Primzahlen und RSA-Verschlüsselung

Michael Fütterer und Jonathan Zachhuber

1 Einiges zu Primzahlen

Ein paar **Definitionen:**

Wir bezeichnen mit $\mathbb{Z}$ die Menge der positiven und negativen *ganzen Zahlen*, also

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

und mit $\mathbb{N}$ die *natürlichen Zahlen*, also

$$\mathbb{N} := \{1, 2, 3, \dots\}.$$

Seien nun a und b zwei ganze Zahlen, also $a, b \in \mathbb{Z}$. Dann sagen wir *a ist ein Teiler von b*, wenn wir ein $k \in \mathbb{Z}$ finden, so dass

$$b = k \cdot a.$$

Wir schreiben dann auch $a \mid b$. Wir nennen a einen *echten Teiler von b*, wenn $a \mid b$ und weder $a = 1$ oder $a = b$ gilt. Wir bezeichnen zwei Zahlen a und b also *teilerfremd*, wenn 1 ihr einziger gemeinsammer Teiler ist, also für alle $l \in \mathbb{Z}$ gilt:

$$l \mid a \text{ und } l \mid b \Rightarrow l = 1.$$

Eine Zahl $p \in \mathbb{Z}$ nennen wir *Primzahl*, wenn sie keine echten Teiler besitzt. Die Menge aller Primzahlen bezeichnen wir mit $\mathbb{P}$.

Seien $a, b, k \in \mathbb{Z}$. Dann sagen wir a sei *kongruent b modulo k*, wenn wir ein $l \in \mathbb{Z}$ finden, so dass

$$a - b = k \cdot l,$$

also $k \mid a - b$. Wir schreiben hierfür auch

$$a \equiv b \bmod k.$$

Anschaulich heißt das gerade, dass a und b bei Division durch k den selben Rest lassen. Zum Beispiel sind alle geraden Zahlen kongruent 0 modulo 2 und alle ungeraden kongruent 1 modulo 2.

Sei $n \in \mathbb{N}$. Dann bezeichnen wir mit $\varphi(n)$ die *Anzahl der Zahlen zwischen 0 und n, die teilerfremd zu n sind.* Zum Beispiel gilt für jede Primzahl p:

$$\varphi(p) = p - 1,$$

da alle natürlichen Zahlen zwischen 0 und p zu p teilerfremd sind. Das sind genau $p - 1$ Stück.

Satz 1 (Fundamentalsatz der Arithmetik): Jede natürliche Zahl (außer 1) besitzt eine eindeutige Zerlegung in Primfaktoren, also: Sei $1 \neq d \in \mathbb{N}$. Dann gibt es eindeutig bestimmte $p_1, \ldots, p_n \in \mathbb{P}$ (für ein $n \in \mathbb{N}$) mit $p_1 \leq \cdots \leq p_n$ und

$$d = p_1 \cdots p_n.$$

Beweis: Wir zeigen erst, dass jedes d eine solche Darstellung besitzt. Wir betrachten dazu die Menge der echten Teiler von d. Entweder diese ist leer, dann sind wir fertig, denn dann ist d selbst schon prim. Wenn nicht, dann finden wir in ihr ein kleinstes Element p (denn die Menge besteht ja nur aus endlich vielen natürlichen Zahlen). p ist prim, denn hätte p einen echten Teiler, so wäre er kleiner als p und auch ein Teiler von d und p wäre gar nicht der kleinste Teiler gewesen! Also nennen wir p ab jetzt p_1 und betrachten nun $\frac{d}{p}$ (das ist wieder eine ganze Zahl, denn p war ja ein Teiler von d). Durch identisches Vorgehen finden wir so p_2 und alle weiteren Primteiler (da die Zahlen beim Teilen immer kleiner werden, geht das nur endlich oft: Irgendwann erhalten wir auf jeden Fall eine Primzahl nach dem Teilen und sind fertig!).

