Grundzüge der Informatik 1

Vorlesung 7

Überblick

Überblick

- Wiederholung Teile & Herrsche Verfahren
- Integer Multiplikation
  - Einfacher Algorithmus („Schulmethode“)
  - Einfacher Teile & Herrsche Algorithmus
  - Verbesserter Teile & Herrsche Algorithmus
- Matrix Multiplikation
  - Einfacher Algorithmus
  - Einfacher Teile & Herrsche Algorithmus

Algorithmenentwurf durch Rekursion

Teile & Herrsche Verfahren

- Idee: Teile die Eingabe in mehrere gleich große Teile auf
- Löse das Problem rekursiv auf den einzelnen Teilen
- Füge die Teile zu einer Lösung des Gesamtproblems zusammen

- Beispiel: Sortieren durch Aufteilen in zwei Teile

15	**7**	**6**	**13**	**25**	**4**	**9**	**12**

Schritt 1:
Aufteilen der
Eingabe

Algorithmenentwurf durch Rekursion

Teile & Herrsche Verfahren

- Idee: Teile die Eingabe in mehrere gleich große Teile auf
- Löse das Problem rekursiv auf den einzelnen Teilen
- Füge die Teile zu einer Lösung des Gesamtproblems zusammen

- Beispiel: Sortieren durch Aufteilen in zwei Teile

Schritt 1:
Aufteilen der
Eingabe

15	7	6	13	25	4	9	12

15	7	6	13

25	4	9	12

Algorithmenentwurf durch Rekursion

Teile & Herrsche Verfahren

- Idee: Teile die Eingabe in mehrere gleich große Teile auf
- Löse das Problem rekursiv auf den einzelnen Teilen
- Füge die Teile zu einer Lösung des Gesamtproblems zusammen

- Beispiel: Sortieren durch Aufteilen in zwei Teile

Schritt 2:
Rekursiv Sortieren

15	7	6	13	25	4	9	12

6	7	13	15

4	9	12	25

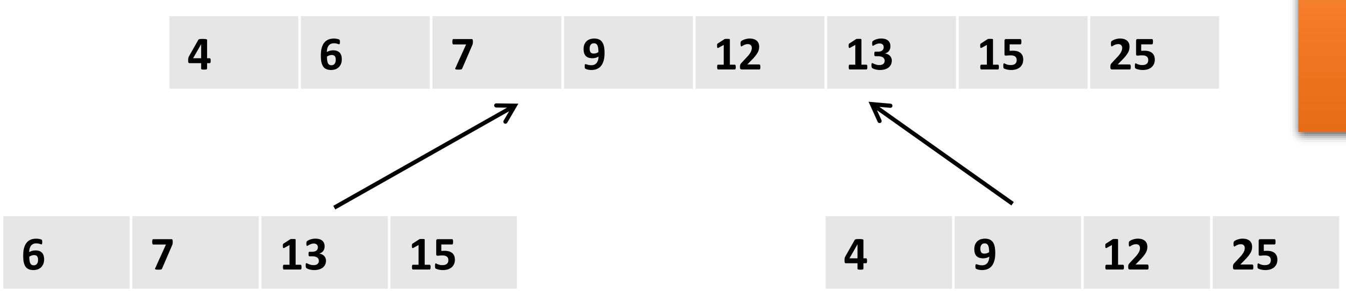
Algorithmenentwurf durch Rekursion

Teile & Herrsche Verfahren

- Idee: Teile die Eingabe in mehrere gleich große Teile auf
- Löse das Problem rekursiv auf den einzelnen Teilen
- Füge die Teile zu einer Lösung des Gesamtproblems zusammen

- Beispiel: Sortieren durch Aufteilen in zwei Teile

Schritt 3:
Zusammenfügen



Laufzeitanalyse
- MergeSort

Laufzeit als Rekursion (n Zweierpotenz)

- $$T(n) \leq \begin{cases} 1 & \text{, falls } n=1 \\ 2\,T(n/2) + cn & \text{, falls } n>1 \end{cases}$$

- Wobei c eine genügend große Konstante ist.

Universität zu Köln

Laufzeitanalyse
- MergeSort

Laufzeit als Rekursion (n Zweierpotenz)

- $T(n) \leq \begin{cases} 1 & \text{, falls } n=1 \\ 2\,T(n/2) + cn & \text{, falls } n>1 \end{cases}$

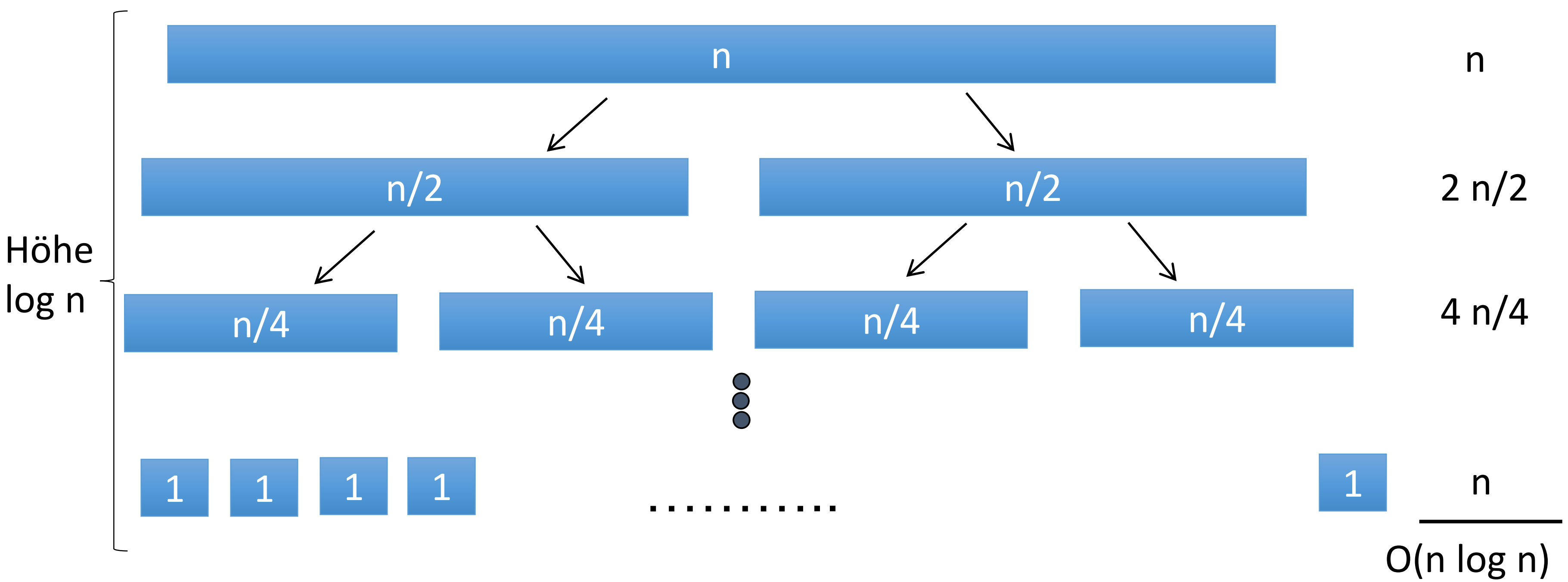
- Wobei c eine genügend große Konstante ist.

