

# Datenstrukturen und Algorithmen

## Vorlesung 6: Mastertheorem (K4)

Joost-Pieter Katoen

Lehrstuhl für Informatik 2  
Software Modeling and Verification Group

<http://moves.rwth-aachen.de/teaching/ss-15/dsal/>

30. April 2015

# Übersicht

- 1 Lösen von Rekursionsgleichungen
  - Substitutionsmethode
  - Rekursionsbäume
  - Mastertheorem

# Übersicht

- 1 Lösen von Rekursionsgleichungen
  - Substitutionsmethode
  - Rekursionsbäume
  - Mastertheorem

# Rekursionsgleichungen

## Rekursionsgleichung

Für rekursive Algorithmen wird die Laufzeit meistens durch [Rekursionsgleichungen](#) beschrieben.

# Rekursionsgleichungen

## Rekursionsgleichung

Für rekursive Algorithmen wird die Laufzeit meistens durch [Rekursionsgleichungen](#) beschrieben.

Eine [Rekursionsgleichung](#) ist eine Gleichung oder eine Ungleichung, die eine Funktion durch ihre eigenen Funktionswerte für kleinere Eingaben beschreibt.

# Rekursionsgleichungen

## Rekursionsgleichung

Für rekursive Algorithmen wird die Laufzeit meistens durch **Rekursionsgleichungen** beschrieben.

Eine **Rekursionsgleichung** ist eine Gleichung oder eine Ungleichung, die eine Funktion durch ihre eigenen Funktionswerte für kleinere Eingaben beschreibt.

## Beispiele

- ▶  $T(n) = T(\lceil (n-1)/2 \rceil) + 1$  Binäre Suche
- ▶  $T(n) = T(n-1) + n - 1$  Bubblesort
- ▶  $T(n) = 2 \cdot T(n/2) + n - 1$  Mergesort
- ▶  $T(n) = 7 \cdot T(n/2) + c \cdot n^2$  Strassen's Matrixmultiplikation

Die zentrale Frage ist: Wie **löst** man solche Rekursionsgleichungen?

# Die Substitutionsmethode

## Substitutionsmethode

Die Substitutionsmethode besteht aus zwei Schritten:

1. **Rate** die Form der Lösung, durch z. B.:
  - ▶ Scharfes Hinsehen, kurze Eingaben ausprobieren und einsetzen
  - ▶ Betrachtung des Rekursionsbaums

# Die Substitutionsmethode

## Substitutionsmethode

Die Substitutionsmethode besteht aus zwei Schritten:

1. **Rate** die Form der Lösung, durch z. B.:
  - ▶ Scharfes Hinsehen, kurze Eingaben ausprobieren und einsetzen
  - ▶ Betrachtung des Rekursionsbaums
2. **Vollständige Induktion**, um die Konstanten zu finden und zu zeigen, dass die Lösung funktioniert.



# Die Substitutionsmethode: Beispiel

## Beispiel

Betrachte folgende Rekursionsgleichung:

$$T(1) = 1$$

$$T(n) = 2 \cdot T(n/2) + n \quad \text{für } n > 1.$$

# Die Substitutionsmethode: Beispiel

## Beispiel

Betrachte folgende Rekursionsgleichung:

$$T(1) = 1$$

$$T(n) = 2 \cdot T(n/2) + n \quad \text{für } n > 1.$$

- Wir vermuten als Lösung  $T(n) \in O(n \cdot \log n)$ .

# Die Substitutionsmethode: Beispiel

## Beispiel

Betrachte folgende Rekursionsgleichung:

$$T(1) = 1$$

$$T(n) = 2 \cdot T(n/2) + n \quad \text{für } n > 1.$$

- ▶ Wir vermuten als Lösung  $T(n) \in O(n \cdot \log n)$ .
- ▶ Dazu müssen wir  $T(n) \leq c \cdot n \cdot \log n$  zeigen, für ein geeignetes  $c > 0$ .

# Die Substitutionsmethode: Beispiel

## Beispiel

Betrachte folgende Rekursionsgleichung:

$$T(1) = 1$$

$$T(n) = 2 \cdot T(n/2) + n \quad \text{für } n > 1.$$

- ▶ Wir vermuten als Lösung  $T(n) \in O(n \cdot \log n)$ .
- ▶ Dazu müssen wir  $T(n) \leq c \cdot n \cdot \log n$  zeigen, für ein geeignetes  $c > 0$ .
- ▶ Bestimme, ob für ein geeignetes  $n_0$  und für  $n \geq n_0$  gilt, dass  $T(n) \leq c \cdot n \cdot \log n$ .

# Die Substitutionsmethode: Beispiel

## Beispiel

Betrachte folgende Rekursionsgleichung:

$$T(1) = 1$$

$$T(n) = 2 \cdot T(n/2) + n \quad \text{für } n > 1.$$

- ▶ Wir vermuten als Lösung  $T(n) \in O(n \cdot \log n)$ .
- ▶ Dazu müssen wir  $T(n) \leq c \cdot n \cdot \log n$  zeigen, für ein geeignetes  $c > 0$ .
- ▶ Bestimme, ob für ein geeignetes  $n_0$  und für  $n \geq n_0$  gilt, dass  $T(n) \leq c \cdot n \cdot \log n$ .
- ▶ Stelle fest, dass  $T(1) = 1 \leq c \cdot 1 \cdot \log 1 = 0$  **verletzt** ist.

