Karlsruher Institut für Technologie

Algorithmen 1

Übung 1
Asymptotik, Pseudocode & Master-Theorem

Organisatorisches

Karlsruher Institut für Technologie

- Tretet gern dem Discord Server bei `https://scale.iti.kit.edu/teaching/2022ss/algo1/start`

Organisatorisches

- Tretet gern dem Discord Server bei `https://scale.iti.kit.edu/teaching/2022ss/algo1/start`

Literatur zur Vorlesung

- Vorlesung richtet sich nicht strikt nach einem konkreten Buch

Organisatorisches

- Tretet gern dem Discord Server bei `https://scale.iti.kit.edu/teaching/2022ss/algo1/start`

Literatur zur Vorlesung

- Vorlesung richtet sich nicht strikt nach einem konkreten Buch
- Folien sollten zum Verständnis ausreichend detailliert sein

Organisatorisches

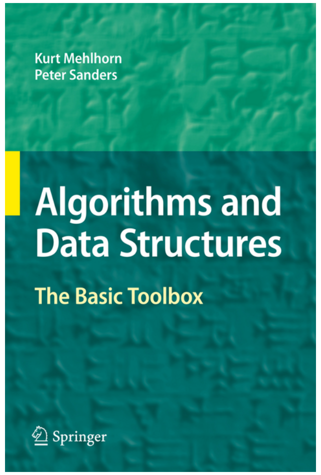
- Tretet gern dem Discord Server bei `https://scale.iti.kit.edu/teaching/2022ss/algo1/start`

Literatur zur Vorlesung

- Vorlesung richtet sich nicht strikt nach einem konkreten Buch
- Folien sollten zum Verständnis ausreichend detailliert sein

Für weitergehende Recherche

- Algorithms and Data Structures – The Basic Toolbox
  Kurt Mehlhorn, Peter Sanders



Organisatorisches

Karlsruher Institut für Technologie

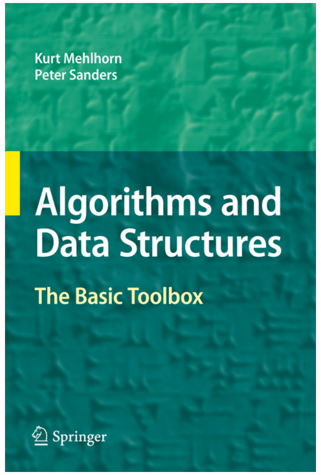
- Tretet gern dem Discord Server bei `https://scale.iti.kit.edu/teaching/2022ss/algo1/start`

Literatur zur Vorlesung

- Vorlesung richtet sich nicht strikt nach einem konkreten Buch
- Folien sollten zum Verständnis ausreichend detailliert sein

Für weitergehende Recherche

- Algorithms and Data Structures – The Basic Toolbox
  Kurt Mehlhorn, Peter Sanders
- Algorithmen und Datenstrukturen: Die Grundwerkzeuge
  Martin Dietzfelbinger, Kurt Mehlhorn, Peter Sanders



Organisatorisches

Karlsruher Institut für Technologie

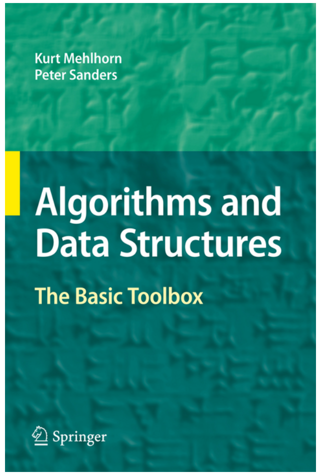
- Tretet gern dem Discord Server bei `https://scale.iti.kit.edu/teaching/2022ss/algo1/start`

Literatur zur Vorlesung

- Vorlesung richtet sich nicht strikt nach einem konkreten Buch
- Folien sollten zum Verständnis ausreichend detailliert sein

Für weitergehende Recherche

- Algorithms and Data Structures – The Basic Toolbox
  Kurt Mehlhorn, Peter Sanders
- Algorithmen und Datenstrukturen: Die Grundwerkzeuge
  Martin Dietzfelbinger, Kurt Mehlhorn, Peter Sanders



