technische universität
dortmund

LS 2 /
Informatik

Datenstrukturen, Algorithmen und Programmierung 2 (DAP2)

Teile & Herrsche

Integer Multiplikation

- Problem: Multipliziere zwei n-Bit Integer
- Eingabe: Zwei n-Bit Integer X, Y
- Ausgabe: $2n$-Bit Integer Z mit $Z = XY$

Annahmen:

- Wir können n-Bit Integer in $\mathbf{\Theta}(n)$ (worst case) Zeit addieren
- Wir können n-Bit Integer in $\mathbf{\Theta}(n+k)$ (worst case) Zeit mit 2^k multiplizieren

Teile & Herrsche

Laufzeit Schulmethode

- n Multiplikationen mit 2^k für ein $k \leq n$
- $n-1$ Additionen im worst-case:

$$\underbrace{11\ldots111}_{n\text{-Bit}} \cdot \underbrace{11\ldots111}_{n\text{-Bit}}$$

- Jede Addition $\boldsymbol{\Theta}(n)$ Zeit
- Insgesamt $\boldsymbol{\Theta}(n^2)$ Laufzeit

Teile & Herrsche

Laufzeit Schulmethode

- n Multiplikationen mit 2^k für ein $k \leq n$
- $n-1$ Additionen im worst-case:

$$\underbrace{11\ldots111}_{n\text{-Bit}} \cdot \underbrace{11\ldots111}_{n\text{-Bit}}$$

- Jede Addition $\boldsymbol{\Theta}(n)$ Zeit
- Insgesamt $\boldsymbol{\Theta}(n^2)$ Laufzeit

Bessere Laufzeit mit Teile & Herrsche?