Vereinfachung

- c=1

Laufzeitanalyse – grafische Darstellung - MergeSort

- Auflösen von $T(n) \leq 2\,T(n/2) + n$ (Intuition)
- $T(1) = 1$

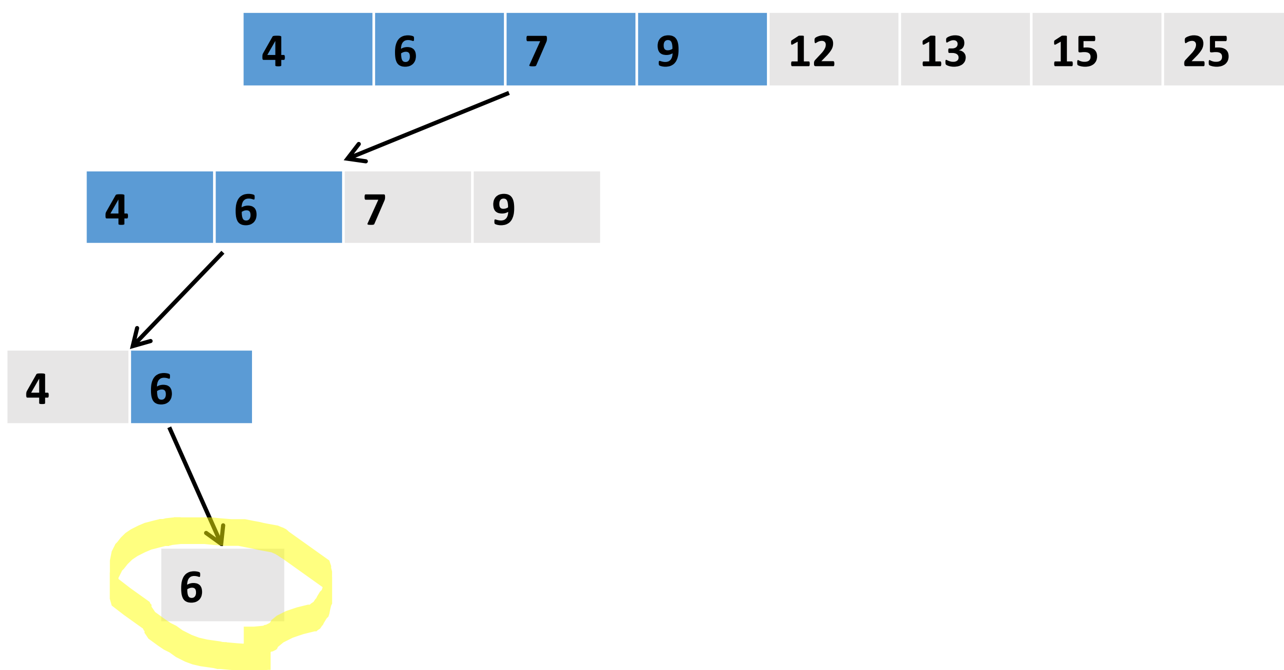


Universität zu Köln

Teile & Herrsche Prinzip

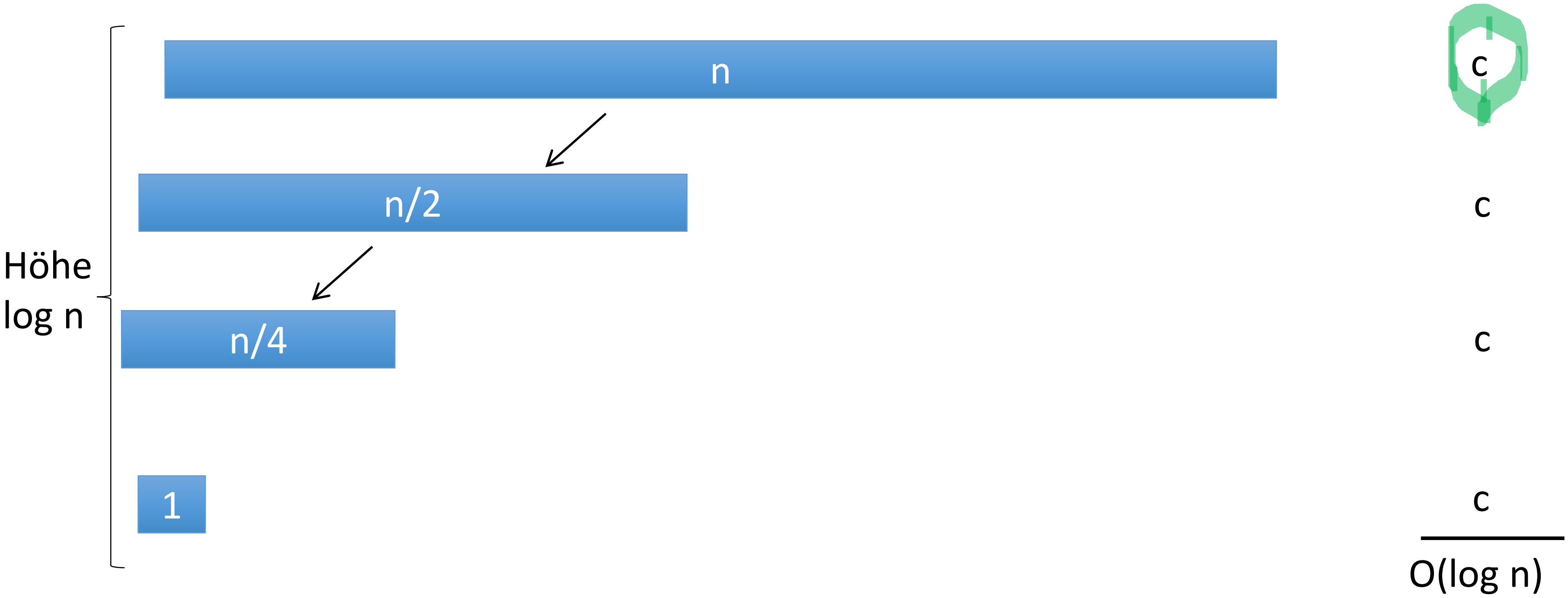
Teile & Herrsche Ansatz (hier: Suche nach 6):

- Teile in der Mitte und suche rekursiv, in dem eindeutigen Teil, der das gesuchte Element enthalten kann



Laufzeitanalyse – grafische Darstellung - Binäre Suche

- Auflösen von $T(n) \leq T(n/2) + c$ (Intuition)
- $T(1) = c$



Teile & Herrsche Algorithmen

Wodurch unterscheiden sich Teile & Herrsche Algorithmen?

- Die Anzahl der Teilprobleme
- Die Größe der Teilprobleme
- Den Algorithmus für das Zusammensetzen der Teilprobleme
- Den Rekursionsabbruch

Wann lohnt sich Teile & Herrsche?

- Kann durch Laufzeitanalyse vorhergesagt werden

Laufzeitanalyse
- Rekursionen

Laufzeiten der Form

$$T(n) = a\,T(n/b) + f(n)$$

(und $T(1) = O(1)$)

Laufzeitanalyse
- Rekursionen

Laufzeiten der Form

$$T(n) = a\,T(n/b) + f(n)$$

(und $T(1) = O(1)$)

Laufzeitanalyse
- Rekursionen

Laufzeiten der Form

$$T(n) = a\,T(n/b) + f(n)$$

Anzahl Unterprobleme

(und $T(1) = O(1)$)

Universität zu Köln

Laufzeitanalyse
- Rekursionen

Laufzeiten der Form

$$T(n) = a\,T(n/b) + f(n)$$

Anzahl Unterprobleme

(und $T(1) = O(1)$)

Größe der Unterprobleme
(bestimmt Höhe des Rekursionsbaums)

Universität zu Köln

Laufzeitanalyse
- Rekursionen

Laufzeiten der Form

$$T(n) = a\, T(n/b) + f(n)$$

Anzahl Unterprobleme

Aufwand für Aufteilen und Zusammenfügen

Größe der Unterprobleme
(bestimmt Höhe des Rekursionsbaums)

(und $T(1) = O(1)$)

Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Integer Multiplikation

- Problem: Multipliziere zwei n-Ziffer Integer
- Eingabe: zwei n-Ziffer Integer X und Y
- Ausgabe: 2n-Ziffer Integer Z mit Z=XY

Darstellung im Rechner

- Wir nehmen an, dass jede Ziffer eine Speicherzelle benötigt!
- Wir können zwei n-Ziffer Integer in $\Theta(n)$ Zeit addieren
- Wir können ein n-Ziffer Integer mit Zehnerpotenz 10^k in $\Theta(n+k)$ Zeit multiplizieren

Universität zu Köln

Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Laufzeit Schulmethode

- n-1 Multiplikationen mit 10^k für $k \leq n$
- n-1 Additionen im Worst-Case:

$$\underbrace{999\ldots99}_{\text{n Ziffern}} \qquad \underbrace{999\ldots99}_{\text{n Ziffern}}$$

9999 · 1111

```
9999
 9999
  9999
   9999
-----------
11108889
```


Universität zu Köln

Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Laufzeit Schulmethode

- n-1 Multiplikationen mit 10^k für $k \leq n$
- n-1 Additionen im Worst-Case:

$$\underbrace{999\ldots99}_{n\ \text{Ziffern}} \qquad \underbrace{999\ldots99}_{n\ \text{Ziffern}}$$

- Jede Addition $\Theta(n)$ Z
- Jede Multiplika ion mit Zweierpotenz Θ(n) Zeit
- Insgesamt $\Theta(n^2)$

Bessere Laufzeit mit Teile & Herrsche?

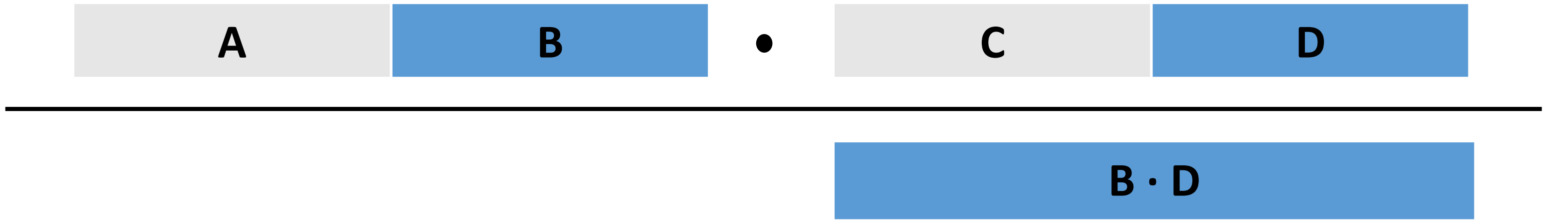
Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Integer Multiplikation

