

13:45 Partnerbörse mit Martino

Kap 8 Rekursion

Aufruf $\hat{=}$ Rekursion

Selbstaufwurf möglich

"Rekursion" (direkt)

Beispiele: ggT

Türme von Hanoi

gelöst wie folgt



1 Start 2 Ziel 3 Helf

→ Schicht $n-1$ Scheiben von 1 nach 3

unterste Scheibe von 1 nach 2



nur dies

"selbst gemacht"

→ Schicht $n-1$ Scheiben von 3 nach 2

|
delegiert

Wichtig bei Rekursion

- Rekursionstiefe



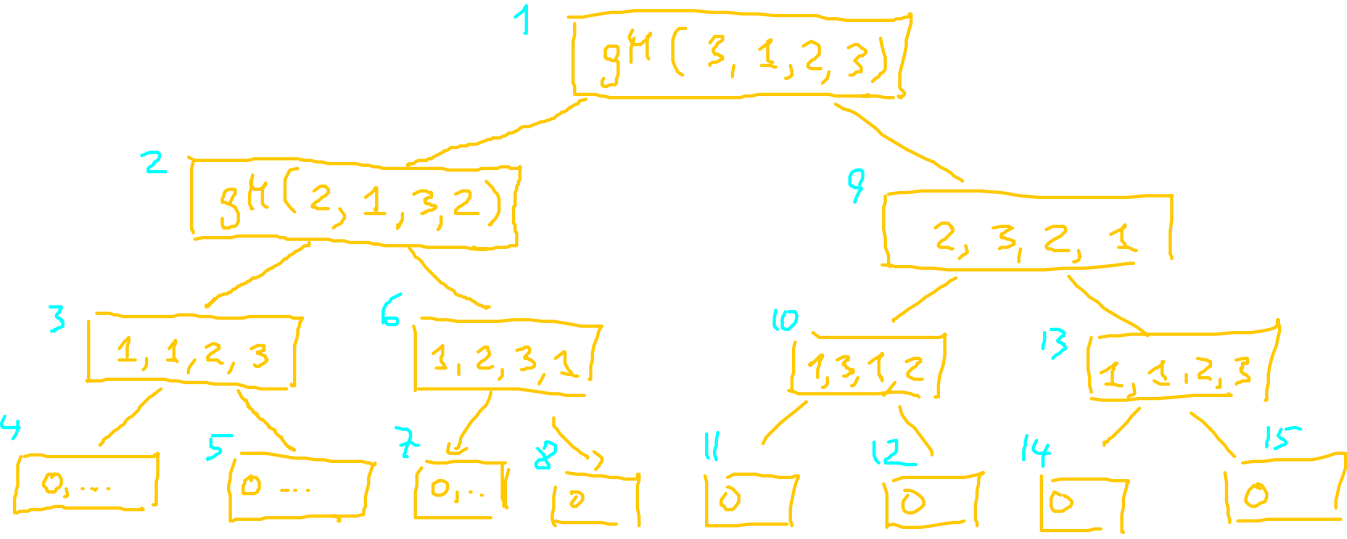
Maß für Speicher

- # rekursiver Aufrufe



Maß für Laufzeit

Beispiel getHoves (3, 1, 2, 3)



- 1 →→ 2
- 1 →→ 3
- 2 →→ 3
- 1 →→ 2
- 3 →→ 1
- 3 →→ 2
- 1 →→ 2

↑
Rekursionsbaum

Rekursionstiefe = 4

rek. Aufrufe = 15

Run-Time-Stack



Korrektheitsbeweis mit Induktion und # Schritte

Rekursion & Induktion gehen Hand in Hand

SATZ: $R(n) :=$ Rekursionstiefe bei n Schritten

$T(n) :=$ # rek. Aufrufe bei n Schritten

Es gilt: $R(n) = n+1$
 $T(n) = 2^{n+1} - 1$

$Z(n) :=$ Anzahl Zeile $Z(n) = 2^n - 1$

Beweis mit Induktion

$n=0 \Rightarrow 1$ Aufruf 0 Zeile \checkmark

IV. Beh sei richtig für $0, 1, 2, \dots, n-1$ bleiben

Schluss auf n

$R(n) = 1 + R(n-1)$
 $\stackrel{IV}{=} 1 + n \checkmark$



$T(n) = 1 + 2 \cdot T(n-1)$
 $\stackrel{IV}{=} 1 + 2 \cdot (2^{n-1+1} - 1) = 1 + 2^{n+1} - 2$
 $= 2^{n+1} - 1 \checkmark$

$Z(n) = Z(n-1) + 1 + Z(n-1)$
 $= 2^{n-1} - 1 + 1 + 2^{n-1} - 1 = 2^n - 1 \checkmark$

zu :

Rekursion bricht erst dann ab
wenn # schreiben = 0 ist

⇒ im Rek. Baum sind alle Ebenen
voll besetzt mit Knoten

⇒ linker Teilbaum und rechter Teilbaum
haben gleiche Knotenzahl und gleiche
Rek. tiefe \square

8.4.3. Die Ackermann Funktion

Beispiel einer sehr stark wachsenden
rekursiv def. Fkt

$$a(m, n) := \begin{cases} n+1 & \text{falls } m=0 \\ a(m-1, 1) & \text{falls } m>0, n=0 \\ a(m-1, a(m, n-1)) & \text{falls } m, n>0 \end{cases}$$