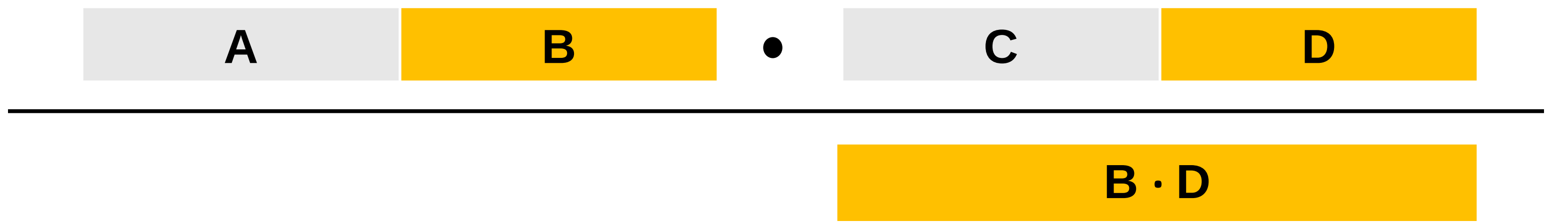
Teile & Herrsche

Integer Multiplikation

A	B	•	C	D

Teile & Herrsche

Integer Multiplikation



Teile & Herrsche

Integer Multiplikation

A B • C D

B · D

\+ B · C

Teile & Herrsche

Integer Multiplikation

A **B** • **C** **D**

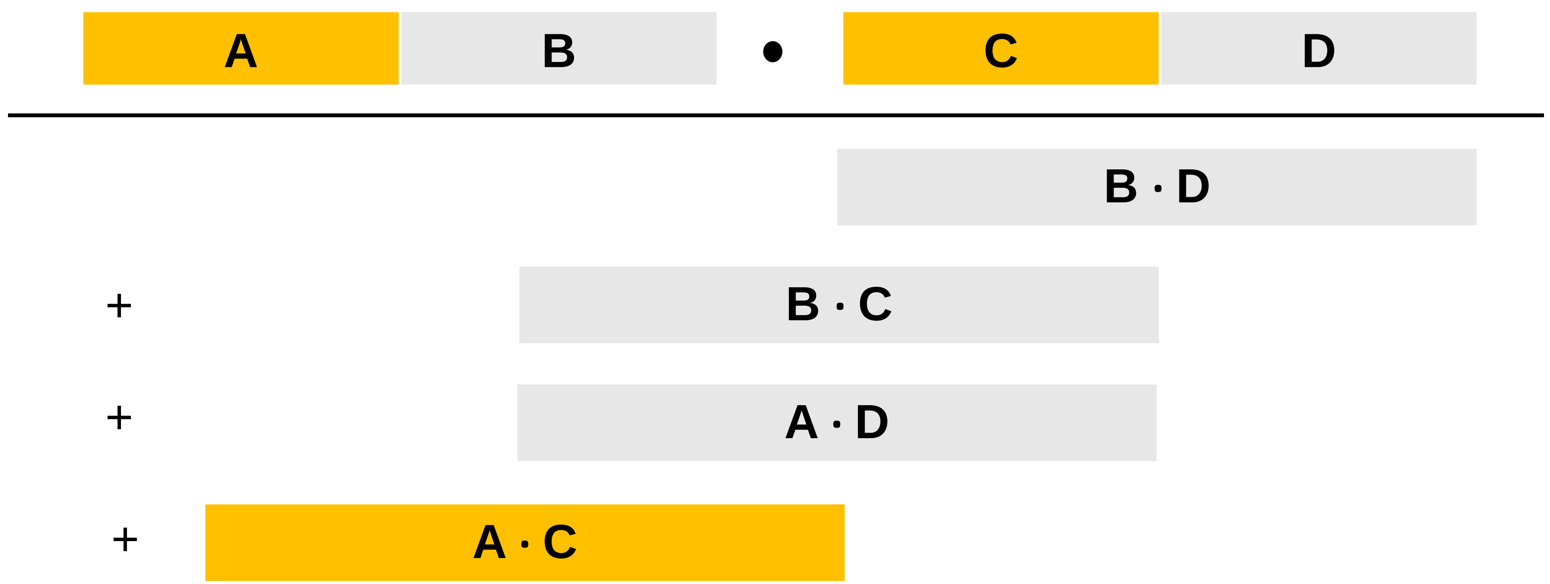
B · D

\+ **B · C**

\+ **A · D**

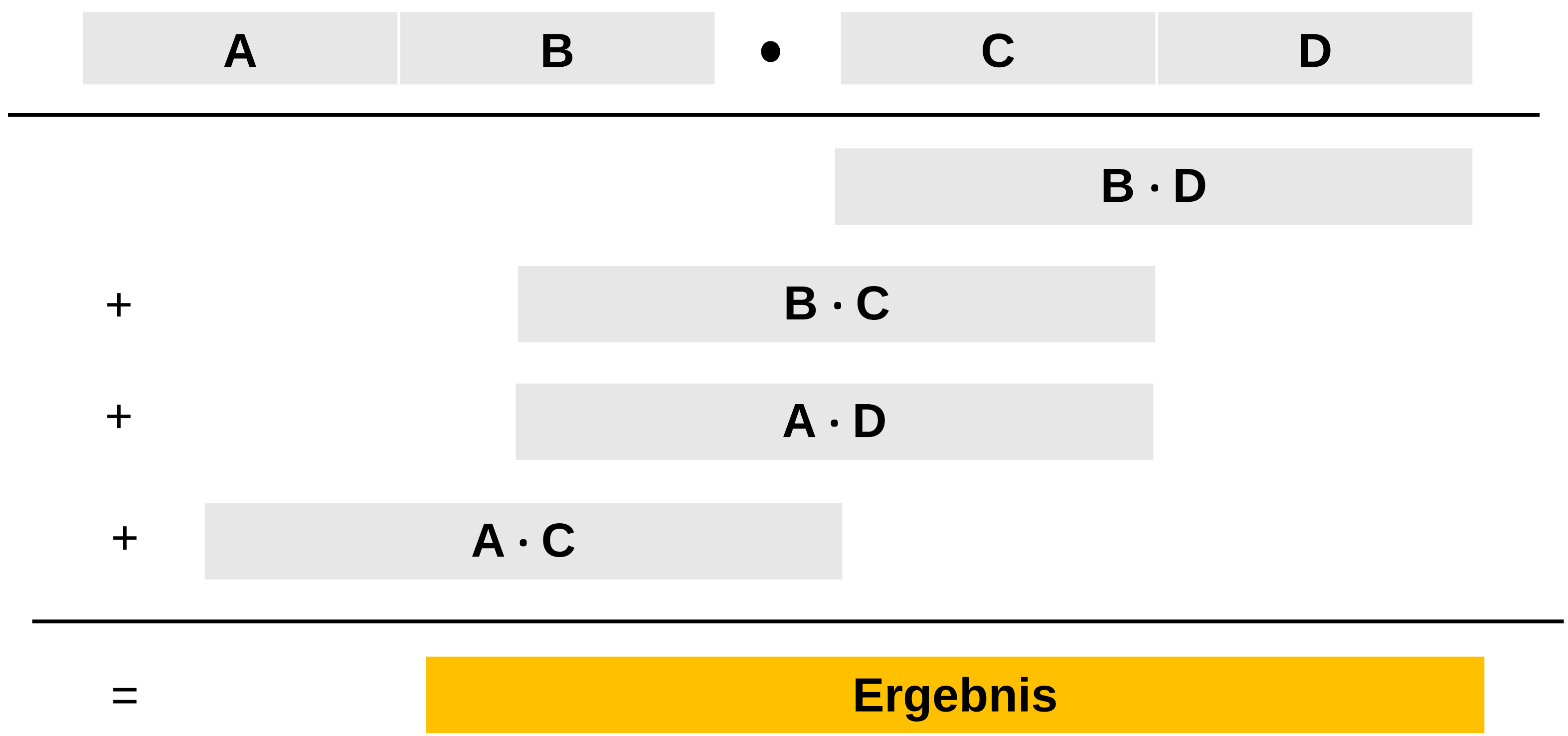
Teile & Herrsche

Integer Multiplikation



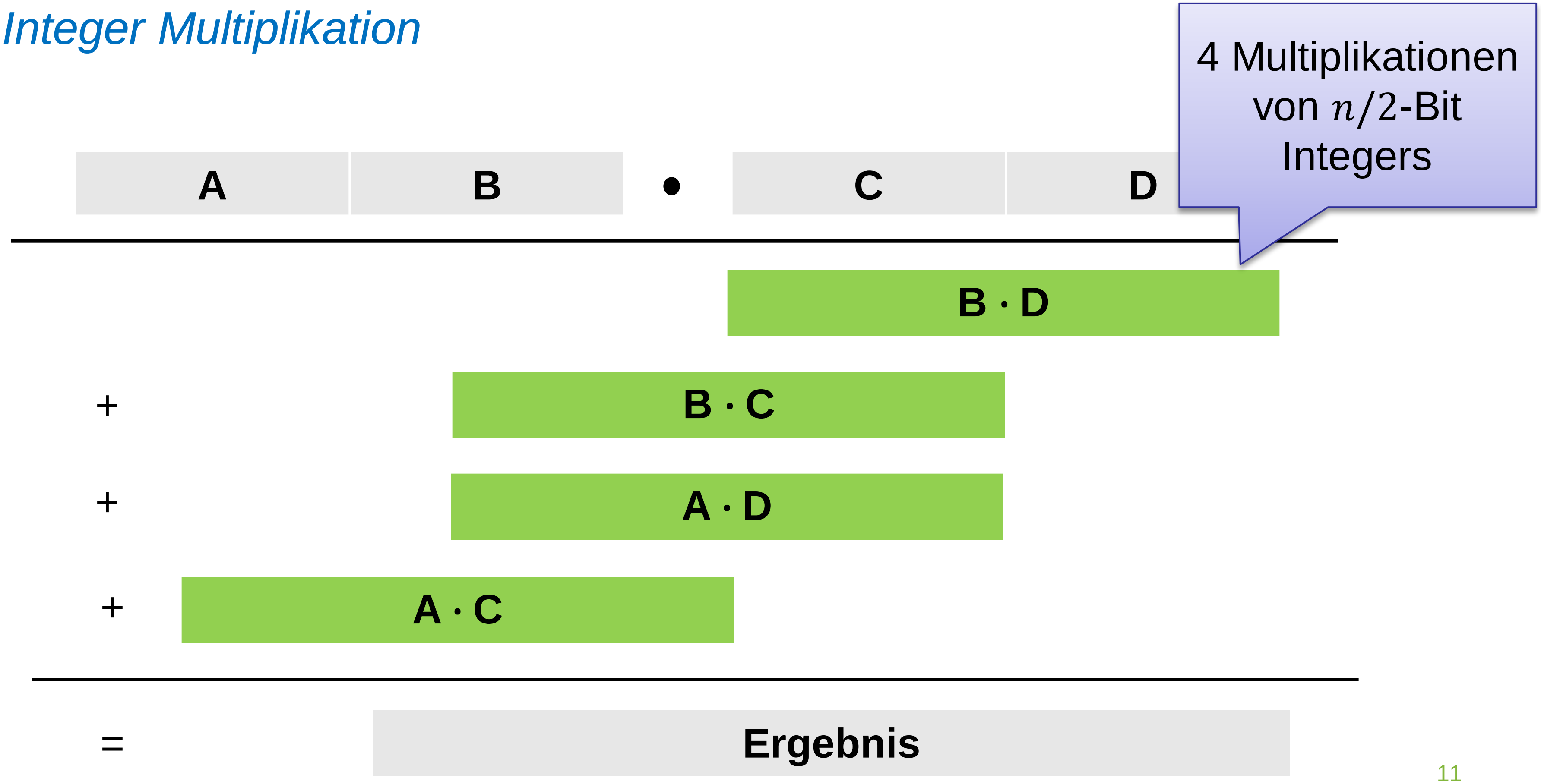
Teile & Herrsche

Integer Multiplikation



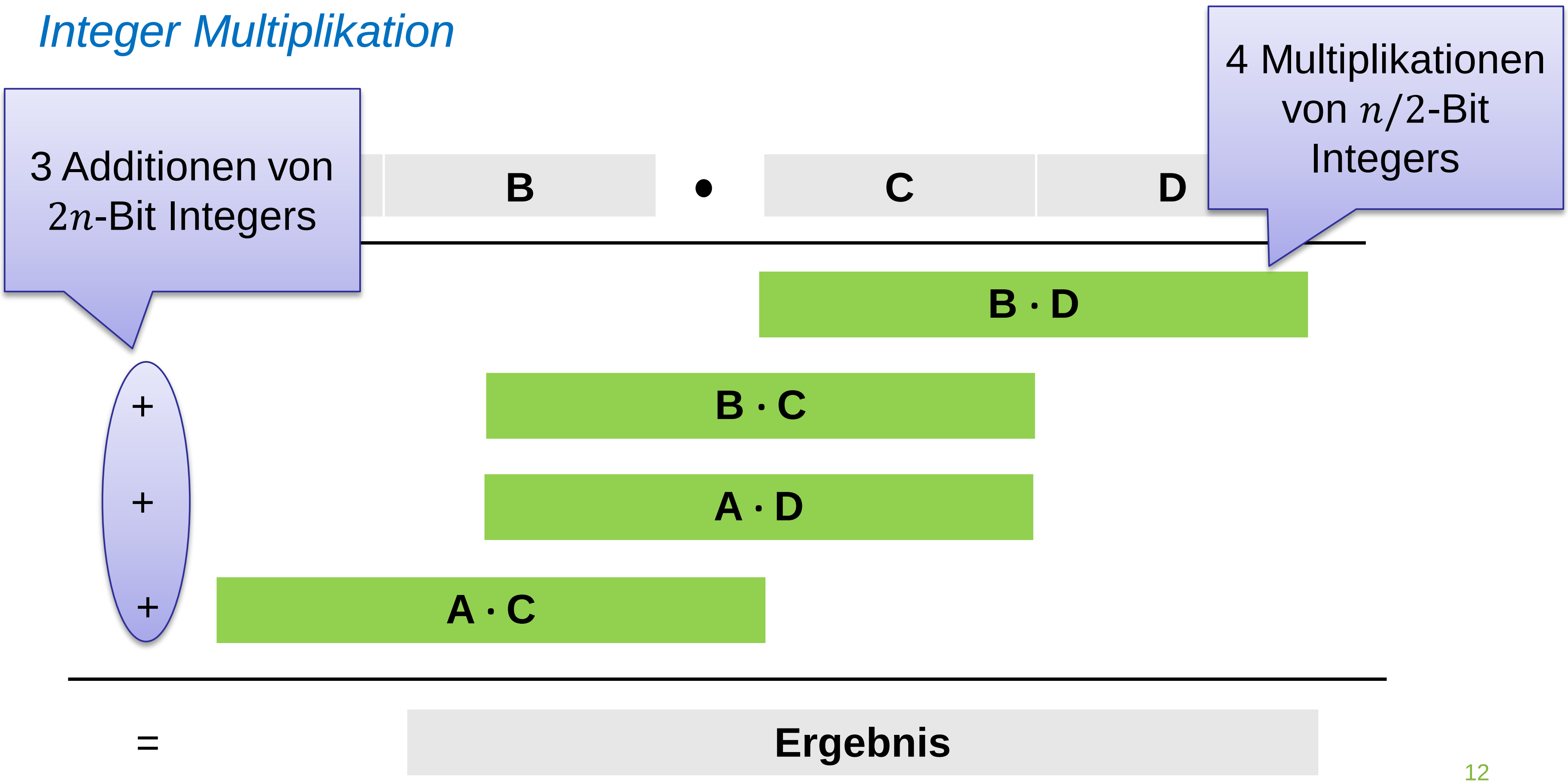
Teile & Herrsche

Integer Multiplikation



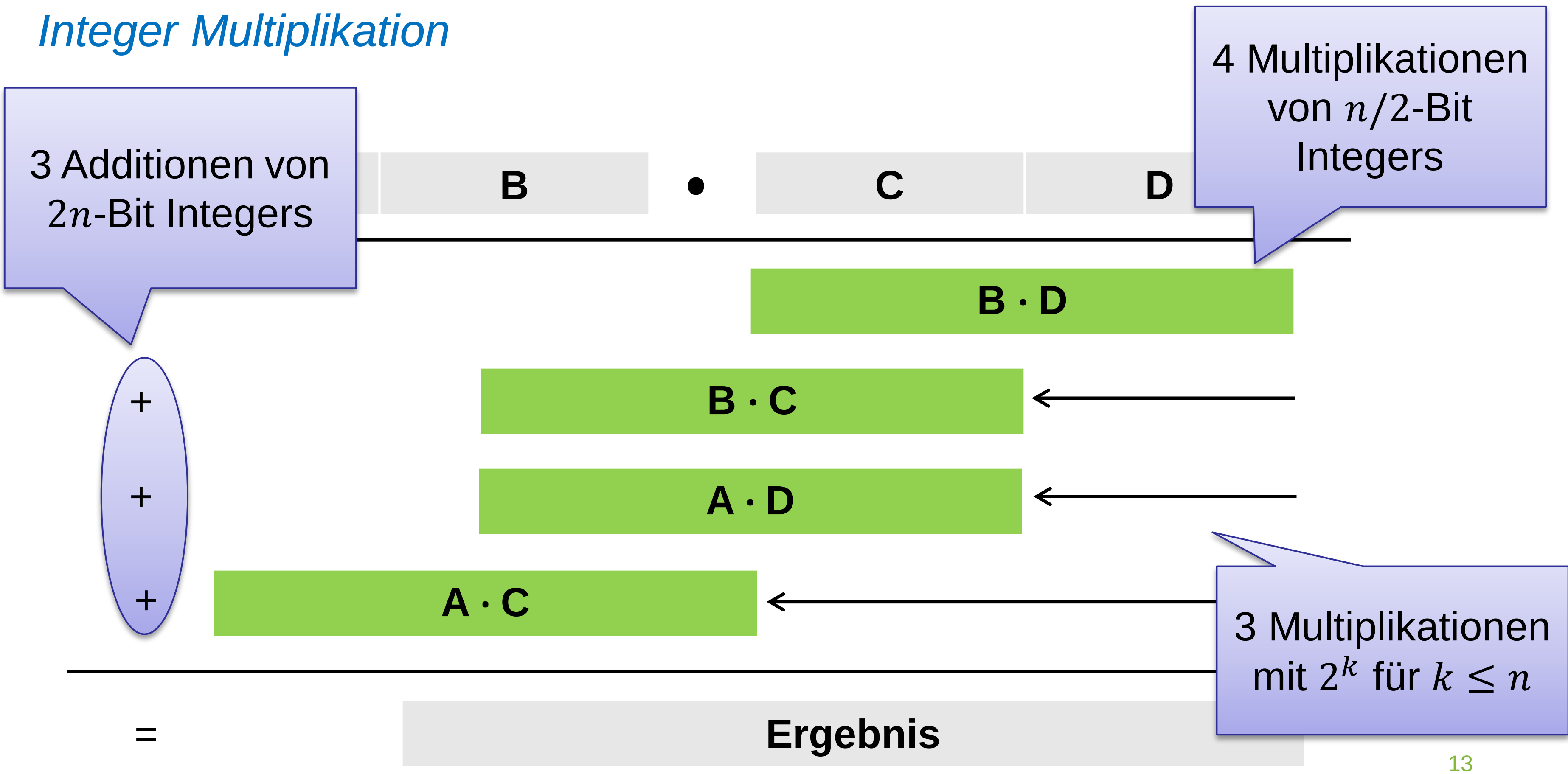
Teile & Herrsche

Integer Multiplikation



Teile & Herrsche

Integer Multiplikation



Teile & Herrsche

Beispiel Multiplikation Schulmethode

$$T(n) = 4 \cdot T(n/2) + cn$$

Anzahl Unterprobleme

Aufwand für Aufteilen und
Zusammenfügen

Größe der Unterprobleme
(bestimmt Höhe des Rekursionsbaums)

- (und $T(1) = \text{const}$)

(n Zweierpotenz)

Teile & Herrsche

Laufzeit einfaches Teile & Herrsche

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

- c geeignete Konstante

Welche bestmögliche Laufzeit ergibt sich?

