

Oft muss man bei der Analyse von Algorithmen Rekursionsgleichungen lösen

z.B. bei Mergesort

$$C(2n) = 2 \cdot C(n) + 2n - 1$$

↑
Vergleiche
im Worst
Case

$2 \times$
Vergleiche
für jeweils
das halbe
Array, denn
sind beide
Hälften sortiert

Vergleiche zum
Verschmelzen der
sortierten Hälften

} Rekurs.
Gleich

$$C(2) = 1$$

↑
hat hoc gelöst, es war $C(n) \in O(n \log n)$

10.3 Beschleunigung durch Aufteilung (Divide and Conquer)

allgemeine Sätze für die Lösung von Rekursionsgleichungen

SATZ: Aufteilungs-Beschleunigungs-Satz (einfache Variante)

Seien $a > 0$, b, c natürliche Zahlen und sei folgende

Rekursionsgleichung gegeben

- $f(a \cdot n) \leq \underbrace{b \cdot f(n)}_{\substack{\text{b Teilprobleme} \\ \text{der Größe} \\ \text{n lösen}}} + \underbrace{c \cdot n}_{\substack{\text{Aufwand zum} \\ \text{Aufteilen und} \\ \text{Zusammensehen} \\ \text{der gesamten Lösung} \\ \text{aus Teillösungen}}}$

↑
Aufwand zur Lösung eines Problems der Größe $a \cdot n$

- $f(1) \leq \frac{c}{a}$

Dann gilt:

$$f(n) \in \begin{cases} O(n) & \text{wenn } a > b \\ O(n \log n) & a = b \\ O(n^{\log_a b}) & a < b \end{cases}$$

Beispiel: Mergesort $C(2n) = 2 \cdot C(n) + 2n - 1$
 $C(2) = 1$

$$\left. \begin{array}{l} f(2n) \leq 2 \cdot f(n) + 2n \\ f(1) \leq 1 \end{array} \right\} \text{SATZ} \Rightarrow f(n) \in O(n \log n)$$

Beweis: nehmen an, dass n eine Potenz von a ist
 also $n = a^k$ (später allgemeine n wie bei Mergesort)

Beh: Für $n = a^q$ gilt

$$f(n) \leq \frac{c}{a} \cdot n \sum_{i=0}^q \left(\frac{b}{a}\right)^i$$

Beweis durch Induktion nach q

Ind. Anfang $\underline{q=0}$ \Rightarrow rechte Seite ist $\frac{c}{a}$
 $n=1$ } \Rightarrow Beh

Ind. Ann. \square sei richtig für $0, 1, 2, \dots, q$

Schluss auf $q+1$

$$\begin{aligned} f(a^{q+1}) &\stackrel{\text{Rek. Gleich}}{\leq} b \cdot \underline{f(a^q)} + c \cdot a^q \\ &\stackrel{\text{I.V.}}{\leq} b \left[\frac{c}{a} \cdot a^q \sum_{i=0}^q \left(\frac{b}{a}\right)^i \right] + c \cdot a^q \\ &= \frac{c}{a} a^{q+1} \frac{b}{a} \sum_{i=0}^q \left(\frac{b}{a}\right)^i + \frac{c}{a} \cdot a^{q+1} \\ &= \frac{c}{a} a^{q+1} \sum_{i=0}^q \left(\frac{b}{a}\right)^{i+1} + \frac{c}{a} \cdot a^{q+1} \\ &= \frac{c}{a} a^{q+1} \sum_{i=1}^{q+1} \left(\frac{b}{a}\right)^i + \frac{c}{a} a^{q+1} \cdot \left(\frac{b}{a}\right)^0 \\ &= \frac{c}{a} a^{q+1} \cdot \sum_{i=1}^{q+1} \left(\frac{b}{a}\right)^i \quad \checkmark \end{aligned}$$

$i=0$

Betrachte die 3 Fälle

$$1. \quad a > b \quad \Rightarrow \quad \sum_{i=0}^q \underbrace{\left(\frac{b}{a}\right)^i}_{< 1} \leq \underbrace{\sum_{i=0}^{\infty} \left(\frac{b}{a}\right)^i}_{\text{geometrische Reihe}} = \frac{1}{1 - \frac{b}{a}} = \frac{a}{a-b} =: k_1$$

$$\Rightarrow f(n) \leq \frac{c}{a} \cdot k_1 \cdot n \in O(n)$$

2. $a=b$

$$f(n) \leq \frac{c}{a} \cdot n \sum_{i=0}^q \underbrace{\left(\frac{b}{a}\right)^i}_1 = \frac{c}{a} \cdot n \cdot (1 + \log_a n)$$

$$= \frac{c}{a} \cdot n \log_a n + \frac{c}{a} \cdot n$$

für $n \geq a$ ist $\log_a n \geq 1$

$$\Rightarrow f(n) \leq \frac{c}{a} \cdot n \log_a n + \frac{c}{a} \cdot n \log_a n \quad \text{für } n \geq a$$

