3.8 Der AKS-Algorithmus

Der Algorithmus wird hier in der Version nach Lenstra/Bernstein beschrieben. Diese ist nicht auf möglichst effiziente Ausführung getrimmt, sondern auf einen möglichst einfachen Nachweis der Polynomialität.

Eingabe

Eingegeben wird eine natürliche Zahl $n \geq 2$.

Die Länge der Eingabe wird durch die Zahl ℓ der Bits in der Darstellung von n zur Basis 2 gemessen, also durch

$$\ell = \begin{cases} \lceil {}^2\log n \rceil, & \text{falls } n \text{ keine Zweierpotenz,} \\ k+1, & \text{falls } n = 2^k. \end{cases}$$

Ausgabe

Ausgegeben wird ein Boolescher Wert, sinngemäß ausgedrückt durch „ZUSAMMENGESETZT“ oder „PRIM“.

Schritt 1

Zweierpotenzen werden vorab abgefangen –

- Falls $n = 2$: Ausgabe „PRIM“, **Ende**.
- (Sonst) Falls n Zweierpotenz: Ausgabe „ZUSAMMENGESETZT“, **Ende**.

  Diesen Fall erkennt man daran, dass ${}^2\log n$ ganzzahlig ist.

Von jetzt an kann man annehmen, dass n keine Zweierpotenz, also $\ell = \lceil {}^2\log n \rceil$, ist.

Schritt 2

Eine große Zahl $N \in \mathbb{N}$ wird vorherberechnet, und zwar

$$N = 2n \cdot (n-1)(n^2-1)(n^3-1)\cdots(n^{4\ell^2}-1) = 2n \cdot \prod_{i=1}^{4\ell^2}(n^i-1).$$

Diese Zahl ist zwar riesengroß, aber, und das ist hier entscheidend:

- Die Zahl $4\ell^2$ der Multiplikationen ist polynomial in ℓ.
- Da
  $$N \leq 2n \cdot n^{\sum_{i=1}^{4\ell^2} i} = 2n \cdot n^{\frac{4\ell^2(4\ell^2+1)}{2}} \leq 2n \cdot n^{16\ell^4},$$
  ist $k := \lceil {}^2\log N \rceil \leq 1 + (16\ell^4+1) \cdot \ell$ polynomial in ℓ.

Die Zahl k wird im folgenden weiterverwendet; es ist $N < 2^k$, und k ist die kleinste natürliche Zahl mit dieser Eigenschaft.

Anforderungen

Es sind jetzt natürliche Zahlen r und s zu finden, die folgende Anforderungen erfüllen:

1. r ist zu n teilerfremd.
2. Im Intervall $[1, \ldots, s]$ hat n keinen Primteiler.
3. Für jeden Teiler $d \mid \frac{\varphi(r)}{q}$ – wobei $q = \mathrm{Ord}_r\, n$ – gilt
$$\binom{\varphi(r) + s - 1}{s} \geq n^{2d \cdot \lfloor \frac{\varphi(r)}{d} \rfloor}.$$
4. Das eigentliche Primzahlkriterium:
$$(X + a)^n \equiv X^n + a \pmod{(n, X^r - 1)}$$
für alle $a = 1, \ldots, s$.

Schritt 3

r wird als die kleinste Primzahl genommen, die kein Teiler von N ist; insbesondere ist r dann auch kein Teiler von n; insbesondere ist die Bedingung 1 erfüllt. Warum wird r mit polynomialem Aufwand gefunden?

Nach einer Erweiterung des Primzahlsatzes gilt für die Summe $\vartheta(x)$ der Logarithmen der Primzahlen $\leq x$,
$$\vartheta(x) = \sum_{p \leq x,\, p \text{ prim}} \ln p\,,$$
die asymptotische Relation
$$\vartheta(x) \sim x$$
sowie die Fehlerabschätzung von Rosser/Schoenfeld
$$x \cdot \left(1 - \frac{1}{\ln x}\right) < \vartheta(x) < x \cdot \left(1 - \frac{1}{2 \ln x}\right) \quad \text{für } x \geq 41.$$
Daher ist
$$\prod_{p \leq 2k,\, p \text{ prim}} p = e^{\vartheta(2k)} > 2^k > N.$$
Es können also nicht alle Primzahlen $< 2k$ Teiler von N sein.

