Der euklidische Algorithmus

Die Definition des ggT

Der grösste gemeinsame Teiler (ggT) zweier ganzer Zahlen a und b, die nicht gleichzeitig null sind, ist die grösste positive ganze Zahl, die sowohl a als auch b ohne Rest teilt. Formal:

$$\mathrm{ggT}(a,b) = \max\{k \in \mathbb{N} \mid k \text{ teilt } a \text{ und } b\}$$

Die englische Bezeichnung für den ggT lautet *greatest common divisor* und wird entsprechend mit *gcd* abgekürzt.

Die Methode aus dem Mathematikunterricht

Vorgehen: Um den ggT zweier natürlicher Zahlen zu bestimmen, zerlegt man beide Zahlen in ihre Primfaktoren. Anschliessend multipliziert man alle gemeinsamen Primfaktoren, wobei jeder Faktor so oft berücksichtigt wird, wie er in beiden Zerlegungen mindestens vorkommt.

Haben die beiden Zahlen keine gemeinsamen Primfaktoren, so sind sie teilerfremd und ihr ggT ist 1.

Beispiel:

32 340	
2	16 170
2	8 085
3	2 695
5	539
7	77
7	11
11	1

18 200	
2	9 100
2	4 550
2	2 275
5	455
5	91
7	13
13	1

Somit ist $\mathrm{ggT}(32\,340, 18\,200) = 2 \cdot 2 \cdot 5 \cdot 7 = 140$

Der Nachteil

Bei grossen Zahlen kann die Primfaktorzerlegung aufwendig sein. Man muss geeignete Primteiler finden und ihre Vielfachheiten bestimmen.

Der euklidische Algorithmus, den wir gleich kennenlernen werden, berechnet den ggT dagegen direkt durch wiederholte Division mit Rest. Dazu müssen die Zahlen nicht in Primfaktoren zerlegt werden.

Kopfrechnungen

(a) $\mathrm{ggT}(12, 18) =$

(b) $\mathrm{ggT}(18, 12) =$

(c) $\mathrm{ggT}(371, 371) =$

(d) $\mathrm{ggT}(371, 1) =$

(e) $\mathrm{ggT}(371, 0) =$

Eigenschaften des ggTs

Die obigen Beispiele machen die unten formulierten Eigenschaften der Funktion $\mathrm{ggT}(a, b)$ plausibel. Für die folgenden Überlegungen genügt es, nichtnegative ganze Zahlen zu betrachten. Negative Zahlen können wir bei der ggT-Berechnung durch ihre Beträge ersetzen.

(1) Vertauschbarkeit: $\mathrm{ggT}(a, b) = \mathrm{ggT}(b, a)$

(2) Gleiche Zahlen: $\mathrm{ggT}(a, a) = a$

(3) Eine Zahl und 1: $\mathrm{ggT}(a, 1) = 1$

(4) Eine Zahl und 0: $\mathrm{ggT}(a, 0) = a$

Division mit Rest

Die folgende Eigenschaft ist zentral für den euklidischen Algorithmus:

(5) $\mathrm{ggT}(a, b) = \mathrm{ggT}(b, a \bmod b) \quad$ mit $a \bmod b = a - \left\lfloor \frac{a}{b} \right\rfloor \cdot b$

Beispiel: Der ggT von $a = 30$ und $b = 20$ ist 10

Der ggT von $b = 20$ und $r = 30 - \left\lfloor \frac{30}{20} \right\rfloor \cdot 20$ ist ebenfalls 10.

Beweis:

Um den Beweis zu verstehen, musst du einsehen, dass für a, b, $c \in \mathbb{Z}$ jeder gemeinsame Teiler von a und b auch ein gemeinsamer Teiler von $a - c \cdot b$ ist.

- Jeder Teiler von a und b teilt $a \bmod b = a - \left\lfloor \frac{a}{b} \right\rfloor \cdot b$ und natürlich b.
- Jeder Teiler von b und $a \bmod b = a - \left\lfloor \frac{a}{b} \right\rfloor \cdot b$ teilt a und natürlich b.
- Daher haben (a, b) und $(b, a \bmod b)$ genau dieselben gemeinsamen Teiler und folglich denselben ggT.

Anwendung von $\mathrm{ggT}(a, b) = \mathrm{ggT}(b, a \bmod b)$

In der Tabelle wird der Divisionsrest $r = a \bmod b$ berechnet und in der nächsten Zeile a durch b und b durch r ersetzt.

a	b	$a \bmod b = r$
32	44	$32 - 0 \cdot 44 = 32$
44	32	$44 - 1 \cdot 32 = 12$
32	12	$32 - 2 \cdot 12 = 8$
12	8	$12 - 1 \cdot 8 = 4$
8	4	$8 - 2 \cdot 4 = 0$
4	0	–

Beobachtungen:

(a) Ist $a < b$, werden die Zahlen ein einziges Mal „vertauscht“.

(b) Jeder Divisionsrest $r = a \bmod b$ ist kleiner als der vorherige.

(c) Da der ggT bei jedem Schritt gleich bleibt, kann man ihn aus $(4, 0)$ ablesen.

(d) Da die Reste bei jedem Schritt kleiner aber nicht negativ werden, erreicht man nach endlich vielen Schritten den Rest null.

Der euklidische Algorithmus

Input: Zwei nichtnegative Zahlen a, b, die nicht gleichzeitig null sind.

Solange $b \neq 0$ gilt:
berechne den Rest $r = a \bmod b$
ersetze a durch b
ersetze b durch r.

Output: a

- **Warum funktioniert der Algorithmus?**
  Bei jedem Schritt $(a, b) \to (b, a \bmod b)$ bleibt der ggT unverändert.
  Wenn $b = 0$ gilt, ist $\mathrm{ggT}(a, 0) = a$
- **Warum endet der Algorithmus?**
  Die Reste sind nichtnegative Zahlen und werden bei jedem Schritt kleiner. Daher wird nach endlich vielen Schritten der Rest null.

Python-Code des euklidischen Algorithmus

Version 1:

```
def gcd(a, b):
    while b != 0:
        r = a % b
        a = b
        b = r
    return a
```

Version 2:

```
def gcd(a, b):
    while b != 0:
        r = a % b
        a, b = b, r
    return a
```

Version 3:

```
def gcd(a, b):
    while b != 0:
        a, b = b, a % b
    return a
```

Die Mehrfachzuweisung in Version 2 und 3 funktioniert nur, weil die Ausdrücke rechts vom Gleichheitszeichen *vor* den Zuweisungen ausgeführt werden.

Was ist ein Algorithmus?

Ein *Algorithmus* ist eine Vorschrift zur Lösung eines Problems, die

- endlich
- eindeutig und
- effektiv (ausführbar)

ist.