Modulares Rechnen

Werner M. Seiler
Universität Kassel

3. Dezember 2013

1 Einführung

Das Rechnen mit ganzen Zahlen ist an vielen Stellen in der Mathematik, aber auch bei vielen Anwendungen von großer Bedeutung. Ein wichtiger Aspekt (insbesondere im Kontext der Computeralgebra) ist dabei, daß wir mit ganzen Zahlen *exakt* rechnen können, d.h. es treten keine Rundungsfehler auf. Ein Nachteil liegt allerdings darin, daß die auftretenden Zahlen immer größer werden: wenn wir zwei n-stellige Zahlen miteinander multiplizieren, so besitzt das Ergebnis in der Regel $2n$ Stellen. Im Extremfall können die Zahlen bei einer längeren Rechnung dann so groß werden, daß sie nicht mehr innerhalb des verfügbaren Speicherplatzes dargestellt werden können. Aber selbst wenn dies nicht passiert, ist es offensichtlich, daß die Ausführung arithmetischer Operationen wie Additionen oder Multiplikationen immer teurer wird, je länger die auftretenden Zahlen werden.

Das modulare Rechnen bietet einen Ausweg aus diesem Problem und ist von fundamentaler Bedeutung für den Entwurf effizienter Algorithmen in der Computeralgebra. Hier wird durchgehend in einem festen und vor allem *endlichen* Zahlenbereich gearbeitet, so daß arithmetische Operationen immer denselben Aufwand benötigen und es nie zu einem Überlauf kommen kann. Die modulare Arithmetik besitzt aber auch zahlreiche andere Anwendungen, vor allem in der Kryptographie und der Kodierungstheorie.

In diesem kurzem Skript werden die folgenden Themen behandelt. Zunächst wiederholen wir die schon aus der Grundschule bekannte Division mit Rest für ganze Zahlen, da sie die Grundlage des gesamten modularen Rechnens bildet. Dann führen wir mittels der bei der Division auftretenden Reste einen neuen Zahlenbereich ein und definieren darauf eine Addition und eine Multiplikation. Es

stellt sich heraus, daß das Rechnen in diesem neuen Bereich in vielem der vertrauten Arithmetik in $\mathbb{Z}$ sehr ähnlich ist, daß aber auch einige neue Phänomene auftreten: z.B. ist es viel häufiger möglich zu dividieren. Um die Divisionen aber effektiv durchführen zu können, benötigen wir das ebenfalls schon aus der Schule bekannte Konzept eines größten gemeinsamen Teilers sowie den (Erweiterten) Euklidischen Algorithmus zu dessen Berechnung.

Wenn die modulare Arithmetik zur Beschleunigung ganzzahliger Rechnungen eingesetzt wird, dann müssen wir zum Schluß auch wieder das gesuchte ganzzahlige Ergebnis aus den modularen Ergebnissen rekonstruieren. Hierzu ist der Chinesische Restesatz ein unverzichtbares Werkzeug. Zum Abschluß behandeln wir noch den sogenannten Kleinen Satz von Fermat, der unter anderem in der Kryptographie benötigt wird.

Drei Anhänge enthalten noch etwas ergänzendes Material. Der erste führt den Begriff einer Äquivalenzrelation ein, der eine mathematisch befriedigendere Definition der modularen Rechenbereiche ermöglicht. Der zweite verallgemeinert die Frage der Inversenberechnung zum Lösen beliebiger linearer Gleichungen in modularen Rechenbereichen. Der dritte behandelt eine iterative Formulierung des (Erweiterten) Euklidischen Algorithmus.

2 Division mit Rest

Eine schon aus der Grundschule bekannte Beobachtung ist, daß die Division ganzer Zahlen in der Regel nicht aufgeht. Wenn man nicht gleich die rationalen Zahlen einführen will, muß man eine Division mit Rest durchführen. Aus mathematischer Sicht liegt dieser der folgende Satz zugrunde.

Satz 1. *Seien $x \in \mathbb{N}_0$ und $y \in \mathbb{N}$[1] zwei natürliche Zahlen. Dann existieren zwei eindeutig bestimmte natürliche Zahlen $q, r \in \mathbb{N}_0$, so daß gilt $x = qy + r$ und $0 \leq r < y$. Wir scheiben $r = x \bmod y$ (gesprochen x modulo y) bzw. $q = x \operatorname{div} y$.*

Beweis. Wir beweisen zunächst die *Existenz* eines Paares $q, r \in \mathbb{N}_0$ mit den gewünschten Eigenschaften. Falls $x < y$, so ist offensichtlich $q = 0$ und $r = x$ eine Lösung. Wenn $x \geq y$, dann setzen wir $x_1 = x - y$ und betrachten dasselbe Problem für x_1, y. Nehmen wir an, daß es Zahlen $q_1, r_1 \in \mathbb{N}_0$ gibt mit $x_1 = q_1 y + r_1$ und $0 \leq r_1 < y$. Offensichtlich ist dann $q = q_1 + 1$ und $r = r_1$ eine

[1] Hier und im folgenden gilt stets, daß $0 \notin \mathbb{N}$; 1 ist die kleinste natürliche Zahl. Wollen wir die Null hinzunehmen, so schreiben wir $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$.

Lösung unseres Ausgangsproblems. Falls $x_1 < y$, können wir natürlich $q_1 = 0$ und $r_1 = x_1$ nehmen. Andernfalls benutzen wir dieselbe Idee noch einmal: wir setzen $x_2 = x_1 - y$ und hoffen, daß es Zahlen $q_2, r_2 \in \mathbb{N}_0$ gibt mit $x_2 = q_2 y + r_2$ und $0 \leq r_2 < y$. Wenn das der Fall ist, können wir aus q_2, r_2 leicht erst q_1, r_1 und dann q, r konstruieren.

Wieder gilt, daß wir für $x_2 < y$ sofort eine Lösung angeben können. Wenn aber $x_2 \geq y$, dann iterieren wir und betrachten $x_3 = x_2 - y$, $x_4 = x_3 - y$ usw. Wir sind fertig mit dem Existenzbeweis, wenn wir zeigen können, daß wir irgendwann einen Wert $x_k < y$ erreichen. Dazu bemerken wir, daß offensichtlich gilt $x > x_1 > x_2 > x_3 > \cdots$; wir haben es mit einer streng fallenden Folge nicht negativer Zahlen zu tun, deren Abstand immer $y > 0$ ist. Dies bedeutet aber, daß die Folge nicht beliebig lang werden kann, sondern irgendwann abbrechen muß mit einem Wert $0 \leq x_k < y$. Unser Problem besitzt dann offensichtlich die Lösung $q = k$ und $r = x_k$.

Zum Schluß zeigen wir noch die *Eindeutigkeit* der oben konstruierten Lösung. Dazu nehmen wir zunächst an, daß es zwei verschiedene Lösungen gäbe, also Zahlen $q_1, q_2, r_1, r_2 \in \mathbb{N}_0$, so daß gilt

$$x = q_1 y + r_1 = q_2 y + r_2 \qquad \text{und} \qquad 0 \leq r_1, r_2 < y \; .$$

Elementare Umformungen der linken Gleichung sowie Ausnützen der rechten Ungleichungen führen dann zu den Relationen

$$(q_1 - q_2) y = r_2 - r_1 \qquad \text{und} \qquad -y < r_2 - r_1 < y \; .$$

Die linke Gleichung besagt nun, daß die ganze Zahl $r_2 - r_1$ ein Vielfaches von y ist. Andererseits soll diese Zahl nach der rechten Abschätzung zwischen $-y$ und y liegen (dabei sind die Werte $\pm y$ nicht erlaubt!). Beiden Bedingungen lassen sich gleichzeitig nur erfüllen, wenn $r_2 - r_1 = 0$ und damit auch $q_1 - q_2 = 0$ gilt. Damit können aber unsere beiden Lösungen nicht verschieden sein. □

Bemerkung 2. Wir haben Satz 1 nur für den Fall formuliert, daß x, y nicht negativ sind. Mit minimalen Änderungen erhält man eine Formulierung für beliebige ganze Zahlen $x, y \in \mathbb{Z}$. Dabei muß natürlich stets vorausgesetzt werden, daß $y \neq 0$.

Der Beweis von Satz 1 ist konstruktiv: er führt uns sofort zu einem *Algorithmus* zur Berechnung der Zahlen q, r. Genau genommen, zeigt der Beweis uns sogar zwei verschiedene Möglichkeiten. Der rekursive Algorithmus 1 folgt aus den Überlegungen am Anfang des Beweises: wenn $x > y$, dann dividieren wir $x - y$ durch y und addieren 1 zu dem so erhaltenen q.

Algorithmus 1 Division mit Rest (rekursive Form) `DivRest1`

Eingabe: zwei natürliche Zahlen $x, y \in \mathbb{N}$

Ausgabe: zwei natürliche Zahlen $q, r \in \mathbb{N}_0$ mit $x = qy + r$ und $0 \leq r < y$

1: **if** $x < y$ **then**
2: **return** $(0, x)$
3: **else**
4: $(q, r) \leftarrow$ `DivRest1`$(x - y, y)$
5: **return** $(q + 1, r)$
6: **end if**

Die Überlegungen gegen Ende des Existenzbeweises zeigen, daß auch der iterative Algorithmus 2 korrekt ist: wir ziehen so lange y von x ab, bis wir eine Zahl r kleiner als y erhalten, und zählen dabei die Anzahl q der Subtraktionen.

Algorithmus 2 Division mit Rest (iterative Form) `DivRest2`

Eingabe: zwei natürliche Zahlen $x, y \in \mathbb{N}$

Ausgabe: zwei natürliche Zahlen $q, r \in \mathbb{N}_0$ mit $x = qy + r$ und $0 \leq r < y$

1: $q \leftarrow 0$
2: **while** $x \geq y$ **do**
3: $x \leftarrow x - y; \quad q \leftarrow q + 1$
4: **end while**
5: **return** (q, x)

Bemerkung 3. Ein ganz wesentlicher Aspekt von Satz 1 ist die Eindeutigkeitsaussage: es gibt nur eine mögliche Wahl für q und r. Dies haben wir dadurch erreicht, daß wir verlangt haben, daß r der Abschätzung $0 \leq r < y$ genügt. Ohne diese Forderung wären mit jeder Lösung q, r immer auch $q + 1, r - y$ und $q - 1, r + y$ weitere Lösungen, so daß es unendlich viele Möglichkeiten gäbe.