Mit einem Aufwand, der höchstens quadratisch in $2k$, also auch polynomial in in ℓ ist, erhält man daher – etwa mit dem Sieb des Eratosthenes – die Liste aller Primzahlen $\leq r$.

Schritt 4

$s := r$. Die Bedingung 2 ist nicht ohne weiteres erfüllt. Daher wird folgendes Verfahren durchgeführt:

- Die (aus Schritt 3 bekannte) Liste der Primzahlen $p < r$ wird durchlaufen.
  - Falls $p = n$: Ausgabe „PRIM“, **Ende**.
    [Das kann nur für „kleine“ n vorkommen, da n exponentiell mit ℓ wächst, r aber nur polynomial.]
  - (Sonst) Falls $p|n$: Ausgabe „ZUSAMMENGESETZT“, **Ende**.

Falls dieser Punkt im Algorithmus noch erreicht wird, ist die Bedingung 2 für s erfüllt.

Bedingung 3

Der Nachweis der Bedingung 3 beginnt mit der Beobachtung, dass $q := \mathrm{Ord}_r\, n > 4\ell^2$.

> Sonst wäre $n^i \equiv 1 \pmod r$ für ein i mit $1 \le i \le 4\ell^2$, also $r \mid n^i - 1 \mid N$, Widerspruch.

Ist nun d ein Teiler von $\frac{\varphi(r)}{q}$, so

$$\begin{aligned} d &\le \frac{\varphi(r)}{q} < \frac{\varphi(r)}{4\ell^2}\,, \\ 2d \cdot \lfloor\sqrt{\frac{\varphi(r)}{d}}\rfloor &\le 2d \cdot \sqrt{\frac{\varphi(r)}{d}} = \sqrt{4d\varphi(r)} < \frac{\varphi(r)}{\ell} < \frac{\varphi(r)}{{}^2\log n}\,, \\ n^{2d\cdot\lfloor\sqrt{\frac{\varphi(r)}{d}}\rfloor} &< n^{\frac{\varphi(r)}{{}^2\log n}} = 2^{\varphi(r)}. \end{aligned}$$

Andererseits ist, da $\varphi(r) \ge 2$,

$$\binom{\varphi(r)+s-1}{s} = \binom{\varphi(r)+r-1}{r} = \binom{2\varphi(r)}{\varphi(r)+1} \ge 2^{\varphi(r)}.$$

Also ist die Bedingung 3 erfüllt.

Schritt 5

Nun wird die Bedingung 4,

$$(X+a)^n \equiv X^n + a \pmod{(n, X^r - 1)}$$

in einer Schleife für $a = 1, \ldots, r$ überprüft. Die Anzahl der Schleifendurchläufe ist höchstens r, also $\le 2k$, also polynomial in ℓ. Bei jedem Schleifendurchlauf wird zweimal binär potenziert, also insgesamt höchstens 4ℓ Mal

multipliziert; multipliziert werden dabei Polynome von einem Grad $< r$ – der also polynomial in ℓ ist – mit Koeffizienten, deren Größe $< n$, deren Bitlänge also auch polynomial in ℓ ist.

- Falls für ein a die Bedingung 4 verletzt ist, wird „ZUSAMMENGESETZT“ ausgegeben, **Ende**.

Ansonsten ist für alle a die Bedingung 4 erfüllt, also n nach dem AKS-Kriterium eine Primzahlpotenz.

Schritt 6

Nun ist noch zu entscheiden, ob n eine echte Primzahlpotenz ist. Da die Primzahlen $\leq r$ keine Teiler von n sind, ist in einer Schleife über t mit $1 \leq t <^r \log n$ zu prüfen:

- Falls $\sqrt[t]{n}$ ganzzahlig: Ausgabe „ZUSAMMENGESETZT“, **Ende**.

Die Zahl der Schleifendurchläufe ist $\leq \ell$, und die Prüfung in jedem Durchlauf ist ebenfalls mit polynomialem Aufwand durchzuführen, wenn man $\lfloor \sqrt[t]{n} \rfloor$ mit einer binären Suche im Intervall $[1 \ldots n-1]$ sucht.

Falls der Algorithmus bis an diese Stelle kommt, wird „PRIM“ ausgegeben, **Ende**.

Damit ist gezeigt:

Hauptsatz 1 *Der AKS-Algorithmus bestimmt, ob n eine Primzahl ist, mit einem Aufwand, der polynomial von $\log n$ abhängt.*