# Die Substitutionsmethode: Beispiel

## Beispiel

Betrachte folgende Rekursionsgleichung:

$$T(1) = 1$$

$$T(n) = 2 \cdot T(n/2) + n \quad \text{für } n > 1.$$

- ▶ Wir vermuten als Lösung  $T(n) \in O(n \cdot \log n)$ .
- ▶ Dazu müssen wir  $T(n) \leq c \cdot n \cdot \log n$  zeigen, für ein geeignetes  $c > 0$ .
- ▶ Bestimme, ob für ein geeignetes  $n_0$  und für  $n \geq n_0$  gilt, dass  $T(n) \leq c \cdot n \cdot \log n$ .
- ▶ Stelle fest, dass  $T(1) = 1 \leq c \cdot 1 \cdot \log 1 = 0$  **verletzt** ist.
- ▶ Es gilt:  $T(2) = 4 \leq c \cdot 2 \log 2$  und  $T(3) = 5 \leq c \cdot 3 \log 3$  für  $c \geq 2$

# Die Substitutionsmethode: Beispiel

## Beispiel

Betrachte folgende Rekursionsgleichung:

$$T(1) = 1$$

$$T(n) = 2 \cdot T(n/2) + n \quad \text{für } n > 1.$$

- ▶ Wir vermuten als Lösung  $T(n) \in O(n \cdot \log n)$ .
- ▶ Dazu müssen wir  $T(n) \leq c \cdot n \cdot \log n$  zeigen, für ein geeignetes  $c > 0$ .
- ▶ Bestimme, ob für ein geeignetes  $n_0$  und für  $n \geq n_0$  gilt, dass  $T(n) \leq c \cdot n \cdot \log n$ .
- ▶ Stelle fest, dass  $T(1) = 1 \leq c \cdot 1 \cdot \log 1 = 0$  **verletzt** ist.
- ▶ Es gilt:  $T(2) = 4 \leq c \cdot 2 \log 2$  und  $T(3) = 5 \leq c \cdot 3 \log 3$  für  $c \geq 2$
- ▶ **Überprüfe** dann durch Substitution und Induktion (s. nächste Folie)

# Die Substitutionsmethode: Beispiel

## Beispiel

Betrachte folgende Rekursionsgleichung:

$$T(1) = 1$$

$$T(n) = 2 \cdot T(n/2) + n \quad \text{für } n > 1.$$

- ▶ Wir vermuten als Lösung  $T(n) \in O(n \cdot \log n)$ .
- ▶ Dazu müssen wir  $T(n) \leq c \cdot n \cdot \log n$  zeigen, für ein geeignetes  $c > 0$ .
- ▶ Bestimme, ob für ein geeignetes  $n_0$  und für  $n \geq n_0$  gilt, dass  $T(n) \leq c \cdot n \cdot \log n$ .
- ▶ Stelle fest, dass  $T(1) = 1 \leq c \cdot 1 \cdot \log 1 = 0$  **verletzt** ist.
- ▶ Es gilt:  $T(2) = 4 \leq c \cdot 2 \log 2$  und  $T(3) = 5 \leq c \cdot 3 \log 3$  für  $c \geq 2$
- ▶ **Überprüfe** dann durch Substitution und Induktion (s. nächste Folie)
- ▶ Damit gilt für jedes  $c \geq 2$  und  $n \geq n_0 > 1$ , dass  $T(n) \leq c \cdot n \cdot \log n$ .



# Die Substitutionsmethode: Beispiel

## Beispiel

$$T(n) = 2 \cdot T(n/2) + n \text{ für } n > 1, \text{ und } T(1) = 1$$

$$T(n) = 2 \cdot T(n/2) + n \quad | \text{ Induktionshypothese}$$

$$\leq 2(c \cdot n/2 \cdot \log n/2) + n$$

$$= c \cdot n \cdot \log n/2 + n$$

$$\begin{array}{l} | \text{ log-Rechnung: } (\log \equiv \log_2) \\ | \log n/2 = \log n - \log 2 \end{array}$$

$$= c \cdot n \cdot \log n - c \cdot n \cdot \log 2 + n$$

$$\leq c \cdot n \cdot \log n - c \cdot n + n$$

$$| \text{ mit } c > 1 \text{ folgt sofort:}$$

$$\leq c \cdot n \cdot \log n$$

# Raten der Lösung durch Iteration

## Grundidee

Wiederholtes Einsetzen der Rekursionsgleichung in sich selbst, bis man ein Muster erkennt.

# Raten der Lösung durch Iteration

## Grundidee

Wiederholtes Einsetzen der Rekursionsgleichung in sich selbst, bis man ein Muster erkennt.

## Beispiel

$$\begin{aligned}T(n) &= 3 \cdot T(n/4) + n && | \text{ Einsetzen} \\&= 3 \cdot (3 \cdot T(n/16) + n/4) + n && | \text{ Nochmal einsetzen} \\&= 9 \cdot (3 \cdot T(n/64) + n/16) + 3 \cdot n/4 + n && | \text{ Vereinfachen} \\&= 27 \cdot T(n/64) + \left(\frac{3}{4}\right)^2 \cdot n + \left(\frac{3}{4}\right)^1 \cdot n + \left(\frac{3}{4}\right)^0 \cdot n\end{aligned}$$

# Raten der Lösung durch Iteration

## Grundidee

Wiederholtes Einsetzen der Rekursionsgleichung in sich selbst, bis man ein Muster erkennt.