A) $\mathbf{O}(n)$

B) $\mathbf{O}(n \log n)$

C) $\mathbf{O}(n^2)$

D) $\mathbf{O}(n^2 \log n)$

Teile & Herrsche

Laufzeit einfaches Teile & Herrsche

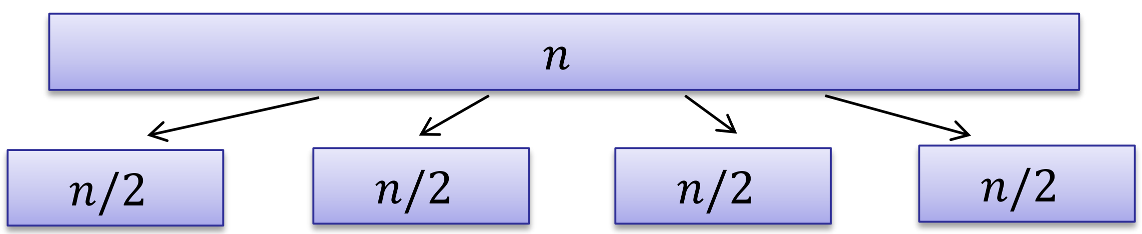
$$T(n) \leq \begin{cases} 4\,T(n/2) + cn & \text{, falls } n > 1 \\ c & \text{, falls } n = 1 \end{cases} \quad \text{c geeignete Konstante}$$

$\boxed{\quad n \quad}$ $\qquad cn$

Teile & Herrsche

Laufzeit einfaches Teile & Herrsche

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



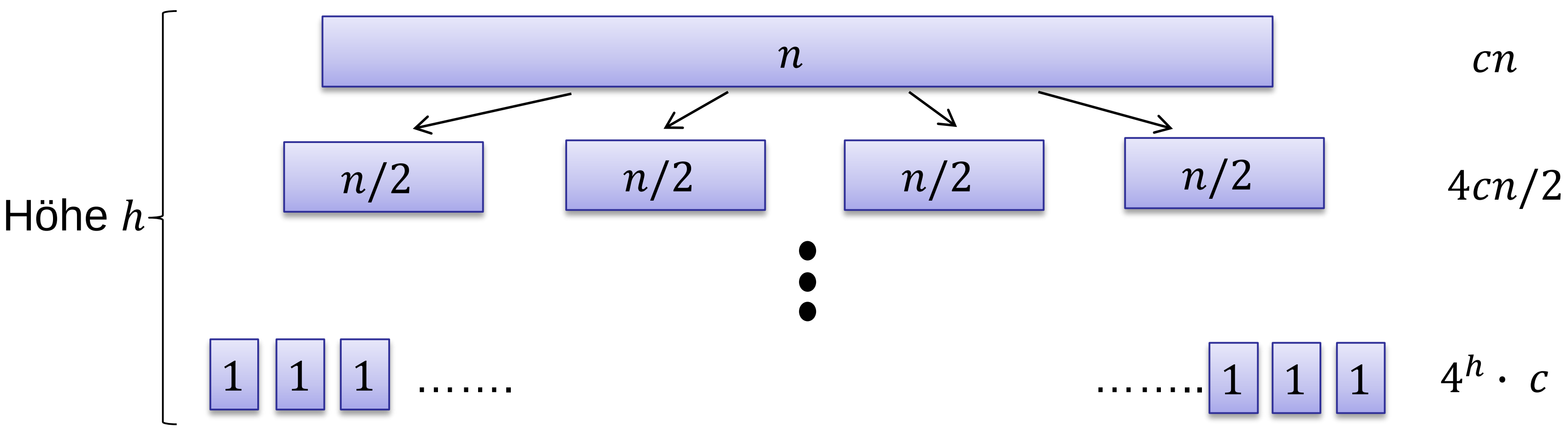
cn

$4cn/2$

Teile & Herrsche

Laufzeit einfaches Teile & Herrsche

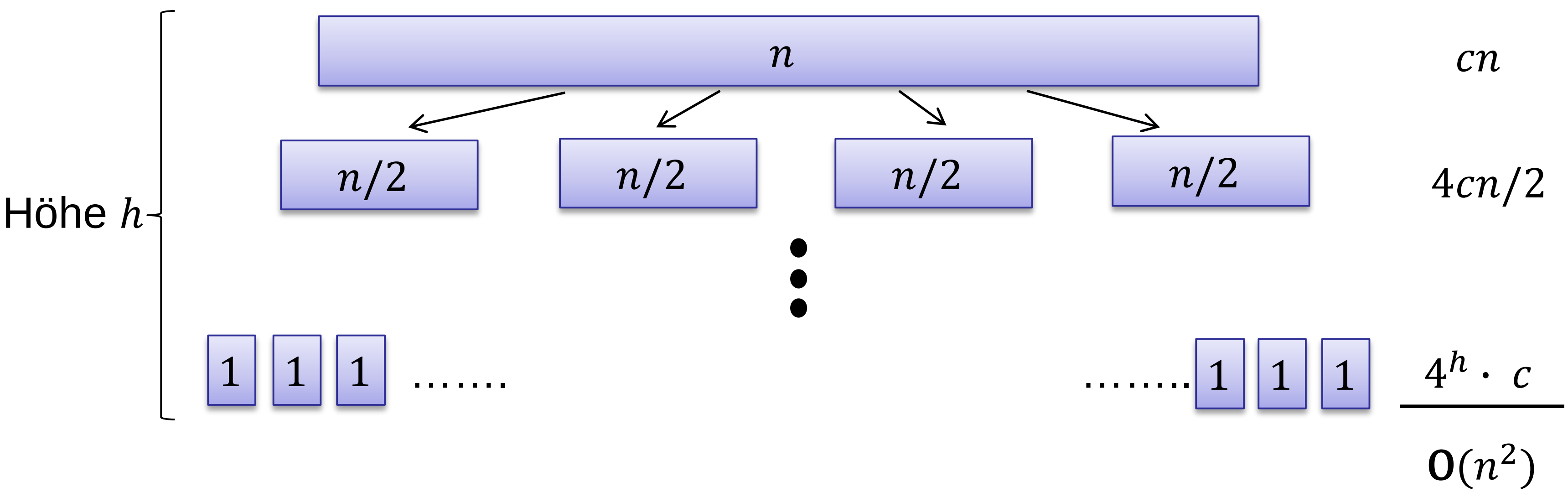
$$T(n) \leq \begin{cases} 4\,T(n/2) + cn & \text{, falls } n > 1 \\ c & \text{, falls } n = 1 \end{cases} \quad \text{c geeignete Konstante}$$



Teile & Herrsche

Laufzeit einfaches Teile & Herrsche

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



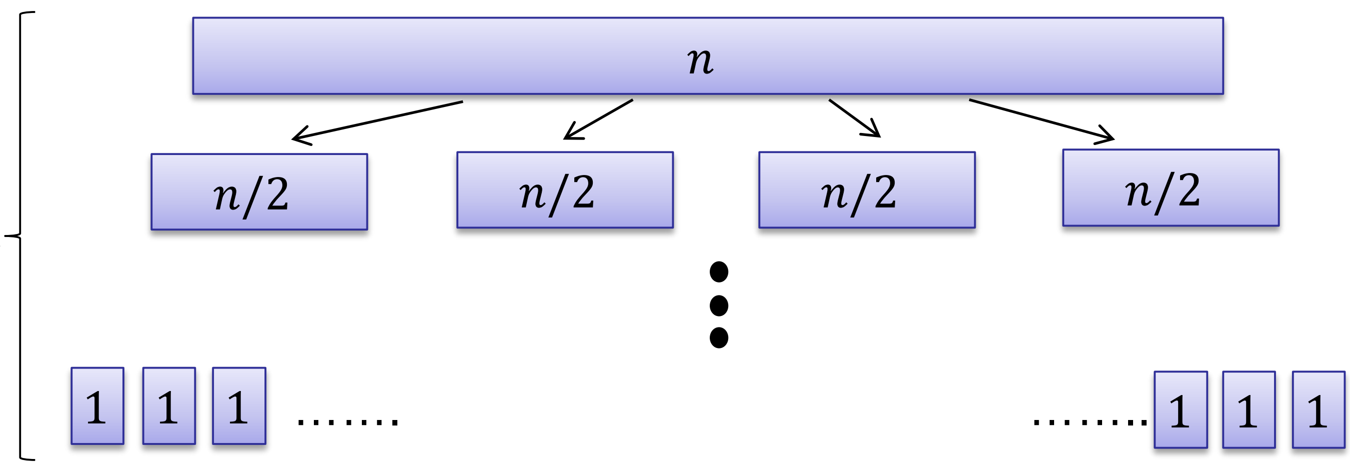
Höhe des Baums: $h = \log n$

Teile & Herrsche

Laufzeit einfaches Teile & Herrsche

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

Höhe h



cn

$4cn/2$

$4^h \cdot c$

$\mathbf{O}(n^2)$

Nichts gewonnen!

Höhe des Baums: $h = \log n$

Teile & Herrsche

Satz 8

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

Beweis

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. Wir zeigen $T(n) \leq cn^2$.

Teile & Herrsche

Satz 8

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

Beweis

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. Wir zeigen $T(n) \leq cn^2$.
- (I.A.) Die Laufzeit zur Multiplikation von zwei 1-Bit Zahlen ist höchstens c.