Konventionen

- Algorithmen sollen immer so effizient wie möglich sein

Organisatorisches

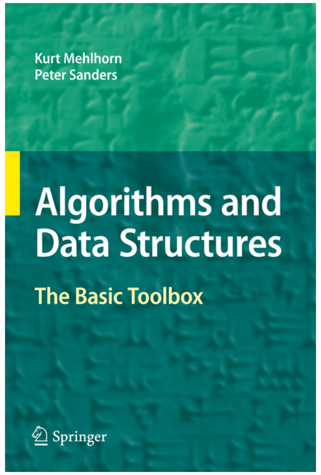
- Tretet gern dem Discord Server bei `https://scale.iti.kit.edu/teaching/2022ss/algo1/start`

Literatur zur Vorlesung

- Vorlesung richtet sich nicht strikt nach einem konkreten Buch
- Folien sollten zum Verständnis ausreichend detailliert sein

Für weitergehende Recherche

- Algorithms and Data Structures – The Basic Toolbox
  Kurt Mehlhorn, Peter Sanders
- Algorithmen und Datenstrukturen: Die Grundwerkzeuge
  Martin Dietzfelbinger, Kurt Mehlhorn, Peter Sanders



Konventionen

- Algorithmen sollen immer so effizient wie möglich sein
- Wenn nicht anders angegeben, ist $\log(\cdot)$ immer zur Basis 2

Asymptotik

- Werkzeug zur Klassifizierung der Laufzeit von Algorithmen

Asymptotik

- Werkzeug zur Klassifizierung der Laufzeit von Algorithmen
- Abstrahieren von Implementierungs- bzw. Hardwaredetails

Asymptotik

- Werkzeug zur Klassifizierung der Laufzeit von Algorithmen
- Abstrahieren von Implementierungs- bzw. Hardwaredetails
- Vereinfacht die Analyse

Asymptotik

- Werkzeug zur Klassifizierung der Laufzeit von Algorithmen
- Abstrahieren von Implementierungs- bzw. Hardwaredetails
- Vereinfacht die Analyse

Landau-Notation, O-Notation, big-O notation

Asymptotik

- Werkzeug zur Klassifizierung der Laufzeit von Algorithmen
- Abstrahieren von Implementierungs- bzw. Hardwaredetails
- Vereinfacht die Analyse

Landau-Notation, *O*-Notation, big-*O* notation

$f(n) \in O(g(n))$ $f(n)$ wächst max. so schnell wie $g(n)$ $\exists c \, \exists n_0 \, \forall n > n_0 \; f(n) \leq c \cdot g(n)$

Asymptotik

- Werkzeug zur Klassifizierung der Laufzeit von Algorithmen
- Abstrahieren von Implementierungs- bzw. Hardwaredetails
- Vereinfacht die Analyse

Landau-Notation, O-Notation, big-O notation

$f(n) \in O(g(n))$ $f(n)$ wächst max. so schnell wie $g(n)$ $\exists c \, \exists n_0 \, \forall n > n_0 \; f(n) \leq c \cdot g(n)$

- n bezeichnet die Größe der Eingabe

Asymptotik

- Werkzeug zur Klassifizierung der Laufzeit von Algorithmen
- Abstrahieren von Implementierungs- bzw. Hardwaredetails
- Vereinfacht die Analyse

Landau-Notation, *O*-Notation, big-*O* notation

$f(n) \in O(g(n))$ $f(n)$ wächst max. so schnell wie $g(n)$ $\exists c \exists n_0 \forall n > n_0\ f(n) \leq c \cdot g(n)$

- n bezeichnet die Größe der Eingabe
- Intuitiv: Wenn die Eingabe ausreichend groß ist (mindestens n_0), dann braucht ein Algorithmus mit Laufzeit $f(n)$ (ungefähr) höchstens so lang wie einer mit Laufzeit $g(n)$.

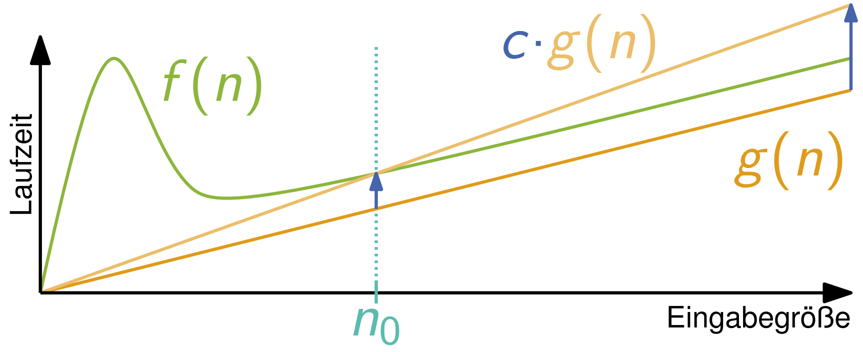
Asymptotik

- Werkzeug zur Klassifizierung der Laufzeit von Algorithmen
- Abstrahieren von Implementierungs- bzw. Hardwaredetails
- Vereinfacht die Analyse