## Beispiel

$$\begin{aligned}
 T(n) &= 3 \cdot T(n/4) + n && | \text{ Einsetzen} \\
 &= 3 \cdot (3 \cdot T(n/16) + n/4) + n && | \text{ Nochmal einsetzen} \\
 &= 9 \cdot (3 \cdot T(n/64) + n/16) + 3 \cdot n/4 + n && | \text{ Vereinfachen} \\
 &= 27 \cdot T(n/64) + \left(\frac{3}{4}\right)^2 \cdot n + \left(\frac{3}{4}\right)^1 \cdot n + \left(\frac{3}{4}\right)^0 \cdot n
 \end{aligned}$$

Wir nehmen  $T(1) = c$  an und erhalten:  $T(n) = \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{4}\right)^i \cdot n + c \cdot n^{\log_4 3}$

# Raten der Lösung durch Iteration

## Grundidee

Wiederholtes Einsetzen der Rekursionsgleichung in sich selbst, bis man ein Muster erkennt.

## Beispiel

$$\begin{aligned}
 T(n) &= 3 \cdot T(n/4) + n && | \text{ Einsetzen} \\
 &= 3 \cdot (3 \cdot T(n/16) + n/4) + n && | \text{ Nochmal einsetzen} \\
 &= 9 \cdot (3 \cdot T(n/64) + n/16) + 3 \cdot n/4 + n && | \text{ Vereinfachen} \\
 &= 27 \cdot T(n/64) + \left(\frac{3}{4}\right)^2 \cdot n + \left(\frac{3}{4}\right)^1 \cdot n + \left(\frac{3}{4}\right)^0 \cdot n
 \end{aligned}$$

Wir nehmen  $T(1) = c$  an und erhalten:  $T(n) = \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{4}\right)^i \cdot n + c \cdot n^{\log_4 3}$

Diese Aussage kann mit Hilfe der Substitutionsmethode gezeigt werden.

# Raten der Lösung durch Rekursionsbäume

## Grundidee

Stelle das Ineinander-Einsetzen als Baum dar, indem man Buch über das **aktuelle Rekursionsargument** und **die nicht-rekursiven Kosten** führt.

# Raten der Lösung durch Rekursionsbäume

## Grundidee

Stelle das Ineinander-Einsetzen als Baum dar, indem man Buch über das **aktuelle Rekursionsargument** und **die nicht-rekursiven Kosten** führt.

## Rekursionsbaum

1. Jeder **Knoten** stellt die Kosten eines Teilproblems dar.

# Raten der Lösung durch Rekursionsbäume

## Grundidee

Stelle das Ineinander-Einsetzen als Baum dar, indem man Buch über das **aktuelle Rekursionsargument** und **die nicht-rekursiven Kosten** führt.

## Rekursionsbaum

1. Jeder **Knoten** stellt die Kosten eines Teilproblems dar.
  - ▶ Die Wurzel stellt die zu analysierenden Kosten  $T(n)$  dar.



# Raten der Lösung durch Rekursionsbäume

## Grundidee

Stelle das Ineinander-Einsetzen als Baum dar, indem man Buch über das **aktuelle Rekursionsargument** und die **nicht-rekursiven Kosten** führt.

## Rekursionsbaum

1. Jeder **Knoten** stellt die Kosten eines Teilproblems dar.
  - ▶ Die Wurzel stellt die zu analysierenden Kosten  $T(n)$  dar.
  - ▶ Die Blätter stellen die Kosten der Basisfälle dar, z. B.  $T(0)$  oder  $T(1)$ .

# Raten der Lösung durch Rekursionsbäume

## Grundidee

Stelle das Ineinander-Einsetzen als Baum dar, indem man Buch über das **aktuelle Rekursionsargument** und **die nicht-rekursiven Kosten** führt.

## Rekursionsbaum

1. Jeder **Knoten** stellt die Kosten eines Teilproblems dar.
  - ▶ Die Wurzel stellt die zu analysierenden Kosten  $T(n)$  dar.
  - ▶ Die Blätter stellen die Kosten der Basisfälle dar, z. B.  $T(0)$  oder  $T(1)$ .
2. Wir summieren die Kosten innerhalb jeder **Ebene** des Baumes.

# Raten der Lösung durch Rekursionsbäume

## Grundidee

Stelle das Ineinander-Einsetzen als Baum dar, indem man Buch über das **aktuelle Rekursionsargument** und **die nicht-rekursiven Kosten** führt.

## Rekursionsbaum

1. Jeder **Knoten** stellt die Kosten eines Teilproblems dar.
  - ▶ Die Wurzel stellt die zu analysierenden Kosten  $T(n)$  dar.
  - ▶ Die Blätter stellen die Kosten der Basisfälle dar, z. B.  $T(0)$  oder  $T(1)$ .
2. Wir summieren die Kosten innerhalb jeder **Ebene** des Baumes.
3. Die **Gesamtkosten** := summieren über **die Kosten aller Ebenen**.

# Raten der Lösung durch Rekursionsbäume

## Grundidee

Stelle das Ineinander-Einsetzen als Baum dar, indem man Buch über **das aktuelle Rekursionsargument** und **die nicht-rekursiven Kosten** führt.