Teile & Herrsche

Satz 8

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

Beweis

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. Wir zeigen $T(n) \ \leq \ cn^2$.
- (I.A.) Die Laufzeit zur Multiplikation von zwei 1-Bit Zahlen ist höchstens c.
- (I.V.) Für jedes $m < n$, m Zweierpotenz, ist die Laufzeit zur Multiplikation von zwei m-Bit Zahlen $c \cdot m^2$.

Teile & Herrsche

Satz 8

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

Beweis

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. Wir zeigen $T(n) \leq cn^2$.
- (I.A.) Die Laufzeit zur Multiplikation von zwei 1-Bit Zahlen ist höchstens c.
- (I.V.) Für jedes $m < n$, m Zweierpotenz, ist die Laufzeit zur Multiplikation von zwei m-Bit Zahlen $c \cdot m^2$.
- (I.S.) Betrachte eine Multiplikation von zwei n-Bit Zahlen (n Zweierpotenz). Es gilt $T(n) \leq 4\,T(n/2) + cn$. Nach (I.V.) gilt dann $T(n) \leq 4c(n/2)^2 + cn = cn^2 + cn$.

Teile & Herrsche

Satz 8

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

Beweis

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. Wir zeigen $T(n) \;\leq\; cn^2$.
- (I.A.) Die Laufzeit zur Multiplikation von zwei 1-Bit Zahlen ist höchstens c.
- (I.V.) Für jedes $m < n$, m Zweierpotenz, ist die Laufzeit zur Multiplikation von zwei m-Bit Zahlen $c \cdot m^2$.
- (I.S.) Betrachte eine Multiplikation von zwei n-Bit Zahlen (n Zweierpotenz). Es gilt $T(n) \leq 4\,T(n/2) + cn$. Nach (I.V.) gilt dann $T(n) \leq 4c(n/2)^2 + cn = \boxed{cn^2 + cn}$.

Funktioniert nicht !!!!!

Teile & Herrsche

Satz 8

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

Beweis (neuer Versuch)

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. O.b.d.A. sei $c \geq T(2)$. Wir zeigen $\boxed{T(n) \leq cn^2 - cn.}$

Trick: Die Funktion etwas verkleinern!!

Teile & Herrsche

Satz 8

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

Beweis (neuer Versuch)

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. O.b.d.A. sei $c \geq T(2)$. Wir zeigen $T(n) \leq cn^2 - cn$.
- (I.A.) Die Laufzeit zur Multiplikation von zwei 2-Bit Zahlen ist höchstens $T(2) \leq c \leq 2c$.

Teile & Herrsche

Satz 8

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

Beweis (neuer Versuch)

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. O.b.d.A. sei $c \geq T(2)$. Wir zeigen $T(n) \leq cn^2 - cn$.
- (I.A.) Die Laufzeit zur Multiplikation von zwei 2-Bit Zahlen ist höchstens $T(2) \leq c \leq 2c$.
- (I.V.) Für jedes $m < n$, m Zweierpotenz, ist die Laufzeit zur Multiplikation von zwei m-Bit Zahlen $cm^2 - cm$.

Teile & Herrsche

Satz 8

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

Beweis (neuer Versuch)

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. O.b.d.A. sei $c \geq T(2)$. Wir zeigen $T(n) \leq cn^2 - cn$.
- (I.A.) Die Laufzeit zur Multiplikation von zwei 2-Bit Zahlen ist höchstens $T(2) \leq c \leq 2c$.
- (I.V.) Für jedes $m < n$, m Zweierpotenz, ist die Laufzeit zur Multiplikation von zwei m-Bit Zahlen $cm^2 - cm$.
- (I.S.) Betrachte eine Multiplikation von zwei n-Bit Zahlen (n Zweierpotenz). Es gilt $T(n) \leq 4T(n/2) + cn$.

Teile & Herrsche

Satz 8

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

Beweis (neuer Versuch)

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. O.b.d.A. sei $c \geq T(2)$. Wir zeigen $T(n) \leq cn^2 - cn$.
- (I.A.) Die Laufzeit zur Multiplikation von zwei 2-Bit Zahlen ist höchstens $T(2) \leq c \leq 2c$.
- (I.V.) Für jedes $m < n$, m Zweierpotenz, ist die Laufzeit zur Multiplikation von zwei m-Bit Zahlen $cm^2 - cm$.
- (I.S.) Betrachte eine Multiplikation von zwei n-Bit Zahlen (n Zweierpotenz). Es gilt $T(n) \leq 4T(n/2) + cn$.
  Nach (I.V.) gilt dann $T(n) \leq 4c(n/2)^2 - 4c(n/2) + cn = cn^2 - cn$.

Teile & Herrsche

Satz 8

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

Beweis (neuer Versuch)

- Wir nehmen an, dass n eine Zweierpotenz ist. Induktion über n. O.b.d.A. sei $c \geq T(2)$. Wir zeigen $T(n) \leq cn^2 - cn$.
- (I.A.) Die Laufzeit zur Multiplikation von zwei 2-Bit Zahlen ist höchstens $T(2) \leq c \leq 2c$.
- (I.V.) Für jedes $m < n$, m Zweierpotenz, ist die Laufzeit zur Multiplikation von zwei m-Bit Zahlen $cm^2 - cm$.
- (I.S.) Betrachte eine Multiplikation von zwei n-Bit Zahlen (n Zweierpotenz). Es gilt $T(n) \leq 4T(n/2) + cn$.
  Nach (I.V.) gilt dann $T(n) \leq 4c(n/2)^2 - 4c(n/2) + cn = cn^2 - cn$.

Teile & Herrsche

Verbesserte Integer Multiplikation

A B • C D

B · D

\+ B · C

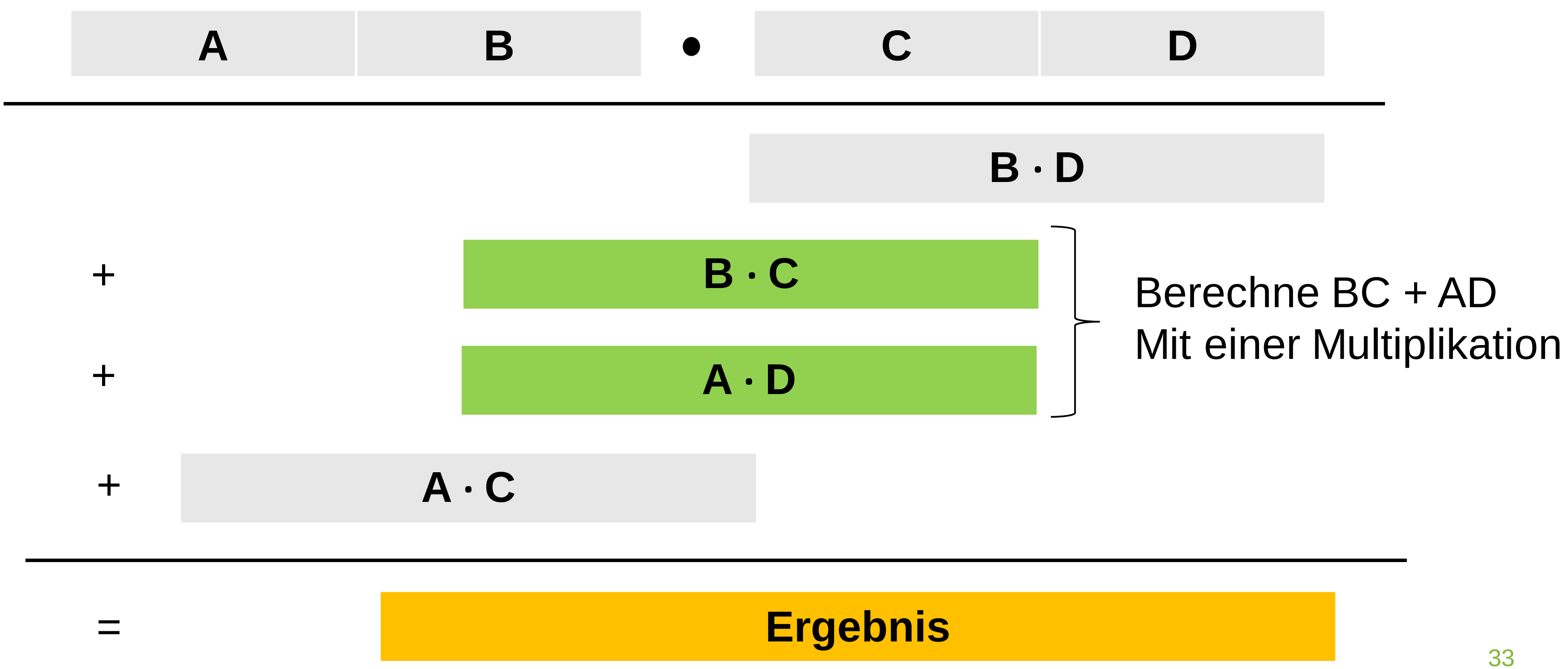
\+ A · D

\+ A · C

= **Ergebnis**

Teile & Herrsche

Verbesserte Integer Multiplikation



Teile & Herrsche

Verbesserte Integer Multiplikation

A	B	•	C	D

B · D

\+ B · C

\+ A · D

\+ A · C

= **Ergebnis**

Berechne BC + AD
Mit einer Multiplikation

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

Teile & Herrsche

Beispiel Schnelle Multiplikation

$$T(n) = 3 \cdot T(n/2) + cn$$

Anzahl Unterprobleme

Aufwand für Aufteilen und Zusammenfügen

Größe der Unterprobleme
(bestimmt Höhe des Rekursionsbaums)

- (und $T(1) = \text{const}$)

(n Zweierpotenz)

Teile & Herrsche

Aufwand verbesserte Integer Multiplikation

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

Laufzeit

$$T(n) = \begin{cases} 3T(n/2) + cn & \text{, falls } n > 1 \\ c & \text{, falls } n = 1 \end{cases} \quad c \text{ geeignete Konstante}$$

