Prof. Dr. M. Helbig
Mathematik – Übung

Hochschule **Rosenheim**
University of Applied Sciences

Chinesischer Restsatz, Kryptographie

Chinesischer Restsatz. Für welche Zahlen $x \in \mathbb{Z}$ gilt

1. $x \equiv 3 \pmod 2$,
   $x \equiv 3 \pmod 5$,
   $x \equiv 3 \pmod 7$.

2. $x \equiv 1 \pmod 2$,
   $x \equiv 3 \pmod 4$.

3. Was sind die jeweils kleinsten positiven x in a) und b)?

Lösung.

1.1 $2 \cdot 5 \cdot 7 = PRIM$

$k_1 = 35 \quad k_2 = 14 \quad k_3 = 10$

$35x_1 \equiv 1 \bmod 2 \quad 14x_2 \equiv 1 \bmod 5 \quad 10x_3 \equiv 1 \bmod 7$

$x_1 = 1 \qquad x_2 = 4 \qquad x_3 = 5$

$x = 35 \cdot 1 \cdot 3 + 14 \cdot 4 \cdot 3 + 10 \cdot 5 \cdot 3 = 423$

$L = \{423 + 70z \mid z \in \mathbb{Z}\}$

Eigener Lösungsversuch.

Rechnen mit großen Zahlen mit CRS. In C kann man mittels des Datentyps `unsigned long long` (nur) 64-bit Zahlen darstellen, also Zahlen x mit

$$0 \leq x \leq 2^{64} - 1 = 18.446.744.073.709.551.615.$$

In der Kryptographie braucht man aber viel größere Zahlen (z.B. in RSA hat der Modul mindestens 1024 Bit, also 309 Dezimalstellen!). Eine Möglichkeit zur Darstellung und dem Rechnen mit großen Zahlen bietet der Chinesische Restsatz:

Wenn $m_1, m_2, \ldots, m_n$ paarweise teilerfremd (und groß, aber noch im Datentyp darstellbar!) sind und $m = m_1 m_2 \cdots m_n$ (riesengroß!), so kann jede Zahl x mit $0 \leq x \leq m$ eindeutig durch ihre Reste $a_k = x \bmod m_k$ $(1 \leq k \leq n)$ repräsentiert werden:

$$x \leftrightarrow (a_1, \ldots, a_n).$$

Beispiel: $m_1 = 9$, $m_2 = 8$. Dann ist etwa $39 = (3, 7)$, denn $39 = 3 \pmod 9$ und $39 = 7 \pmod 8$. Umgekehrt kann zu jedem Tupel mithilfe des chinesischen Restsatzes die Zahl rekonstruiert werden: man erhält $x = 39$ als eindeutige Lösung von

$$x \equiv 3 \,(\text{mod } 9),$$
$$x \equiv 7 \,(\text{mod } 8).$$

Wir addieren und multiplizieren jetzt x, y nicht direkt, sondern rechnen mit den Resten: Sind $x \leftrightarrow (a_1, \ldots, a_n)$, $y \leftrightarrow (b_1, \ldots, b_n)$ zwei Zahlen, so entspricht

$$x + y \leftrightarrow ((a_1 + b_1) \bmod m_1, \ldots, (a_n + b_n) \bmod m_n),$$
$$x \cdot y \leftrightarrow ((a_1 \cdot b_1) \bmod m_1, \ldots, (a_n \cdot b_n) \bmod m_n).$$

Fazit: Anstatt mit den großen Zahlen x, y zu rechnen, wird mit den kleineren Resten gerechnet und am Ende das Ergebnis mit dem chinesischen Restsatz zurückgerechnet. (Die einzelnen Reste können parallel berechnet werden, Stichworte: Nebenläufigkeit und Multicore-Rechner! In der Praxis verwendet man für die Module m_k Zahlen der Form $2^l - 1$, da sich die Modulo-Rechnung für diese Zahlen binär gut implementieren lässt.)

Jetzt zur eigentlichen **Aufgabe** - wir rechnen leider nur exemplarisch mit kleinen Zahlen :'-(. Angenommen, Ihre Lieblingsprogrammiersprache kann nur 4-bit Zahlen x mit

$$0 \leq x \leq 2^4 - 1 = 15$$

darstellen. Sie möchten aber auch 9-bit Zahlen x mit

$$0 \leq x \leq 2^9 - 1 = 511$$

addieren und multiplizieren.

1. Wählen Sie passende Module, um nach obigem Verfahren
2. $459 + 510$ und
3. $459 \cdot 510$ zu berechnen.

Lösung.

Eigener Lösungsversuch.

Lineare Kryptographie. Es wird mit $E(x) = (a \cdot x + b) \bmod 26$ verschlüsselt. Dabei ist $0 \leq x < 26$ die Nummer des zu verschlüsselnden Buchstabens (0-25 entspricht a-z).

1. Sei $a = 3$, $b = 2$. Verschlüsseln Sie „informatik“ und entschlüsseln Sie „ismvkhob“.
2. Geben Sie die Entschlüsselung als Funktion $D(x)$ an. Welche Zahlen $0 \leq a, b < 26$ sind für eine eindeutige Ver-/Entschlüsselung geeignet?
3. Gibt es Zahlen $0 \leq a, b < 26$, so dass $E(x) = D(x)$ für alle $0 \leq x < 26$?

Lösung.

Eigener Lösungsversuch.