

Zwei *divide-and-conquer* Algorithmen der Arithmetik

- KARATSUBA-Multiplikation von Polynomen (und von ganzen Zahlen)

Multiplikation zweier Polynome vom Grad $\deg f, \deg g < 2n$

$$f(x) = \sum_{i=0}^{2n-1} f_i x^i$$

$$g(x) = \sum_{j=0}^{2n-1} g_j x^j$$

$$(f * g)(x) = \sum_{k=0}^{2n-2} \left(\sum_{0 \leq i \leq k} f_i \cdot g_{k-i} \right) x^k$$

wird zurückgeführt auf drei Multiplikationen von Polynomen vom Grad $< n$

$$\begin{aligned} f(x) &= a(x) + x^n b(x) \\ g(x) &= c(x) + x^n d(x) \\ f(x)g(x) &= a(x)c(x) + x^n (a(x)d(x) + b(x)c(x)) + x^{2n}b(x)d(x) \\ &= u(x) + x^n (w(x) - u(x) - v(x)) + x^{2n}v(x) \end{aligned}$$

wobei

$$\begin{aligned} u(x) &:= a(x)c(x) \\ v(x) &:= b(x)d(x) \\ w(x) &:= (a(x) + b(x))(c(x) + d(x)) \end{aligned}$$

Diese Idee ist rekursiv anzuwenden, bis man Polynome vom Grad 0 (Konstanten) oder Polynome eines kleinen Grades zu multiplizieren hat.

Die Idee überträgt sich wörtlich auf die Multiplikation ganzer Zahlen (Übertrag beachten!)

```
karatsuba := proc(f,g,x,m)
local a,b,c,d,u,v,w,deg;
if m=0 then RETURN(f*g) fi;
deg := 2^(m-1);
a := sum('coeff(f,x,i)*x^i', 'i'=0..deg-1);
b := sum('coeff(f,x,i+deg)*x^i', 'i'=0..deg-1);
c := sum('coeff(g,x,i)*x^i', 'i'=0..deg-1);
d := sum('coeff(g,x,i+deg)*x^i', 'i'=0..deg-1);
u := karatsuba(a,c,x,m-1);
v := karatsuba(b,d,x,m-1);
w := karatsuba(a+b,c+d,x,m-1);
RETURN(expand(u+(w-u-v)*x^deg+v*x^(2*deg)));
end;
kara_mult := proc(f,g,var)
local df,dg,bound;
df := degree(f,var);
dg := degree(g,var);
bound := ceil(simplify(log[2](max(df,dg)+1)));
karatsuba(f,g,var,bound);
end;
```

Komplexitätsanalyse (für $n = 2^m$):

$$t_K(n) := \begin{cases} \text{(worst-case) Anzahl der Additionen und Multiplikationen} \\ \text{im Koeffizientenbereich} \\ \text{bei input-Grad } < n \end{cases}$$

$$t_K(2n) \leq 3 \cdot t_K(n) + 8n, \quad t(1) = 1$$

Lösung der divide-and-conquer-Rekursion

$$t(2n) = 3 \cdot t(n) + \Theta(n), \quad t(1) = \Theta(1)$$

führt zu

$$\Rightarrow t_K(n) \in \Theta(n^{\log_2 3})$$

Beachte: $\log_2 3 = 1.584962501 \dots$

- Siehe HEUN, GA, Abschnitt 7.6. für eine genaue Analyse im Fall der Multiplikation ganzer Zahlen (Theorem 7.45)

$$t_K(n) \leq 41 \cdot n^{\log 3}$$

und Optimierung des Rekursionsabbruchs (Theorem 7.46)

$$t_K(n) \leq 11 \cdot n^{\log 3}$$

- Sei der Arbeit von

A. A. KARATSUBA, Y. P. OFMAN, Multiplication of multidigit numbers on automata, *Dokl. Acad. Nauk SSR* (145 (1962), 293–294

ist es gelungen, für die Komplexität der Multiplikation ganzer Zahlen eine obere Schranke von $\mathcal{O}(n \log n \log \log n)$ zu finden:

A. SCHÖNHAGE, V. STRASSEN, Schnelle Multiplikation grosser Zahlen, *Computing* 7 (1971), 281–292,

basierend auf der Technik der Schnellen Fourier-Transformation (FFT).

- Siehe D. E. KNUTH, TAOCP vol. 2 für Details.

