Euklidischer Algorithmus
Proseminar Informatik "Algorithms Unplugged"
am Institut für Theoretische Informatik
Leibniz Universität Hannover

Bharat Ahuja
Wintersemester 2014/15

Der euklidische Algorithmus ist ein Algorithmus, mit dem sich der größte gemeinsame Teiler (*g.g.T.*) zweier natürlicher Zahlen berechnen lässt. Diesen Algorithmus hat Euklid ca. 300 *v.Chr.* in *Buch VII – Die Elemente* (Proposition 1 und 2) als einen geometrischen Algorithmus vorgestellt.

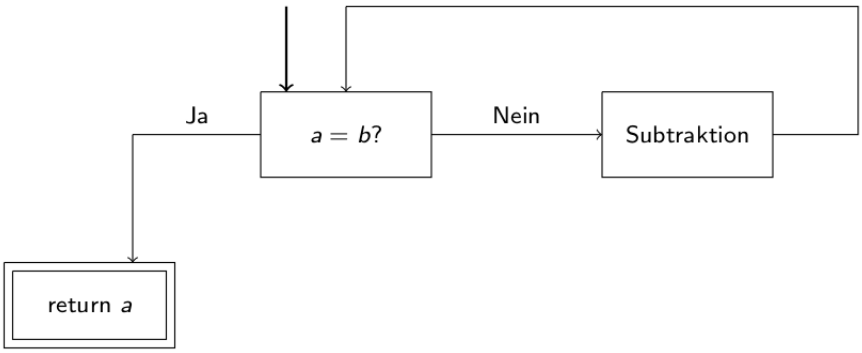
Die Berechnung vom *g.g.T.* ist relevant für die Gruppierung von Objekten und das Prüfen auf Teilerfremdheit von Zahlen.
Die Teilerfremdheit zweier Zahlen kann man alternativ durch das Vergleichen der Primfaktoren überprüfen.

$$ggT(a,b) = \prod_{i \in \mathbb{N}} p_i^{min(a_i,b_i)}$$

wobei $p_i \in \mathbb{P}$. Die Bestimmung der Primfaktorzerlegung einer Zahl liegt aber in NP.

Euklid hat damals den folgenden Algorithmus vorgestellt.

LangsamEuklid: Gegeben $a, b \in \mathbb{N}$
1: **while** $a \neq b$ **do**
2: Falls a größer ist als $b, a \leftarrow a - b$
3: Falls b größer ist als $a, b \leftarrow b - a$
4: **end while**
5: Gib den gemeinsamen Wert der Zahlen aus

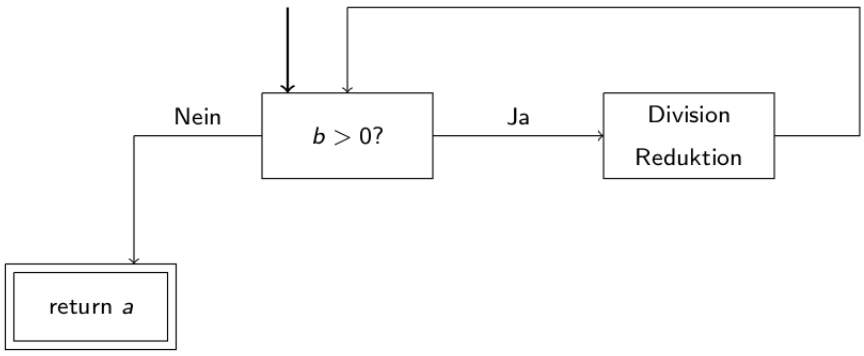


Dieser Algorithmus braucht viele überflüssige Schritte, die man weglassen möchte, wenn eine der beiden Zahlen wesentlich größer ist.

Den Algorithmus sollte man deswegen verbessern und so erhalten wir den modernen Euklidischen Algorithmus.

EUKLID: Gegeben $a, b \in \mathbb{N}$

1: **while** $b > 0$ **do**
2: berechne q, r mit $a = q \cdot b + r$, wobei $0 \leq r < b$
3: $a \leftarrow b, b \leftarrow r$
4: **end while**
5: **return** a



$2 \cdot \log_2 a$ ist eine obere Schranke an der Anzahl an Iterationen von EUKLID.

Dieser Algorithmus ist natürlich viel effizienter als LANGSAMEUKLID, wo die obere Schranke an Iterationen $a + b$ lautet, weil man mehrfache Subtraktionen zusammengefasst hat.

Betrachtet man den Worst-Case für diesen Algorithmus, dann stellt man fest, je schneller die Reste fallen, desto früher terminiert der Algorithmus. D.h. die Fibonacci-Zahlen bilden den ungünstigsten Fall für diesen Algorithmus, weil alle Quotienten den kleinstmöglichen Wert annehmen, und dadurch die Reste sehr langsam fallen.

Im Kern ist der Euklidische Algorithmus effizient, weil nach jeder Iteration immer nur der Divisionsrest relevant ist.

Zur Veranschaulichung : Prüft man die Zahlen 938631347 und 2 auf Teilerfremdheit, so achtet man intuitiv auch nur auf den Divisionsrest.

$$\Rightarrow T_b = \frac{1}{b} \sum_{0<k\leq b} T(k, b)$$

wobei T_b die durchschnittliche Anzahl an Divisionen ist, wenn man eine zufällige Zahl auf Teilerfremdheit mit b prüfen möchte.

Ausserdem gilt,

$$T_n \approx H_n$$

wobei H_n die n-te harmonische Zahl ist.

Literaturverzeichnis :

- *The Art of Computer Programming: Band 1* Kapitel 1.1
- *The Art of Computer Programming: Band 2* Kapitel 4.5.2