Chinesischer Restsatz

Seien $a_i \in R$ paarweise teilerfremd, $I_i = Ra_i$, $a = \prod_i a_i$ und $I = Ra$.
Wir betrachten den Homomorphismus $f : R \to \prod_{i=1}^n R/I_i$,
$x \mapsto (x + I_i)_{1 \leq i \leq n}$.

Es gilt $\ker(f) = \cap_{i=1}^n I_i = \prod_i I_i = I$, so daß $g : R/I \to \prod_{i=1}^n R/I_i$, $g(x+I) \mapsto f(x)$ injektiv ist.

Thm: Der Monomorphismus g ist ein Isomorphismus, liefert

$$R/I \cong \prod_{i=1}^{n} R/I_i.$$

Bew: Es bleibt die Surjektivität zu zeigen. Setze $b_i = \prod_{j \neq i} a_j$. Die b_i haben keinen gemeinsamen Primteiler und daraus folgt $R = \sum_i Rb_i$. Es gibt also $\lambda_j \in R$ mit $1 = \sum_j \lambda_j b_j$. Sei $(c_i + I_i)_i \in \prod_{i=1}^n R/I_i$ mit $c_i \in R$. Setze $c = \sum_j c_j \lambda_j b_j$. Wegen $a_i \mid b_j$ für alle $j \neq i$ gilt $1 = \sum_j \lambda_j b_j = \lambda_i b_i \bmod I_i$ und $c = c_i \bmod I_i$. Folglich $g(c) = (c_i + I_i)_i$ und g ist surjekiv. $\square$

Chinesischer Restsatz

Im Beweis gilt $\lambda_i = b_i^{-1} \bmod a_i$.

Formel von Garner: $a = a_1 a_2$ mit a_1, a_2 prim und teilerfremd. Sei $c = ((c_1 - c_2)(a_2^{-1} \bmod a_1) \bmod a_1)a_2 + c_2$. Dann $c = c_1 \bmod a_1$ und $c = c_2 \bmod a_2$.

Formel von Garner kann induktiv auf beliebig viele a_i verallgemeinert werden.

Englische Abkürzung: CRT (Chinese Remainder Theorem).

Beispiele

Es gilt $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/3\mathbb{Z} \times \mathbb{Z}/5\mathbb{Z} \cong \mathbb{Z}/30\mathbb{Z}$. Wegen $1 = -3 \cdot 5 + 2 \cdot 5 + 2 \cdot 3$ entspricht das Element $(1,2,3)$ dem Wert
$1 \cdot (-3 \cdot 5) + 2 \cdot (2 \cdot 5) + 3 \cdot (2 \cdot 3) = -7 = 23 \bmod 30$.
Check: $23 = 1 \bmod 2$, $23 = 2 \bmod 3$ und $23 = 3 \bmod 5$.

Regel: Für Polynome in $k[x]$ gilt $g(x) \bmod (x - x_0) = g(x_0)$.

Es gilt $k[x]/(x-1)k[x] \times k[x]/(x+1)k[x] \cong k[x]/(x^2-1)k[x]$.
Elemente der linken Seite werden durch Paare $(c_1, c_2) \in k \times k$ repräsentiert.
Wegen $1 = (x+1)/2 - (x-1)/2$ entspricht das Element (c_1, c_2) dem Wert $c = c_1(x+1)/2 - c_2(x-1)/2 = (c_1 + c_2 + (c_1 - c_2)x)/2 \bmod x^2 - 1$.
Check: Für $x = 1$ ergibt sich $c(1) = c_1$ und für $x = -1$ ergibt sich $c(-1) = c_2$.

Lagrange Interpolation

Ist chinesischer Restsatz für lineare Polynome.

Seien $x_i \in k$ für $1 \leq i \leq n$ paarweise verschieden, und $y_i \in k$. Wollen $f \in k[x]$ mit $\deg(f) < n$ und $f(x_i) = y_i$ finden. Ein solches f ist dann notwendigerweise eindeutig bestimmt.

Methode: $b_i(x) = \prod_{j \neq i}(x - x_j)$ und $\lambda_i = \prod_{j \neq i}(x_i - x_j)^{-1}$. Dann gilt $1 = \lambda_i b_i(x_i) = \lambda_i b_i(x) \bmod (x - x_i)$ und $0 = \lambda_j b_j(x_i) = \lambda_j b_j(x) \bmod (x - x_i)$.
Folglich $1 = \sum_{j=1}^n \lambda_j b_j(x)$ beziehungsweise $1 - \sum_{j=1}^n \lambda_j b_j(x) = 0$ zunächst modulo jedem $x - x_i$, wegen der Teilerfremdheit der $x - x_i$ dann auch modulo $\prod_i (x - x_i)$, und dann exakt in $k[x]$ aus Gradgründen. Es gilt nämlich $\deg(\sum_{j=1}^n \lambda_j b_j(x)) < \deg(\prod_i (x - x_i))$, und $1 - \sum_{j=1}^n \lambda_j b_j(x) = 0 \bmod \prod_i (x - x_i)$ impliziert $1 - \sum_{j=1}^n \lambda_j b_j(x) = 0$. Wie im Beweis des chinesischen Restsatz ist schließlich $f(x) = \sum_i y_i \lambda_i b_i(x)$ das gesuchte Polynom.

Shamir's Secret Sharing

Aufgabe:

n Teilnehmer sollen sich ein Geheimnis teilen (z.B. soll ein Masterschlüssel aufgeteilt werden).
Genau dann, wenn $\geq t$ Teilnehmer zusammenarbeiten, sollen sie das Geheimnis rekonstruieren können.

