

Schnelle Multiplikation (eine Vorlesung zur Computer-Algebra)

Achill Schürmann

Institut für Algebra und Geometrie
Universität Magdeburg

Sommersemester 2003

Schnelle Multiplikation

1. Vorbemerkungen

— R bezeichnet stets einen kommutativen Ring mit Einselement 1.

— Polynome $f = \sum_{i=0}^n f_i x^i \in R[x]$ vom Grad $\leq n$ werden mit dem Koeffizientenvektor $(f_0, \dots, f_n) \in R^{n+1}$ identifiziert.

Schnelle Multiplikation

1 - 1

2. Karatsuba's Algorithmus

Bemerkung 2.1. Für die „klassische Multiplikation“ von zwei Polynomen $f, g \in R[x]$ vom Grad m und n benötigt man $(m+1)(n+1)$ Multiplikationen und m Additionen, insgesamt also $O(mn)$ Ringoperationen.

Beispiel 2.1. $f = f_1x + f_0, g = g_1x + g_0$, mit $f_i, g_i \in R$.
 $\Rightarrow fg = f_1g_1x^2 + (f_1g_0 + f_0g_1)x + f_0g_0$

Mit $h := (f_1 + f_0)(g_1 + g_0)$ gilt

$$f_1g_0 + f_0g_1 = h - f_1g_1 - f_0g_0.$$

Die Berechnung von fg lässt sich daher auch mit nur 3 Multiplikationen (und 4 Additionen) bewerkstelligen!

Schnelle Multiplikation

2 - 2

$O(n^{1.59})$ Ringoperationen.

Satz 2.1. Karatsuba's Algorithmus benötigt maximal $9n^{\log_3 3}$, bzw.

Schnelle Multiplikation

2 - 3

Korollar 2.1. Karatsuba's Algorithmus benötigt für die Multiplikation von ganzen Zahlen, die sich mit maximal n Worten (z.B. zu je 64 Bit) darstellen lassen, $O(n^{1.59})$ Wortoperationen.

Sind $x_0, \dots, x_n \in R$ paarweise verschieden, so ist V invertierbar und der Koeffizientenvektor $f = (f_0, \dots, f_n)$ ist wegen

$$f = V^{-1} (f(x_0), \dots, f(x_n))^T$$

eindeutig durch den Vektor $(f(x_0), \dots, f(x_n))$ bestimmt, bzw. interpolierbar.

Die $n + 1$ Paare $(x_i, f(x_i))$ ergeben eine modulare Darstellung von f .

Zur Multiplikation von Polynomen $f, g \in R[x]$ vom Grad $\leq n$ benötigt man in modularer Darstellung bezüglich $2n + 1$ paarweise verschiedener Punkte nur $2n + 1$, also $O(n)$ Multiplikationen in R . Jede „schnelle“ Transformation in und aus einer modularen Darstellung ermöglicht daher auch eine „schnelle“ Multiplikation!

3. Zur Erinnerung: Darstellung von Polynomen, Evaluation und Interpolation

Zu paarweise verschiedenen $x_0, \dots, x_n \in R$ lässt sich der Vektor $(f(x_0), \dots, f(x_n))$ von $f \in R[x]$ mit Hilfe der Vandermondeschen Matrix

$$V = \text{VDM}(x_0, \dots, x_n) = \begin{pmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{pmatrix}$$

mittels

$$(f(x_0), \dots, f(x_n))^T = V \cdot (f_0, \dots, f_n)^T$$

(aus dem Koeffizientenvektor) evaluieren.

4. Ringe mit primitiven n -ten Einheitswurzeln

Definition 4.1. Sei $n \in \mathbb{N}$.

(i) $w \in R$ ist eine n -te Einheitswurzel, wenn $w^n = 1$.

(ii) $w \in R$ ist eine primitive n -te Einheitswurzel, wenn w n -te Einheitswurzel, $n \cdot 1 \in R$ eine Einheit und $w^t \neq 1$, für alle Primteiler t von n , kein Nullteiler ist.

Beispiel 4.1. Für $R = \mathbb{C}$ und $n \in \mathbb{N}$ ist

$$\{z \in \mathbb{C} : z = e^{\frac{k2\pi i}{n}}, k = 1, \dots, n\}$$

die Gruppe der n -ten Einheitswurzeln. Dabei ist $z = e^{\frac{k2\pi i}{n}}$ genau dann primitiv, wenn $\text{ggT}(k, n) = 1$.

Beispiel 4.2. Für eine Primzahlpotenz q und $n \in \mathbb{N}$ mit $\text{ggT}(q, n) = 1$ besitzt der endliche Körper $\text{GF}(q)$ (mit q Elementen) genau dann eine primitive n -te Einheitswurzel, falls $n|q - 1$ (Übung).