Teile & Herrsche

Aufwand verbesserte Integer Multiplikation

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

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

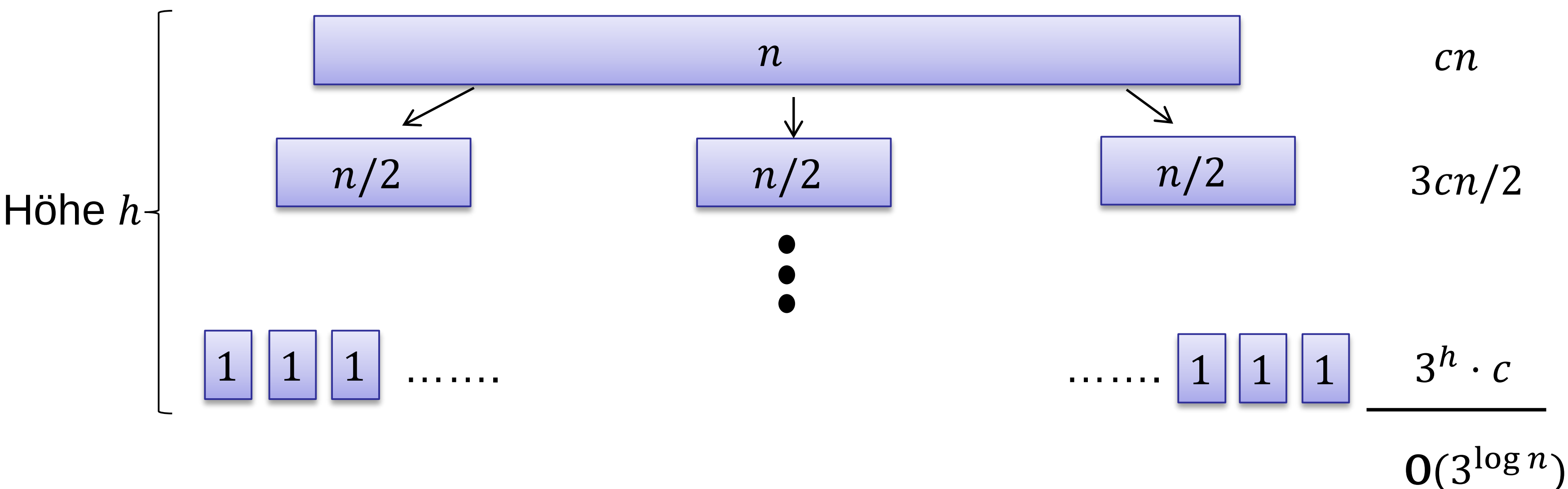
Laufzeit

$$T(n) = \begin{cases} 3T(n/2) + cn & \text{, falls } n > 1 \\ c & \text{, falls } n = 1 \end{cases} \quad c \text{ geeignete Konstante}$$

Teile & Herrsche

Laufzeit verbesserte Integer Multiplikation

$$T(n) = \begin{cases} 3T(n/2) + cn & \text{, falls } n > 1 \\ c & \text{, falls } n = 1 \end{cases} \qquad c \text{ geeignete Konstante}$$

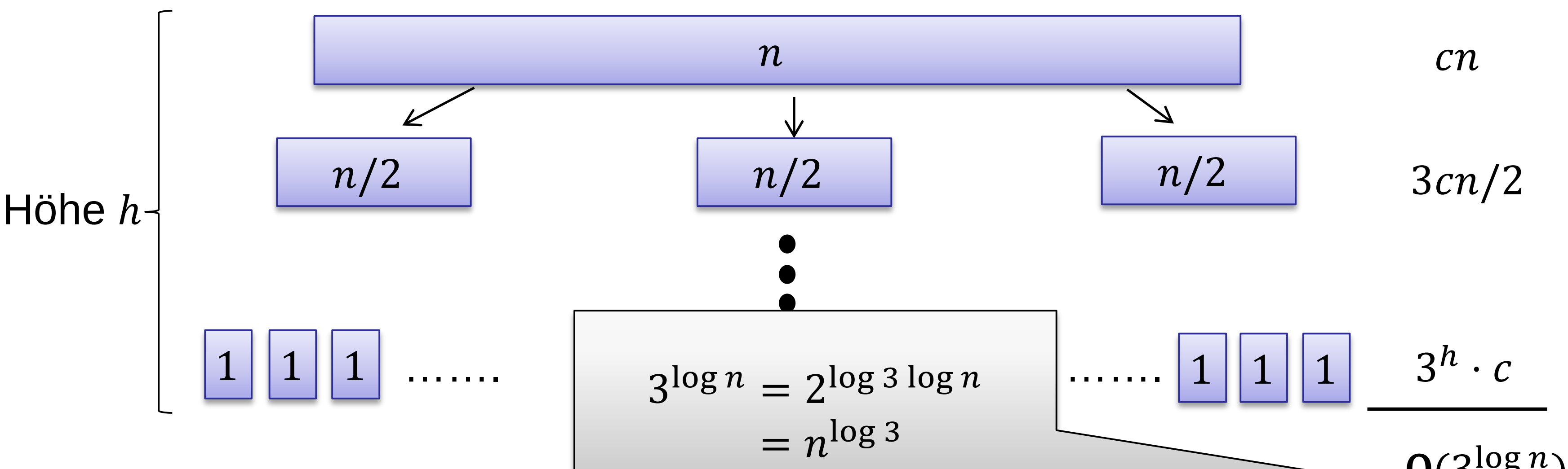


Höhe des Baums: $h = \log n$

Teile & Herrsche

Laufzeit verbesserte Integer Multiplikation

$$T(n) = \begin{cases} 3T(n/2) + cn & \text{, falls } n > 1 \\ c & \text{, falls } n = 1 \end{cases} \quad c \text{ geeignete Konstante}$$



Höhe des Baums: $h = \log n$

Teile & Herrsche

Satz 9

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

Beweis

- Wir zeigen den Satz nur, wenn n eine Zweierpotenz ist.

Teile & Herrsche

Satz 9

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

Beweis

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

Teile & Herrsche

Satz 9

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

Beweis

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

Teile & Herrsche

Satz 9

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

Beweis

- Wir zeigen den Satz nur, wenn n eine Zweierpotenz ist.
- Induktion über n. Sei $T(4) \leq c$. Wir zeigen $T(n) \leq c \cdot 3^{\log n} - 2cn$.
- (I.A.) Es gilt $T(4) \leq c = c \cdot 3^{\log 4} - 2 \cdot c \cdot 4$.
- (I.V.) Für jedes $m < n$, m Zweierpotenz, ist die Laufzeit für die Multiplikation zweier m-Bit Zahlen höchstens $c\ 3^{\log m} - 2cm$.

Teile & Herrsche

Satz 9

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

Beweis

- Wir zeigen den Satz nur, wenn n eine Zweierpotenz ist.
- Induktion über n. Sei $T(4) \le c$. Wir zeigen $T(n) \le c \cdot 3^{\log n} - 2cn$.
- (I.A.) Es gilt $T(4) \le c = c \cdot 3^{\log 4} - 2 \cdot c \cdot 4$.
- (I.V.) Für jedes $m < n$, m Zweierpotenz, ist die Laufzeit für die Multiplikation zweier m-Bit Zahlen höchstens $c\ 3^{\log m} - 2cm$.
- (I.S.) Betrachte die Multiplikation von zwei n-Bit Zahlen (n Zweierpotenz). Es gilt $T(n) = 3T(n/2) + cn \le c \cdot 3^{\log n} - 6c\ (n/2) + cn = c \cdot 3^{\log n} - 2cn$.

Teile & Herrsche

Satz 9

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

Beweis

- Wir zeigen den Satz nur, wenn n eine Zweierpotenz ist.
- Induktion über n. Sei $T(4) \leq c$. Wir zeigen $T(n) \leq c \cdot 3^{\log n} - 2cn$.
- (I.A.) Es gilt $T(4) \leq c = c \cdot 3^{\log 4} - 2 \cdot c \cdot 4$.
- (I.V.) Für jedes $m < n$, m Zweierpotenz, ist die Laufzeit für die Multiplikation zweier m-Bit Zahlen höchstens $c\ 3^{\log m} - 2cm$.
- (I.S.) Betrachte die Multiplikation von zwei n-Bit Zahlen (n Zweierpotenz). Es gilt $T(n) = 3T(n/2) + cn \leq c \cdot 3^{\log n} - 6c\ (n/2) + cn = c \cdot 3^{\log n} - 2cn$.

Teile & Herrsche

Satz 10

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

Beweis

- Die Laufzeit folgt aus Satz 9 wegen $1.59 \geq \log 3$. Korrektheit zeigen wir per Induktion über n.

Teile & Herrsche

Satz 10

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

Beweis

- Die Laufzeit folgt aus Satz 9 wegen $1.59 \geq \log 3$. Korrektheit zeigen wir per Induktion über n.
- (I.A.) Die Multiplikation zweier 1-Bit Zahlen wird vom Rechner korrekt ausgeführt.

Teile & Herrsche

Satz 10

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

Beweis

- Die Laufzeit folgt aus Satz 9 wegen $1.59 \geq \log 3$. Korrektheit zeigen wir per Induktion über n.
- (I.A.) Die Multiplikation zweier 1-Bit Zahlen wird vom Rechner korrekt ausgeführt.
- (I.V.) Die Multiplikation zweier m-Bit Zahlen für $m < n$ ist korrekt.

Teile & Herrsche

Satz 10

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

Beweis

- Die Laufzeit folgt aus Satz 9 wegen $1.59 \geq \log 3$. Korrektheit zeigen wir per Induktion über n.
- (I.A.) Die Multiplikation zweier 1-Bit Zahlen wird vom Rechner korrekt ausgeführt.
- (I.V.) Die Multiplikation zweier m-Bit Zahlen für $m < n$ ist korrekt.
- (I.S.) Nach (I.V.) werden die Produkte AC, BD, $(A+B)(C+D)$ korrekt berechnet.

Teile & Herrsche

Satz 10

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

Beweis

- Die Laufzeit folgt aus Satz 9 wegen $1.59 \geq \log 3$. Korrektheit zeigen wir per Induktion über n.
- (I.A.) Die Multiplikation zweier 1-Bit Zahlen wird vom Rechner korrekt ausgeführt.
- (I.V.) Die Multiplikation zweier m-Bit Zahlen für $m < n$ ist korrekt.
- (I.S.) Nach (I.V.) 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.

Teile & Herrsche

Satz 10

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

Beweis

- Die Laufzeit folgt aus Satz 9 wegen $1.59 \geq \log 3$. Korrektheit zeigen wir per Induktion über n.
- (I.A.) Die Multiplikation zweier 1-Bit Zahlen wird vom Rechner korrekt ausgeführt.
- (I.V.) Die Multiplikation zweier m-Bit Zahlen für $m < n$ ist korrekt.
- (I.S.) Nach (I.V.) 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.