A	B	•	C	D

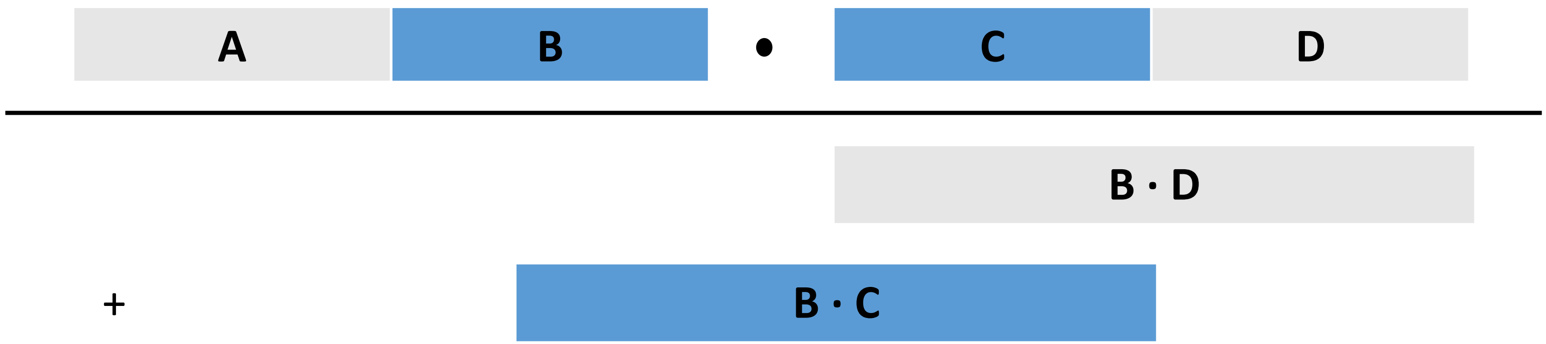
Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Integer Multiplikation



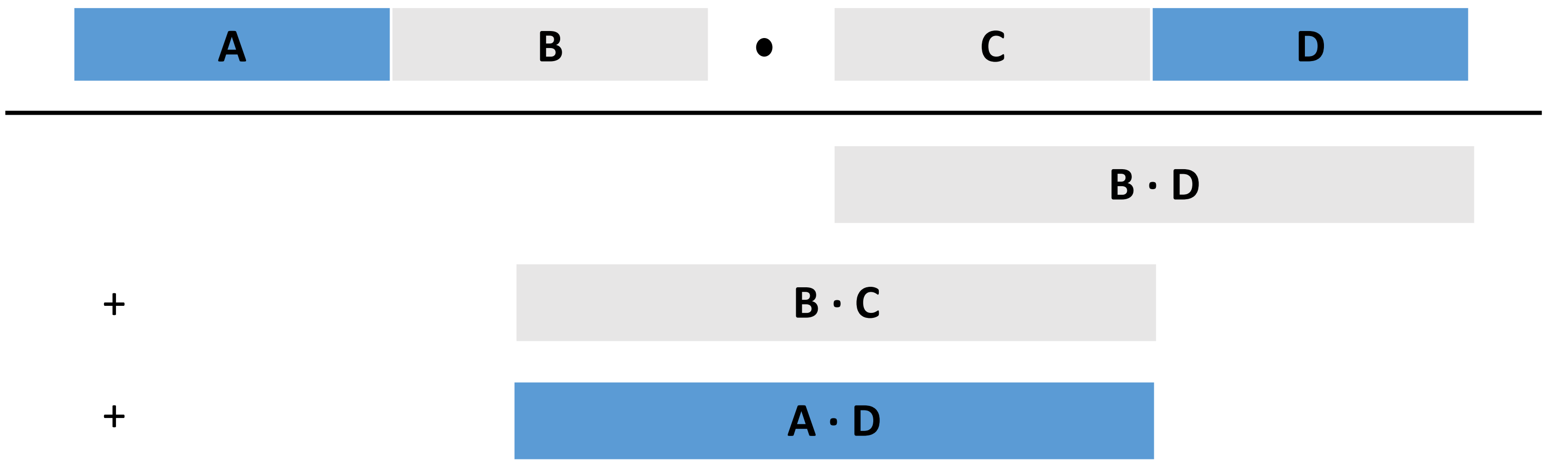
Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Integer Multiplikation



Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

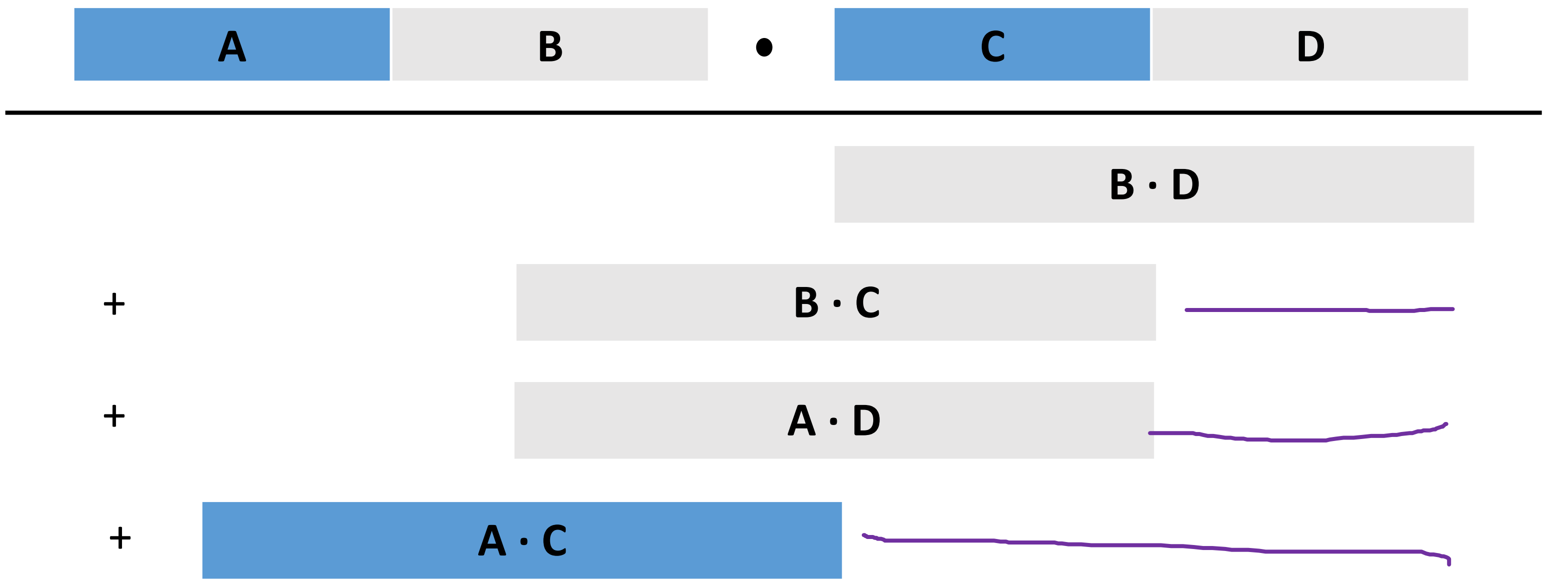
Integer Multiplikation



Universität zu Köln

Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Integer Multiplikation



Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Integer Multiplikation

A	B	•	C	D

B · D

\+ B · C

\+ A · D

\+ A · C

= **Ergebnis**

Universität zu Köln

Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Integer Multiplikation

4 Multiplikationen von n/2-Ziffer Integers

A B • C D

B · D

\+ B · C

\+ A · D

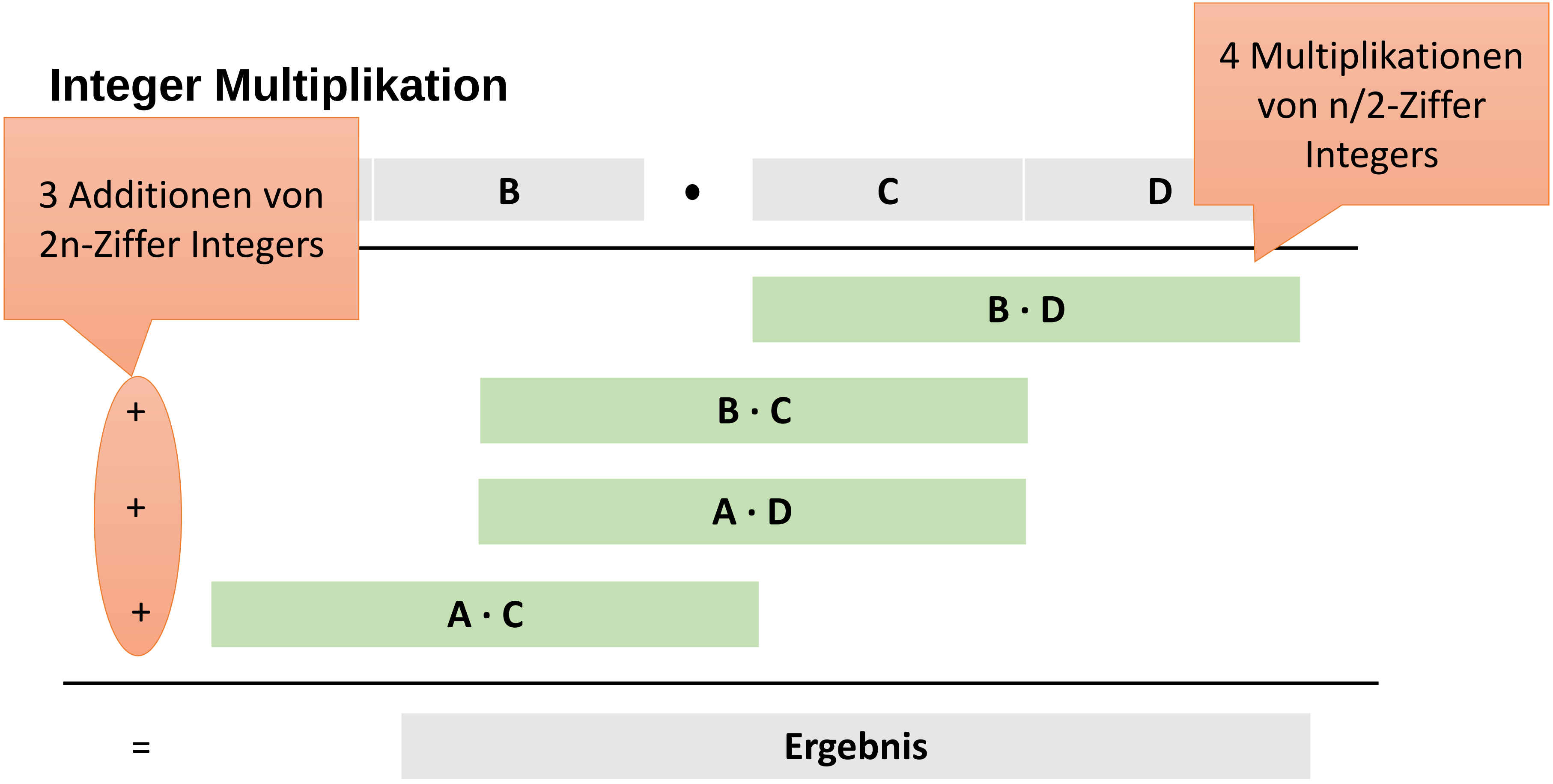
\+ A · C

= Ergebnis

Universität zu Köln

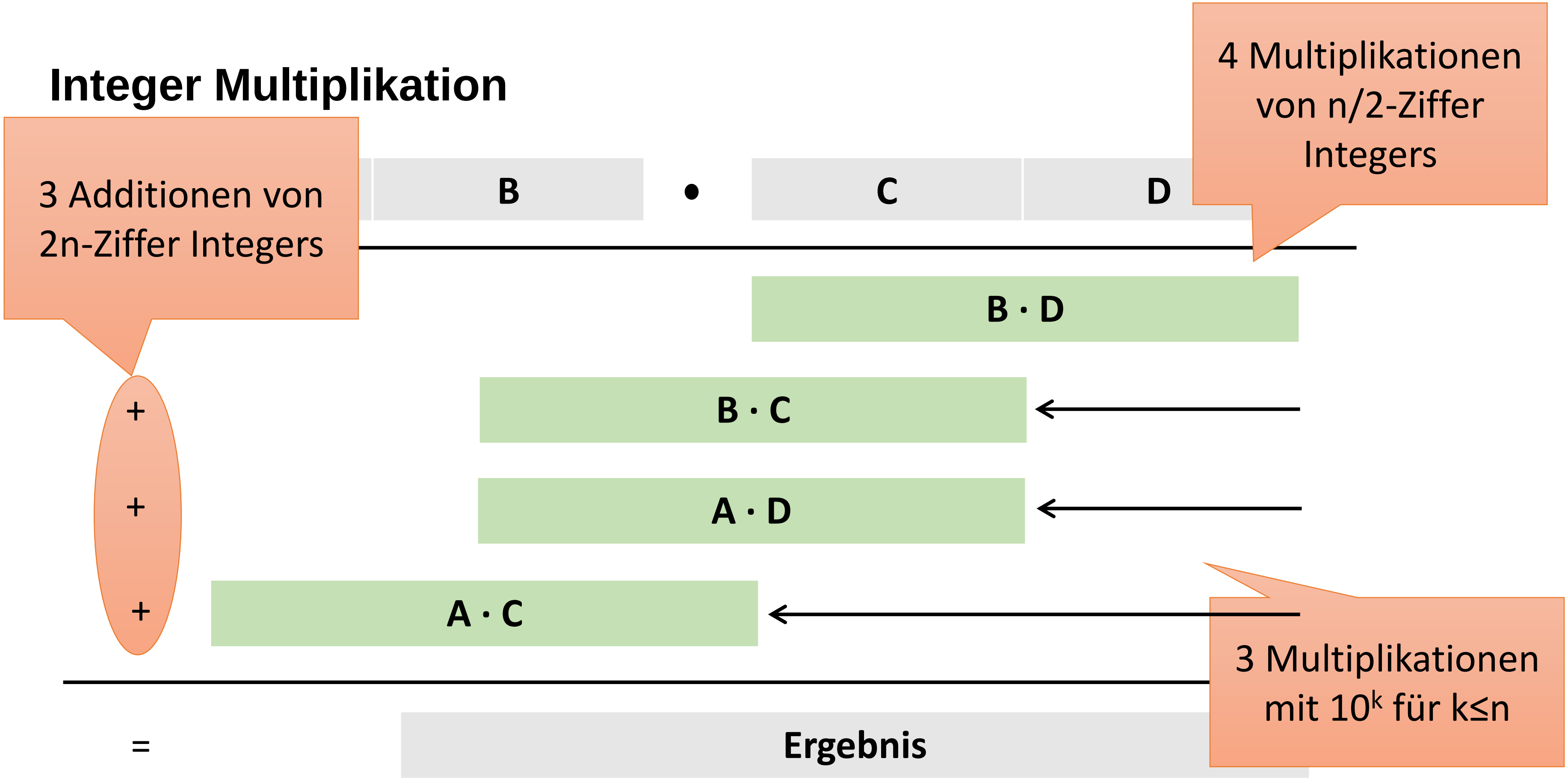
Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Integer Multiplikation



Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Integer Multiplikation



Laufzeitanalyse
- Einfache Integer Multiplikation

Einfache Integer Multiplikation

$$T(n) = 4\ T(n/2) + c\ n$$

Anzahl Unterprobleme

Aufwand für Aufteilen und
Zusammenfügen

Größe der Unterprobleme
(bestimmt Höhe des Rekursionsbaums)

(und $T(1) = O(1)$)

Universität zu Köln

Laufzeitanalyse
- Einfache Integer Multiplikation

Laufzeit einfache Integer Multiplikation

$$T(n) \leq \begin{cases} 4\,T(n/2) + cn & , n>1 \\ c & , n=1 \end{cases}$$

c geeignete Konstante

Welche bestmögliche Schranke für die Worst-Case Laufzeit ergibt sich?

A) O(n)
B) O(n log n)
C) $O(n^2)$
D) $O(n^2 \log n)$

Universität zu Köln

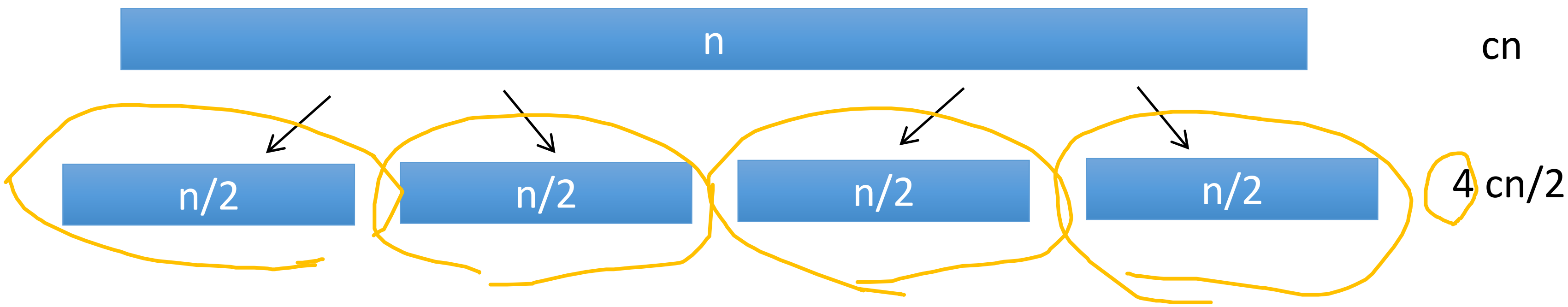
Laufzeitanalyse – grafische Darstellung
- Einfache Integer Multiplikation