Lemma 4.1. Für einen Ring R mit primitiver n -ter Einheitswurzel w und $l \in \mathbb{N}$, $1 < l < n$, gilt:

(i) $w^l - 1$ ist kein Nullteiler in R

(ii) $\sum_{0 \leq j < n} w^{lj} = 0$

Lemma 5.1. Sei $n \in \mathbb{N}$ und $w \in R$ eine primitive n -te Einheitswurzel. Dann ist w^{-1} eine primitive n -te Einheitswurzel und $V^w \cdot V^{w^{-1}} = nE_n$ (wobei E_n die $n \times n$ Einheitsmatrix ist). Insbesondere gilt also

$$DFT_{-1}^w = n^{-1} DFT^{w^{-1}}$$

5. Diskrete und Schnelle Fourier Transformation

Definition 5.1. Für eine primitive n -te Einheitswurzel $w \in R$ ist die Diskrete Fourier Transformation $DFT_w : R^n \mapsto R^n$ definiert durch

$$DFT_w(f) = DFT_w(f_0, \dots, f_{n-1}) = (f(1), f(w), f(w^2), \dots, f(w^{n-1}))^T$$

Bemerkung 5.1. Mit der Vandermondaschen Matrix

$$V^w = VDM(1, w, w^2, \dots, w^{n-1}) = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ \vdots & w & w^2 & \dots & w^{n-1} \\ \vdots & w^2 & w^4 & \dots & w^{2(n-1)} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ \vdots & w^{n-1} & w^{2(n-1)} & \dots & w^{(n-1)^2} \end{pmatrix}$$

gilt

$$DFT_w(f) = V^w \cdot (f_0, \dots, f_{n-1})^T$$

ALGORITHMUS (SCHNELLE FOURIER TRANSFORMATION)

Input: $f \in R[x]$ vom Grad $< n = 2^k$, $k \in \mathbb{N} \cup \{0\}$,

eine primitive n -te Einheitswurzel $w \in R$ und w_2, \dots, w_{n-1} .

Output: $DFT_w(f) = (f(1), f(w), \dots, f(w^{n-1})) \in R^n$.

1. if $n = 1$ then return f_0

2. Berechne $r_0 = \sum_{0 \leq j < \frac{n}{2}} f_j + f_{j+\frac{n}{2}}$ und $r_1 = \sum_{0 \leq j < \frac{n}{2}} f_j - f_{j+\frac{n}{2}}$

3. Berechne rekursiv $DFT_{w_2}(r_0)$ und $DFT_{w_2}(r_1)$

4. return $(r_0(1), r_1(1), r_0(w_2), r_1(w_2), \dots, r_0(w_{n-2}), r_1(w_{n-2}))$

Satz 5.1. Sei $n = 2^k$, $k \in \mathbb{N}$, w eine primitive n -te Einheitswurzel und $f \in R[x]$ ein Polynom vom Grad $< n$. Dann berechnet der SFT-Algorithmus $DFT_w(f)$ und benötigt dazu $n \log n$ Additionen und $\frac{n}{2} \log n$ Multiplikationen von Potenzen von w , also insgesamt $\frac{3}{2}n \log n$ Ringoperationen.

Definition 5.2. Ein Ring R unterstützt den SFT-Algorithmus, falls es in R für alle $k \in \mathbb{N}$ eine 2^k -te primitive Einheitswurzel gibt.

Beispiel 5.1. \mathbb{C} unterstützt den SFT-Algorithmus.

Lemma 6.1. Für Polynome $f, g \in R[x]$ vom Grad $< n$ gilt

$$DFT_w(f * g) = DFT_w(f) \cdot DFT_w(g),$$

wobei „ \cdot “ die komponentenweise Multiplikation (von Vektoren aus R^n) bezeichnet.

6. Die (schnelle) Faltung

Im folgenden wird ein „schneller“ Algorithmus zur Multiplikation von Polynomen in $R[x]/\langle x^n - 1 \rangle$ beschrieben. R ist dabei SFT-unterstützend.

Definition 6.1. Die Faltung von zwei Polynomen $f = \sum_{0 \leq i < n} f_i x^i$ und $g = \sum_{0 \leq j < n} g_j x^j$ in $R[x]$ ist definiert als

$$h = f * g = \sum_{0 \leq k < n} h_k x^k \in R[x] \quad \text{mit} \quad h_k = \sum_{i+j \equiv k \pmod n} f_i g_j$$

Bemerkung 6.1. Für $f, g \in R[x]$ gilt

$$f * g \equiv fg \pmod{x^n - 1}.$$

ALGORITHMUS (SCHNELLE FALTUNG)

Input: $f, g \in R[x]$ vom Grad $< n = 2^k$, $k \in \mathbb{N} \cup \{0\}$, und eine primitive n -te Einheitswurzel $w \in R$

Output: $f * g \in R[x]$.

1. Berechne w^2, \dots, w^{n-1} .

2. $a = DFT_w(f)$, $b = DFT_w(g)$

3. $c = a \cdot b$

4. return $DFT_{w^{-1}}^{-1}(c) = \frac{1}{n} DFT_{w^{-1}}^{-1}(c)$

Satz 6.1. Ist R ein SFT-unterstützender Ring und $n = 2^k$, $k \in \mathbb{N}$, dann benötigt die schnelle Faltung zur Multiplikation von zwei Polynomen f, g in $R[x]/\langle x^n - 1 \rangle$, bzw. in $R[x]$ mit $\text{grad}(fg) > n$, maximal $\frac{7}{9}n \log n + O(n)$ Ringoperationen.

7. Ausblicke, Zusammenfassung

Bemerkung 7.1. Die schnellste bekannte Methode zur Multiplikation ganzer Zahlen von SCHÖNHAGE und STRASSEN (1971) „basiert“ auf dem SFT-Algorithmus.

| | |
|------------------------------|---|
| Algorithmus | $O(n^2)$ |
| „Multiplikationszeit“ $M(n)$ | $O(n^2)$ klassisch Karatsuba $O(n^{1.59})$ SFT Multiplikation $O(n \log n)$ Schönhage & Strassen $O(n \log n \log \log n)$ |