Teile & Herrsche

Matrixmultiplikation

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

Zeile x Spalte

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

Teile & Herrsche

Matrixmultiplikation

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

Zeile x Spalte

- $5 \cdot 2 + 0 \cdot 2 + 0 \cdot 1 + 3 \cdot 0 = 10$

Teile & Herrsche

Matrixmultiplikation

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

Teile & Herrsche

Matrixmultiplikation

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

Teile & Herrsche

Matrixmultiplikation

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

Teile & Herrsche

Matrixmultiplikation

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

Teile & Herrsche

Matrixmultiplikation

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

Teile & Herrsche

Matrixmultiplikation

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

$$\begin{pmatrix} x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} \\ x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} \\ x_{3,1} & x_{3,2} & x_{3,3} & x_{3,4} \\ x_{4,1} & x_{4,2} & x_{4,3} & x_{4,4} \end{pmatrix} \bullet \begin{pmatrix} y_{1,1} & y_{1,2} & y_{1,3} & y_{1,4} \\ y_{2,1} & y_{2,2} & y_{2,3} & y_{2,4} \\ y_{3,1} & y_{3,2} & y_{3,3} & y_{3,4} \\ y_{4,1} & y_{4,2} & y_{4,3} & y_{4,4} \end{pmatrix} = \begin{pmatrix} z_{1,1} & z_{1,2} & z_{1,3} & z_{1,4} \\ z_{2,1} & z_{2,2} & z_{2,3} & z_{2,4} \\ z_{3,1} & z_{3,2} & z_{3,3} & z_{3,4} \\ z_{4,1} & z_{4,2} & z_{4,3} & z_{4,4} \end{pmatrix}$$

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)

1. $\boxed{\textbf{new array } Z[1..n][1..n]}$
2. **for** $i \leftarrow 1$ **to** n **do**
3. **for** $j \leftarrow 1$ **to** n **do**
4. $Z[i][j] \leftarrow 0$
5. **for** $k \leftarrow 1$ **to** n **do**
6. $Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$
7. **return** Z

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)

1. **new array** $Z[1..n][1..n]$
2. $\boxed{\textbf{for } i \leftarrow 1 \textbf{ to } n \textbf{ do}}$
3. **for** $j \leftarrow 1$ **to** n **do**
4. $Z[i][j] \leftarrow 0$
5. **for** $k \leftarrow 1$ **to** n **do**
6. $Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$
7. **return** Z

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)

1. **new array** $Z[1..n][1..n]$
2. **for** $i \leftarrow 1$ **to** n **do**
3. $\boxed{\textbf{for } j \leftarrow 1 \textbf{ to } n \textbf{ do}}$
4. $Z[i][j] \leftarrow 0$
5. **for** $k \leftarrow 1$ **to** n **do**
6. $Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$
7. **return** Z

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)

1. **new array** $Z[1..n][1..n]$
2. **for** $i \leftarrow 1$ **to** n **do**
3. **for** $j \leftarrow 1$ **to** n **do**
4. $\boxed{Z[i][j] \leftarrow 0}$
5. **for** $k \leftarrow 1$ **to** n **do**
6. $Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$
7. **return** Z

Teile & Herrsche

$\text{MatrixMultiplikation}(\text{Array } X, Y, n)$

$\textbf{new array } Z[1..n][1..n]$

$\textbf{for } i \leftarrow 1 \textbf{ to } n \textbf{ do}$

$\quad \textbf{for } j \leftarrow 1 \textbf{ to } n \textbf{ do}$

$\quad\quad Z[i][j] \leftarrow 0$

$\quad\quad \boxed{\textbf{for } k \leftarrow 1 \textbf{ to } n \textbf{ do}}$

$\quad\quad\quad Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$

$\textbf{return } Z$

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)

1. **new array** $Z[1..n][1..n]$
2. **for** $i \leftarrow 1$ **to** n **do**
3. **for** $j \leftarrow 1$ **to** n **do**
4. $Z[i][j] \leftarrow 0$
5. **for** $k \leftarrow 1$ **to** n **do**
6. $\boxed{Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]}$
7. **return** Z

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)

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

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n) Laufzeit

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

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)

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

Laufzeit

$\mathbf{O}(n^2)$

<u>Ausnahme (Rechenmodell):</u>
Dynamische Initialisierung eines Feldes benötigt Zeit proportional zur Größe des Feldes

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)	Laufzeit
1. **new array** $Z[1..n][1..n]$	$\mathbf{O}(n^2)$
2. **for** $i \leftarrow 1$ **to** n **do**	$\mathbf{O}(n)$
3. **for** $j \leftarrow 1$ **to** n **do**	
4. $Z[i][j] \leftarrow 0$	
5. **for** $k \leftarrow 1$ **to** n **do**	
6. $Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$	
7. **return** Z	

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)	Laufzeit
1. **new array** $Z[1..n][1..n]$	$\mathbf{O}(n^2)$
2. **for** $i \leftarrow 1$ **to** n **do**	$\mathbf{O}(n)$
3. **for** $j \leftarrow 1$ **to** n **do**	$\mathbf{O}(n^2)$
4. $Z[i][j] \leftarrow 0$	
5. **for** $k \leftarrow 1$ **to** n **do**	
6. $Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$	
7. **return** Z	

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)	Laufzeit
1. **new array** $Z[1..n][1..n]$	$\mathbf{O}(n^2)$
2. **for** $i \leftarrow 1$ **to** n **do**	$\mathbf{O}(n)$
3. **for** $j \leftarrow 1$ **to** n **do**	$\mathbf{O}(n^2)$
4. $Z[i][j] \leftarrow 0$	$\mathbf{O}(n^2)$
5. **for** $k \leftarrow 1$ **to** n **do**	
6. $Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$	
7. **return** Z	

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)	Laufzeit
new array $Z[1..n][1..n]$	$\mathbf{O}(n^2)$
for $i \leftarrow 1$ **to** n **do**	$\mathbf{O}(n)$
for $j \leftarrow 1$ **to** n **do**	$\mathbf{O}(n^2)$
$Z[i][j] \leftarrow 0$	$\mathbf{O}(n^2)$
for $k \leftarrow 1$ **to** n **do**	$\mathbf{O}(n^3)$
$Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$	
return Z	

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)	Laufzeit
1. **new array** $Z[1..n][1..n]$	$\mathbf{O}(n^2)$
2. **for** $i \leftarrow 1$ **to** n **do**	$\mathbf{O}(n)$
3. **for** $j \leftarrow 1$ **to** n **do**	$\mathbf{O}(n^2)$
4. $Z[i][j] \leftarrow 0$	$\mathbf{O}(n^2)$
5. **for** $k \leftarrow 1$ **to** n **do**	$\mathbf{O}(n^3)$
6. $Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$	$\mathbf{O}(n^3)$
7. **return** Z	

Teile & Herrsche

MatrixMultiplikation(Array X, Y, n)	Laufzeit
1. **new array** $Z[1..n][1..n]$	$\mathbf{O}(n^2)$
2. **for** $i \leftarrow 1$ **to** n **do**	$\mathbf{O}(n)$
3. **for** $j \leftarrow 1$ **to** n **do**	$\mathbf{O}(n^2)$
4. $Z[i][j] \leftarrow 0$	$\mathbf{O}(n^2)$
5. **for** $k \leftarrow 1$ **to** n **do**	$\mathbf{O}(n^3)$
6. $Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$	$\mathbf{O}(n^3)$
7. **return** Z	$\mathbf{O}(1)$

Teile & Herrsche

$\text{MatrixMultiplikation}(\text{Array } X, Y, n)$	Laufzeit
$\textbf{new array } Z[1..n][1..n]$	$\mathbf{O}(n^2)$
$\textbf{for } i \leftarrow 1 \textbf{ to } n \textbf{ do}$	$\mathbf{O}(n)$
$\quad \textbf{for } j \leftarrow 1 \textbf{ to } n \textbf{ do}$	$\mathbf{O}(n^2)$
$\quad\quad Z[i][j] \leftarrow 0$	$\mathbf{O}(n^2)$
$\quad\quad \textbf{for } k \leftarrow 1 \textbf{ to } n \textbf{ do}$	$\mathbf{O}(n^3)$
$\quad\quad\quad Z[i][j] \leftarrow Z[i][j] + X[i][k] \cdot Y[k][j]$	$\mathbf{O}(n^3)$
$\textbf{return } Z$	$\mathbf{O}(1)$
	$\mathbf{O}(n^3)$

Teile & Herrsche

Matrixmultiplikation

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

Aufwand

- 8 Multiplikationen von $n/2 \times n/2$ Matrizen
- 4 Additionen von $n/2 \times n/2$ Matrizen

Teile & Herrsche

Matrixmultiplikation

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

Aufwand

- 8 Multiplikationen von $n/2 \times n/2$ Matrizen
- 4 Additionen von $n/2 \times n/2$ Matrizen

- *Laufzeit*

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

Welche bestmögliche Laufzeit ergibt sich?