## Rekursionsbaum

1. Jeder **Knoten** stellt die Kosten eines Teilproblems dar.
  - ▶ Die Wurzel stellt die zu analysierenden Kosten  $T(n)$  dar.
  - ▶ Die Blätter stellen die Kosten der Basisfälle dar, z. B.  $T(0)$  oder  $T(1)$ .
2. Wir summieren die Kosten innerhalb jeder **Ebene** des Baumes.
3. Die **Gesamtkosten** := summieren über **die Kosten aller Ebenen**.

## Wichtiger Hinweis

Ein Rekursionsbaum ist sehr nützlich, um eine Lösung zu raten, die dann mit Hilfe der Substitutionsmethode überprüft werden kann.

# Raten der Lösung durch Rekursionsbäume

## Grundidee

Stelle das Ineinander-Einsetzen als Baum dar, indem man Buch über **das aktuelle Rekursionsargument** und **die nicht-rekursiven Kosten** führt.

## Rekursionsbaum

1. Jeder **Knoten** stellt die Kosten eines Teilproblems dar.
  - ▶ Die Wurzel stellt die zu analysierenden Kosten  $T(n)$  dar.
  - ▶ Die Blätter stellen die Kosten der Basisfälle dar, z. B.  $T(0)$  oder  $T(1)$ .
2. Wir summieren die Kosten innerhalb jeder **Ebene** des Baumes.
3. Die **Gesamtkosten** := summieren über **die Kosten aller Ebenen**.

## Wichtiger Hinweis

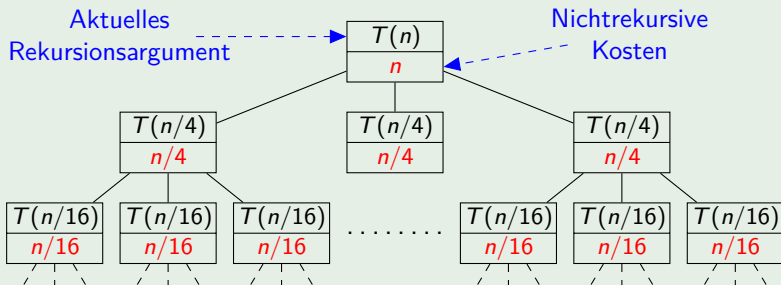
Ein Rekursionsbaum ist sehr nützlich, um eine Lösung zu raten, die dann mit Hilfe der Substitutionsmethode überprüft werden kann.

**Der Baum selber reicht jedoch meistens nicht als Beweis.**

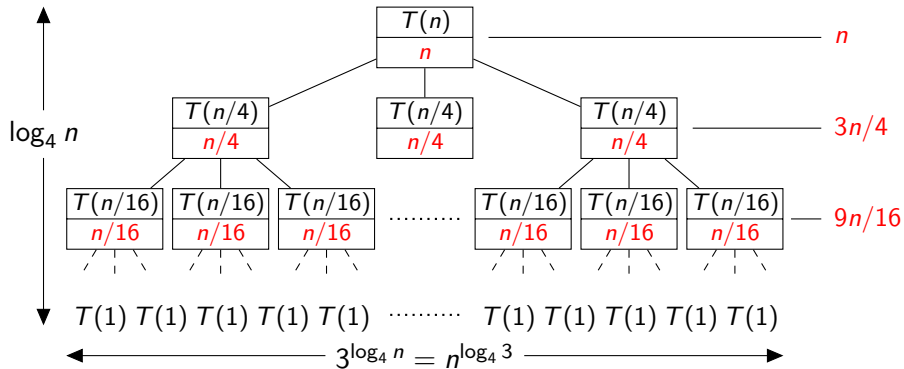
# Rekursionsbaum: Beispiel

## Beispiel

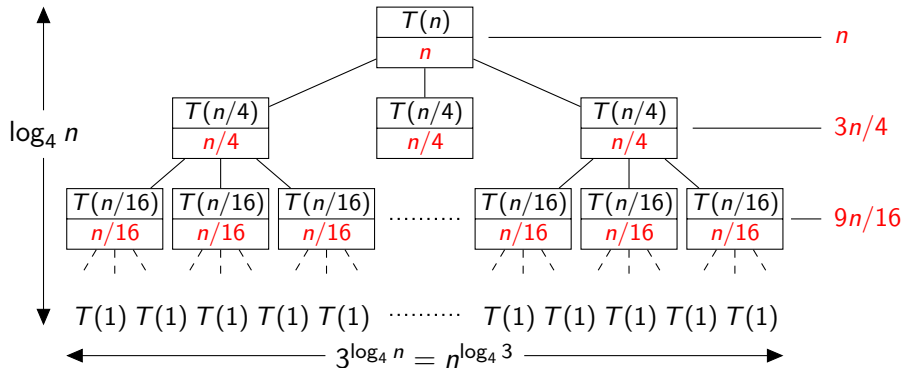
Der Rekursionsbaum von  $T(n) = 3 \cdot T(n/4) + n$  sieht etwa so aus:



# Rekursionsbaum: Beispiel



# Rekursionsbaum: Beispiel



$$T(n) = \underbrace{\sum_{i=0}^{\log_4 n - 1}}_{\text{Summe über alle Ebenen}} \underbrace{\left(\frac{3}{4}\right)^i \cdot n}_{\text{Kosten pro Ebene}} + \underbrace{c \cdot n^{\log_4 3}}_{\text{Gesamtkosten für die Blätter mit } T(1) = c}$$