- Auflösen von $T(n) \leq 4\ T(n/2) + cn$ (Intuition)
- $T(1) = c$

Universität zu Köln

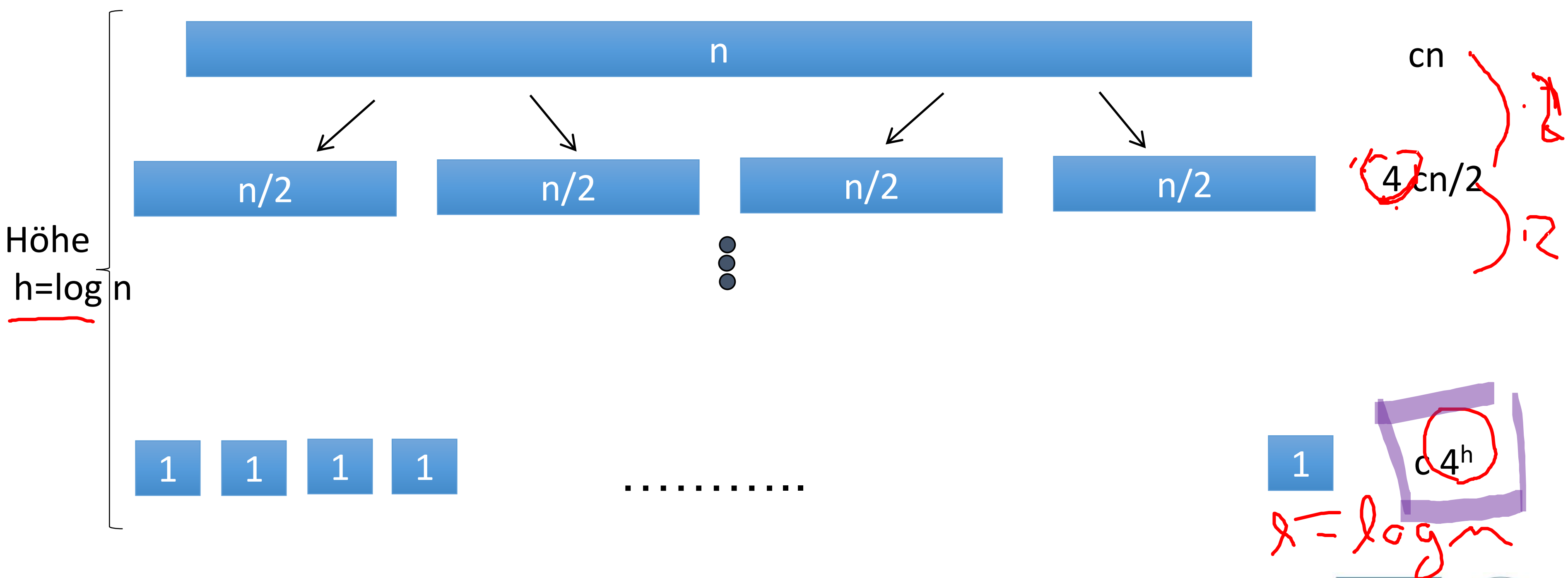
Laufzeitanalyse – grafische Darstellung
- Einfache Integer Multiplikation

- Auflösen von $T(n) \leq 4\ T(n/2) + cn$ (Intuition)
- $T(1) = c$



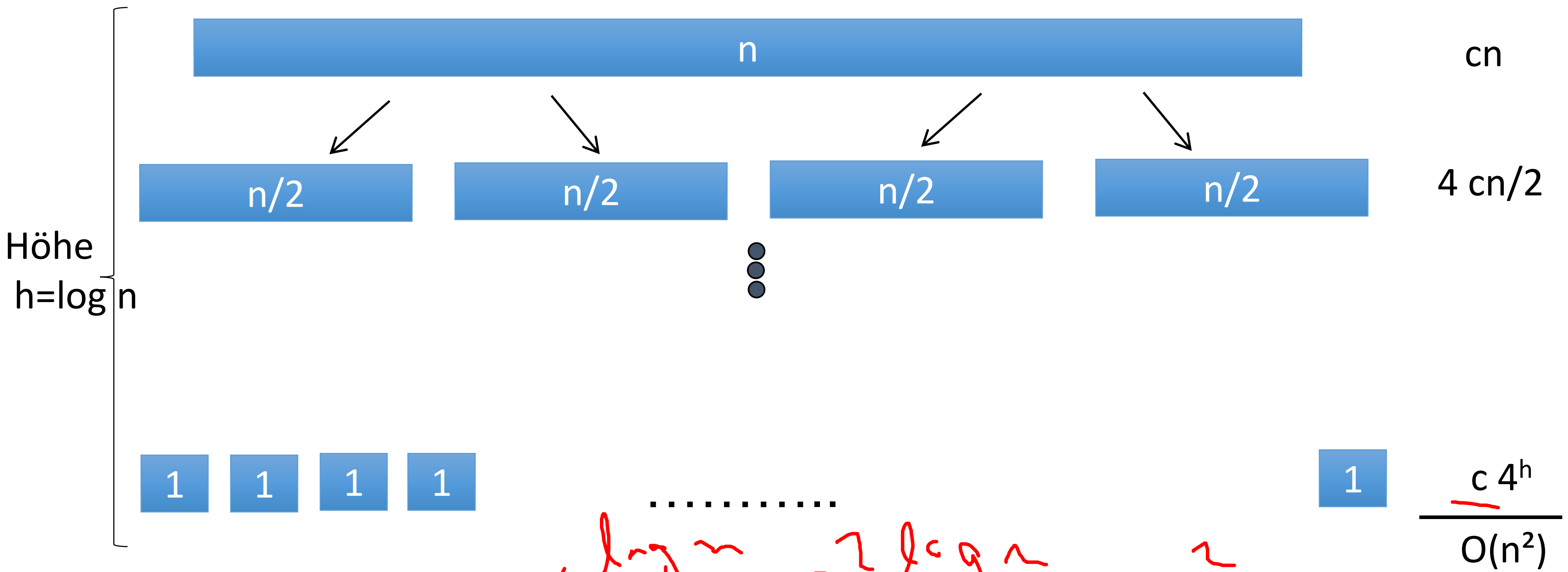
Laufzeitanalyse – grafische Darstellung - Einfache Integer Multiplikation

- Auflösen von $T(n) \leq 4\,T(n/2) + cn$ (Intuition)
- $T(1) = c$



Laufzeitanalyse – grafische Darstellung - Einfache Integer Multiplikation

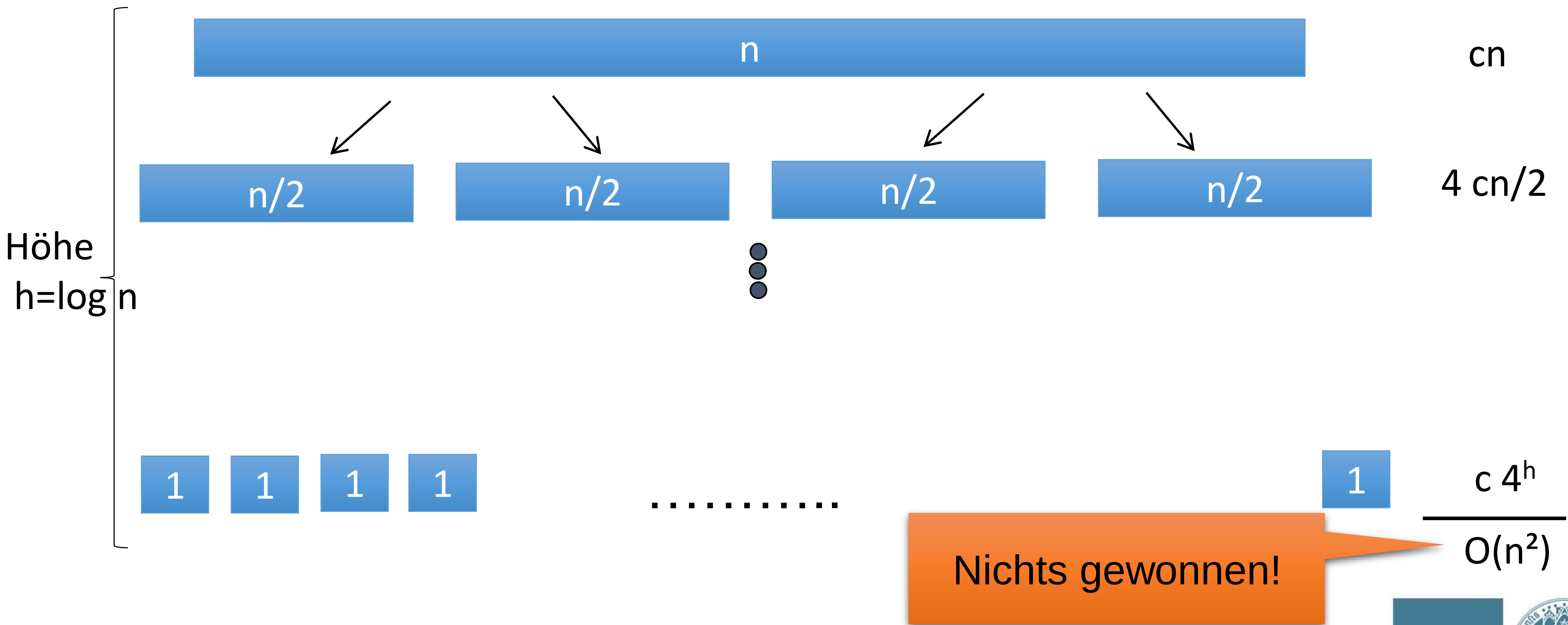
- Auflösen von $T(n) \leq 4\,T(n/2) + cn$ (Intuition)
- $T(1) = c$



$4^{\log n} = 2^{2\log n} = n^2$

Laufzeitanalyse – grafische Darstellung - Einfache Integer Multiplikation

- Auflösen von $T(n) \leq 4\,T(n/2) + cn$ (Intuition)
- $T(1) = c$



Laufzeitanalyse
- Einfache Integer Multiplikation

Satz 7.1

Die Multiplikation zweier n-Bit Zahlen mit dem einfachen Teile & Herrsche Verfahren hat Laufzeit $O(n^2)$.

