Oft muss man bei der Analyse von Algorithmen Rekursionsgleichungen lösen

z.B. bei Mergesort

$$C(2n) = \underbrace{2 \cdot C(n)}_{\text{2x \# Vergleiche für jeweils das halbe Array, danach sind beide Hälften sortiert}} + \underbrace{2n-1}_{\text{\# Vergleiche beim Verschmelzen der sortierten Hälfte}}$$

↑ # Vergleiche im Worst Case

} Rekurs. Gleich

$$C(2) = 1$$

↑ Ad 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

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

  $\uparrow$ 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 $\quad \overset{a}{C(2n)} = \overset{b}{2} \cdot C(n) + 2n - 1$

$C(2) = 1$

$\swarrow$

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

Beweis: nehmen an, dass n eine Potenz von a ist

also $n = a^k$ (spare allgemeine n wie bei Mergesort)

Beh: Für $n = a^q$ gilt

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

Beweis durch Induktion nach q

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

$n = 1$

Ind. Vor. $\square$ sei wahr für $0, 1, 2, \ldots, q$

Schluss auf $q+1$

$$f(a^{q+1}) \overset{\text{Rek. Glied}}{\leq} b \cdot \underline{f(a^q)} + c \cdot a^q$$

$$\overset{\text{I.V.}}{\leq} b \left[\frac{c}{a} \cdot a^q \sum_{i=0}^{q} \left(\frac{b}{a}\right)^i \right] + c \cdot a^q$$

$$= \frac{c}{a} a^{q+1} \frac{b}{a} \sum_{i=0}^{q} \left(\frac{b}{a}\right)^i + \frac{c}{a} \cdot a^{q+1}$$

$$= \frac{c}{a} a^{q+1} \sum_{i=0}^{q} \left(\frac{b}{a}\right)^{i+1} + \frac{c}{a} \cdot a^{q+1}$$

$$= \frac{c}{a} a^{q+1} \sum_{i=1}^{q+1} \left(\frac{b}{a}\right)^i + \frac{c}{a} a^{q+1} \cdot \left(\frac{b}{a}\right)^0$$

$$= \frac{c}{a} a^{q+1} \cdot \sum_{i=0}^{q+1} \left(\frac{b}{a}\right)^i \quad \checkmark$$

$i = 0$

Betrachte die 3 Fälle

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

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

2. $\underline{a = b}$

$\log_a n$

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

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

für $n \geq a$ ist $\log_a n \geq 1$

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

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

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

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

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

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

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

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

(Abschätzung durch geometrische Reihe)

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

$n = a^q$

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

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

$$= \frac{c}{a} k_2 \, n^{\log_a b} \in O\left(n^{\log_a b}\right) \qquad \square$$

SATZ Aufteilungs-Bewältigungs-Satz (alte meine Version)

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

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

↳ war c·n im speziellen Satz

$f(1)$ constant

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

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

   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 $g(n) \in \Omega(n^{\log_a b + \varepsilon})$ für eine Konstante $\varepsilon > 0$

   und erfüllt $g(n)$ die Regularitätsbedingung

   $$b \cdot g\left(\frac{n}{a}\right) \leq c \cdot g(n)$$ für eine Konstante $c < 1$ und alle $n \geq n_0$

   so ist $f(n) \in \Theta(g(n))$

Man 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})

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

2. Ist Wachstum von $g(n)$ genauso wie $n^{\log_a b}$
so kommt gegenüber von 1 ein $\log_2$-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. $f(\underset{a}{3}n) = \underset{b}{9} \cdot f(n) + \overset{g(n)}{\sim n}$

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

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

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

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

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

$\Rightarrow f(n) \in \Theta(\underbrace{n^{\log_a b}}_{1} \cdot \log n) = \Theta(\log n)$

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

$a = 4 \qquad b = 3 \qquad g(n) = n \log n$

$n^{\log_a b} = n^{\log_4 3} \approx n^{0,793}$

$g(n) \in \Omega(n^{\log_a b + \varepsilon})$ mit $\varepsilon = 0,2$

$\Rightarrow$ Fall 3

Regularitätsbedingung überprüfen

$b \cdot g\left(\frac{n}{a}\right) \leq c \cdot g(n) \quad \forall n \geq n_0$ ($c < 1$, c, n_0 geeignet gewählt)

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

$\frac{3}{4} < 1$

erfüllt

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

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

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

$n^{\log_a b} = n$ ABER $g(n)$ ist nicht um einen Faktor n^{ε} größer als n

da $\log n \in O(n^{\varepsilon})$

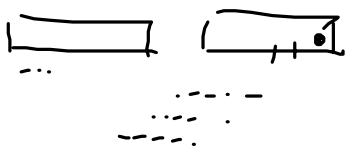
$\Rightarrow$ können Fall 3 nicht anwenden

dieses Beispiel wird nicht vom Satz abgedeckt

10.3.2 Multiplikation von Dualzahlen

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

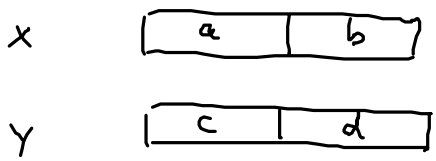
x, y seien n stellige Dualzahlen

normale Mult.  Schulmethode $\Theta(n^2)$ Operationen

Σ

Aufteilung + Beschleunigung anwenden

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



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

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

$$\Rightarrow x \cdot y = \underline{ac}\, 2^n + \underline{(ad + bc)}\, 2^{n/2} + \underline{b \cdot d}$$

$\uparrow$ Mult von $n/2$ Zahlen

$\equiv$ Additionen

Shifts

$O(n)$

Aufwand $T(n)$ genügt Rekursionsgleichung

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

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

nicht besser als Schul-Mult.

Wissen b unter 4 bringen:

Trick: ⊕ ⊕

$u := (a+b)(c+d)$ ← Problem $a+b$, $c+d$ können $n/2+1$ lang sein

$v := a \cdot c$

$w := b \cdot d$ ← shift →

$z := v\,2^n + (u - v - w)\,2^{n/2} + w$ Beh. $\boxed{z = x \cdot y}$

↑

3 Mult von kleineren Zahlen + Additionen + Shifts

$\leq c \cdot n$

Beispiel im Dezimalsystem

$x = 4217 \quad y = 5236$

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

$u = 5988 = 5192$

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

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

$z = 21840000 + (\underbrace{5192 - 2184 - 612}_{2396}) \cdot 10^2 + 612$

$= 21840000$
$+ \quad 239600$
$+ \quad 612$

$\hat{=}$ 22080212

Ansatz

$a+b$ = [α | $\bar{a}$] ($n/2$ Bits) $\qquad$ $c+d$ = [β | $\bar{c}$] ($n/2$ Bits)

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

- $\alpha\beta$: Bits mult
- $\cdot 2^n$: Shiften
- $\alpha \cdot \bar{c}$, $\beta \cdot \bar{a}$: $n/2$ Zahl wird mit Bit mult, $O(n)$
- $\bar{a}\bar{c}$: einzige Mult von $n/2$ Zahlen

$\Rightarrow$ 1 Mult von $n/2$ Zahlen + linearer Aufwand

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

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

aktuell bei Matrixmult: $O(n^{2,81})$ statt $O(n^3)$ ← Schulmethode

Rek. Gleich. $T(2n) = 7\,T(n) + 14\,n^2$


l
Rechteck
h/2
h/2