Eine Variante von Satz 1 erreicht die Eindeutigkeit der Lösung durch die Forderung $-y/2 \leq r \leq y/2$; so daß hier die möglichen Reste symmetrisch um die Null verteilt sind. In *MuPAD* erhält man die Zahlen q und r aus Satz 1 durch die beiden Befehle `x div y` bzw. `x mod y` (alternativ anstelle von letzterem auch `modp(x,y)`). Dies entspricht gerade der oben eingeführten mathematischen Notation $r = x \bmod y$ bzw. $q = x \operatorname{div} y$. Der Befehl `mods(x,y)` realisiert die Variante mit symmetrisch verteilten Resten (es gibt aber keinen eigenen Befehl für das zugehörige q).

3 Modulare Arithmetik

Wir wählen uns nun eine beliebige aber feste natürliche Zahl $m \in \mathbb{N}$ und ersetzen jede ganze Zahl immer durch ihren Rest bei der Division mit m. Man sieht schnell, daß man mit diesen Resten ganz ähnlich rechnen kann, wie man es von den ganzen Zahlen selbst gewohnt ist.

Lemma 4. *Seien $x, y \in \mathbb{Z}$ und $m \in \mathbb{N}$. Dann gilt*

$$\big((x \bmod m) + (y \bmod m)\big) \bmod m = (x + y) \bmod m \,, \tag{1a}$$

$$\big((x \bmod m) \cdot (y \bmod m)\big) \bmod m = (x \cdot y) \bmod m \,. \tag{1b}$$

Beweis. Der Beweis erfolgt durch elementares Nachrechnen. Nach Satz 1 gibt es eindeutig bestimmte Zahlen q_x, q_y, r_x, r_y mit $0 \le r_x, r_y < m$, so daß gilt: $x = q_x m + r_x$ und $y = q_y m + r_y$. Damit ist $x + y = (q_x + q_y)m + (r_x + r_y)$. Wenn wir nun auf beiden Seiten die Reste bei der Division durch m betrachten, so erhalten wir links $(x + y) \bmod m$ und rechts $(r_x + r_y) \bmod m$, da wir Vielfache von m ignorieren dürfen. Der zweite Ausdruck ist aber gerade die linke Seite von (1a) und wir sind fertig mit der ersten Aussage.

Der Beweis der zweiten Aussage geht analog: geschicktes Ausklammern ergibt diesmal $x \cdot y = (q_x q_y m + q_x r_y + q_y r_x)m + r_x \cdot r_y$. Ganzzahlige Division durch m führt nun sofort zu (1b). □

Beispiel 5. Ein solches Rechnen mit Resten ist uns wohl vertraut von Uhrzeiten,[2] wo $m = 24$ gilt. Geht man um 22:00 für 30 Stunden aus dem Haus, dann kommt man um 4:00 zurück. Dieses Ergebnis kann man auf zwei Arten erhalten. Entweder man rechnet gemäß der rechten Seite von (1a) $22+30 = 52$ und $52 \bmod 24 = 4$ oder gemäß der linken Seite zunächst $22 \bmod 24 = 22$ bzw. $30 \bmod 24 = 6$ und dann $(22 + 6) \bmod 24 = 4$. Man beachte, daß die linke Seite zwar komplizierter aussieht, aber daß man dafür hier im allgemeinen mit kleineren Zahlen rechnet

Wenn wir alle Zahlen immer durch ihre Reste modulo m ersetzen, dann macht es nach Lemma 4 offensichtlich keinen Unterschied, ob wir mit der Zahl x oder mit $x + m$ oder aber mit $x - 42m$ arbeiten, da wir in jedem Fall den gleichen Rest erhalten. Deshalb fassen wir alle ganzen Zahlen, die modulo m den gleichen Rest

[2] Ein anderes Beispiel sind Winkel mit $m = 360$.

wie x ergeben, zusammen in der Menge[3]

$$\begin{aligned}[x]_m &= \{y \in \mathbb{Z} : m \text{ teilt } y - x\} \\ &= \{x, x \pm m, x \pm 2m, x \pm 3m, \dots\} \subset \mathbb{Z} \ .\end{aligned} \tag{2}$$

Man nennt $[x]_m$ die *Restklasse modulo* m der Zahl x.

Offensichtlich kann es nur m verschiedene Restklassen modulo m geben entsprechend den m verschiedenen möglichen Werten für den Rest bei der ganzzahligen Division durch m. Allerdings kann man eine Restklasse auf viele verschiedene (genau genommen, unendlich viele) Arten beschreiben:

$$[x]_m = [x \pm m]_m = [x \pm 2m]_m = [x \pm 3m]_m = \cdots \ .$$

Man nennt x den *Vertreter* (oder *Repräsentanten*) der Restklasse $[x]_m$; jedes Element von $[x]_m$ kann als Vertreter gewählt werden. In vielen Fällen ist es aber am natürlichsten, für den Vertreter x einer Restklasse eine Zahl $0 \leq x < m$ zu wählen entsprechend unserer Formulierung von Satz 1. Aber manchmal ist es auch praktisch (oder unvermeidlich) mit anderen Vertretern zu arbeiten.

Wir betrachten nun die Menge

$$\mathbb{Z}_m = \big\{[0]_m, [1]_m, [2]_m, \dots, [m-1]_m\big\}$$

aller verschiedenen Restklassen. Eigentlich sind also die Elemente von $\mathbb{Z}_m$ selbst wieder Mengen. Wir wollen sie jetzt aber als eine neue Art von Zahlen auffassen und eine Addition sowie eine Multiplikation auf der Menge $\mathbb{Z}_m$ einführen. Eine naheliegende Definition besteht darin, diese Operationen einfach aus der bekannten Arithmetik auf $\mathbb{Z}$ abzuleiten:

$$[x]_m + [y]_m = [x + y]_m \ , \qquad [x]_m \cdot [y]_m = [xy]_m \ . \tag{3}$$

(Wir benutzen für die neue Addition und Multiplikation dieselbe Notation wie für die vertrauten Operationen in $\mathbb{Z}$).

Da Restklassen etwas kompliziertere Objekte sind als die „normalen“ Zahlen, müssen wir uns erst einmal überlegen, ob (3) überhaupt Sinn macht. Wir haben ja gerade gesehen, daß wir dieselbe Restklasse durch viele verschiedene Vertreter beschreiben können. Unsere Definition der Arithmetik in (3) hängt aber explizit von den gewählten Vertretern ab. Wir müssen also zeigen, daß für jede Wahl der

[3]Eigentlich arbeitet man hier mit einer sogenannten Äquivalenzrelation. Im Anhang A wird dieser Begriff näher erläutert.

Vertreter immer dasselbe Ergebnis herauskommt: wenn $[x]_m = [\hat{x}]_m$ und $[y]_m = [\hat{y}]_m$ gilt, dann muß auch $[x+y]_m = [\hat{x}+\hat{y}]_m$ und $[xy]_m = [\hat{x}\hat{y}]_m$ gelten.

Dies folgt aber sofort aus einer elementaren Rechnung. Wenn $[x]_m = [\hat{x}]_m$, dann muß es eine ganze Zahl $q_x \in \mathbb{Z}$ geben, so daß $\hat{x} = x + q_x m$. Entsprechend gilt $\hat{y} = y + q_y m$ für ein $q_y \in \mathbb{Z}$. Das bedeutet aber $\hat{x} + \hat{y} = x + y + (q_x + q_y)m$ und damit in der Tat $[x+y]_m = [\hat{x}+\hat{y}]_m$. Genauso folgt $[xy]_m = [\hat{x}\hat{y}]_m$ aus der Gleichung $\hat{x}\hat{y} = xy + (xq_y + yq_x + q_x q_y m)m$. Also sind die in (3) eingeführte Addition und Multiplikation *wohldefiniert*. Scheinbar unterscheiden sich diese neuen arithmetischen Operationen nicht sonderlich von den vertrauten in $\mathbb{Z}$. Wenn man sich aber einige konkrete Rechnungen anschaut, fallen schnell neben vielen Gemeinsamkeiten auch einige Unterschiede auf.

Beispiel 6. In $\mathbb{Z}$ spielen die beiden Zahlen 0 und 1 eine besondere Rolle: wenn wir 0 zu einer Zahl x hinzuaddieren oder aber x mit 1 multiplizieren, dann bekommen wir gerade wieder x als Ergebnis. Man überprüft leicht, daß in $\mathbb{Z}_m$ dasselbe für $[0]_m$ bzw. $[1]_m$ (aber keine anderen Elemente) gilt.

Wenn wir zwei ganze Zahlen $x, y \in \mathbb{Z}$ multiplizieren, dann kann als Ergebnis nur dann Null herauskommen, wenn entweder x oder y selbst Null ist. In $\mathbb{Z}_m$ sieht das manchmal anders aus. So gilt in $\mathbb{Z}_8$, daß $[2]_8, [4]_8 \neq [0]_8$; trotzdem erhalten wir für ihr Produkt $[2]_8 \cdot [4]_8 = [8]_8 = [0]_8$. Solche Zahlen nennt man *Nullteiler*.

In $\mathbb{Z}$ sind die einzigen Zahlen, durch die wir immer teilen können, 1 und -1, da $1/1 = 1$ und $1/(-1) = -1$. Für alle anderen Zahlen $x \in \mathbb{Z} \setminus \{\pm 1\}$ gilt, daß $1/x \notin \mathbb{Z}$, bzw. anders ausgedrückt, daß es keine Zahl $y \in \mathbb{Z}$ gibt mit $xy = 1$. In $\mathbb{Z}_m$ ist es viel häufiger möglich zu teilen. Wenn wir z.B. wieder zu $\mathbb{Z}_8$ gehen, so gilt $[3]_8 \cdot [3]_8 = [9]_8 = [1]_8$; der Kehrwert von $[3]_8$ ist also gerade wieder $[3]_8$! Aber durch $[2]_8$ können wir auch in $\mathbb{Z}_8$ nicht teilen, denn für beliebige $[x]_8$ gilt $[2]_8 \cdot [x]_8 = [2x]_8 \neq [1]_8$, da $2x$ immer gerade ist und damit nie Rest 1 bei Division durch 8 ergeben kann. In $\mathbb{Z}_5$ besitzt dagegen jede Zahl außer $[0]_5$ einen Kehrwert: $[1]_5 \cdot [1]_5 = [1]_5$, $[2]_5 \cdot [3]_5 = [3]_5 \cdot [2]_5 = [1]_5$ und $[4]_5 \cdot [4]_5 = [1]_5$. Damit gehen in $\mathbb{Z}_5$ alle Divisionen auf und wir können rechnen wie mit den rationalen Zahlen.