Aufteilen des Geheimnisses:

Sei $y_0 \in k$ das Geheimnis und $x_0 \in k$ öffentlich bekannt.

1. Wähle ein zufälliges Polynom $f \in k[x]$ vom Grad $t-1$ mit $f(x_0) = y_0$.
2. Wähle paarweise verschiedene $x_1, \dots, x_n \in k$ mit $x_i \neq x_0$ und berechne $y_i = f(x_i)$.
3. Gebe (x_i, y_i) für $1 \leq i \leq n$ an den i-ten Teilnehmer.

Shamir's Secret Sharing

Rekonstruktion des Geheimnisses:

$\geq t$ Teilnehmer können das Polynom f eindeutig aus ihren Werten (x_i, y_i) mit Lagrange Interpolation rekonstruieren, und damit den Wert $y_0 = f(x_0)$ berechnen.

Bei $< t$ Teilnehmer ist das Polynom nicht eindeutig bestimmt, $f(x_0)$ kann theoretisch jeden beliebigen Wert aus k annehmen (denn es gibt f mit $\deg(f) \leq t-1$ und $f(x_0) = y'_0$ für beliebige y'_0 wegen der Lagrange Interpolation).

Liefert also informationstheoretische Sicherheit!

Langzahlarithmetik

Müssen mit großen ganzen Zahlen rechnen.

Seien $f, g \in \mathbb{Z}^{\geq 1}$ mit $b = \log(f) \geq \log(g)$. Aufwand in Bitoperationen:

- $f+g,\ f-g$: $O(\log(f) + \log(g)) = O(b)$
- fg: $O(\log(f)\log(g)) = O(b^2)$.
- f^c: $O(\log(c) b^2)$.
- $f = sg + r$: $O(\log(s)\log(g)) = O(b^2)$.
- $\gcd\{f, g\}$: $O(b^2)$.

Analoges gilt für $f, g \in k[x]$ mit $\log$ ersetzt durch $\deg$ und Aufwand in Operationen in k.

Langzahlarithmetik

Es gibt asymptotisch bessere Algorithmen. Die Multiplikation kann z.B. in $O(b^{1.58})$ oder auch $O(b\log(b)^2)$ ausgeführt werden. Spielt in der Kryptographie keine so große Rolle, da die Zahlen nicht groß genug sind.

Es gibt Bibliotheken für die Langzahlarithmetik in $\mathbb{Z}$ und $\mathbb{Z}/n\mathbb{Z}$ bzw. für $k[x]$ und $k[x]/hk[x]$ (z.B. GMP, NTL, Kash, ...).

Langzahlarithmetik mit CRT

Rechnen in $\mathbb{Z}/n\mathbb{Z}$ für $n = pq$ mit p, q Primzahlen, teilerfremd und $\log(p) \approx \log(q)$:

- $\mathbb{Z}/n\mathbb{Z} \cong \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$.
- Statt in $\mathbb{Z}/n\mathbb{Z}$ in $\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$ rechnen.
- Zwischen $\mathbb{Z}/n\mathbb{Z}$ und $\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$ mit mod und Garner hin- und herschalten.

Multiplikation in $\mathbb{Z}/p\mathbb{Z}$ bzw. $\mathbb{Z}/q\mathbb{Z}$ viermal so schnell wie in $\mathbb{Z}/n\mathbb{Z}$. Daher doppelt so schnell in $\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$ wie in $\mathbb{Z}/n\mathbb{Z}$.

Exponentiationen für zufälligen Exponenten noch besser, $b = \log_2(n)$:

- In $\mathbb{Z}/n\mathbb{Z}$: $\approx 3b/2$ b-Bit Multiplikationen.
- In $\mathbb{Z}/p\mathbb{Z}$: $\approx 3b/4$ $b/2$-Bit Multiplikationen.
- In $\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$: $\approx 3b/2$ $b/2$-Bit Multiplikationen.

Daher $3b/2 \cdot b^2$ mit $3b/2 \cdot b^2/4$ vergleichen, also 3-4 mal schneller.

Euler Phi Funktion

Sei $R \in \{\mathbb{Z}, k[x]\}$ für $k = \mathbb{F}_q$ und $n \in R \setminus \{0\}$. Dann $\phi(n) = \#(R/nR)^\times$.

Beispiel:

- Für $R = \mathbb{Z}$ ist $\phi(n)$ die Anzahl der zu n teilerfremden Zahlen m mit $1 \leq m \leq |n| - 1$ (beachte „$1 = \lambda m + \mu n$“).

Eigenschaften:

- $\phi(n) = 0$ genau dann, wenn $n \in R^\times$.
- $\phi(n^e) = \#(R/nR)^\times \cdot \#(R/nR)^{e-1}$ für n Primelement.
- Speziell $\phi(n^e) = (n-1)n^{e-1}$ für $n \in \mathbb{Z}$ Primzahl und $e \in \mathbb{Z}^{\geq 0}$.
- Speziell $\phi(n^e) = (q^d - 1)q^{d(e-1)}$ für $n \in k[x]$ Primpolynom mit $\deg(n) = d$ und $e \in \mathbb{Z}^{\geq 0}$.
- $\phi(n_1 n_2) = \phi(n_1)\phi(n_2)$ für teilerfremde Nichteinheiten $n_1, n_2 \in R$.