# Rekursionsbaum: Beispiel

Eine obere Schranke für die Komplexität erhält man nun folgendermaßen:

$$T(n) = \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{4}\right)^i \cdot n + c \cdot n^{\log_4 3} \quad | \text{ Vernachlässigen kleinerer Terme}$$

$$< \sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^i \cdot n + c \cdot n^{\log_4 3} \quad | \text{ Geometrische Reihe}$$

$$< \frac{1}{1 - (3/4)} \cdot n + c \cdot n^{\log_4 3} \quad | \text{ Umformen}$$

$$< 4 \cdot n + c \cdot n^{\log_4 3} \quad | \text{ Asymptotische Ordnung bestimmen}$$

setze ein, dass  $\log_4 3 < 1$

$$T(n) \in O(n).$$

# Korrektheit

Wir können die Substitutionsmethode benutzen, um die Vermutung zu bestätigen, dass:

$T(n) \in O(n)$  eine obere Schranke von  $T(n) = 3 \cdot T(n/4) + n$  ist.

# Korrektheit

Wir können die Substitutionsmethode benutzen, um die Vermutung zu bestätigen, dass:

$T(n) \in O(n)$  eine obere Schranke von  $T(n) = 3 \cdot T(n/4) + n$  ist.

$$T(n) = 3 \cdot T(n/4) + n \quad | \text{ Induktionshypothese}$$

$$\leq 3d \cdot n/4 + n$$

$$= \frac{3}{4}d \cdot n + n$$

$$= \left( \frac{3}{4}d + 1 \right) \cdot n \quad | \text{ mit } d \geq 4 \text{ folgt sofort:}$$

$$\leq d \cdot n$$

# Korrektheit

Wir können die Substitutionsmethode benutzen, um die Vermutung zu bestätigen, dass:

$T(n) \in O(n)$  eine obere Schranke von  $T(n) = 3 \cdot T(n/4) + n$  ist.

$$T(n) = 3 \cdot T(n/4) + n \quad | \text{ Induktionshypothese}$$

$$\leq 3d \cdot n/4 + n$$

$$= \frac{3}{4}d \cdot n + n$$

$$= \left( \frac{3}{4}d + 1 \right) \cdot n \quad | \text{ mit } d \geq 4 \text{ folgt sofort:}$$

$$\leq d \cdot n$$

Und wir stellen fest, dass es ein  $n_0$  gibt, sodass  $T(n_0) \leq d \cdot n_0$  ist.

# Mastertheorem

## Allgemeines Format der Rekursionsgleichung

Eine Rekursionsgleichung für die Komplexitätsanalyse sieht meistens folgendermaßen aus:

$$T(n) = b \cdot T\left(\frac{n}{c}\right) + f(n)$$

wobei  $b > 0$ ,  $c > 1$  gilt und  $f(n)$  eine gegebene Funktion ist.

# Mastertheorem

## Allgemeines Format der Rekursionsgleichung

Eine Rekursionsgleichung für die Komplexitätsanalyse sieht meistens folgendermaßen aus:

$$T(n) = b \cdot T\left(\frac{n}{c}\right) + f(n)$$

wobei  $b > 0$ ,  $c > 1$  gilt und  $f(n)$  eine gegebene Funktion ist.

## Intuition:

- ▶ Das zu analysierende Problem teilt sich jeweils in  $b$  Teilprobleme auf.

# Mastertheorem

## Allgemeines Format der Rekursionsgleichung

Eine Rekursionsgleichung für die Komplexitätsanalyse sieht meistens folgendermaßen aus:

$$T(n) = b \cdot T\left(\frac{n}{c}\right) + f(n)$$

wobei  $b > 0$ ,  $c > 1$  gilt und  $f(n)$  eine gegebene Funktion ist.

## Intuition:

- ▶ Das zu analysierende Problem teilt sich jeweils in  $b$  Teilprobleme auf.
- ▶ Jedes dieser Teilprobleme hat die Größe  $\frac{n}{c}$ .

# Mastertheorem

## Allgemeines Format der Rekursionsgleichung

Eine Rekursionsgleichung für die Komplexitätsanalyse sieht meistens folgendermaßen aus:

$$T(n) = b \cdot T\left(\frac{n}{c}\right) + f(n)$$

wobei  $b > 0$ ,  $c > 1$  gilt und  $f(n)$  eine gegebene Funktion ist.

## Intuition:

- ▶ Das zu analysierende Problem teilt sich jeweils in  $b$  Teilprobleme auf.
- ▶ Jedes dieser Teilprobleme hat die Größe  $\frac{n}{c}$ .
- ▶ Die Kosten für das Aufteilen eines Problems und Kombinieren der Teillösungen sind  $f(n)$ .



# Das Mastertheorem

$$T(n) = b \cdot T\left(\frac{n}{c}\right) + f(n) \quad \text{mit } b \geq 1 \text{ und } c > 1.$$

- ▶ Anzahl der Blätter im Rekursionsbaum:  $n^E$  mit  $E = \log b / \log c$ .

# Das Mastertheorem

$$T(n) = b \cdot T\left(\frac{n}{c}\right) + f(n) \quad \text{mit } b \geq 1 \text{ und } c > 1.$$

- Anzahl der Blätter im Rekursionsbaum:  $n^E$  mit  $E = \log b / \log c$ .

## Mastertheorem

Wenn

Dann

1.  $f(n) \in O(n^{E-\varepsilon})$  für ein  $\varepsilon > 0$

$$T(n) \in \Theta(n^E)$$

# Das Mastertheorem

$$T(n) = b \cdot T\left(\frac{n}{c}\right) + f(n) \quad \text{mit } b \geq 1 \text{ und } c > 1.$$

- Anzahl der Blätter im Rekursionsbaum:  $n^E$  mit  $E = \log b / \log c$ .