Bemerkung 7. In *MuPAD* können wir den Rechenbereich $\mathbb{Z}_m$ mit dem „Befehl“ `Dom::IntegerMod` erzeugen und dann auch mit seinen Elementen, also den Restklassen, gemäß der oben besprochenen Regeln rechnen. So erhalten wir $\mathbb{Z}_8$ durch die Anweisung `Z8:=Dom::IntegerMod(8)`. Die Restklasse $[3]_8$ erzeugen wir dann mit `Z8(3)`; *MuPAD* schreibt das Ergebnis in der Form `3 mod 8`. Zum Rechnen können wir die vertrauten Operatoren `+` und `*` benutzen. So liefert die Eingabe `Z8(3)*Z8(3)` als Ausgabe `1 mod 8`.

4 Größter Gemeinsamer Teiler

Die Rechnungen in Beispiel 6 führen natürlich zu der Frage, ob wir auf eine einfache Weise erkennen können, welche Elemente in $\mathbb{Z}_m$ invertierbar sind (und wie wir dann auch den Kehrwert berechnen können). Es stellt sich heraus, daß die Antwort in einer wohlvertrauten Operation liegt, nämlich der Berechnung des größten gemeinsamen Teilers zweier ganzer Zahlen.

Definition 8. Seien $x, y \in \mathbb{Z} \setminus \{0\}$ zwei ganze Zahlen. Wir sagen x *teilt* y, geschrieben $x \mid y$, wenn ein $q \in \mathbb{Z}$ existiert mit $y = qx$. Andernfalls schreiben wir $x \nmid y$. Die Zahl $d \in \mathbb{N}$ ist ein *größter gemeinsamer Teiler* von x, y, geschrieben $d = \mathrm{ggT}(x, y)$, falls (i) $d \mid x$ und $d \mid y$ (d.h.: d ist ein gemeinsamer Teiler der Zahlen x, y) und (ii) für jede andere Zahl $c \in \mathbb{N}$ mit $c \mid x$ und $c \mid y$ gilt $c \mid d$ (d.h.: jeder andere gemeinsame Teiler von x, y ist auch ein Teiler von d).

Bemerkung 9. Die Bedingung (ii) in Definition 8 erscheint etwas ungewöhnlich. Man erwartet eher eine Forderung der Form $c \leq d$; dies legt ja auch die Bezeichnung *größter* gemeinsamer Teiler nahe (man beachte aber, daß aus $c \mid d$ immer folgt $c \leq d$). In der Tat wäre dies auch genauso korrekt und für unsere Zwecke genauso gut. Die obige Formulierung hat den Vorteil, daß sie es vermeidet, die $<$-Relation zu benutzen. Damit kann sie auch auf andere Objekte angewandt werden, bei denen solche Vergleiche nicht möglich sind. Zum Beispiel kann Definition 8 ohne Probleme auch für Polynome in einer Variablen t benutzt werden, wo es keine $<$-Relation gibt.

Satz 10. *Seien $x, y \in \mathbb{Z} \setminus \{0\}$ zwei ganze Zahlen. Dann existiert ein eindeutig bestimmter größter gemeinsamer Teiler $d = \mathrm{ggT}(x, y)$ und es gilt*

$$d = \min \{n \in \mathbb{N} : \textit{es gibt } s, t \in \mathbb{Z} \textit{ so daß } n = sx + ty\} . \tag{4}$$

Beweis. Zuerst zeigen wir, daß es höchstens einen größten gemeinsamen Teiler geben kann (d.h.: wenn es überhaupt eine Zahl $d \in \mathbb{N}$ gibt, die die beiden Bedingungen in Definition 8 erfüllt, dann ist sie eindeutig und wir dürfen von *dem* größten gemeinsamen Teiler sprechen). Die Eindeutigkeit folgt aber sofort aus der Bedingung (ii). Wenn d und d' zwei größte gemeinsame Teiler sind, dann muß einerseits $d \mid d'$ gelten (da d' ein größter gemeinsamer Teiler und d ein weiterer gemeinsamer Teiler ist) und andererseits $d' \mid d$ (da d ein größter gemeinsamer Teiler und d' ein weiterer gemeinsamer Teiler ist). Diese Teilbarkeiten können aber nur dann gleichzeitig gelten, wenn $d = d'$.

Zur Abkürzung bezeichnen wir die auf der rechten Seite von (4) auftretende Menge mit $\mathcal{M}$. Die Menge $\mathcal{M}$ ist sicherlich nicht leer, da mit $\alpha = |x|/x$ bzw. $\beta = |y|/y$ gilt $\alpha x + \beta y = |x| + |y| \in \mathbb{N}$. Wir zeigen nun, daß die Zahl $d = \min \mathcal{M}$ die definierenden Eigenschaften eines größten gemeinsamen Teilers erfüllt.

Nach Satz 1 gibt es eindeutige Zahlen $q, r \in \mathbb{Z}$, so daß $x = qd + r$ und $0 \leq r < d$. Wir zeigen nun, daß hier $r = 0$ und damit $d \mid x$ gelten muß. Dazu formen wir unter Ausnutzung der Definition von d um und erhalten

$$r = x - qd = x - q(sx + ty) = (1 - qs)x - qty$$

für geeignete ganze Zahlen s, t. Da $1 - qs$ und $-qt$ ganze Zahlen sind, bedeutet dies $r \in \mathcal{M}$, falls $r \neq 0$. Nun haben wir d als das Minimum von $\mathcal{M}$ definiert und gleichzeitig soll gelten $r < d$. Dieser Widerspruch löst sich nur auf, wenn $r = 0$. Analog zeigt man, daß $d \mid y$ gilt, d.h. d ist ein gemeinsamer Teiler von x, y.

Sei nun $c \in \mathbb{N}$ ein weiterer gemeinsamer Teiler von x, y, d.h. es gilt $c \mid x$ und $c \mid y$. Dann folgt aber aus $d = sx + ty$ sofort auch $c \mid d$. Also ist d in der Tat ein größter gemeinsamer Teiler von x, y. □

Man beachte, daß im Gegensatz zu unserer Vorgehensweise bei Satz 1 dieser Beweis *nicht* konstruktiv ist. Da die Menge $\mathcal{M}$ unendlich groß ist, können wir sie nicht explizit berechnen, um dann ihr Minimum zu bestimmen. Satz 10 liefert uns also kein Berechnungsverfahren für den größten gemeinsamen Teiler.

Dafür folgt aus Satz 10 eine sehr wichtige Eigenschaft des größten gemeinsamen Teilers, die unsere Frage nach der Existenz von Inversen in $\mathbb{Z}_m$ beantworten wird: es gibt immer ganze Zahlen $s, t \in \mathbb{Z}$ mit $\mathrm{ggT}\,(x, y) = sx + ty$ (man nennt dies auch das *Lemma von Bézout*). Solche Zahlen s, t heißen *Bézout-Koeffizienten* und sind nicht eindeutig bestimmt, da für ein beliebiges $r \in \mathbb{Z}$ stets gilt $sx + ty = (s + ry)x + (t - rx)y$. Umgekehrt kann man auch zeigen, daß wenn s', t' weitere Bézout-Koeffizienten für x, y sind, dann gibt es ein $r \in \mathbb{Z}$ mit $s' - s = ry$ und $t - t' = rx$.

Beispiel 11. Offensichtlich gilt $\mathrm{ggT}(12, 42) = 6$. Bei solch kleinen Zahlen ist es kein Problem mit etwas Herumrechnen mögliche Bézout-Koeffizienten anzugeben: $4 \cdot 12 - 1 \cdot 42 = -3 \cdot 12 + 1 \cdot 42 = 6$. Wir werden im nächsten Abschnitt sehen, wie man sie systematisch berechnen kann.

Definition 12. Zwei ganze Zahlen $x, y \in \mathbb{Z} \setminus \{0\}$ heißen *teilerfremd* oder *relativ prim*, wenn $\mathrm{ggT}\,(x, y) = 1$.

Diese Begriffsbildung ist ziemlich natürlich, da 1 jede ganze Zahl teilt und wenn $\mathrm{ggT}\,(x, y) = 1$ gilt, so gibt es offensichtlich außer diesem trivialen Teiler keine weiteren gemeinsame Teiler von x, y. Teilerfremde Zahlen kann man nun analog zu Satz 10 charakterisieren.

Korollar 13. *Zwei ganze Zahlen $x, y \in \mathbb{Z} \setminus \{0\}$ sind genau dann teilerfremd, wenn es Zahlen $s, t \in \mathbb{Z}$ gibt mit $sx + ty = 1$.*

Beweis. Nehmen wir zunächst an, daß x, y teilerfremd sind. Nach Definition gilt dann $\mathrm{ggT}\,(x, y) = 1$ und die Existenz von s, t folgt nun sofort aus Satz 10.

Umgekehrt gelte $sx + ty = 1$ für ganze Zahlen $s, t \in \mathbb{Z}$. Dann folgt in der Notation des Beweises von Satz 10, daß $1 \in \mathcal{M}$. Da aber $\mathcal{M} \subseteq \mathbb{N}$, muß 1 offensichtlich auch das Minimum von $\mathcal{M}$ sein und damit nach Satz 10 der größte gemeinsame Teiler von x, y. □

Nach diesen Vorarbeiten können wir jetzt endlich sagen, für welche Restklassen $[k]_m \in \mathbb{Z}_m$ eine inverse Restklasse $[\ell]_m = [k]_m^{-1}$ mit $[k]_m \cdot [\ell]_m = [1]_m$ existiert.

