  für  $n \in \mathbb{N}$  berechnen zu können. Aus der Definition lassen sich folgende Eigenschaften schlussfolgern<sup>5</sup>:

- Ist  $n$  eine Primzahl, dann besitzt diese nur 1 und sich selbst als Teiler. Daher gilt für alle Primzahlen  $p$ :  $\varphi(p) = p - 1$
- Für zwei teilerfremde Zahlen  $m$  und  $n$  gilt:  $\varphi(m \cdot n) = \varphi(m) \cdot \varphi(n)$
- Ist  $p$  eine Primzahl und  $k \in \mathbb{N}$ , so ist  $\varphi(p^k) = p^k - p^{k-1} = p^{k-1}(p - 1)$ .

Mit Hilfe dieser Eigenschaften lässt sich nun für jede natürliche Zahl  $n$  der Wert  $\varphi(n)$  auf Grundlage seiner Primfaktorzerlegung berechnen.

**i Beispiel:** Sei die Zahl  $n = 1000 = 10^3 = 2^3 \cdot 5^3$  gegeben. Dann ist

$$\varphi(1000) = \varphi(2^3 \cdot 5^3) = \varphi(2^3) \cdot \varphi(5^3) = 2^{3-1}(2 - 1) \cdot 5^{3-1}(5 - 1) = 400.$$

Demnach existieren insgesamt 400 natürliche Zahlen, die kleiner als 1000 und zu ihr teilerfremd sind.

<sup>5</sup> Beweisidee der letzten Eigenschaft: Zähle die durch  $p$  teilbaren Zahlen und subtrahiere sie von  $p^k$ .

Ein Sonderfall des Satzes von Euler-Fermat ergibt sich, wenn der Modul eine Primzahl ist.

### Kleiner Satz von Fermat<sup>6</sup>:

Sei  $p$  eine Primzahl. Für alle  $a \in \mathbb{Z}$  mit  $\text{ggT}(a, p) = 1$  (d.h. nicht Vielfaches von  $p$ ) gilt

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

Beweis: Der Beweis nutzt geschickt eine Menge von Vielfachen  $\{1a, 2a, \dots, (p-1)a\}$  und gliedert sich in mehrere Teile:

- Die Zahlen  $1a, 2a, \dots, (p-1)a$  sind je nicht kongruent („inkongruent“) zu 0. Denn jede dieser Zahlen lässt sich schreiben als  $ia$  mit  $0 < i < p$ . Da aufgrund von  $0 < i < p$  sowie  $\text{ggT}(a, p) = 1$  keiner der Faktoren kongruent zu null ist<sup>7</sup>, ist auch das Produkt inkongruent zu 0.
- Die Zahlen aus  $\{1a, 2a, \dots, (p-1)a\}$  sind je paarweise inkongruent, denn angenommen es würde  $ia \equiv ja \pmod{p}$  für  $0 < i, j < p$  gelten. Dann wäre  $0 \equiv ia - ja = (i-j)a \pmod{p}$  und somit  $i = j$  wegen  $\text{ggT}(a, p) = 1$  und  $0 < i, j < p$ .
- Damit liegen die  $p-1$ -vielen Zahlen der Menge  $\{1a, 2a, \dots, (p-1)a\}$  paarweise verschieden in den  $p-1$ -vielen Restklassen  $[1]_p, \dots, [p-1]_p$ . Wir können damit schreiben:

$$\begin{aligned} (1a) \cdot \dots \cdot (p-1)a &\equiv 1 \cdot \dots \cdot (p-1) \pmod{p} \\ \Rightarrow (p-1)! \cdot a^{p-1} &\equiv (p-1)! \pmod{p} \end{aligned}$$

Da  $p$  eine Primzahl ist und daher kein Teiler von  $(p-1)!$  ist, darf gekürzt werden zu  $a^{p-1} \equiv 1 \pmod{p}$ .

<sup>6</sup> Mit Hilfe dieses Satzes lassen sich auch große Primzahlen finden: Vermutet man, dass eine (große) Zahl eine Primzahl ist, dann muss auch diese Zahl die obige Bedingung erfüllen. Erfüllt sie das nicht, ist sie sicherlich keine Primzahl. Erfüllt sie die Eigenschaft, so könnte sie prim sein – muss es aber nicht. Je mehr verschiedene  $a$  getestet werden, desto höher ist die Wahrscheinlichkeit, dass es sich um eine Primzahl handelt (Fermatscher Primzahltest).

<sup>7</sup> Wäre  $a = 0$ , so wäre  $\text{ggT}(a, p) = p$  (vgl. Definition der Teilbarkeit).