## Mastertheorem

Wenn

Dann

1.  $f(n) \in O(n^{E-\varepsilon})$  für ein  $\varepsilon > 0$

$$T(n) \in \Theta(n^E)$$

2.  $f(n) \in \Theta(n^E)$

$$T(n) \in \Theta(n^E \cdot \log n)$$

# Das Mastertheorem

$$T(n) = b \cdot T\left(\frac{n}{c}\right) + f(n) \quad \text{mit } b \geq 1 \text{ und } c > 1.$$

- Anzahl der Blätter im Rekursionsbaum:  $n^E$  mit  $E = \log b / \log c$ .

## Mastertheorem

Wenn

Dann

- |   |                                     |
|---|-------------------------------------|
| 1. $f(n) \in O(n^{E-\varepsilon})$ für ein $\varepsilon > 0$  | $T(n) \in \Theta(n^E)$              |
| 2. $f(n) \in \Theta(n^E)$   | $T(n) \in \Theta(n^E \cdot \log n)$ |
| 3. $f(n) \in \Omega(n^{E+\varepsilon})$ für ein $\varepsilon > 0$ und<br>$b \cdot f(n/c) \leq d \cdot f(n)$ für ein $d < 1$<br>und $n$ hinreichend groß | $T(n) \in \Theta(f(n))$             |

# Das Mastertheorem

$$T(n) = b \cdot T\left(\frac{n}{c}\right) + f(n) \quad \text{mit } b \geq 1 \text{ und } c > 1.$$

- ▶ Anzahl der Blätter im Rekursionsbaum:  $n^E$  mit  $E = \log b / \log c$ .

## Mastertheorem

Wenn

Dann

- |   |                                     |
|---|-------------------------------------|
| 1. $f(n) \in O(n^{E-\varepsilon})$ für ein $\varepsilon > 0$  | $T(n) \in \Theta(n^E)$              |
| 2. $f(n) \in \Theta(n^E)$   | $T(n) \in \Theta(n^E \cdot \log n)$ |
| 3. $f(n) \in \Omega(n^{E+\varepsilon})$ für ein $\varepsilon > 0$ und<br>$b \cdot f(n/c) \leq d \cdot f(n)$ für ein $d < 1$<br>und $n$ hinreichend groß | $T(n) \in \Theta(f(n))$             |

- ▶ Bemerke, dass das Mastertheorem nicht alle Fälle abdeckt.

# Das Mastertheorem verstehen

In jedem der 3 Fälle wird die Funktion  $f(n)$  mit  $n^E = n^{\log_c b}$  verglichen.

# Das Mastertheorem verstehen

In jedem der 3 Fälle wird die Funktion  $f(n)$  mit  $n^E = n^{\log_c b}$  verglichen.

## Mastertheorem: Intuition

Wenn

Dann

1.  $f(n)$  polynomial kleiner ist als  $n^E$

$T(n) \in \Theta(n^E)$

# Das Mastertheorem verstehen

In jedem der 3 Fälle wird die Funktion  $f(n)$  mit  $n^E = n^{\log_c b}$  verglichen.

## Mastertheorem: Intuition

Wenn

Dann

- |   |                                     |
|---|-------------------------------------|
| 1. $f(n)$ polynomial kleiner ist als $n^E$  | $T(n) \in \Theta(n^E)$              |
| 2. $f(n)$ und $n^E$ die gleiche Größe haben | $T(n) \in \Theta(n^E \cdot \log n)$ |



# Das Mastertheorem verstehen

In jedem der 3 Fälle wird die Funktion  $f(n)$  mit  $n^E = n^{\log_c b}$  verglichen.

## Mastertheorem: Intuition

Wenn

Dann

- |  |                                     |
|--|-------------------------------------|
| 1. $f(n)$ polynomial kleiner ist als $n^E$   | $T(n) \in \Theta(n^E)$              |
| 2. $f(n)$ und $n^E$ die gleiche Größe haben  | $T(n) \in \Theta(n^E \cdot \log n)$ |
| 3. $f(n)$ ist polynomial größer als $n^E$ und erfüllt $b \cdot f(n/c) \leq d \cdot f(n)$ | $T(n) \in \Theta(f(n))$             |

# Das Mastertheorem verstehen

In jedem der 3 Fälle wird die Funktion  $f(n)$  mit  $n^E = n^{\log_c b}$  verglichen.

## Mastertheorem: Intuition

Wenn

Dann

- |  |                                     |
|--|-------------------------------------|
| 1. $f(n)$ polynomial kleiner ist als $n^E$   | $T(n) \in \Theta(n^E)$              |
| 2. $f(n)$ und $n^E$ die gleiche Größe haben  | $T(n) \in \Theta(n^E \cdot \log n)$ |
| 3. $f(n)$ ist polynomial größer als $n^E$ und erfüllt $b \cdot f(n/c) \leq d \cdot f(n)$ | $T(n) \in \Theta(f(n))$             |

## Nicht abgedeckte Fälle:

1.  $f(n)$  ist kleiner als  $n^E$ , jedoch nicht polynomiell kleiner.

# Das Mastertheorem verstehen

In jedem der 3 Fälle wird die Funktion  $f(n)$  mit  $n^E = n^{\log_c b}$  verglichen.