Satz 14. *Sei $m \in \mathbb{N}$ und $k \in \mathbb{Z} \setminus \{0\}$. Die Restklasse $[k]_m$ besitzt genau dann ein Inverses in $\mathbb{Z}_m$, wenn k und m teilerfremd sind.*

Beweis. Nehmen wir zunächst an, daß k und m teilerfremd sind. Dann existieren nach Korollar 13 ganze Zahlen $s, t \in \mathbb{Z}$ mit $sk + tm = 1$. Ganzzahlige Division durch m ergibt nun

$$[s]_m \cdot [k]_m + [t]_m \cdot [m]_m = [sk + tm]_m = [1]_m$$

Da $[m]_m = [0]_m$ in $\mathbb{Z}_m$, bedeutet dies $[s]_m \cdot [k]_m = [1]_m$, d.h. die Restklasse $[s]_m$ ist das gesuchte Inverse zu $[k]_m$.

Umgekehrt existiere ein inverses Element $[s]_m$ zu der Restklasse $[k]_m$, d.h. es gelte $[s]_m \cdot [k]_m = [sk]_m = [1]_m$. Diese Gleichung bedeutet nach der Definition der Restklassen modulo m, daß es eine ganze Zahl $t \in \mathbb{Z}$ gibt mit $sk + tm = 1$. Damit folgt aus Korollar 13, daß die beiden Zahlen k, m teilerfremd sind. □

Bemerkung 15. Man beachte, daß obiger Beweis uns sogar konkret sagt, wie man das Inverse $[k]_m^{-1}$ berechnet, wenn es existiert: wir müssen lediglich Bézout-Koeffizienten $s, t \in \mathbb{Z}$ zu k, m finden; es gilt dann $[k]_m^{-1} = [s]_m$. Die Tatsache, daß Bézout-Koeffizienten nicht eindeutig sind, bereitet hier kein Probleme. Wie wir oben gesehen haben, ist jedes andere Paar $s', t' \in \mathbb{Z}$ von Bézout-Koeffizienten von der Form $s' = s + rm$ und $t' = t - rk$ mit einer ganzen Zahl $r \in \mathbb{Z}$. Damit gilt aber in $\mathbb{Z}_m$, daß $[s]_m = [s']_m$.

Bemerkung 16. Eine Zahl $p \in \mathbb{N} \setminus \{1\}$ heißt bekanntlich *Primzahl*, wenn sie nur durch 1 und sich selbst teilbar ist; die Menge alle Primzahlen wird mit $\mathbb{P}$ bezeichnet. Sei nun $k \in \mathbb{Z}$ eine ganze Zahl, die nicht durch p teilbar ist; offensichtlich gilt dann $\mathrm{ggT}\,(k,p) = 1$. Wenn aber k durch p teilbar ist, dann ist $[k]_p = [0]_p$ in $\mathbb{Z}_p$. Also besitzt in $\mathbb{Z}_p$ jede Restklasse außer $[0]_p$ ein Inverses. In Beispiel 6 haben wir dies für $p = 5$ explizit überprüft.

Wenn aber unser Modulus $m \in \mathbb{N}$ keine Primzahl ist, dann existieren ganze Zahlen $1 < k, \ell < m$ mit $m = k\ell$. Da nun offensichtlich $[k]_m, [\ell]_m \neq [0]_m$ in $\mathbb{Z}_m$ gilt, stellen wir erstens fest, daß es in diesem Fall in $\mathbb{Z}_m$ Nullteiler gibt, denn $[k]_m \cdot [\ell]_m = [m]_m = [0]_m$. Zweitens können zu $[k]_m, [\ell]_m$ nach Satz 14 keine Inverse in $\mathbb{Z}_m$ existieren, da $\mathrm{ggT}\,(k,m) = k$ und entsprechend für ℓ. Dies erklärt die in Beispiel 6 beobachteten Eigenschaften von $[2]_8$.

5 Euklidischer Algorithmus

Wir haben nun die Existenz von Inversen in $\mathbb{Z}_m$ auf den größten gemeinsamen Teiler zurückgeführt. Insbesondere zeigte sich, daß das Inverse direkt aus den Bézout-Koeffizienten folgt. Also stellt sich für einen Computeralgebraiker die Frage, wie man den größten gemeinsamen Teiler und die zugehörigen Bézout-Koeffizienten explizit berechnen kann. Unser Existenzbeweis für den größten gemeinsamen Teiler im letzten Abschnitt war ja nicht konstruktiv.

Einer der fundamentalen Algorithmen der Computeralgebra ist der *Euklidische Algorithmus* zur Berechnung des größten gemeinsamen Teilers. Er beruht auf dem folgenden einfachen Ergebnis.

Lemma 17. *Seien $x, y \in \mathbb{N}$ zwei natürliche Zahlen mit $x \leq y$ und $x \nmid y$. Dann gilt*

$$\mathrm{ggT}\,(x,y) = \mathrm{ggT}\,(y \bmod x, x)\ . \tag{5}$$

Beweis. Wir zeigen sogar mehr, als das Lemma aussagt, nämlich daß *jeder* gemeinsame Teiler von x, y auch ein gemeinsamer Teiler von $\hat{y} = y \bmod x$ und x ist und umgekehrt.

Nehmen wir zunächst an, daß $d \mid x$ und $d \mid y$. Offensichtlich gilt für ein geeignet gewähltes $q \in \mathbb{N}$, daß $y = qx + \hat{y}$. Aber daraus folgt sofort, daß d auch ein Teiler von $\hat{y} = y - qx$ ist. Wenn umgekehrt d ein gemeinsamer Teiler von $\hat{y}, x$ ist, dann folgt wieder aus $y = qx + \hat{y}$, daß d auch y teilt. □

Dieses Ergebnis führt sofort zu dem Euklidischen Algorithmus 3. Wir geben hier nur eine rekursive Formulierung an; man kann aber leicht auch eine iterative Variante entwerfen.

Algorithmus 3 Euklidischer Algorithmus `EA`

Eingabe: zwei natürliche Zahlen $x, y \in \mathbb{N}$ mit $x \leq y$
Ausgabe: $d = \mathrm{ggT}\,(x, y)$

```
1: if x | y then
2:     return x
3: else
4:     return EA(y mod x, x)
5: end if
```

Satz 18. *Der Euklidische Algorithmus 3 ist korrekt und terminiert für alle zulässigen Eingaben x, y.*

Beweis. Die Korrektheit folgt sofort aus Lemma 17. Für die Terminierung stellen wir fest, daß bei jedem rekursiven Aufruf von `EA` die Summe der Argumente echt kleiner wird. Da es sich dabei um eine natürliche Zahl handelt, kann es nicht zu unendlich vielen Aufrufen kommen. □

Beispiel 19. Wir wollen $\mathrm{ggT}\,(35, 126)$ mit dem Euklidischen Algorithmus bestimmen. Dies führt rekursiv zu den folgenden Rechnungen:

$$\begin{aligned}
126 = 3 \cdot 35 + 21 \quad &\Longrightarrow \quad \mathrm{ggT}\,(35, 126) &&= \mathrm{ggT}\,(21, 35) \\
35 = 1 \cdot 21 + 14 \quad &\Longrightarrow \quad \mathrm{ggT}\,(21, 35) &&= \mathrm{ggT}\,(14, 21) \\
21 = 1 \cdot 14 + 7 \quad &\Longrightarrow \quad \mathrm{ggT}\,(14, 21) &&= \mathrm{ggT}\,(7, 14) \\
14 = 2 \cdot 7 \quad &\Longrightarrow \quad \mathrm{ggT}\,(7, 14) &&= 7
\end{aligned}$$

Also gilt $\mathrm{ggT}\,(35, 126) = 7$.

Eigentlich müssten wir nun eine Analyse der *Komplexität* des Algorithmus durchführen. Dazu versucht man, die Anzahl der rekursiven Aufrufe entweder im durchschnittlichen oder im schlimmsten Fall abzuschätzen. Wir verzichten aber hier darauf. Stattdessen überlegen wir uns, wie wir Algorithmus 3 erweitern können, so daß er auch mögliche Bézout-Koeffizienten mitberechnet. Dies führt zu dem *Erweiterten Euklidischen Algorithmus* 4.

Algorithmus 4 Erweiterter Euklidischer Algorithmus `EEA`

Eingabe: $x, y \in \mathbb{N}$ mit $x \leq y$
Ausgabe: $s, t \in \mathbb{Z}$ mit $\mathrm{ggT}\,(x, y) = sx + ty$

1: **if** $x \mid y$ **then**
2: **return** $(1, 0)$
3: **else**
4: $(s', t') \leftarrow \texttt{EEA}(y \texttt{ mod } x, x)$
5: $s \leftarrow t' - s' \cdot (y \texttt{ div } x)$; $t \leftarrow s'$
6: **return** (s, t)
7: **end if**

Satz 20. *Der Erweiterte Euklidische Algorithmus 4 ist korrekt und terminiert für alle zulässigen Eingaben x, y.*

Beweis. Die Terminierung folgt aus demselben Argument wie bei dem einfachen Euklidischen Algorithmus 3. Für die Korrektheit erinnern wir uns daran. daß gilt

$$y = (y \operatorname{div} x) \cdot x + (y \bmod x)\ .$$

Falls wir nun in dem letzten Rekursionsschritt für den größten gemeinsamen Teiler $d = \mathrm{ggT}\,(x, y) = \mathrm{ggT}\,(y \bmod x, x)$ die Darstellung $d = s'(y \bmod x) + t'x$ gefunden haben, so ergibt sich durch Einsetzen die neue Darstellung

$$d = \big(t' - s'(y \operatorname{div} x)\big) \cdot x + s'y\ .$$

Offensichtlich berechnet Algorithmus 4 in Zeile 5 genau diese Koeffizienten. □

Beispiel 21. Wir behandeln wieder dieselben Zahlenwerte wie in Beispiel 19. Die Zwischenergebnisse bei der Anwendung des Erweiterten Euklidischen Algorithmus 4 lassen sich dabei gut in einer Tabelle darstellen.

x	y	s	t	
35	**126**	**-7**	**2**	$(-7 = -1 - 3 \cdot 2)$
21	35	2	-1	$(2 = 1 - 1 \cdot (-1))$
14	21	-1	1	$(-1 = 0 - 1 \cdot 1)$
7	14	1	0	

Unser Endergebnis steht hier gleich in der ersten Zeile:

$$\mathrm{ggT}\,(35, 126) = -7 \cdot 35 + 2 \cdot 126 = 7\ .$$

Im Laufe der Rekursion ergeben sich die Werte in der linken Hälfte der Tabelle von oben nach unten: jede Zeile enthält die Argumente für den nächsten Aufruf. Die rechte Hälfte der Tabelle füllt sich von unten nach oben mit den Ergebnissen des entsprechenden Rekursionsschritts. Gemäß Zeile 5 von Algorithmus 4 ist der zweite Wert immer gerade der erste Wert der darunter stehenden Zeile und der erste Werte ergibt sich aus den beiden vorherigen Werten zusammen mit $y \operatorname{div} x$ (wurde bereits in Beispiel 19 berechnet). Man beachte, daß nicht nur in der ersten, sondern in *jeder* Zeile gilt $sx + ty = \operatorname{ggT}(x, y)$.