- STRASSENS Matrix-Multiplikation

Ausgangspunkt: die Multiplikation von zwei (2×2) -Matrizen läßt sich mit 7 Multiplikationen im Koeffizientenbereich ausführen — statt mit 8 Multiplikationen nach “Schulmethode”:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix} \\ = \begin{pmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{pmatrix}$$

Wichtig: das gilt über jedem Ring!

Man berechnet zunächst 7 Produkte

$$c_1 = (a_{12} - a_{22})(b_{21} + b_{22})$$

$$c_2 = (a_{11} + a_{22})(b_{11} + b_{22})$$

$$c_3 = (a_{11} - a_{21})(b_{11} + b_{12})$$

$$c_4 = (a_{11} + a_{12})b_{22}$$

$$c_5 = a_{11}(b_{12} - b_{22})$$

$$c_6 = a_{22}(b_{21} - b_{11})$$

$$c_7 = (a_{21} + a_{22})b_{11}$$

und danach die d_{11}, \dots, d_{22} durch

$$d_{11} = c_1 + c_2 - c_4 + c_6 \quad d_{12} = c_4 + c_5$$

$$d_{21} = c_6 + c_7 \quad d_{22} = c_2 - c_3 + c_5 - c_7$$

Der Witz der (rekursiven) Angelegenheit: die Koeffizienten $a_{ij}, b_{i,j}, \dots$ können selbst wieder Matrizen sein, d.h.

Mit STRASSENS Idee erreicht man:

Reduktion einer Matrixmultiplikation für zwei $(2n \times 2n)$ -Matrizen auf 7 Matrixmultiplikationen von $(n \times n)$ -Matrizen, dazu 18 Additionen/Subtraktionen von $(n \times n)$ -Matrizen

```

strassen := proc(A::matrix,B::matrix,m)
local dim,A11,A12,A21,A22,B11,B12,B21,B22,
      C1,C2,C3,C4,C5,C6,C7,D11,D12,D21,D22;
if m=0 then RETURN(evalm(A*B)) fi;
dim := 2^(m-1);
A11 := submatrix(A,1..dim,1..dim);
B22 := submatrix(B,dim+1..2*dim,dim+1..2*dim); ... (etc.)
C1 := strassen(evalm(A12-A22),evalm(B21+B22),m-1);
C2 := strassen(evalm(A11+A22),evalm(B11+B22),m-1);
C3 := strassen(evalm(A11-A21),evalm(B11+B12),m-1);
C4 := strassen(evalm(A11+A12),B22,m-1);
C5 := strassen(A11,evalm(B12-B22),m-1);
C6 := strassen(A22,evalm(B21-B11),m-1);
C7 := strassen(evalm(A21+A22),B11,m-1);
D11 := evalm(C1+C2-C4+C6);
D12 := evalm(C4+C5);
D21 := evalm(C6+C7);
D22 := evalm(C2-C3+C5-C7);
RETURN(stackmatrix(concat(D11,D12),concat(D21,D22)));
end;

```

Komplexitätsanalyse (für $n = 2^m$):

$$t_S(n) := \begin{cases} \text{(maximale) Anzahl der arithmetischen Operationen im Koeffizien-} \\ \text{tenbereich für die Multiplikation von zwei } (n \times n)\text{-Matrizen} \end{cases}$$

$$t_S(2n) = 7 \cdot t(n) + 18 \cdot n^2, \quad t_S(1) = 1$$

$$\Rightarrow t_S(n) \in \Theta(n^{\log_2 7})$$

Beachte: $\log_2 7 = 2.81 \dots!$

Traditionelle Matrixmultiplikation benötigt für die Multiplikation von zwei $(n \times n)$ -Matrizen

$n^2 \cdot n$ Multiplikationen und $n^2 \cdot (n - 1)$ Additionen von Koeffizienten

ist also ein $\mathcal{O}(n^3)$ -Verfahren.

Seit

V. STRASSEN, Gaussian elimination is not optimal,
Numerische Mathematik 13 (1969), 354–356

ist es gelungen, Verfahren bis zu $\mathcal{O}(n^{2.38\dots})$ zu finden (COPPERSMITH,
WINOGRAD, 1990)

- Siehe Abschnitt 7.8 in HEUN, GA, für eine genaue Analyse (Theorem 7.50)

$$t_S(n) \leq \frac{141}{5} \cdot n^{\log 7}$$

und Optimierung des Abbruchpunktes für die Rekursion (Theorem 7.51)

$$t_S(n) \leq 4.62 \cdot n^{\log 7}$$

- Siehe Kapitel 31.2 in CORMEN/LEISERSON/RIVEST für einen Rekonstruktionsversuch der STRASSENSchen Idee.