## Mastertheorem: Intuition

Wenn

Dann

- |  |                                     |
|--|-------------------------------------|
| 1. $f(n)$ polynomial kleiner ist als $n^E$   | $T(n) \in \Theta(n^E)$              |
| 2. $f(n)$ und $n^E$ die gleiche Größe haben  | $T(n) \in \Theta(n^E \cdot \log n)$ |
| 3. $f(n)$ ist polynomial größer als $n^E$ und erfüllt $b \cdot f(n/c) \leq d \cdot f(n)$ | $T(n) \in \Theta(f(n))$             |

## Nicht abgedeckte Fälle:

- $f(n)$  ist kleiner als  $n^E$ , jedoch nicht polynomiell kleiner.
- $f(n)$  ist größer als  $n^E$ , jedoch nicht polynomiell größer.

# Das Mastertheorem verstehen

In jedem der 3 Fälle wird die Funktion  $f(n)$  mit  $n^E = n^{\log_c b}$  verglichen.

## Mastertheorem: Intuition

Wenn

Dann

- |  |                                     |
|--|-------------------------------------|
| 1. $f(n)$ polynomial kleiner ist als $n^E$   | $T(n) \in \Theta(n^E)$              |
| 2. $f(n)$ und $n^E$ die gleiche Größe haben  | $T(n) \in \Theta(n^E \cdot \log n)$ |
| 3. $f(n)$ ist polynomial größer als $n^E$ und erfüllt $b \cdot f(n/c) \leq d \cdot f(n)$ | $T(n) \in \Theta(f(n))$             |

## Nicht abgedeckte Fälle:

- $f(n)$  ist kleiner als  $n^E$ , jedoch nicht polynomiell kleiner.
- $f(n)$  ist größer als  $n^E$ , jedoch nicht polynomiell größer.
- $f(n)$  ist polynomiell größer als  $n^E$ , erfüllt nicht  $b \cdot f(n/c) \leq d \cdot f(n)$ .

# Anwendung des Mastertheorems

## Beispiel

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

# Anwendung des Mastertheorems

## Beispiel

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

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n$ ;  $E = \log 4 / \log 2 = 2$ .

# Anwendung des Mastertheorems

## Beispiel

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

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Da  $f(n) = n \in O(n^{2-\epsilon})$ , gilt Fall 1:  $T(n) \in \Theta(n^2)$

# Anwendung des Mastertheorems

## Beispiel

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

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Da  $f(n) = n \in O(n^{2-\epsilon})$ , gilt Fall 1:  $T(n) \in \Theta(n^2)$

## Beispiel

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



# Anwendung des Mastertheorems

## Beispiel

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

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Da  $f(n) = n \in O(n^{2-\epsilon})$ , gilt Fall 1:  $T(n) \in \Theta(n^2)$

## Beispiel

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

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n^2$ ;  $E = \log 4 / \log 2 = 2$ .

# Anwendung des Mastertheorems

## Beispiel

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

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Da  $f(n) = n \in O(n^{2-\epsilon})$ , gilt Fall 1:  $T(n) \in \Theta(n^2)$

## Beispiel

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

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n^2$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Da  $f(n) = n^2 \notin O(n^{2-\epsilon})$ , gilt Fall 1 nicht.

# Anwendung des Mastertheorems

## Beispiel

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

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Da  $f(n) = n \in O(n^{2-\epsilon})$ , gilt Fall 1:  $T(n) \in \Theta(n^2)$

## Beispiel

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

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n^2$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Da  $f(n) = n^2 \notin O(n^{2-\epsilon})$ , gilt Fall 1 nicht.
- ▶ Aber weil  $f(n) = n^2 \in \Theta(n^2)$ , gilt Fall 2:  $T(n) \in \Theta(n^2 \cdot \log n)$

# Anwendung des Mastertheorems

## Beispiel

$$T(n) = 4 \cdot T(n/2) + n^3$$

# Anwendung des Mastertheorems

## Beispiel

$$T(n) = 4 \cdot T(n/2) + n^3$$

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n^3$ ;  $E = \log 4 / \log 2 = 2$ .

# Anwendung des Mastertheorems

## Beispiel

$$T(n) = 4 \cdot T(n/2) + n^3$$

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n^3$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Wegen  $E = 2$  gelten Fälle 1 und 2 offenbar **nicht**.

# Anwendung des Mastertheorems

## Beispiel

$$T(n) = 4 \cdot T(n/2) + n^3$$

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n^3$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Wegen  $E = 2$  gelten Fälle 1 und 2 offenbar **nicht**.
- ▶ Da  $f(n) = n^3 \in \Omega(n^{2+\varepsilon})$  für  $\varepsilon = 1$ , könnte Fall 3 gelten.

# Anwendung des Mastertheorems

## Beispiel

$$T(n) = 4 \cdot T(n/2) + n^3$$

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n^3$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Wegen  $E = 2$  gelten Fälle 1 und 2 offenbar **nicht**.
- ▶ Da  $f(n) = n^3 \in \Omega(n^{2+\varepsilon})$  für  $\varepsilon = 1$ , könnte Fall 3 gelten.
- ▶ Überprüfe: gilt  $f(n/2) \leq \frac{d}{4} \cdot f(n)$  für ein  $d < 1$  und hinreichend grosse  $n$ ?