Bsp:

$$\begin{aligned} a(1, 3) &= a(0, a(1, 2)) \\ &= a(0, a(0, a(1, 1))) \\ &= a(0, a(0, a(0, a(1, 0)))) \\ &= a(0, a(0, a(0, \underbrace{a(0, 1)}))) \end{aligned}$$



$$a(0, n) > n$$

$$a(1, n) > n+1$$

$$a(2, n) > 2n$$

$$a(3, n) > 2^n$$

$$a(4, n) > 2^{2^{\dots^2}} \left. \vphantom{2^{2^{\dots^2}}} \right\} n$$

$$a(5, n) \text{ ? keine geschlossene Formel, } a(5, 4) > 10^{10.000}$$

SATZ: für alle m, n terminiert der Aufruf $a(m, n)$
nach endlich vielen Schritten

Beweis: vollst. Induktion (geschloßelte Induktion)

Induktion nach m (äußere Induktion)

Ind. auf $m=0 \Rightarrow$ terminieren für alle n

Ind. hyp: Aufruf terminiert für Werte $< m$ in erster Komp
und für alle n

Schluss auf m

Zeige: voraus $a(m, n)$ terminiert für alle n
(m jetzt fest)

Ind. nach n (innere Induktion)

Ind. hyp. $n = 0$

$$a(m, 0) = \underbrace{a(m-1, 1)}$$

terminiert nach äußerer Ind. hyp.

Ind. hyp: $a(m, k)$ terminiert für $k = 0, 1, 2, \dots, n-1$

Schluss auf n

$$a(m, n) \stackrel{\text{Def}}{=} a(m-1, \underbrace{a(m, n-1)})$$

terminiert nach innerer Ind. Vor.
und liefert eine Zahl N

$$= \underbrace{a(m-1, N)}$$

terminiert nach äußerer Ind. Ann. \square

8.1.4 Ackers Funktion

Beispiel für einfache rekursive Fkt,

von der unbekannt ist ob sie bei allen Aufrufen terminiert

Algorithmus erzeugt Folge von Zahlen

$$a_0 > 0, a_1, a_2, a_3, \dots \quad a_0 \in \mathbb{N}$$

$$\text{mit } a_n := \begin{cases} a_{n-1}/2 & \text{falls } a_{n-1} \text{ gerade} \\ 3 \cdot a_{n-1} + 1 & \text{sonst} \end{cases}$$

und bricht ab, wenn $a_n = 1$ ist

$$\begin{array}{cccccccccccc} a_0 = 3 & \rightarrow & 10 & \rightarrow & 5 & \rightarrow & 16 & \rightarrow & 8 & \rightarrow & 4 & \rightarrow & 2 & \rightarrow & 1 \\ & & a_1 & & a_2 & & a_3 & & a_4 & & a_5 & & a_6 & & a_7 \\ & & & & & & & & & & & & & & 0 \end{array}$$

Wlans Funktion

$$\text{wlans}(a) := \begin{cases} 0 & a = 1 \\ 1 + \text{wlans}(a/2) & a \text{ gerade} \quad a \in \mathbb{N} \\ 1 + \text{wlans}(3a+1) & a \text{ ungerade} \quad a \in \mathbb{N} \end{cases}$$

$$\text{wlans}(3) = 7$$

$\text{wlans}(a)$ zählt Länge der Folge bis man bei 1 ist

8.2 Wo Rekursion zu vermeiden ist

Hinweis: Rekursion oder Iteration

↓
schlecht, wenn → gut, wenn Problem

Rec-Time-Stack
≙ Rek. Tiefe
zu groß wird

eine rekursive Struktur hat
(Türme von Hanoi)

8.2.1 Fakultät

$$\text{fak}(n) = \begin{cases} 1 & n = 0 \\ n \cdot \text{fak}(n-1) & n > 0 \end{cases} \quad n \in \mathbb{N}$$

Rekursionstiefe: $n+1$ ← Stack enthält $n-1$ Kopien
des Parameters n

einfache iterative Lösung: $n! = n(n-1)(n-2) \dots 1 \quad n \geq 1$

rekursiv

if ($n == 0$) fak = 1;
else
fak = $n * \text{fak}(n-1)$;

einfach mit Iteration

fak = 1;
for (k = 2; k <= n; k++) {
 fak *= k;
}

8.2.2. Berechnung ggT

gegen Rekursion: Rek. Baum ist Liste

dafür

Rek. Tiefe klein

$$1 + 2 \log(\max\{a, b\})$$

in Summe: okay

8.2.3. Türme von Hanoi

Rek. Baum ist keine Liste \Rightarrow Iteration möglicherweise schwierig zu realisieren

Rek. Tiefe, # rek. Aufrufe sind groß

$$n+1$$

$$2^{n+1} - 1$$

Struktur des Problems ist rekursiv

← rekursiv

← iterativ

allerdings ist $Z(n) = 2^n - 1$

und jeder Zug muss berechnet werden

in Summe: okay

8.2.4. Berechnung der Fibonacci Zahlen

$$f_0, f_1, f_2, \dots$$

Zahlenfolge mit Bildungsgesetz

Satz: $T_{fib}(n) \geq 2^{\lfloor n/2 \rfloor}$

↑

rek. Aufrufe wächst exponentiell

Beweis: $T_{fib}(\underline{n+1}) \geq 2 \cdot T_{fib}(n-1)$

$$\geq \underset{IV}{2} \cdot 2^{\lfloor n-1/2 \rfloor} = 2 \cdot \underline{\underline{2^{\lfloor (n+1)/2 \rfloor}}}$$

Es gibt einfache iterative Lösung (Skript)