A) $\mathbf{O}(n^2)$

B) $\mathbf{O}(n^2 \log n)$

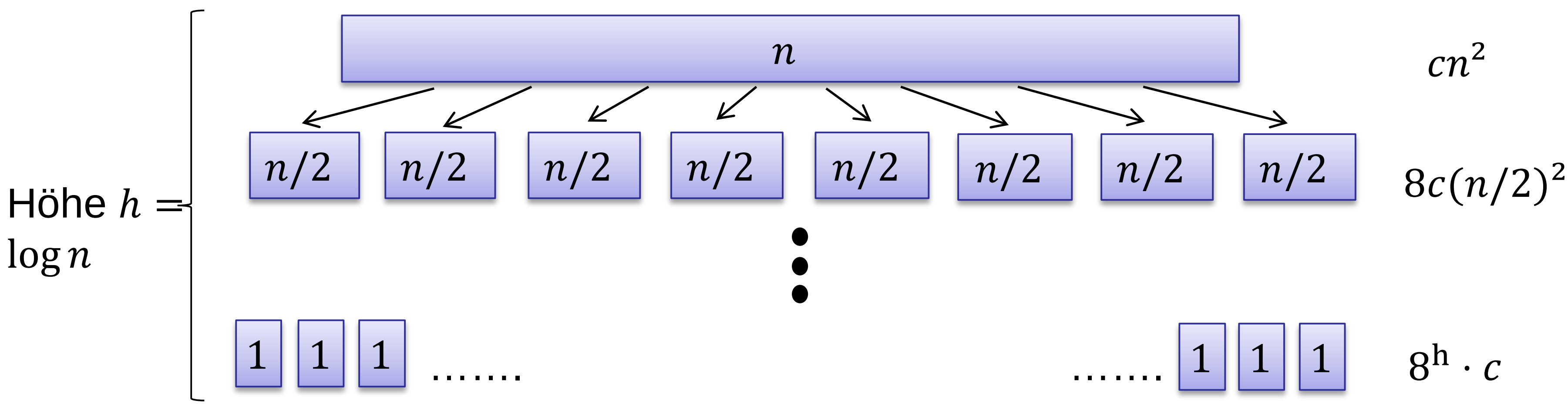
C) $\mathbf{O}(n^3)$

D) $\mathbf{O}(n^3 \log n)$

Teile & Herrsche

Laufzeit einfache Matrixmultiplikation

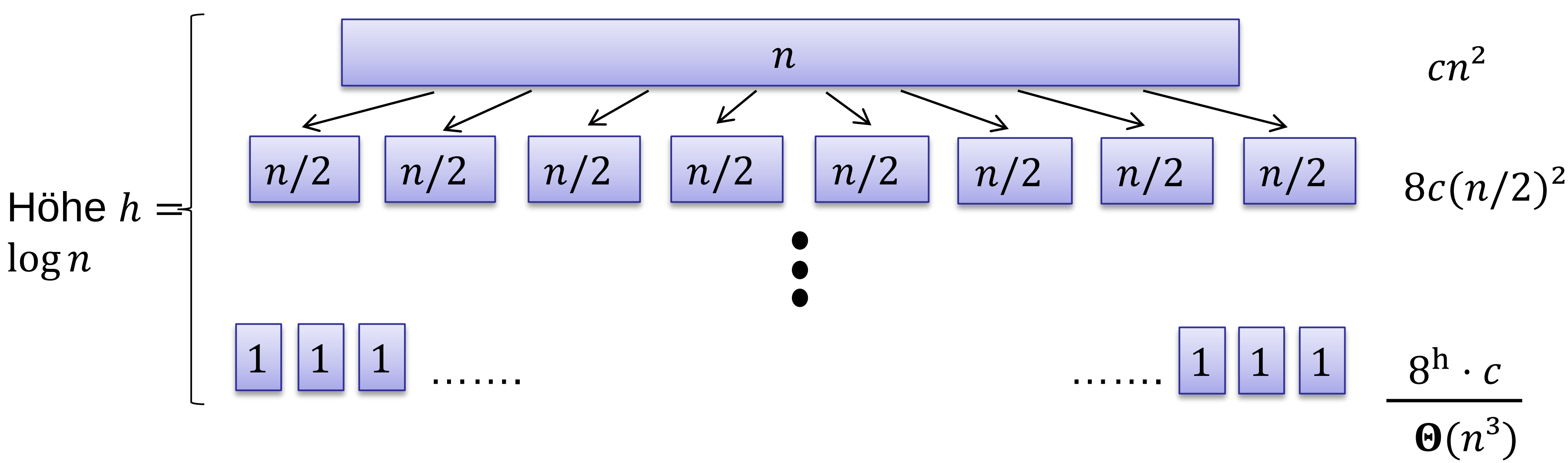
$$T(n) = \begin{cases} 8T(n/2) + cn^2 & ,n > 1 \\ c & ,n = 1 \end{cases} \quad c \text{ geeignete Konstante}$$



Teile & Herrsche

Laufzeit einfache Matrixmultiplikation

$$T(n) = \begin{cases} 8T(n/2) + cn^2 & , n > 1 \\ c & , n = 1 \end{cases} \quad c \text{ geeignete Konstante}$$



Teile & Herrsche

Matrixmultiplikation

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

Trick (wie bei Integer Multiplikation)

- $P_1 = A \cdot (F - H)$ $\quad P_5 = (A + D) \cdot (E + H)$
- $P_2 = (A + B) \cdot H$ $\quad P_6 = (B - D) \cdot (G + H)$
- $P_3 = (C + D) \cdot E$ $\quad P_7 = (A - C) \cdot (E + F)$
- $P_4 = D \cdot (G - E)$

Teile & Herrsche

Matrixmultiplikation

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

Trick (wie bei Integer Multiplikation)

- $P_1 = A \cdot (F - H)$ $\quad P_5 = (A + D) \cdot (E + H)$
- $P_2 = (A + B) \cdot H$ $\quad P_6 = (B - D) \cdot (G + H)$
- $P_3 = (C + D) \cdot E$ $\quad P_7 = (A - C) \cdot (E + F)$
- $P_4 = D \cdot (G - E)$

$$AE + BG = P_4 + P_5 + P_6 - P_2$$

$$AF + BH = P_1 + P_2$$

$$CE + DG = P_3 + P_4$$

$$CF + DH = P_1 + P_5 - P_3 - P_7$$

Teile & Herrsche

Matrixmultiplikation

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

Trick (wie bei Integer Multiplikation)

- $P_1 = A \cdot (F - H)$ $\quad P_5 = (A + D) \cdot (E + H)$
- $P_2 = (A + B) \cdot H$ $\quad P_6 = (B - D) \cdot (G + H)$
- $P_3 = (C + D) \cdot E$ $\quad P_7 = (A - C) \cdot (E + F)$
- $P_4 = D \cdot (G - E)$

$$AE + BG = P_4 + P_5 + P_6 - P_2$$

$$AF + BH = P_1 + P_2$$

$$CE + DG = P_3 + P_4$$

$$CF + DH = P_1 + P_5 - P_3 - P_7$$

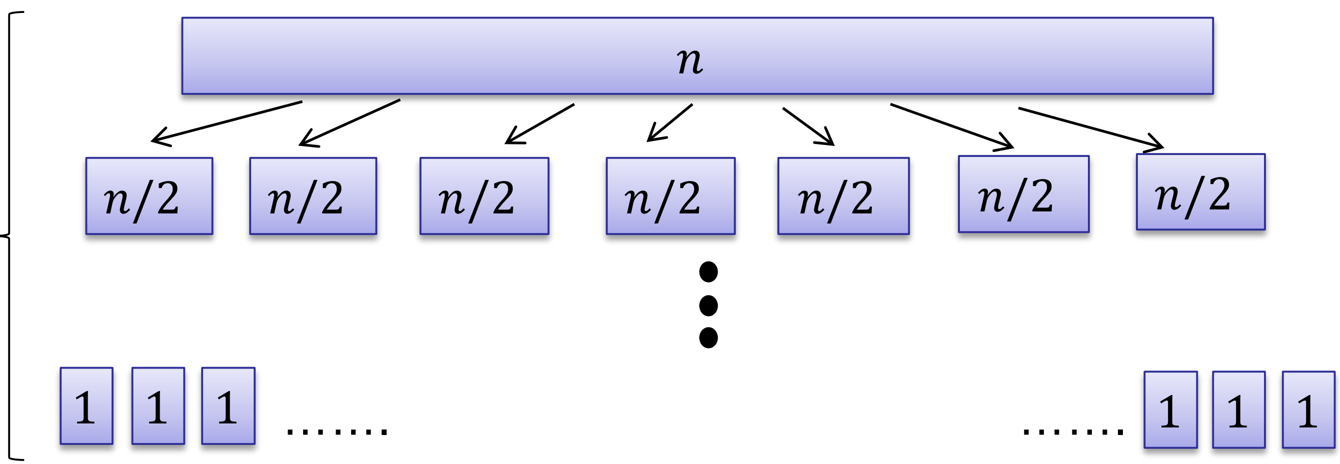
Nur 7 Multiplikationen !!!!!

Teile & Herrsche

Laufzeit schnelle Matrixmultiplikation

$$T(n) = \begin{cases} 7T(n/2) + cn^2 & ,n > 1 \\ c & ,n = 1 \end{cases} \quad c \text{ geeignete Konstante}$$

Höhe $h = \log n$



cn^2

$7c(n/2)^2$

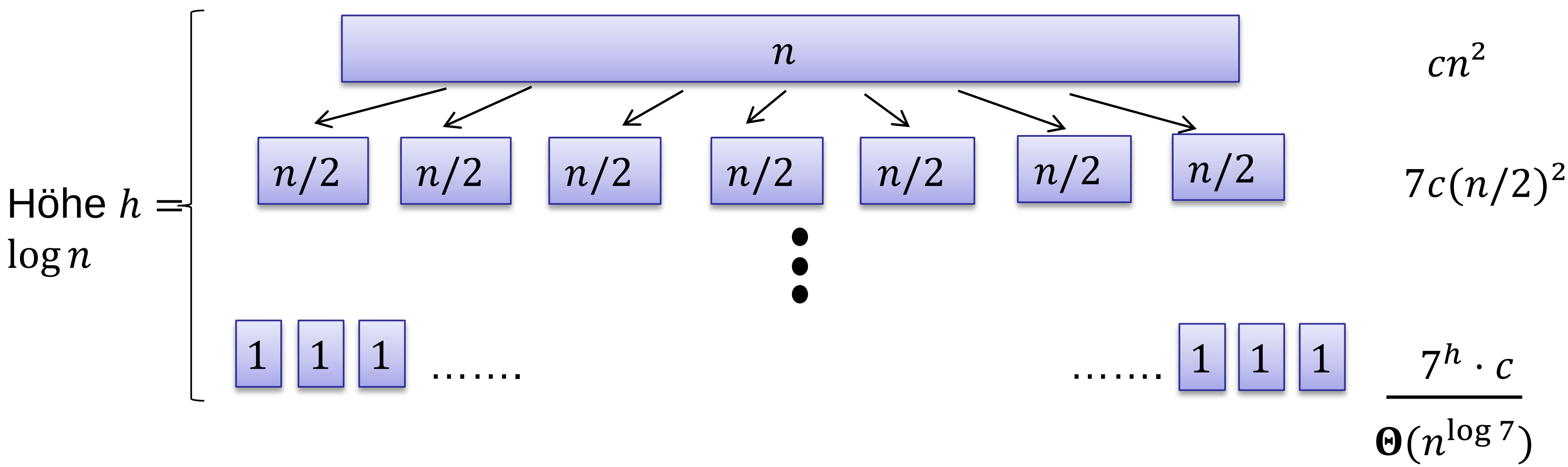
$7^h \cdot c$

$\mathbf{\Theta}(7^{\log n})$

Teile & Herrsche

Laufzeit schnelle Matrixmultiplikation

$$T(n) = \begin{cases} 7T(n/2) + cn^2 & , n > 1 \\ c & , n = 1 \end{cases} \quad c \text{ geeignete Konstante}$$



Teile & Herrsche

Satz 11

- Zwei $n \times n$-Matrizen können mit Hilfe des Teile & Herrsche Verfahrens in $\mathbf{O}(n^{2.81})$ worst case Laufzeit multipliziert werden.