Bemerkung 22. Da man auch für verschiedene andere mathematische Objekte wie z.B. Polynome in einer Variablen größte gemeinsame Teiler berechnen kann, gibt es in *MuPAD* verschiedene Befehle dafür. Der uns interessierende Fall ganzer Zahlen wird dabei von dem Befehl `igcd` (nach der englischen Bezeichnung *integer greatest common divisor*) abgedeckt. Der Aufruf `igcd(35,126)` liefert dann auch das erwartete Ergebnis `7`. Wenn man zusätzlich zwei Bézout-Koeffizienten berechnen will, kann man den Befehl `igcdex` benutzen, hinter dem sich der Erweiterte Euklidische Algorithmus verbirgt (`ex` steht hier für *extended*). Das Resultat ist eine Folge von drei Zahlen: der größte gemeinsame Teiler und zwei Bézout-Koeffizienten. So erhält man mit dem Aufruf `igcdex(35,126)` das Ergebnis `7,-7,2`. Schließlich findet man in der Bibliothek `numlib`, die Befehle für verschiedene Probleme der Zahlentheorie enthält, noch den Befehl `numlib::igcdmult`,[4] mit dem auch der größte gemeinsame Teiler von mehr als zwei Zahlen gefunden werden kann. Hierbei handelt es sich ebenfalls um eine „erweiterte“ Form, d.h. es werden neben dem größten gemeinsamen Teiler auch entsprechende Bézout-Koeffizienten berechnet und in Form einer Liste zurückgegeben. So liefert z.B. der Aufruf `numlib::igcdmult(35,126,50)` das Ergebnis $[1, 49, -14, 1]$. Dies ist so zu interpretieren, daß gilt:

$$\operatorname{ggT}(35, 126, 50) = 1 = 49 \cdot 35 - 14 \cdot 126 + 1 \cdot 50 \; .$$

[4]Wenn man häufig Befehle aus einer bestimmten Bibliothek aufruft, dann wird es mit der Zeit lästig, immer wieder den Bibliotheksnamen (hier also `numlib`) einzugeben. Hier schafft die Anweisung `use(numlib)` Abhilfe. Sie „exportiert“ alle Befehle aus der Bibliothek, so daß man anschließen auch einfach `igcdmult(35,126,50)` ohne ein vorheriges `numlib` eingeben kann. Man muß aber aufpassen: wenn man zu viele Bibliotheken exportiert, dann kann es zu Namenskollisionen kommen (in einem solchen Fall gibt *MuPAD* eine Warnung aus und exportiert den entsprechenden Befehl nicht).

6 Chinesischer Restesatz

Eine wichtige Anwendung des modularen Rechnens besteht in der Beschleunigung von Berechnungen mit ganzen Zahlen. Bei einer solchen Berechnung möchte man aber zum Schluß auch wieder das richtige ganzzahlige Ergebnis erhalten und nicht nur seinen Rest bezüglich irgendeines Modulus $m \in \mathbb{N}$. In der Praxis wird man auch mit verschiedenen Moduli $m_1, m_2, \ldots, m_k \in \mathbb{N}$ gerechnet haben und möchte dann aus den verschiedenen Ergebnissen das Endergebnis berechnen. Dieses Problem kann man auch so beschreiben: gegeben Zahlen $\ell_1, \ell_2, \ldots, \ell_k$, suchen wir eine Zahl $x \in \mathbb{Z}$, so daß gilt $x \in [\ell_1]_{m_1}$, $x \in [\ell_2]_{m_2}$,..., $x \in [\ell_k]_{m_k}$ (hier betrachten wir die Restklassen $[\ell_i]_{m_i}$ wieder als Mengen und nicht als Zahlen).

Wie wir gleich sehen werden, besitzt dieses Problem im allgemeinen viele Lösungen (zumindest wenn die gewählten Moduli gewisse Bedingungen erfüllen), aber nur eine davon genügt der Ungleichung $0 \leq x < m = m_1 \cdot m_2 \cdots m_k$. Da sich bereits in den Schriften der alten Chinesen eine Lösung dieses Problems finden läßt, heißt der nachfolgende Satz auch *Chinesischer Restesatz*. Sein Beweis ist wieder konstruktiv, d.h. er liefert uns ein konkretes Verfahren zur Berechnung der Lösung x, das sich leicht in *MuPAD* implementieren läßt. Zusammen mit dem (Erweiterten) Euklidischen Algorithmus ist es von fundamentaler Bedeutung für viele Bereiche der Computeralgebra, aber auch der Kryptographie.

Satz 23. *Seien $\ell_1, \ell_2, \ldots, \ell_k \in \mathbb{Z}$ beliebig und $m_1, m_2, \ldots, m_k \in \mathbb{N} \setminus \{1\}$ so gewählt, daß für $i \neq j$ stets $\mathrm{ggT}\,(m_i, m_j) = 1$ gilt (d.h. die Zahlen $m_1, \ldots, m_k$ seien paarweise teilerfremd). Dann gibt es genau eine ganze Zahl $0 \leq x < m = m_1 \cdot m_2 \cdots m_k$ mit $x \in [\ell_1]_{m_1}$, $x \in [\ell_2]_{m_2}$,..., $x \in [\ell_k]_{m_k}$.*

Beweis. Für den Beweis der *Existenz* einer Lösung $0 \leq x < m$ führen wir zunächst für jedes $1 \leq i \leq k$ die Zahl

$$\mu_i = \frac{m}{m_i} = m_1 \cdots m_{i-1} m_{i+1} \cdots m_k \in \mathbb{N}$$

ein. Dann behaupten wir, daß μ_i und m_i stets teilerfremd sind. Wir beweisen diese Behauptung nur für den Fall $k = 3$; der allgemeine Fall ergibt sich dann durch Iteration (genauer gesagt: durch vollständige Induktion). Wir wollen also zeigen, daß aus $\mathrm{ggT}\,(m_1, m_3) = \mathrm{ggT}\,(m_2, m_3) = 1$ folgt, daß auch $\mathrm{ggT}\,(m_1 m_2, m_3) = 1$ gilt. Dazu betrachten wir wieder Bézout-Koeffizienten: nach Satz 10 gibt es unter unseren Annahmen Zahlen $s_1, s_2, t_1, t_2 \in \mathbb{Z}$, so daß

$$s_1 m_1 + t_1 m_3 = 1 = s_2 m_2 + t_2 m_3 \ .$$

Wir multiplizieren nun die erste Gleichung mit $s_2 m_2$ und addieren dann auf beiden Seiten $t_2 m_3$. Dies führt zu

$$s_1 s_2 (m_1 m_2) + (t_1 s_2 m_2 + t_2) m_3 = s_2 m_2 + t_2 m_3 = 1 \; .$$

Daraus folgt aber wiederum nach Satz 10, daß $\mathrm{ggT}\,(m_1 m_2, m_3) = 1$, da natürlich sowohl $s_1 s_2$ als auch $t_1 s_2 m_2 + t_2$ eine ganze Zahl ist.

Nachdem wir also gezeigt haben, daß immer $\mathrm{ggT}\,(\mu_i, m_i) = 1$ gilt, betrachten wir wieder die zugehörigen Bézout-Koeffizienten. Diese sind ganze Zahlen, die wir der Einfachheit halber wieder $s_i, t_i \in \mathbb{Z}$ nennen wollen, mit $s_i \mu_i + t_i m_i = 1$. Mit ihrer Hilfe definieren wir die Zahl

$$\hat{x} = \ell_1 s_1 \mu_1 + \ell_2 s_2 \mu_2 + \cdots + \ell_k s_k \mu_k \tag{6}$$

und behaupten, daß $x = \hat{x} \bmod m$ eine Lösung unseres Problems darstellt. Zunächst erfüllt x sicher die Ungleichung $0 \leq x < m$ nach der Definition des Restes bei der Division durch m. Da sich x und $\hat{x}$ nur durch Vielfache von m unterscheiden und m offensichtlich durch jede der Zahlen m_i teilbar ist, sind die Bedingungen $x \in [\ell_i]_{m_i}$ und $\hat{x} \in [\ell_i]_{m_i}$ äquivalent. Letztere Bedingung ist aber erfüllt wegen

$$[\hat{x}]_{m_i} = \Big[\sum_{j=1}^{k} \ell_j s_j \mu_j\Big]_{m_i} = [\ell_i s_i \mu_i]_{m_i} = [\ell_i]_{m_i} \; .$$

Dabei gilt das zweite Gleichheitszeichen, weil jedes μ_j mit $j \neq i$ durch m_i teilbar ist und damit alle anderen Summanden weggelassen werden können. Und das dritte Gleichheitszeichen folgt aus der Beobachtung, daß nach Konstruktion die Restklasse $[s_i]_{m_i}$ invers zu $[\mu_i]_{m_i}$ ist.

Nachdem wir also eine Lösung x in dem gewünschten Intervall gefunden haben, müssen wir noch deren *Eindeutigkeit* zeigen. Dazu nehmen wir an, daß es eine weitere ganze Zahl $0 \leq y < m$ gäbe mit $y \in [\ell_i]_{m_i}$ für $1 \leq i \leq k$. Da x in denselben Restklassen liegt, bedeutet dies für die Differenz der beiden Lösungen $x - y \in [0]_{m_i}$, d.h. jedes m_i ist ein Teiler von $x - y$. Wir behaupten nun, daß daraus folgt, daß auch m die Differenz $x - y$ teilt. Wenn wir nämlich dies zeigen können, dann sind wir fertig mit unserem Beweis, da wir dann auf einen Widerspruch gestoßen sind: es kann nicht sein, daß sowohl x als auch y zwischen 0 und $m - 1$ liegen und gleichzeitig ihre Differenz ein Vielfaches von m ist. Also kann es keine solche Lösung y geben.