$$= \frac{2c}{a} \cdot n \log_a n = \frac{2c}{a} \log_2 a \cdot \underbrace{\log_2 n}_{\text{Konstant}} \in O(n \log n)$$

$$3. \quad f(n) \leq \frac{c}{a} \cdot n \sum_{i=0}^q \left(\frac{b}{a}\right)^i$$

$$= \frac{c}{a^n} a^n \sum_{i=0}^q \left(\frac{b}{a}\right)^i$$

$$= \frac{c}{a^n} \sum_{i=0}^q b^i a^{n-i} = \frac{c}{a} \left[b^0 a^n + b^1 a^{n-1} + \dots + b^{q-1} a^1 + b^q \cdot a^0 \right]$$

$$= \frac{c}{a^n} \sum_{i=0}^q b^{q-i} a^i$$

$$= \frac{c}{a^n} b^q \sum_{i=0}^q \left(\frac{a}{b}\right)^i$$

$$a < b \Rightarrow \frac{a}{b} < 1 \Rightarrow \sum < k_2 \text{ wie in}$$

Fall 1

(Abschätzung durch geometrische Reihe)

$$\leq \frac{c}{a^n} k_2 \cdot b^q$$

$$b^q = (a^{\log_a b})^q \stackrel{n = a^q}{\downarrow} = (a^{\log_a b})^{\log_a n}$$

$$= (a^{\log_a n})^{\log_a b} = n^{\log_a b}$$

$$= \frac{c}{a^n} k_2 n^{\log_a b} \in O(n^{\log_a b}) \quad \square$$

SATZ Aufteilungs- Beschränkungs-Satz (alt je meine Version)

Seien $a > 0$, $b > 0$ natürliche Zahlen und sei die Rek. Gleich.

$$f(a \cdot n) = b \cdot f(n) + \underbrace{g(n)}$$

↳ was $c \cdot n$ im speziellen Satz

$$f(1) \text{ constant}$$

Dann hat $f(n)$ folgendes asymptotisches Wachstum:

1. Ist $g(n) \in \mathcal{O}(n^{\log_a b - \epsilon})$ für eine Konstante $\epsilon > 0$

$$\text{so ist } f(n) \in \Theta(n^{\log_a b})$$

2. Ist $g(n) \in \Theta(n^{\log_a b})$ so ist $f(n) \in \Theta(n^{\log_a b} \cdot \log n)$

3. Ist $g(n) \in \Omega(n^{\log_a b + \epsilon})$ für eine Konstante $\epsilon > 0$

und erfüllt $g(n)$ die Regularitätsbedingung

$$b \cdot g\left(\frac{n}{a}\right) \leq c \cdot g(n) \text{ für eine Konstante } c < 1$$

und alle $n \geq n_0$

$$\text{so ist } f(n) \in \Theta(g(n))$$

Kann vergleicht Wachstum von $g(n)$ mit $n^{\log_a b}$

1. Wachstum von $g(n)$ ist echt kleiner als $n^{\log_a b}$
(um Faktor n^ϵ)

$$\text{so ist } f(n) = \Theta(n^{\log_a b})$$

2. Ist Wachstum von $g(n)$ genauso wie $n^{\log_a b}$
 so kommt genau über von 1 ein \log_a Faktor beim Wachstum
 von f dazu

3. Ist Wachstum von $g(n)$ echt größer als $n^{\log_a b}$
 (um Faktor n^ϵ) und gilt Regularitätsbed.
 so ist Wachstum von f gleich Wachstum von g

Beispiele:

$$1. \quad f(3n) = 9 \cdot f(n) + \overset{g(n)}{n}$$

$$n^{\log_a b} = n^2 \quad g(n) = n \in \mathcal{O}(n^{2-\epsilon}) \quad \epsilon=1$$

$$\Rightarrow \text{Fall 1 trifft zu} \quad \Rightarrow f(n) \in \Theta(n^{\log_a b}) = \Theta(n^2)$$

$$2. \quad f\left(\frac{3}{2}n\right) = f(n) + 1$$

$$a = \frac{3}{2} \quad b = 1 \quad g(n) = 1$$

$$n^{\log_a b} = n^{\log_{3/2} 1} = n^0 = 1 = g(n) \quad \Rightarrow \text{Fall 2 trifft zu}$$

$$\Rightarrow f(n) \in \Theta\left(\underbrace{n^{\log_a b}}_1 \cdot \log n\right) = \Theta(\log n)$$

$$3. \quad f(4n) = 3f(n) + n \log n$$

$$a=4 \quad b=3 \quad g(n) = n \log n \quad \left. \vphantom{g(n)} \right\} g(n) \in \Omega(n^{\log_a b + \epsilon})$$

$$n^{\log_a b} = n^{\log_4 3} \approx n^{0,793} \quad \text{mit } \epsilon = 0,2$$