Beweis

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. Wir zeigen $T(n) \leq cn^2$.
- Induktionsanfang: Die Laufzeit zur Multiplikation von zwei 1-Ziffer Zahlen ist höchstens c.
- Induktionsannahme: Für jedes $m<n$, m Z … die Laufzeit zur Multiplikation von zwei m-Ziffer Zahlen c …

Funktioniert nicht !!!!!

- Induktionsschluss: Betrachte eine Multiplikation von zwei n-Ziffer Zahlen (n Zweierpotenz). Es gilt $T(n) \leq 4\,T(n/2) + cn$. Nach Ind. Ann. gilt dann $T(n) \leq 4\,c\,(n/2)^2 + cn = cn^2+cn$.

Laufzeitanalyse
- Einfache Integer Multiplikation

Satz 7.1

Die Multiplikation zweier n-Bit Zahlen mit dem einfachen Teile & Herrsche Verfahren hat Laufzeit $O(n^2)$.

Trick: Die Funktion etwas verkleinern

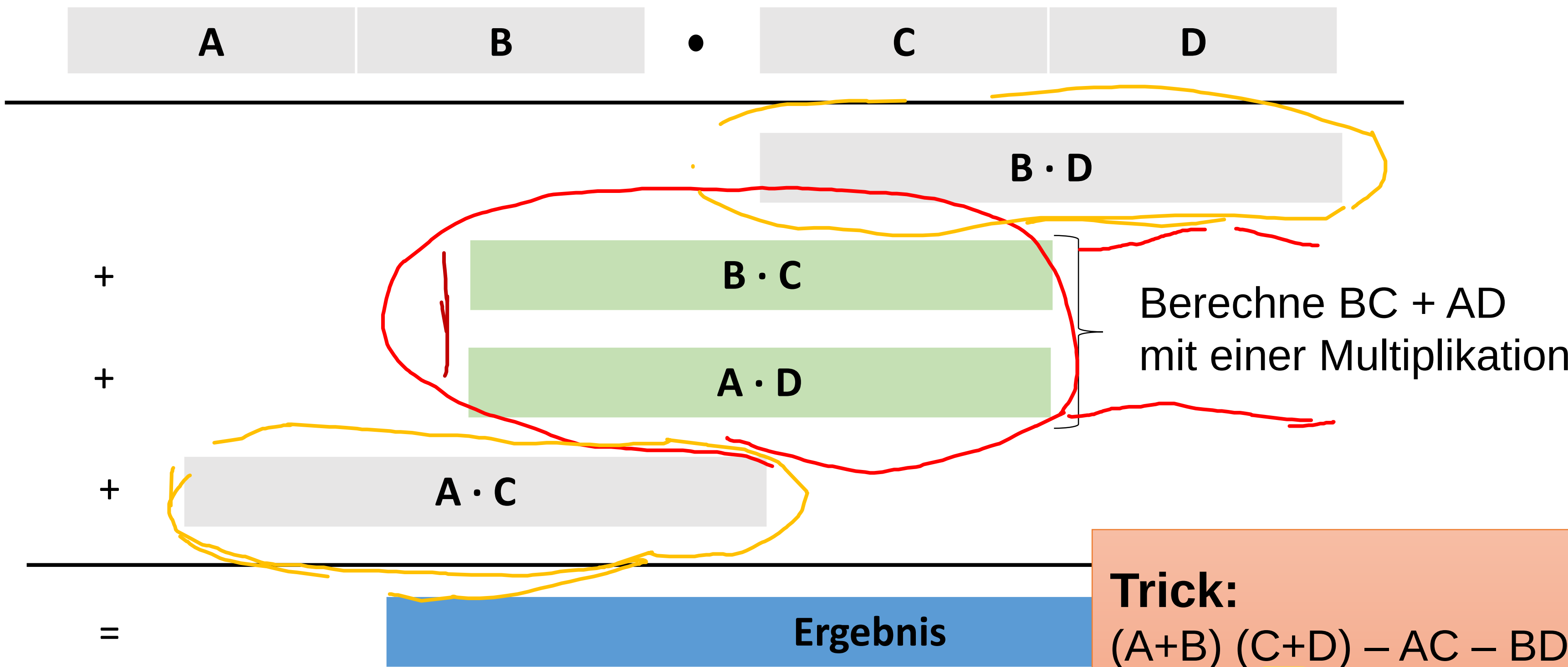
Beweis (neuer Versuch)

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. Wir zeigen $T(n) \leq cn^2 - \boxed{cn}$.
- Induktionsanfang: Die Laufzeit zur Multiplikation von zwei 2-Ziffer Zahlen ist höchstens $T(2) \leq c \leq 2c = 4c-2c$.
- Induktionsannahme: Für jedes $m<n$, m Zweierpotenz, ist die Laufzeit zur Multiplikation von zwei m-Ziffer Zahlen höchstens $cm^2 - cm$.
- Induktionsschluss: Betrachte eine Multiplikation von zwei n-Ziffer Zahlen (n Zweierpotenz). Es gilt $T(n) \leq 4\,T(n/2) + cn$. Nach Induktionsannahme gilt dann $T(n) \leq 4\,c\,(n/2)^2 - 4\,c(n/2) + cn = cn^2 - cn$.

Universität zu Köln

Teile & Herrsche Prinzip
- Multiplikation großer Zahlen

Verbesserte Integer Multiplikation



Trick:
(A+B) (C+D) – AC – BD
= BC + AD

Laufzeitanalyse
- Verbesserte Integer Multiplikation

Verbesserte Integer Multiplikation

$$T(n) = 3\, T(n/2) + c\, n$$

Anzahl Unterprobleme

Aufwand für Aufteilen und Zusammenfügen

Größe der Unterprobleme (bestimmt Höhe des Rekursionsbaums)

(und $T(1) = O(1)$)

Laufzeitanalyse
- Verbesserte Integer Multip

Die Multiplikation (A+B)(C+D) ist eigentlich eine Multiplikation zweier Zahlen mit n/2+1 Ziffern. Diese kann aber in O(n) Zeit auf eine Multiplikation von zwei Zahlen mit n/2 Ziffern zurückgeführt werden.

Aufwand Verbesserte Integer Multiplika

- 3 Multiplikationen der Länge n/2
- [AC, BD, (A+B) (C+D)]
- Konstant viele Additionen und Multiplikationen mit Zweierpotenzen

Laufzeit

$$T(n) \leq \begin{cases} 3\,T(n/2) + cn & , n>1 \\ c & , n=1 \end{cases}$$

c geeignete Konstante

Laufzeitanalyse
- Verbesserte Integer Multiplikation

Auflösen der Rekursionsgleichung (Annahme: n ist Zweierpotenz)

- $T(n) \le 3\,T(n/2) + cn$
- $3T(n/2) \le 3\,(3\,T(n/4) + cn/2) = 9\,T(n/4) + 3cn/2$
- $9T(n/4) \le 9\,(3\,T(n/8) + cn/4) = 27\,T(n/8) + 9cn/4$
- …
- $3^{\log n}\,T(n/n) = 3^{\log n}\,c$