Wir zeigen hier nur, daß das Produkt $m_1 m_2$ ein Teiler von $x - y$ ist; daß auch m ein Teiler ist, folgt dann wieder durch Iteration (genauer gesagt, wieder durch vollständige Induktion). Der Schlüssel ist erneut Satz 10 bzw. die Bézout-Koeffizienten. Da nach Voraussetzung $\mathrm{ggT}\,(m_1, m_2) = 1$ gilt, existieren Zahlen $s, t \in \mathbb{Z}$, so daß $sm_1 + tm_2 = 1$. Da sowohl m_1 als auch m_2 ein Teiler der Differenz $x-y$ ist, gibt es außerdem Zahlen $k_1, k_2 \in \mathbb{Z}$ mit $k_1 m_1 = k_2 m_2 = x-y$. Damit erhalten wir dann

$$\begin{aligned} x - y &= (x - y) \cdot 1 = (x - y) \cdot (sm_1 + tm_2) \\ &= (x - y)sm_1 + (x - y)tm_2 \\ &= k_2 m_2 s m_1 + k_1 m_1 t m_2 = (k_2 s + k_1 t)(m_1 m_2)\ . \end{aligned}$$

Also ist $m_1 m_2$ in der Tat ein Teiler von $x - y$. □

Bemerkung 24. Die Eindeutigkeit der Lösung x unseres Problems haben wir in Satz 23 dadurch erreicht, daß wir gefordert haben $0 \leq x < m$. Wenn wir diese Forderung fallen lassen, dann finden wir unendlich viele Lösungen, nämlich jedes Element der Restklasse $[x]_m$.

Die Voraussetzung, daß in Satz 23 die Moduli $m_1, \ldots, m_k$ paarweise teilerfremd sind, kann nicht weggelassen werden, wie das folgende Beispiel zeigt. Gesucht sei eine Zahl $x \in \mathbb{Z}$ mit $x \in [1]_{15}$ und $x \in [2]_{21}$. Da 3 ein gemeinsamer Teiler von 15 und 21 ist, bedeutet dies aber auch, daß $x \in [1]_3 \cap [2]_3$. Dies ist aber unmöglich, da keine Zahl x bei Division durch 3 gleichzeitig den Rest 1 und den Rest 2 liefern kann.

Beispiel 25. Wir suchen ganze Zahlen $x \in \mathbb{Z}$, die die folgenden drei Bedingungen erfüllen: $x \in [2]_3$, $x \in [3]_4$ und $x \in [1]_7$. Wenn wir mit der Formulierung von Satz 23 vergleichen, behandeln wir also den Fall $k = 3$, $\ell_1 = 2$, $\ell_2 = 3$, $\ell_3 = 1$, $m_1 = 3$, $m_2 = 4$, $m_3 = 7$ und es gibt genau eine Lösung mit $0 \leq x < m = 3 \cdot 4 \cdot 7 = 84$. Zur Berechnung dieser Lösung folgen wir den im Beweis von Satz 23 vorgegebenen Schritten. Zuerst setzen wir $\mu_1 = m/m_1 = 4 \cdot 7 = 28$, $\mu_2 = m/m_2 = 3 \cdot 7 = 21$ und $\mu_3 = m/m_3 = 3 \cdot 4 = 12$. Dann bestimmen wir durch wiederholte Anwendung des Erweiterten Euklidischen Algorithmus die benötigten Bézout-Koeffizienten:

$$\begin{aligned} 1 &= 1 \cdot \mu_1 - 9 \cdot m_1 \qquad \Longrightarrow \qquad s_1 = 1\ , \\ 1 &= 1 \cdot \mu_2 - 5 \cdot m_2 \qquad \Longrightarrow \qquad s_2 = 1\ , \\ 1 &= 3 \cdot \mu_3 - 5 \cdot m_3 \qquad \Longrightarrow \qquad s_3 = 3\ . \end{aligned}$$

Nach (6) ist damit eine Lösung unseres Problems gegeben durch:

$$\hat{x} = \ell_1 s_1 \mu_1 + \ell_2 s_2 \mu_2 + \ell_3 s_3 \mu_3 = 2 \cdot 1 \cdot 28 + 3 \cdot 1 \cdot 21 + 1 \cdot 3 \cdot 12 = 155 \; .$$

Berechung des Restes von $\hat{x}$ modulo $m = 84$ ergibt dann schließlich die gesuchte Lösung $x = 155 \bmod 84 = 71$ zwischen 0 und 83. Die gesamte Lösungsmenge unserer Aufgabe ist $[71]_{84} \subset \mathbb{Z}$.

Bemerkung 26. In *MuPAD* ist eine Implementierung des Chinesischen Restesatzes in der Bibliothek `numlib` in Form des Befehls `numlib::ichrem` (von englisch *integer chinese remainder*) zu finden. Er nimmt zwei Argumente; das erste ist eine Liste mit den Werten $\ell_1, \ldots, \ell_k$ und das zweite eine Liste mit den zugehörigen Moduli $m_1, \ldots, m_k$. Unser obiges Beispiel kann man also mit der Anweisung `numlib::ichrem([2,3,1],[3,4,7])` lösen und bekommt dann auch die Bestätigung, daß in der Tat `71` die Lösung ist.

7 Kleiner Satz von Fermat

Als eine einfache Anwendung des modularen Rechnens beweisen wir jetzt noch den sogenannten *Kleinen Satz von Fermat*. Dieser wird später in der Kryptographie wichtig sein.

Satz 27. *Sei $n \in \mathbb{N}_0$ eine natürliche Zahl und $p \in \mathbb{P}$ eine Primzahl. Dann gilt*

$$n^p \bmod p = n \bmod p \tag{7}$$

(d.h. n^p und n führen auf denselben Rest bei Division durch p: $[n^p]_p = [n]_p$).

Beweis. Wir führen den Beweis mittels einer vollständigen Induktion über n. Für $n = 0$ und $n = 1$ ist die Aussage des Satzes offensichtlich war. Wir nehmen nun an, daß die Aussage für ein beliebiges aber festes $n \in \mathbb{N}$ bereits bewiesen wäre, und zeigen, daß sie dann auch für $n + 1$ gilt. Dazu benötigen wir die binomische Formel, um $(n+1)^p$ zu berechnen:

$$(n+1)^p = n^p + \binom{p}{1} n^{p-1} + \binom{p}{2} n^{p-2} + \cdots + \binom{p}{p-1} n + 1 \; . \tag{8}$$

Dabei benutzten wir die Binomialkoeffizienten[5]

$$\binom{p}{k} = \frac{p!}{k!(p-k)!} = p \cdot \frac{(p-1)!}{k!(p-k)!} \in \mathbb{Z} \; .$$

[5]Anschaulich läßt sich $\binom{p}{k}$ auch interpretieren als die Anzahl der k-elementigen Teilmengen einer p-elementigen Menge.

Man sieht deutlich, daß für $1 \leq k \leq p-1$ stets p ein Teiler von $\binom{p}{k}$ ist. Wenn wir also nur den Rest bei Division durch p betrachten, dann sind alle diese Summanden bedeutungslos und wir erhalten $(n+1)^p \bmod p = (n^p+1) \bmod p$. Da wir angenommen haben, daß unsere Aussage für n bereits bewiesen sei, gilt aber $n^p \bmod p = n \bmod p$ und daraus folgt sofort, daß $(n+1)^p \bmod p = n+1 \bmod p$. Dies stellt jedoch genau unsere Aussage für $n+1$ dar. □

Bemerkung 28. Wenn wir annehmen, daß p kein Teiler von n ist (d.h. insbesondere, daß $[n]_p \neq [0]_p$), dann gilt $\mathrm{ggT}\,(n,p) = 1$. Da also p, n teilerfremd sind, besitzt nach Satz 14 die Restklasse $[n]_p$ ein Inverses in $\mathbb{Z}_p$. Wir können als durch $[n]_p$ teilen und erhalten, daß in diesem Fall gilt $[n^{p-1}]_p = [1]_p$. Häufig wird auch diese Aussage als Kleiner Satz von Fermat bezeichnet. Man beachte, daß wir jetzt auch das Inverse leicht angeben können: $[n]_p^{-1} = [n^{p-2}]_p$.

In dieser Form kann man den Kleinen Satz von Fermat auch „umdrehen“ und als (nicht sonderlich guten) Primzahltest benutzen: Eine Zahl $p \in \mathbb{N}\backslash\{1\}$ ist genau dann prim, wenn für alle $n \in \mathbb{N}$, die nicht durch p teilbar sind, $[n^{p-1}]_p = [1]_p$ gilt. Die eine Richtung haben wir gerade oben bewiesen. Für die andere Richtung nehmen wir an, daß p keine Primzahl sei und daher einen Teiler $1 < k < p$ besitzt. Nach unserer Voraussetzung gilt dann $[k^{p-1}]_p = [1]_p$, d.h. es gibt eine Zahl $\ell \in \mathbb{Z}$ mit $k^{p-1} = 1 + \ell p$. Wenn $p = mk$, dann gilt damit $k^{p-1} = 1 + \ell mk$ oder $k(k^{p-2} - \ell m) = 1$. Damit hätte aber $k > 1$ ein Inverses in $\mathbb{Z}$, was bekanntlich nicht stimmt. Also muß p eine Primzahl sein.

Wenn p keine Primzahl ist, aber trotzdem für ein $n \in \mathbb{N}$ gilt $[n^{p-1}]_p = [1]_p$, dann nennt man p eine *Pseudoprimzahl* bezüglich n. Ein ungerades $p \in \mathbb{N} \setminus \{1\}$, das bezüglich aller $n \in \mathbb{N}$ mit $\mathrm{ggT}\,(p,n) = 1$ eine Pseudoprimzahl ist, heißt *Carmichael-Zahl*. Es gibt unendlich vieler solcher Zahlen; die drei kleinsten sind $561 = 3 \cdot 11 \cdot 17$, $1105 = 5 \cdot 13 \cdot 17$ sowie $1729 = 7 \cdot 13 \cdot 19$.

A Äquivalenzrelationen

Die Definition der Menge $\mathbb{Z}_m$ aller Restklassen modulo $m \in \mathbb{N}$ kann mathematisch befriedigender erfolgen mit Hilfe des Begriffs einer Äquivalenzrelation. Dies ist ein allgemeines (und für die moderne Mathematik sehr wichtiges) Konzept, um auszudrücken, daß verschiedene Elemente einer Menge in einem gewissen Sinne „äquivalent“ oder „gleichwertig“ sind.