Die Eindeutigkeit ist etwas schwieriger zu zeigen. Wir nehmen hierzu an, es gäbe zwei verschiedene Darstellungen, also Primzahlen $p_1, \ldots, p_n$ und $q_1, \ldots, q_m$ (für natürliche Zahlen n und m), so dass

$$p_1 \cdots p_n = d = q_1 \cdots q_m$$

gilt. Ohne Einschränkung können wir $m \geq n$ verlangen. Wenn m also 0 oder 1 ist, ist $m = n$ und d ist entweder 1 oder q_1 und damit sind auch die p's festgelegt und die Darstellung ist eindeutig. Andernfalls stellen wir fest, dass q_1 das Produkt $p_1 \cdots p_n$ teilt und da die p's alle Primzahlen sind, muss eine von ihnen schon q_1 sein (ohne Einschränkung können wir hier wieder p_1 verlangen, sonst sortieren wir die p's einfach um!). Damit gilt also $p_1 = q_1$, also können wir diesen Faktor auf beiden Seiten „wegkürzen“ und das Ganze mit q_2 wiederholen. Wenn wir immer so weiter machen, stellen wir fest, dass tatsächlich $m = n$ und $p_i = q_i$ für alle $i \in \{1, \ldots, m\}$ gilt, die Darstellung also eindeutig ist. □

Bemerkung: Für $d = 1$ findet man keinen Primteiler, das ist dann aber auch eindeutig.

Beobachtung: So eine Primfaktorzerlegung *existiert* immer, ist aber gerade für *große* Zahlen d sehr aufwendig zu bestimmen!

Satz 2 (Euklid): Es gibt unendlich viele Primzahlen.

Beweis:

□

Satz 3 (Euklidischer Algorithmus): Für zwei *teilerfremde* Zahlen $a, b \in \mathbb{Z}$ finden wir zwei Zahlen $k, l \in \mathbb{Z}$, so dass $ak + bl = 1$.

Beispiel: Wir führen das Beispielhaft an 1024 und 753 vor. Wir stellen fest:

$$
\begin{aligned}
1024 &= 1 \cdot 753 + 271 \\
753 &= 2 \cdot 271 + 211 \\
271 &= 1 \cdot 211 + 60 \\
211 &= 3 \cdot 60 + 31 \\
60 &= 1 \cdot 31 + 29 \\
31 &= 1 \cdot 29 + 2 \\
29 &= 14 \cdot 2 + 1 \\
2 &= 2 \cdot 1 + 0
\end{aligned}
$$

Diese Gleichungen können wir nun wieder ineinander einsetzen und erhalten so:

$$\begin{aligned}
1 &= 29 - 14 \cdot 2 \\
&= 29 - 14 \cdot (31 - 1 \cdot 29) = 15 \cdot 29 - 14 \cdot 31 \\
&= 15 \cdot (60 - 1 \cdot 31) - 14 \cdot 31 = 15 \cdot 60 - 29 \cdot 31 \\
&= 15 \cdot 60 - 29 \cdot (211 - 3 \cdot 60) = 102 \cdot 60 - 29 \cdot 211 \\
&= 102 \cdot (271 - 1 \cdot 211) - 29 \cdot 211 = 102 \cdot 271 - 131 \cdot 211 \\
&= 102 \cdot 271 - 131 \cdot (753 - 2 \cdot 271) = 364 \cdot 271 - 131 \cdot 753 \\
&= 364 \cdot (1024 - 1 \cdot 753) - 131 \cdot 753 = 364 \cdot 1024 - 495 \cdot 753
\end{aligned}$$

Also erhalten wir

$$1 = 364 \cdot 1024 - 495 \cdot 753$$

was man vielleicht doch nicht auf den ersten Blick sieht.

Bemerkung: Diese Darstellung ist im Allgemeinen nicht eindeutig. Zum Beispiel gilt $1 = 3 - 2 = 3 \cdot 3 - 4 \cdot 2$.

Der Beweis *wieso* dieser Algorithmus funktioniert (also wieso wir immer nach *endlich* vielen Schritten so eine Darstellung erhalten) ist recht technisch, daher wollen wir ihn hier nicht ausführen. Er basiert aber grundlegend auf der Beobachtung, dass wir für zwei Zahlen $a, b \in \mathbb{Z}$ eine *eindeutige* Zahl r mit $0 \leq r \leq b$ und eine weitere Zahl $t \in \mathbb{Z}$ finden, so dass

$$a = k \cdot b + r$$

gilt (also Division mit Rest). Dann müssen wir nur noch das, was wir in dem Beispiel getan haben, etwas allgemeiner formulieren und sehen, dass unser r in jedem Schritt kleiner wird und folglich nach endlich vielen Schritten tatsächlich 0 sein wird.