Beweis

- Laufzeit und Korrektheit können einfach per Induktion gezeigt werden.

Teile & Herrsche

Zusammenfassung

- Multiplikation und Matrizenmultiplikation sind weitere Beispiel für Teile & Herrsche Algorithmen
- Faustregel: Je weniger rekursive Aufrufe desto schneller
- Trick bei Laufzeitbeweisen: Abziehen von Termen niedriger Ordnung

Teile & Herrsche

Laufzeit der Form

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

Anzahl Unterprobleme

Größe der Unterprobleme
(bestimmt Höhe des
Rekursionsbaums)

Aufwand für das Aufteilen
und Zusammensetzen

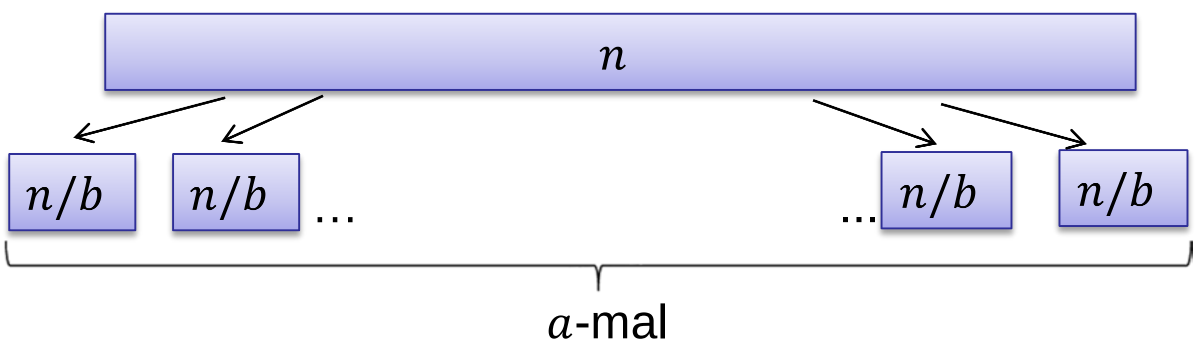
- wobei $T(1)$ konstant ist

Teile & Herrsche

$f(n) = \mathbf{O}(n^k)$ für Konstante k.

Laufzeit der Form

$$T(n) = \begin{cases} a\,T(n/b) + f(n) & , n > 1 \\ 1 & , n = 1 \end{cases}$$



$f(n)$

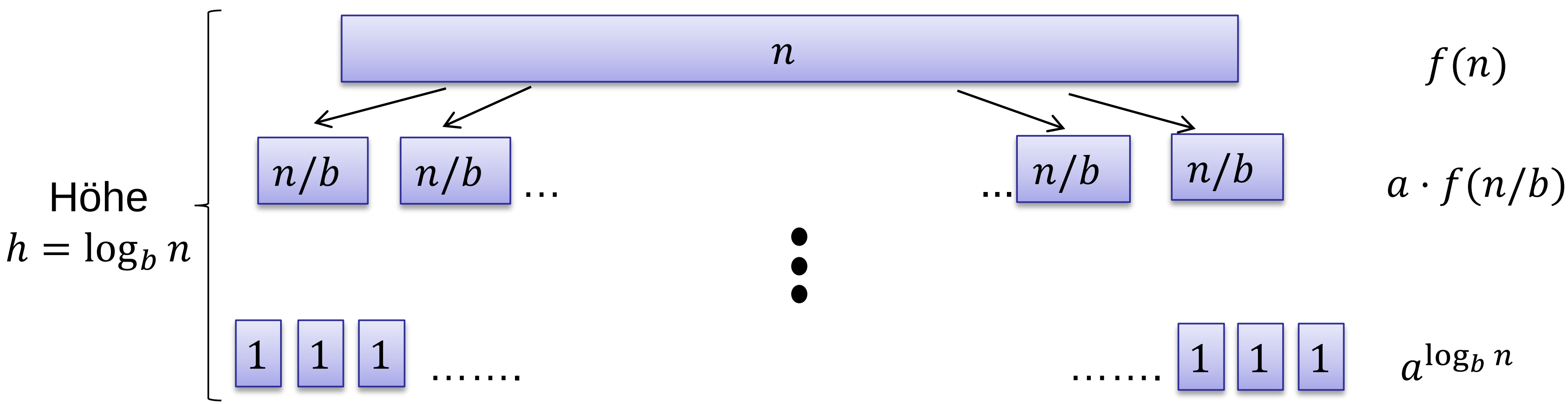
$a \cdot f(n/b)$

Teile & Herrsche

$f(n) = \mathbf{O}(n^k)$ für Konstante k.

Laufzeit der Form

- Setze $\gamma \cdot f(n) = a \cdot f(n/b)$



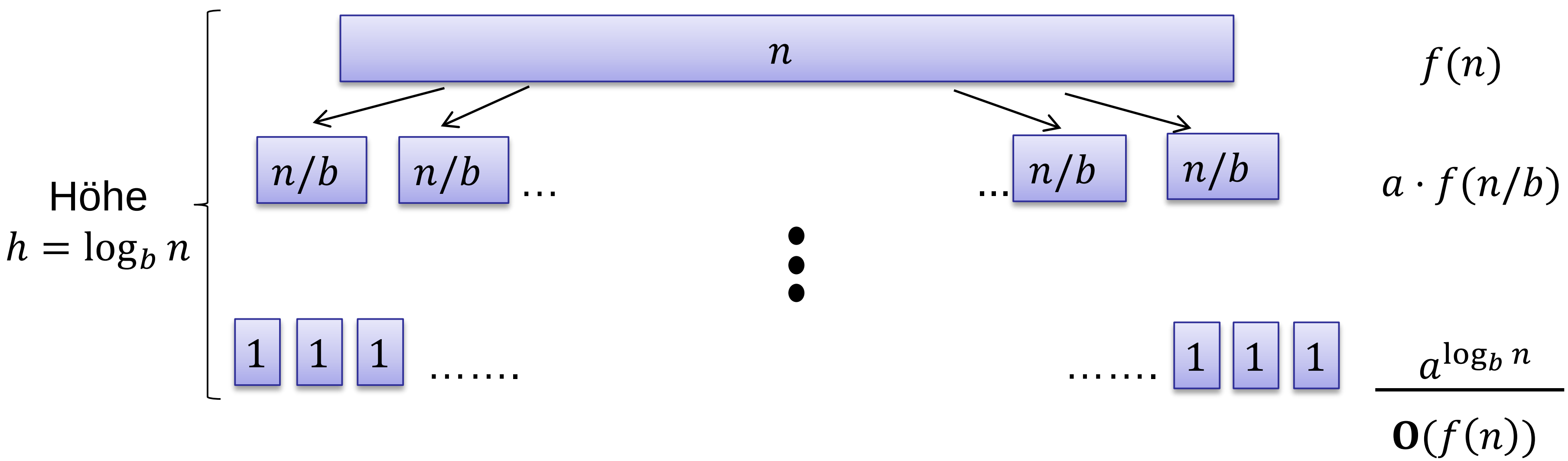
Teile & Herrsche

$f(n) = \mathbf{O}(n^k)$ für Konstante k.

Fall 1:
$\gamma < 1$

Laufzeit der Form

- Setze $\gamma \cdot f(n) = a \cdot f(n/b)$



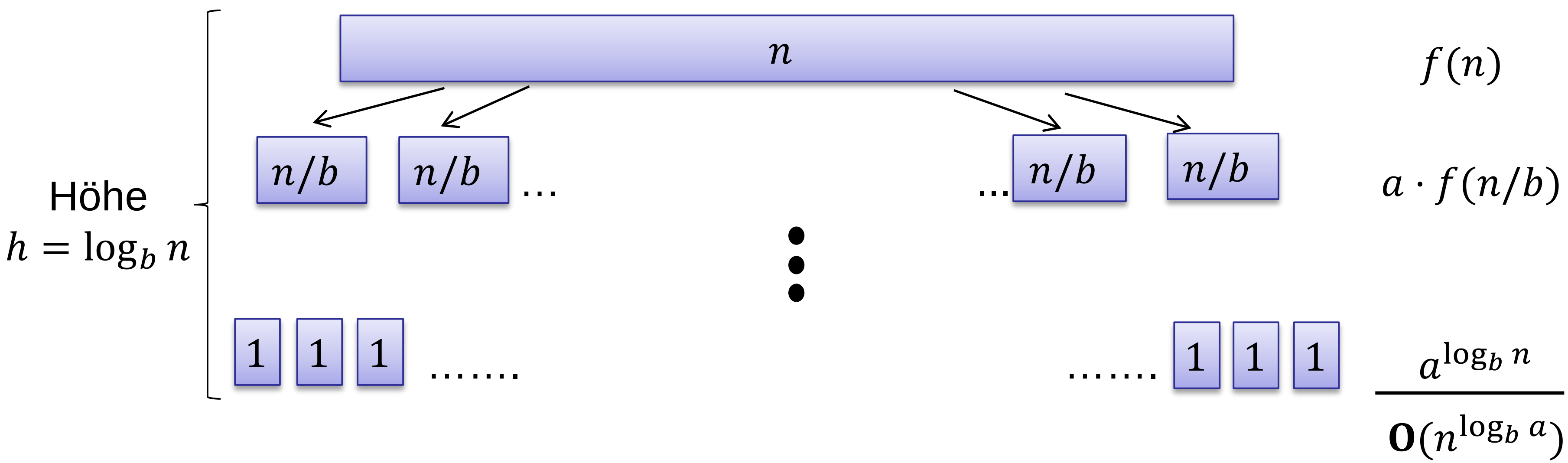
Teile & Herrsche

$f(n) = \mathbf{O}(n^k)$ für Konstante k.

Laufzeit der Form

- Setze $\gamma \cdot f(n) = a \cdot f(n/b)$

Fall 2:
$\gamma > 1$



Teile & Herrsche

$f(n) = \mathbf{O}(n^k)$ für Konstante k.

Laufzeit der Form

- Setze $\gamma \cdot f(n) = a \cdot f(n/b)$

Fall 3:
$\gamma = 1$