Definition 29. Sei $\mathcal{M}$ eine beliebige (nichtleere) Menge. Wir schreiben dann $\mathcal{M}^2 = \mathcal{M} \times \mathcal{M} = \big\{(m,n) : m,n \in \mathcal{M}\big\}$ für das kartesische Produkt von

$\mathcal{M}$ mit sich selbst. Eine beliebige Teilmenge $\mathcal{R} \subseteq \mathcal{M}^2$ heißt *Relation* auf $\mathcal{M}$. Statt $(m,n) \in \mathcal{R}$ schreibt man auch oft kürzer $m\mathcal{R}n$.

Die Relation $\mathcal{R}$ heißt (i) *reflexiv*, falls für alle $m \in \mathcal{M}$ gilt $(m,m) \in \mathcal{R}$, (ii) *symmetrisch*, falls aus $(m,n) \in \mathcal{R}$ stets folgt $(n,m) \in \mathcal{R}$, und (iii) *transitiv*, falls aus $(\ell,m),(m,n) \in \mathcal{R}$ stets folgt $(\ell,n) \in \mathcal{R}$. Eine reflexive, symmetrische und transitive Relation heißt *Äquivalenzrelation*. Für eine solche Relation hat sich die Schreibweise $m \sim n$ (gesprochen m *äquivalent zu* n) anstelle von $(m,n) \in \mathcal{R}$ eingebürgert.

Relationen stellen eine Verallgemeinerung von Funktionen dar. Eine Funktion $f : \mathcal{M} \to \mathcal{M}$ entspricht gerade einer Relation $\mathcal{R} \subseteq \mathcal{M}^2$, bei der es für jedes Element $m \in \mathcal{M}$ genau ein Paar $(m,n) \in \mathcal{R}$ gibt, nämlich $\big(m, f(m)\big)$. Bei einer allgemeinen Relation muß es nicht zu jedem $m \in \mathcal{M}$ ein Paar $(m,n) \in \mathcal{R}$ geben, es kann aber umgekehrt auch mehr als ein Paar mit m als erstem Eintrag geben.

Beispiel 30. Sei $m \in \mathbb{N}$ eine beliebige aber feste natürliche Zahl. Wir definieren auf den ganzen Zahlen $\mathbb{Z}$ eine Relation $\mathcal{R}$, in dem wir sagen, daß $(x,y) \in \mathcal{R}$ gilt, falls die Differenz $x - y$ durch m teilbar ist. Statt $(x,y) \in \mathcal{R}$ schreiben wir dann auch kurz $x \equiv y \bmod m$ und sagen, daß x *kongruent zu* y *modulo* m ist.

Wir wollen nun zeigen, daß $\mathcal{R}$ eine Äquivalenzrelation ist. Zunächst ist $\mathcal{R}$ reflexiv, denn für jede ganze Zahl $x \in \mathbb{Z}$ gilt, daß $0 = x - x$ durch m teilbar ist und damit $(x,x) \in \mathcal{R}$. Die Relation $\mathcal{R}$ ist auch symmetrisch, denn wenn $x - y$ durch m teilbar ist, dann gilt das gleiche für $y - x$ und damit folgt $(y,x) \in \mathcal{R}$ sofort aus $(x,y) \in \mathcal{R}$. Schließlich ist $\mathcal{R}$ auch transitiv, denn wenn $x - y$ und $y - z$ durch m teilbar sind, dann gilt dies auch für $x - z = (x - y) + (y - z)$; also folgt aus $(x,y),(y,z) \in \mathcal{R}$ immer $(x,z) \in \mathcal{R}$.

Beispiel 31. Sei $\mathcal{S}$ die Menge aller Schüler, die an der AG *Computeralgebra* teilnehmen. Wir definieren eine Relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$, in dem wir sagen, daß $(x,y) \in \mathcal{R}$, wenn die Schüler x und y in derselben Klassenstufe sind. Man überprüft leicht, daß dies eine Äquivalenzrelation definiert (Übung!).

Definition 32. Sei $\mathcal{R}$ eine Äquivalenzrelation auf der Menge $\mathcal{M}$ und $m \in \mathcal{M}$ ein beliebiges Element. Dann heißt $[m] = \{n \in \mathcal{M} : n \sim m\} \subseteq \mathcal{M}$ die *Äquivalenzklasse* von m und m ist ein *Vertreter* oder *Repräsentant* dieser Äquivalenzklasse. Die Menge $\mathcal{M}/\mathcal{R} = \big\{[m] : m \in \mathcal{M}\big\}$ aller Äquivalenzklassen heißt *Quotientenmenge* von $\mathcal{M}$ bezüglich der Relation $\mathcal{R}$.

Beispiel 33. Im Prinzip kennt man schon aus der Schule ein wichtiges Beispiel einer Äquivalenzrelation mit der zugehörigen Quotientenmenge. Wir betrachten

die Menge $\mathcal{M} = \mathbb{Z} \times (\mathbb{Z} \setminus \{0\})$. Auf dieser Menge $\mathcal{M}$ definieren wir eine Äquivalenzrelation $\mathcal{R}$,[6] in dem wir sagen, daß genau dann $(p_1, q_1) \sim (p_2, q_2)$ gelten soll, wenn $p_1 q_2 = p_2 q_1$. Die Bedeutung dieser Relation erkennt man leicht, wenn man statt (p, q) die vertrautere Notation p/q benutzt: es handelt sich um Brüche (deshalb haben wir ja auch $q = 0$ ausgeschlossen) und $(p_1, q_1) \sim (p_2, q_2)$ gilt genau dann, wenn $p_1/q_1 = p_2/q_2$. Die zu dieser Äquivalenzrelation gehörende Quotientenmenge kann also mit den rationalen Zahlen $\mathbb{Q}$ identifiziert werden und die Äquivalenzklasse $[p/q]$ enthält alle Brüche, die denselben Wert wie p/q besitzen.

Beispiel 34. Wir fahren fort mit Beispiel 30. Offensichtlich bedeutet die Bedingung, daß $x - y$ durch m teilbar ist, nichts anderes als daß x und y bei Division durch m denselben Rest liefern. Die Äquivalenzklasse $[x]$ enthält also alle ganzen Zahlen, die bei Division durch m denselben Rest ergeben wie x. Daraus folgt auch sofort, daß $[x] = [x + m] = [x + 2m] = [x - m] = \cdots$. Also gibt es genau m verschiedene Äquivalenzklassen und wir können schreiben

$$\mathbb{Z}/\mathcal{R} = \big\{[0], [1], [2], \ldots, [m-1]\big\} \; . \tag{9}$$

Offensichtlich können wir daher $\mathbb{Z}/\mathcal{R}$ mit $\mathbb{Z}_m$ identifizieren. Man beachte, daß wir in (9) nur die natürlichste und einfachste Wahl für die Repräsentanten getroffen haben. Wir hätten genauso gut schreiben können

$$\mathbb{Z}/\mathcal{R} = \big\{[42m], [5m+1], [2-13m], \ldots, [-1]\big\} \; .$$

Wir haben gesehen, daß die Addition und die Multiplikation auf $\mathbb{Z}$ zu einer Addition und Multiplikation auf $\mathbb{Z}_m = \mathbb{Z}/\mathcal{R}$ führt. Dies ist nicht selbstverständlich, sondern beruht darauf, daß sich unsere spezielle Äquivalenzrelation mit der Arithmetik auf $\mathbb{Z}$ „verträgt“; man spricht dann von einer *Kongruenzrelation*. Solche Kongruenzrelationen sind eine allgemeine Technik, um aus „großen“ Rechenbereichen durch Quotientenbildung „kleinere“ zu erhalten.

B Lineare Kongruenzen

In Abschnitt 4 haben wir die Frage nach der Existenz von Inversen in $\mathbb{Z}_m$ beantwortet (Satz 14). Eine Restklasse $[j]_m$ ist genau dann ein Inverses zu $[k]_m$, wenn j eine Lösung der Gleichung $[k]_m \cdot [x]_m = [1]_m$ ist. Offensichtlich ist dies ein Spezialfall des Problems, Gleichungen der Form $[k]_m \cdot [x]_m = [\ell]_m$ für vorgegebene

[6] Man beachte, daß hier bereits die Menge $\mathcal{M}$ aus Paaren (p, q) besteht. Unsere Äquivalenzrelation $\mathcal{R} \subset \mathcal{M}^2$ enthält also Paare $\big((p_1, q_1), (p_2, q_2)\big)$, deren Elemente selbst wieder Paare sind!

Werte von k, ℓ, m zu lösen. In der in Beispiel 30 eingeführten Notation können wir dieses Problem auch als $kx \equiv \ell \bmod m$ schreiben und man spricht dann von einer *linearen Kongruenz*.

Beispiel 35. Wenn wir lineare Gleichungen in $\mathbb{Q}$ (oder $\mathbb{R}$) lösen, dann ist die Situation sehr einfach: vorausgesetzt, daß $a \neq 0$, besitzt die Gleichung $ax = b$ genau eine Lösung, nämlich $x = b/a$. In $\mathbb{Z}_m$ ist die Sachlage etwas komplizierter, wie zwei einfache Beispiele zeigen. Wir wählen $m = 6$. Dann ist die Gleichung $2x \equiv 3 \bmod 6$ unlösbar, da für jedes $x \in \mathbb{Z}$ die Zahl $2x$ modulo 6 einen geraden Rest liefert. Wenn wir dagegen die Gleichung $2x \equiv 4 \bmod 6$ betrachten, passiert etwas ganz anderes: wir haben in $\mathbb{Z}_m$ zwei Lösungen, da sowohl $x = 2$ also auch $x = 5$ die Gleichung erfüllt und $[2]_6 \neq [5]_6$.

Das letzte Beispiel zeigt, daß in $\mathbb{Z}_m$ eine vertraute Rechenregel im allgemeinen nicht mehr gültig ist: aus $ax = ay$ folgt selbst für $a \neq 0$ nicht unbedingt $x = y$. Wie das nächste Lemma zeigt, muß bei dieser Kürzungsregel im modularen Fall unter Umständen der Modulus angepaßt werden.