Landau-Notation, *O*-Notation, big-*O* notation

$f(n) \in O(g(n))$ $f(n)$ wächst max. so schnell wie $g(n)$ $\exists c \exists n_0 \forall n > n_0 \; f(n) \leq c \cdot g(n)$

- n bezeichnet die Größe der Eingabe
- Intuitiv: Wenn die Eingabe ausreichend groß ist (mindestens n_0), dann braucht ein Algorithmus mit Laufzeit $f(n)$ (ungefähr) höchstens so lang wie einer mit Laufzeit $g(n)$.



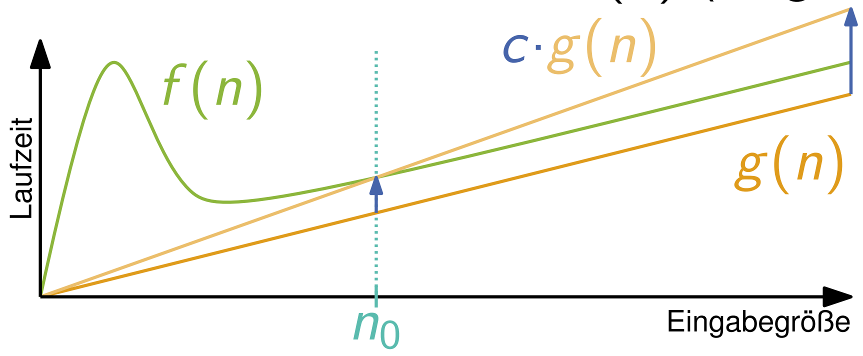
Asymptotik

- Werkzeug zur Klassifizierung der Laufzeit von Algorithmen
- Abstrahieren von Implementierungs- bzw. Hardwaredetails
- Vereinfacht die Analyse

Landau-Notation, *O*-Notation, big-*O* notation

$f(n) \in O(g(n))$ $f(n)$ wächst max. so schnell wie $g(n)$ $\exists c \, \exists n_0 \, \forall n > n_0 \; f(n) \leq c \cdot g(n)$

- n bezeichnet die Größe der Eingabe
- Intuitiv: Wenn die Eingabe ausreichend groß ist (mindestens n_0), dann braucht ein Algorithmus mit Laufzeit $f(n)$ (ungefähr) höchstens so lang wie einer mit Laufzeit $g(n)$.



$f(n) \in \omega(g(n))$ $f(n)$ wächst schneller als $g(n)$

$f(n) \in \Omega(g(n))$ $f(n)$ wächst min. so schnell wie $g(n)$

$f(n) \in \Theta(g(n))$ $f(n)$ und $g(n)$ wachsen gleich schnell

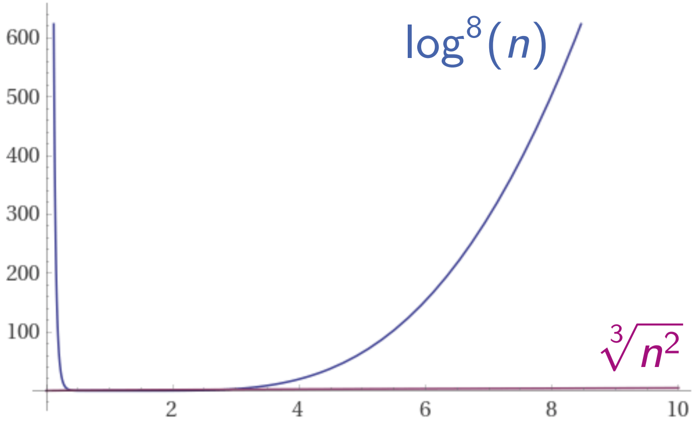
$f(n) \in o(g(n))$ $f(n)$ wächst langsamer als $g(n)$