2 Assymetrische Verschlüsselungsverfahren

Bei symmetrischen Verschlüsselungsverfahren gibt es nur einen Schlüssel, der beiden Partnern bekannt sein muss, der also zuvor ausgetauscht werden muss und dann für sowohl Ver- als auch Entschlüsseln benutzt wird. Assymetrische Verschlüsselungsverfahren hingegen beruhen im Gegensatz dazu nicht auf dem Austausch eines gemeinsamen Schlüssels. Stattdessen besitzt jeder Teilnehmer einen *privaten Schlüssel,* der nur ihm bekannt ist und einen *öffentlichen Schlüssel,* der „allen“ bekannt ist. Was mit einem der beiden verschlüsselt wurde, kann nur mit dem anderen entschlüsselt werden. So kann man mit dem privaten Schlüssel Nachrichten, die mit dem öffentlichen Schlüssel verschlüsselt wurden als einziger lesen. Zusätzlich kann man Nachrichten sogar mit dem privaten Schlüssel *signieren,* um zu beweisen, dass sie von einem selbst stammen (diese können dann nur mit Hilfe des zugehörigen öffentlichen Schlüssels entschlüsselt werden).

Praktisch lässt sich dies durch assymetrische mathematische Verfahren realisieren, also durch Berechnungen, die in eine Richtung leicht gehen aber schwierig „rückgängig“ zu machen sind. Ein Beispiel hierfür ist die Zerlegung einer Zahl in ihre Primfaktoren (das ist sehr aufwendig im Vergleich zum Multiplizieren der einzelnen Faktoren).

Das RSA-Verfahren anhand eines Beispiels:

Wir fangen an, indem wir uns zwei Primzahlen wählen. Im Allgemeinen sollten diese möglichst groß sein (das Produkt sollte von der Größenordnung ca. 600 Stellen sein), aber damit das nicht zu aufwendig wird, nehmen wir mal $p = 17$ und $q = 5$. Dann berechnen wir zuerst

$$N = p \cdot q = 17 \cdot 5 = 85$$

und $\varphi(N) = (p-1) \cdot (q-1) = 16 \cdot 4 = 64$.

Als nächstes wählen wir eine Zahl e, die teilerfremd zu $\varphi(N) = 64$ ist, z.B. $e = 13$. N und e bilden nun gemeinsam unseren öffentlichen Schlüssel.

Auf diese beiden teilerfremden Zahlen können wir nun den Euklidischen Algorithmus anwenden und erhalten so:

$$\begin{aligned} 64 &= 4 \cdot 13 \quad + \quad 12 \\ 13 &= 1 \cdot 12 \quad + \quad 1 \\ 12 &= 12 \cdot 1 + 0 \end{aligned}$$

Wie vorhin erhalten wir:

$$\begin{aligned} 1 &= 13 - 1 \cdot 12 \\ &= 13 - 1 \cdot (64 - 4 \cdot 13) \\ &= 5 \cdot 13 - 1 \cdot 64 \end{aligned}$$

Der Faktor vor $e = 13$ ist unser privater Schlüssel und wir nennen ihn $d = 5$.

Jetzt machen wir uns an das eigentliche Verschlüsseln: Dazu brauchen wir erst eine Nachricht. Angenommen, wir möchten den Text „X“ verschicken, dann müssen wir ihm zu allererst eine Zahl zuordnen, damit wir damit „rechnen“ können. Zum Beispiel können wir einfach die Buchstaben durch ihre Stellen im Alphabet ersetzen: Als Zahl lautet unsere Nachricht also $X = 24$.

Um die Nachricht nun zu verschlüsseln, berechnen wir den Rest von X^e bei Division durch N, also in unserem Fall:

$$24^{13} \equiv 74 \mod 85.$$

Dieser Rest, also 74, ist der Geheimtext.

Der Empfänger kann nun mit Hilfe des geheimen Schlüssels $d = 5$ die Nachricht wieder entschlüsseln. Das geht genauso wie das Verschlüsseln und wir sehen tatsächlich:

$$74^5 \equiv 24 \mod 85,$$

was ja unsere ursprüngliche Nachricht war (24).

Im Moment können wir leider allerhöchstens 85 verschiedene Nachrichten verschicken (denn mehr Reste bei Division durch 85 gibt es nicht!). Aber mit größeren Primzahlen p und q wird auch N größer und das Problem erübrigt sich.

Dass das Verfahren *immer* klappt und wir tatsächlich mit dem beschriebenen Verfahren immer *funktionierende* Schlüssel erhalten, sieht man mit Hilfe des sogenannten *kleinen Satzes von Fermat.*

Diesen Satz können wir an dieser Stelle leider nicht beweisen. Man muss sich dazu zuerst noch ein wenig intensiver mit der Division durch Primzahlen und den dabei auftretenden Resten beschäftigen. Dazu fehlt uns hier leider die Zeit.

Wenn ihr mehr zu dem Thema erfahren wollt, guckt doch mal im Internet, z.B. auf `http://matheprisma.de/Module/RSA/index.htm`.