Lemma 36. *Seien $j, k, \ell \in \mathbb{Z}$ und $m \in \mathbb{N}$. Wenn gilt $kj \equiv k\ell \bmod m$, dann gilt auch $j \equiv \ell \bmod (m/d)$ mit $d = \mathrm{ggT}\,(k, m)$.*

Beweis. Aus der Definition der Kongruenz $\equiv$ folgt, daß es eine Zahl $r \in \mathbb{Z}$ gibt mit $kj = kl + rm$. Nach Satz 10 existieren ferner Bézout-Koeffizienten $s, t \in \mathbb{Z}$, so daß $sk + tm = d$. Da d ein gemeinsamer Teiler von k, m ist, können wir beide Zahlen ohne Rest durch d teilen und setzen $k' = k/d$ bzw. $m' = m/d$. Eine entsprechende Division unserer beiden Gleichungen ergibt dann zum einen $k'j = k'\ell + rm'$ und zum anderen $sk' + tm' = 1$ (was wieder mit Satz 10 nach sich zieht, daß $\mathrm{ggT}\,(k', m') = 1$). Eine Multiplikation der zweiten dieser Gleichungen mit $j - \ell$ ergibt $sk'(j - \ell) + tm'(j - \ell) = j - \ell$. Da aus der ersten folgt, daß m' ein Teiler von $k'(j - \ell)$ ist, stellen wir fest, daß m' ein Teiler beider Summanden auf der linken Seite und damit auch ein Teiler der rechten Seite ist. $m' \mid (j - \ell)$ ist aber nur eine andere Art zu sagen, daß $j \equiv \ell \bmod m'$. □

Mit Hilfe dieses Lemmas können wir nun die Lösbarkeit linearer Kongruenzen genauer untersuchen. Der Beweis des folgenden Satzes ist wieder konstruktiv, da er zeigt, daß die Lösung (falls eine existiert) sich leicht aus einem Bézout-Koeffizienten berechnen läßt.

Satz 37. *Seien $k, \ell \in \mathbb{Z}$ und $m \in \mathbb{N} \setminus \{1\}$. Die lineare Kongruenz $kx \equiv \ell \bmod m$ besitzt genau dann eine Lösung $x \in \mathbb{Z}$, wenn mit $d = \mathrm{ggT}\,(k, m)$ gilt $d \mid \ell$. Wenn*

$\hat{x} \in \mathbb{Z}$ eine Lösung ist, dann ist die gesamte Lösungsmenge gegeben durch $[\hat{x}]_{m/d}$, also der Restklasse von $\hat{x}$ modulo m/d.

Beweis. Da wir es hier wieder mit einer *genau dann, wenn* Aussage zu tun haben, müssen wir zwei Richtungen beweisen. Wir beginnen mit der Rückrichtung, nehmen also an, daß $d \mid \ell$. Dann sind auch $k' = k/d$, $m' = m/d$ sowie $\ell' = \ell/d$ ganze Zahlen und nach der Kürzungsregel aus Lemma 36 gilt $k'x \equiv \ell' \bmod m'$. Im Beweis von Lemma 36 haben wir auch gezeigt, daß $\mathrm{ggT}\,(k', m') = 1$ gilt. Damit ist aber $[k']_{m'}$ in $\mathbb{Z}_{m'}$ invertierbar; es existiert also ein $s \in \mathbb{Z}$ mit $k's \equiv 1 \bmod m'$. Damit aber erfüllt $x = s\ell'$ die Gleichung $k'x \equiv \ell' \bmod m'$. Eine Multiplikation mit d zeigt dann, daß dieses x auch eine Lösung von $kx \equiv \ell \bmod m$ ist.

Nun nehmen wir an, daß ein $x \in \mathbb{Z}$ existiert, daß unsere lineare Kongruenz $kx \equiv \ell \bmod m$ löst. Das bedeutet aber, daß für ein geeignet gewähltes $q \in \mathbb{Z}$ gilt $\ell = kx + qm$. Wenn d ein gemeinsamer Teiler von k, m ist, dann muß d damit auch wie behauptet ℓ teilen.

Nach unseren bisherigen Überlegungen erfüllt jede Lösung unseres Ausgangsproblem auch das reduzierte Problem $k'x \equiv \ell' \bmod m'$ (d.h. es muß in $\mathbb{Z}_{m'}$ gelten, daß $[k']_{m'}[x]_{m'} = [\ell']_{m'}$) und umgekehrt liefert jede Lösung des reduzierten Problems eine des Ausgangsproblem. Damit können wir bereits festhalten, daß mit einer Lösung $x_0 \in \mathbb{Z}$ auch jedes Element der Restklasse $[x_0]_{m'}$ Lösung ist.

Sei nun $x_1 \in \mathbb{Z}$ eine weitere Lösung. Da dann sowohl x_0 als auch x_1 das reduzierte Problem lösen, folgt nach Subtraktion, daß $k'(x_0 - x_1) \equiv 0 \bmod m'$. Da $\mathrm{ggT}\,(k', m') = 1$ gilt nach Lemma 36, daß bereits $x_0 - x_1 \equiv 0 \bmod m'$. Damit liegen aber x_0 und x_1 in derselben Restklassen modulo m' und auch unsere Eindeutigkeitsaussage ist bewiesen. □

Beispiel 38. Wir suchen die Lösung der linearen Kongruenz $33x \equiv 88 \bmod 319$. Mit dem Erweiterten Euklidischen Algorithmus erhalten wir

$$\mathrm{ggT}\,(33, 319) = 10 \cdot 33 - 1 \cdot 319 = 11\ .$$

Also können wir die Kongruenz reduzieren zu $3x \equiv 8 \bmod 29$. Wieder mit dem Erweiterten Euklidischen Algorithmus berechnen wir nun

$$\mathrm{ggT}\,(3, 29) = 10 \cdot 3 - 1 \cdot 29 = 1\ .$$

Also besitzt $[3]_{29}$ ein Inverses in $\mathbb{Z}_{29}$ (das ist auch nicht verwunderlich, da 29 eine Primzahl ist), nämlich $[10]_{29}$ und die Lösungsmenge unserer Kongruenz ist $[10]_{29} \cdot [8]_{29} = [22]_{29}$.

Bemerkung 39. *MuPAD* kann ebenfalls lineare Kongruenzen lösen. Dazu dient der Befehl `numlib::lincongruence` (also mal wieder ein Befehl aus der `numlib`-Bibliothek zur Zahlentheorie). Er benötigt als Eingabe drei Argumente: die Koeffizienten k, ℓ der Kongruenz und natürlich den Modulus m. Die Ausgabe ist allerdings anders organisiert als in unserer obigen Formulierung des Satzes: man erhält – falls das Problem lösbar ist – alle Lösungen in $\mathbb{Z}_m$ und nicht die eindeutige Lösung in $\mathbb{Z}_{m/d}$. Andernfalls lautet die Antwort `FAIL`. So liefert die Eingabe `numlib::lincongruence(33,88,319)` als Ergebnis die Liste `[22, 51, 80, 109, 138, 167, 196, 225, 254, 283, 312]`. Dies entspricht aber unserem Ergebnis $[22]_{29}$, denn es handelt sich gerade um die Elemente der Restklasse $[22]_{29}$, die zwischen 0 und 319 liegen.

C Iterative Form des Euklidischen Algorithmus

In Abschnitt 5 haben wir eine rekursive Form des (Erweiterten) Euklidischen Algorithmus angegeben. Diese ist aus theoretischer Sicht am einfachsten und elegantesten. Unter praktischen Gesichtspunkten ist allerdings häufig eine iterative Form vorzuziehen. Deshalb wollen wir diese hier auch kurz präsentieren.

Im Falle des „einfachen“ Euklidischen Algorithmus 3 ist eine solche Formulierung trivial zu erhalten. Anstelle der rekursiven Aufrufe benutzen wir eine `while`-Schleife, die so lange durchlaufen wird, bis x ein Teiler von y ist. Wenn dies der Fall ist, wird x als Ergebnis zurückgegeben. Bei jedem Durchlauf der Schleife wird x durch $y \bmod x$ und y durch x ersetzt (damit kein Wert verloren geht, muß dieses Vertauschen über eine Hilfsvariable a ablaufen).

Algorithmus 5 Euklidischer Algorithmus (iterative Form)

Eingabe: zwei natürliche Zahlen $x, y \in \mathbb{N}$ mit $x \leq y$
Ausgabe: $d = \mathrm{ggT}\,(x, y)$

1: **while** $x \nmid y$ **do**
2: $\quad a \leftarrow x$
3: $\quad x \leftarrow y \bmod x$
4: $\quad y \leftarrow a$
5: **end while**
6: **return** x

Im Falle des Erweiterten Euklidischen Algorithmus 4 ist die iterative Formulierung etwas aufwendiger. Hier wird deshalb nur eine mögliche Form ohne de-

taillierte Diskussion angegeben. Man erkennt aber leicht die wesentlichen Schritte von Algorithmus 4 wieder (scheinbar wird $y \bmod x$ nicht mehr verwendet; diese Berechnung verbirgt sich aber hinter der Zeile $r_2 \leftarrow a - qr_2$ deren Ergebnis nämlich gerade $r_1 \bmod r_2$ ist).

Algorithmus 6 Erweiterter Euklidischer Algorithmus (iterative Form) `EEA2`

Eingabe: $x, y \in \mathbb{N}$ mit $x \leq y$
Ausgabe: $s, t, d \in \mathbb{Z}$ mit $\mathrm{ggT}(x, y) = d = sx + ty$

1: $s_1 \leftarrow 1; \quad s_2 \leftarrow 0; \quad t_1 \leftarrow 0; \quad t_2 \leftarrow 1; \quad r_1 \leftarrow y; \quad r_2 \leftarrow x$
2: **repeat**
3: $\quad q \leftarrow r_1 \operatorname{div} r_2$
4: $\quad a \leftarrow s_1; \quad s_1 \leftarrow s_2; \quad s_2 \leftarrow a - qs_2$
5: $\quad a \leftarrow t_1; \quad t_1 \leftarrow t_2; \quad t_2 \leftarrow a - qt_2$
6: $\quad a \leftarrow r_1; \quad r_1 \leftarrow r_2; \quad r_2 \leftarrow a - qr_2$
7: **until** $r_2 = 0$
8: **return** (s_1, t_1, r_1)