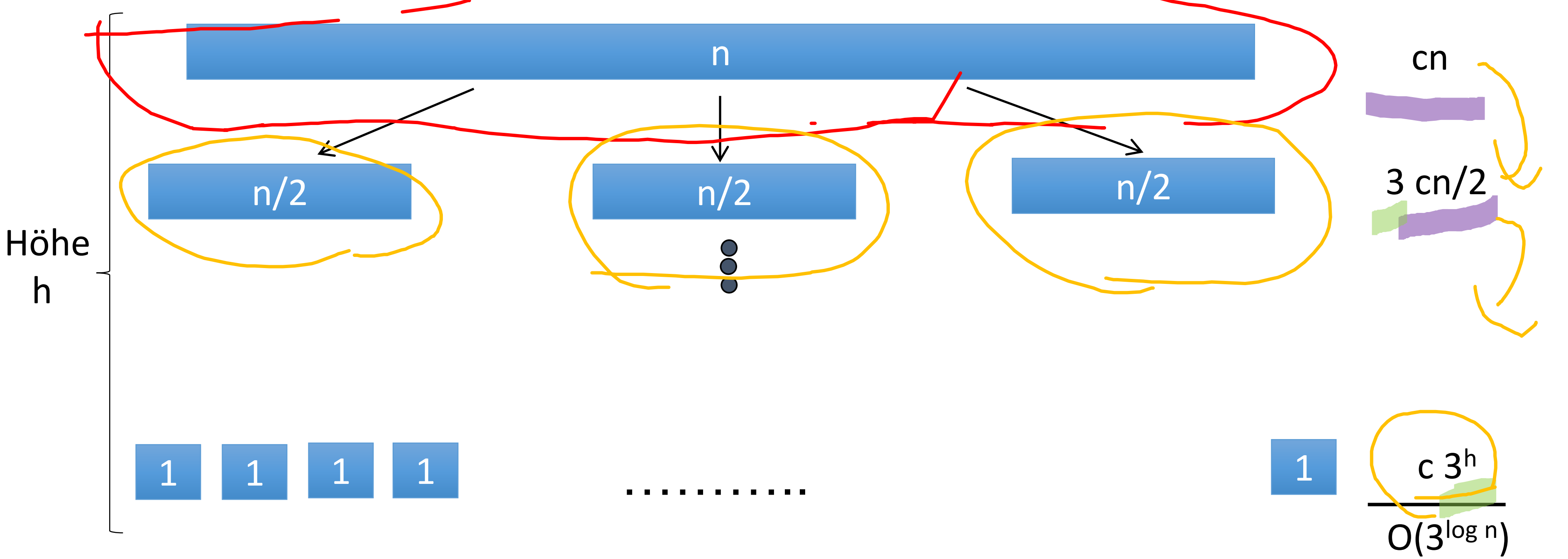
$\frac{3}{2}$

Zusammenfügen

- Mit jeder Rekursionsstufe erhöht sich der Aufwand um einen konstanten Faktor
- Daher entsteht der Hauptaufwand beim Rekursionsabbruch
- Da n in jeder Rekursionsstufe halbiert wird, gibt es genau log n Rekursionsstufen
- Damit ist die Anzahl Aufrufe mit n=1 gerade $3^{\log n}$

Laufzeitanalyse – grafische Darstellung - Verbesserte Integer Multiplikation

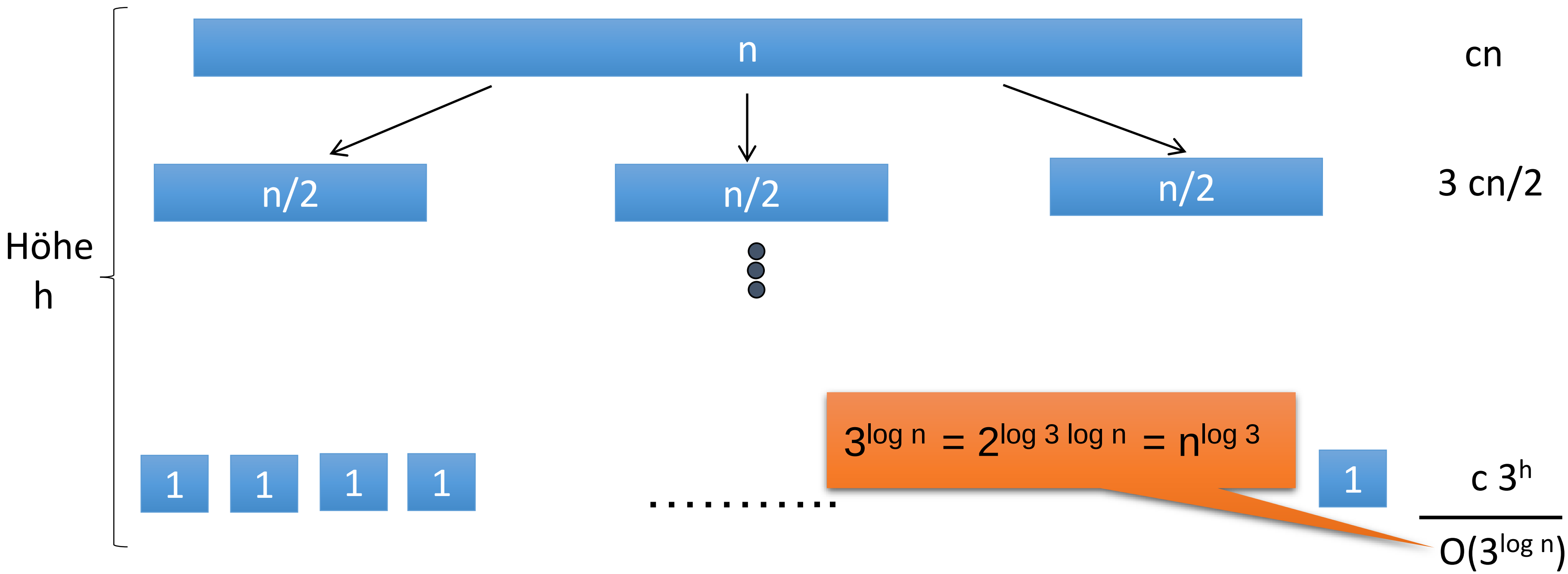
- Auflösen von $T(n) \leq 3\,T(n/2) + cn$ (Intuition)
- $T(1) = c$



$h = \log n$

Laufzeitanalyse – grafische Darstellung - Verbesserte Integer Multiplikation

- Auflösen von $T(n) \leq 3\,T(n/2) + cn$ (Intuition)
- $T(1) = c$



Laufzeitanalyse
- Verbesserte Integer Multiplikation

$\log(n/2) = (\log n) - 1$

Satz 7.2

- Die Laufzeit der verbesserten Integer Multiplikation ist $O(3^{\log n}) = O(n^{\log 3})$.

Beweis

- Wir zeigen den Satz nur, wenn n eine Zweierpotenz ist.
- Induktion über n. Sei $T(4) \le c$. Wir zeigen $T(n) \le c\, 3^{\log n} - 2cn$.
- Induktionsanfang: Es gilt $T(4) \le c = c \cdot 3^{\log 4} - 2 \cdot c \cdot 4$.

$c \cdot 3^2 - 8c$

- Induktionsannahme: Für jedes m<n, m Zweierpotenz, ist die Laufzeit für die Multiplikation zweier m-Ziffer Zahlen höchstens $c\, 3^{\log m} - 2cm$.

$= c$

- Induktionsschluss: Betrachte die Multiplikation von zwei n-Ziffer Zahlen (n Zweierpotenz). Es gilt
  $T(n) \le 3\, T(n/2) + cn \le c \cdot 3^{\log n} - 6c\,(n/2) + cn = c \cdot 3^{\log n} - 2cn.$

Universität zu Köln

Korrektheit
-Verbesserte Integer Multiplikation

Satz 7.3 (Algorithmus von Karazuba)

- Zwei n-Ziffer Integer Zahlen können mit Hilfe des Teile & Herrsche Verfahrens in $O(n^{1.59})$ worst case Laufzeit multipliziert werden.

Beweis

- Die Laufzeit folgt aus Satz 7.2 wegen $1.59 \geq \log 3$. Korrektheit zeigen wir per Induktion über n.
- Induktionsanfang: Die Multiplikation zweier 1-Ziffer Zahlen wird vom Rechner korrekt ausgeführt.

Korrektheit
-Verbesserte Integer Multiplikation

Satz 7.3 (Algorithmus von Karazuba)

- Zwei n-Ziffer Integer Zahlen können mit Hilfe des Teile & Herrsche Verfahrens in $O(n^{1.59})$ worst case Laufzeit multipliziert werden.

Beweis

- Induktionsannahme: Die Multiplikation zweier m-Ziffer Zahlen für m<n ist korrekt.
- Induktionsschluss: Nach Ind. Ann. werden die Produkte AC, BD, (A+B)(C+D) korrekt berechnet.
- Damit folgt die Korrektheit des Algorithmus wegen (A+B) (C+D) – AC – BD = BC + AD und aufgrund unserer Vorüberlegungen.

Universität zu Köln

Teile & Herrsche
- Integer Multiplikation

Zusammenfassung

- Problem: Multiplikation großer Integers
- „Schulmethode" liefert $O(n^2)$ Algorithmus
- Einfache Teile&Herrsche Methode gibt ebenfalls $O(n^2)$
- Verbesserter Teile & Herrsche Algorithmus benötigt 3 statt 4 rekursive Aufrufe
- Dadurch wird Laufzeit zu $O(n^{1.59})$

Teile & Herrsche Prinzip - Matrixmultiplikation

Matrixmultiplikation

$$\begin{pmatrix} 5 & 0 & 0 & 3 \\ 1 & 1 & 1 & 1 \\ 2 & 3 & 0 & 4 \\ 0 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 1 & 3 \\ 7 & 2 & 2 & 0 \\ 0 & 1 & 0 & 0 \\ 2 & 0 & 4 & 0 \end{pmatrix} = \begin{pmatrix} 11 & & & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

Zeile mal Spalte

- $5 \cdot 1 + 0 \cdot 7 + 0 \cdot 0 + 3 \cdot 2 = 11$

Teile & Herrsche Prinzip - Matrixmultiplikation

Matrixmultiplikation

$$\begin{pmatrix} 5 & 0 & 0 & 3 \\ 1 & 1 & 1 & 1 \\ 2 & 3 & 0 & 4 \\ 0 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 1 & 3 \\ 7 & 2 & 2 & 0 \\ 0 & 1 & 0 & 0 \\ 2 & 0 & 4 & 0 \end{pmatrix} = \begin{pmatrix} 11 & 10 & & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