# Anwendung des Mastertheorems

## Beispiel

$$T(n) = 4 \cdot T(n/2) + n^3$$

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n^3$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Wegen  $E = 2$  gelten Fälle 1 und 2 offenbar **nicht**.
- ▶ Da  $f(n) = n^3 \in \Omega(n^{2+\varepsilon})$  für  $\varepsilon = 1$ , könnte Fall 3 gelten.
- ▶ Überprüfe: gilt  $f(n/2) \leq \frac{d}{4} \cdot f(n)$  für ein  $d < 1$  und hinreichend grosse  $n$ ?
- ▶ Dies liefert  $\frac{1}{8}n^3 \leq \frac{d}{4} \cdot n^3$ , und dies gilt für alle  $\frac{1}{2} \leq d < 1$  (und  $n$ )

# Anwendung des Mastertheorems

## Beispiel

$$T(n) = 4 \cdot T(n/2) + n^3$$

- ▶ Somit:  $b = 4$ ,  $c = 2$  und  $f(n) = n^3$ ;  $E = \log 4 / \log 2 = 2$ .
- ▶ Wegen  $E = 2$  gelten Fälle 1 und 2 offenbar **nicht**.
- ▶ Da  $f(n) = n^3 \in \Omega(n^{2+\varepsilon})$  für  $\varepsilon = 1$ , könnte Fall 3 gelten.
- ▶ Überprüfe: gilt  $f(n/2) \leq \frac{d}{4} \cdot f(n)$  für ein  $d < 1$  und hinreichend grosse  $n$ ?
- ▶ Dies liefert  $\frac{1}{8}n^3 \leq \frac{d}{4} \cdot n^3$ , und dies gilt für alle  $\frac{1}{2} \leq d < 1$  (und  $n$ )
- ▶ Somit gilt Fall 3 tatsächlich und wir folgern:

$$T(n) \in \Theta(n^3)$$

# Das Mastertheorem ist nicht immer anwendbar

## Beispiel

$$T(n) = 4 \cdot T(n/2) + \frac{n^2}{\log n}$$

# Das Mastertheorem ist nicht immer anwendbar

## Beispiel

$$T(n) = 4 \cdot T(n/2) + \frac{n^2}{\log n}$$

- ▶ Also gilt:  $b = 4$ ,  $c = 2$  und  $f(n) = n^2 / \log n$ ;  $E = 2$ .

# Das Mastertheorem ist nicht immer anwendbar

## Beispiel

$$T(n) = 4 \cdot T(n/2) + \frac{n^2}{\log n}$$

- ▶ Also gilt:  $b = 4$ ,  $c = 2$  und  $f(n) = n^2 / \log n$ ;  $E = 2$ .

Fall 1 ist **nicht** anwendbar:

$$n^2 / \log n \notin O(n^{2-\varepsilon}), \text{ da } f(n)/n^2 = (\log n)^{-1} \notin O(n^{-\varepsilon}).$$

# Das Mastertheorem ist nicht immer anwendbar

## Beispiel

$$T(n) = 4 \cdot T(n/2) + \frac{n^2}{\log n}$$

► Also gilt:  $b = 4$ ,  $c = 2$  und  $f(n) = n^2 / \log n$ ;  $E = 2$ .

Fall 1 ist **nicht** anwendbar:

$$n^2 / \log n \notin O(n^{2-\varepsilon}), \text{ da } f(n)/n^2 = (\log n)^{-1} \notin O(n^{-\varepsilon}).$$

Fall 2 ist **nicht** anwendbar:  $n^2 / \log n \notin \Theta(n^2)$ .

# Das Mastertheorem ist nicht immer anwendbar

## Beispiel

$$T(n) = 4 \cdot T(n/2) + \frac{n^2}{\log n}$$

► Also gilt:  $b = 4$ ,  $c = 2$  und  $f(n) = n^2 / \log n$ ;  $E = 2$ .

Fall 1 ist **nicht** anwendbar:

$$n^2 / \log n \notin O(n^{2-\varepsilon}), \text{ da } f(n)/n^2 = (\log n)^{-1} \notin O(n^{-\varepsilon}).$$

Fall 2 ist **nicht** anwendbar:  $n^2 / \log n \notin \Theta(n^2)$ .

Fall 3 ist **nicht** anwendbar:

$$f(n) \notin \Omega(n^{2+\varepsilon}), \text{ da } f(n)/n^2 = (\log n)^{-1} \notin O(n^{+\varepsilon}).$$

# Das Mastertheorem ist nicht immer anwendbar

## Beispiel

$$T(n) = 4 \cdot T(n/2) + \frac{n^2}{\log n}$$

- ▶ Also gilt:  $b = 4$ ,  $c = 2$  und  $f(n) = n^2 / \log n$ ;  $E = 2$ .

Fall 1 ist **nicht** anwendbar:

$$n^2 / \log n \notin O(n^{2-\varepsilon}), \text{ da } f(n)/n^2 = (\log n)^{-1} \notin O(n^{-\varepsilon}).$$

Fall 2 ist **nicht** anwendbar:  $n^2 / \log n \notin \Theta(n^2)$ .

Fall 3 ist **nicht** anwendbar:

$$f(n) \notin \Omega(n^{2+\varepsilon}), \text{ da } f(n)/n^2 = (\log n)^{-1} \notin O(n^{+\varepsilon}).$$

⇒ Das Mastertheorem hilft hier überhaupt nicht weiter!

- ▶ Durch Substitution erhält man:  $T(n) \in \Theta(n^2 \cdot \log \log n)$



# Mastertheorem: Beweis