\Rightarrow Fall 3

Regularitätsbedingung überprüfen

$$b \cdot g\left(\frac{n}{a}\right) \leq c \cdot g(n) \quad \forall n \geq n_0 \quad (c, n_0 \text{ geeignet gewählt})$$

$c < 1$

$$b \cdot g\left(\frac{n}{a}\right) = 3 \cdot \frac{n}{4} \log\left(\frac{n}{4}\right) < \frac{3}{4} n \log n \quad \forall n \geq 1$$

$\underbrace{\hspace{10em}}_{c < 1}$

erfüllt

$$\Rightarrow f(n) \in \Theta(g(n)) = \Theta(n \log n)$$

4. $f(2n) = 2f(n) + n \log n$

$$a=2 \quad b=2 \quad g(n) = n \log n$$

$n^{\log_a b} = n$ ABER $g(n)$ ist nicht nur linear Faktor n^ϵ
größer als n
da $\log n \in \mathcal{O}(n^\epsilon)$

\Rightarrow können Fall 3 nicht anwenden

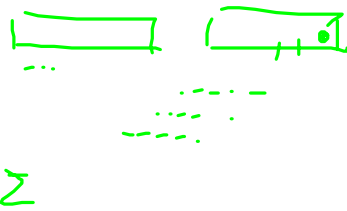
dieses Beispiel wird nicht von Satz abgedeckt

10.3.2 Multiplikation von Dualzahlen

[nur von Bedeutung für Rechnerarithmetiken mit sehr, sehr großen Zahlen]

x, y seien n stellige Dualzahlen

normale Mult.



Schulmethode $\Theta(n^2)$

Operationen

Aufteilung + Beschränkung anwenden

Idee: Multiplikation zurückführen auf Mult. von $n/2$ -langen Zahlen

$$x \quad \boxed{a \quad b} \quad x = a \cdot 2^{n/2} + b$$

$$y \quad \boxed{c \quad d} \quad y = c \cdot 2^{n/2} + d$$

$$\Rightarrow x \cdot y = \underbrace{ac}_{\substack{\uparrow \\ \text{Mult von } n/2 \text{ Zahlen}}} 2^n + \underbrace{(ad + bc)}_{\substack{\text{Addition} \\ \text{Shifts}}} 2^{n/2} + \underbrace{bd}_{\text{Addition}}$$

$\underbrace{\hspace{10em}}_{\Theta(n)}$

Aufwand $T(n)$ genügt Rekursionsgleichung

$$T(2n) \leq 4 \cdot T(n) + c \cdot n$$

einfacher A-B-S: $a=2$ $b=4$ $=0$ $O(n^{\log_2 a^b}) = O(n^2)$

↑
nicht besser als Schul-Kalk.

Wissen b unter 4 bringen:

Trick:

$u := \overset{\oplus}{(a+b)} \overset{\oplus}{(c+d)}$ ← Problem $a+b, c+d$ können $n/2+1$ lang sein
 $v := a \cdot c$
 $w := b \cdot d$ ← Shift
 $z := v \cdot 2^n + (u - v - w) \cdot 2^{n/2} + w$ Beh. $z = x \cdot y$

↑
3 Multi von kleineren Zahlen + Additionen + Shifts
 $\leq C \cdot n$

Beispiel im Dezimalsystem

$x = 4217$ $y = 5236$

$a = 42$ $b = 17$ $c = 52$ $d = 36$ $a+b = 59$ $c+d = 88$

$u = 59088 = 5192$

$v = 40c = 42 \cdot 52 = 2184$

$w = b \cdot d = 17 \cdot 36 = 612$

$z = 21840000 + (\underbrace{5192 - 2184 - 612}_{2396}) \cdot 10^2 + 612$
 $=$
 $\begin{array}{r} 21840000 \\ + 239600 \\ + 612 \\ \hline \end{array}$

Ansatz

$$a+b = \boxed{\alpha \mid \bar{a}}$$

$n/2$ Bits

$$c+d = \boxed{\beta \mid \bar{c}}$$

$n/2$ Bits

$$(a+b)(c+d) = \alpha\beta \cdot 2^n + (\alpha \cdot \bar{c} + \beta \cdot \bar{a}) 2^{n/2} + \bar{a}\bar{c}$$

↑ Bits mult ↑
 ↑
 Strifen
 ↑
 $n/2$ Zahl
 wird
 mit Bit
 mult,
 $O(n)$

↑
 einzige mult von
 $n/2$ Zahlen

=> 1 mult von $n/2$ Zahlen + linearer Aufwand

=> insgesamt $T(2n) = 3 T(n) + c \cdot n$
 $c \leq 9$

=> $T(n) \in O(n^{\log_2 3}) = O(n^{1,59})$

ähnlich bei Matrixmult: $O(n^{2,81})$ statt $O(n^3)$
 ↑
 Strichmethode

rek. Gleich. $T(2n) = 7 T(n) + 14 n^2$