Zeile mal Spalte

- $5 \cdot 1 + 0 \cdot 7 + 0 \cdot 0 + 3 \cdot 2 = 11$

Universität zu Köln

Teile & Herrsche Prinzip
- Matrixmultiplikation

Matrixmultiplikation

$$\begin{pmatrix} 5 & 0 & 0 & 3 \\ 1 & 1 & 1 & 1 \\ 2 & 3 & 0 & 4 \\ 0 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 1 & 3 \\ 7 & 2 & 2 & 0 \\ 0 & 1 & 0 & 0 \\ 2 & 0 & 4 & 0 \end{pmatrix} = \begin{pmatrix} 11 & 10 & 17 & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

Zeile mal Spalte

- $5 \cdot 1 + 0 \cdot 7 + 0 \cdot 0 + 3 \cdot 2 = 11$

Universität zu Köln

Teile & Herrsche Prinzip
- Matrixmultiplikation

Matrixmultiplikation

$$\begin{pmatrix} 5 & 0 & 0 & 3 \\ 1 & 1 & 1 & 1 \\ 2 & 3 & 0 & 4 \\ 0 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 1 & 3 \\ 7 & 2 & 2 & 0 \\ 0 & 1 & 0 & 0 \\ 2 & 0 & 4 & 0 \end{pmatrix} = \begin{pmatrix} 11 & 10 & 17 & 15 \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

Zeile mal Spalte

- $5 \cdot 1 + 0 \cdot 7 + 0 \cdot 0 + 3 \cdot 2 = 11$

Teile & Herrsche Prinzip
- Matrixmultiplikation

Matrixmultiplikation

$$\begin{pmatrix} 5 & 0 & 0 & 3 \\ 1 & 1 & 1 & 1 \\ 2 & 3 & 0 & 4 \\ 0 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 1 & 3 \\ 7 & 2 & 2 & 0 \\ 0 & 1 & 0 & 0 \\ 2 & 0 & 4 & 0 \end{pmatrix} = \begin{pmatrix} 11 & 10 & 17 & 15 \\ 10 & & & \\ & & & \\ & & & \end{pmatrix}$$

Zeile mal Spalte

- $5 \cdot 1 + 0 \cdot 7 + 0 \cdot 0 + 3 \cdot 2 = 11$

Teile & Herrsche Prinzip - Matrixmultiplikation

Matrixmultiplikation

$$\begin{pmatrix} 5 & 0 & 0 & 3 \\ 1 & 1 & 1 & 1 \\ 2 & 3 & 0 & 4 \\ 0 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 1 & 3 \\ 7 & 2 & 2 & 0 \\ 0 & 1 & 0 & 0 \\ 2 & 0 & 4 & 0 \end{pmatrix} = \begin{pmatrix} 11 & 10 & 17 & 15 \\ 10 & 5 & 7 & 3 \\ 31 & 10 & 24 & 6 \\ 2 & 1 & 4 & 0 \end{pmatrix}$$

Teile & Herrsche Prinzip
- Matrixmultiplikation

Matrixmultiplikation

- Problem: Berechne das Produkt zweier $n \times n$-Matrizen
- Eingabe: Matrizen X, Y
- Ausgabe: Matrix $Z = X \cdot Y$

Teile & Herrsche Prinzip
- Matrixmultiplikation

MatrixMultiplikation(X,Y,n)

1. Z = new array [1..n][1..n]
2. **for** i=1 **to** n **do**
3.   **for** j=1 **to** n **do**
4.   Z[i][j] = 0
5.   **for** k=1 **to** n **do**
6.   Z[i][j] = Z[i][j] + X[i][k] Y[k][j]
7. **return** Z

$O(n^2)$

$O(n)$

$O(n^2)$

$O(n^3)$

$O(n)$

$O(n^3)$

Universität zu Köln

Teile & Herrsche Prinzip
- Matrixmultiplikation

Matrixmultiplikation

- $$\begin{pmatrix} A & B \\ C & D \end{pmatrix} \cdot \begin{pmatrix} E & F \\ G & H \end{pmatrix} = \begin{pmatrix} \boxed{AE + BG} & AF + BH \\ CE + DG & CF + DH \end{pmatrix}$$

Laufzeit

- 8 Multiplikationen von n/2×n/2 Matrizen
- 4 Additionen von n/2 ×n/2 Matrizen

$O(n^2)$

- $$T(n) \le \begin{cases} 8T\left(\frac{n}{2}\right) + cn^2 & \text{, für } n > 1 \\ c & \text{, für } n = 1 \end{cases}$$

Universität zu Köln

Teile & Herrsche Prinzip - Matrixmultiplikation

Matrixmultiplikation

- $\begin{pmatrix} A & B \\ C & D \end{pmatrix} \cdot \begin{pmatrix} E & F \\ G & H \end{pmatrix} = \begin{pmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{pmatrix}$

Laufzeit

- 8 Multiplikationen von n/2×n/2 Matrizen
- 4 Additionen von n/2 ×n/2 Matrizen

- $T(n) \leq \begin{cases} 8T\left(\frac{n}{2}\right) + cn^2 & \text{, für } n > 1 \\ c & \text{, für } n = 1 \end{cases}$

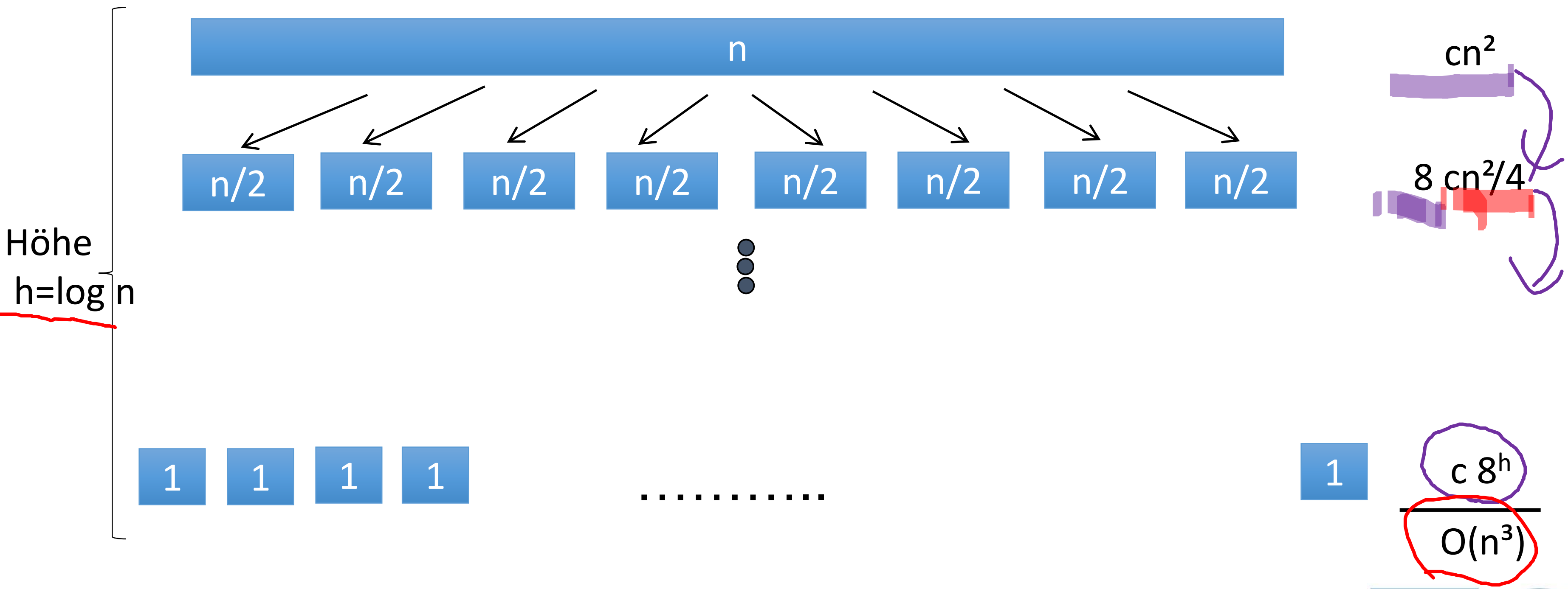
Welche bestmögliche Schranke für die Worst-Case Laufzeit ist korrekt?

A) $O(n^2)$
B) $O(n^2 \log n)$
C) $O(n^3)$
D) $O(n^3 \log n)$

Teile & Herrsche Prinzip - Matrixmultiplikation

$$c\left(\frac{n}{2}\right)^2 = c\,n^2/4$$

- Auflösen von $T(n) \leq 8\,T(n/2) + cn^2$ (Intuition)
- $T(1) = c$



Teile & Herrsche Prinzip - Matrixmultiplikation

Zusammenfassung

- Integer Multiplikation
- Einfacher Teile & Herrsche Algorithmus
- Verbesserter Algorithmen
- Matrix Multiplikation
- Einfacher Teile & Herrsche Algorithmus

Universität zu Köln