Asymptotik – Tipps

- Plots helfen, um ein Gefühl zu bekommen, können aber auch trügerisch sein

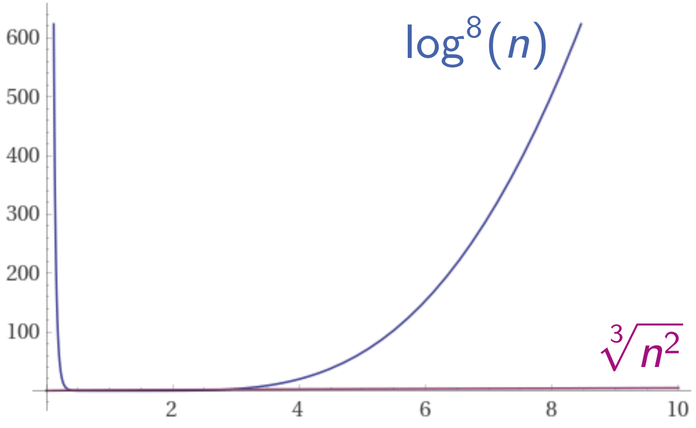
Asymptotik – Tipps

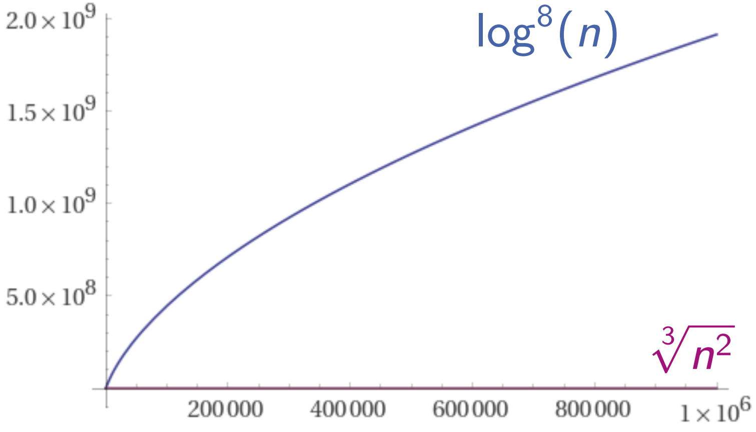
- Plots helfen, um ein Gefühl zu bekommen, können aber auch trügerisch sein



Asymptotik – Tipps

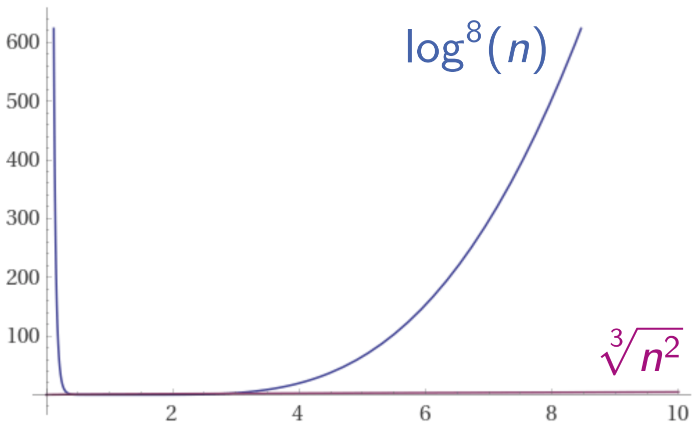
- Plots helfen, um ein Gefühl zu bekommen, können aber auch trügerisch sein

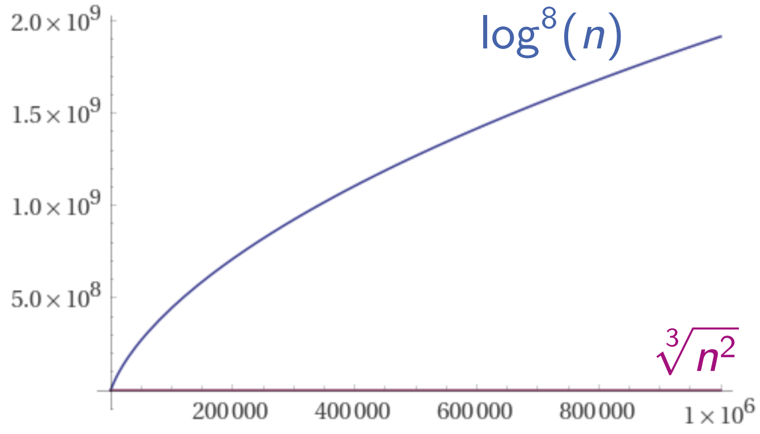


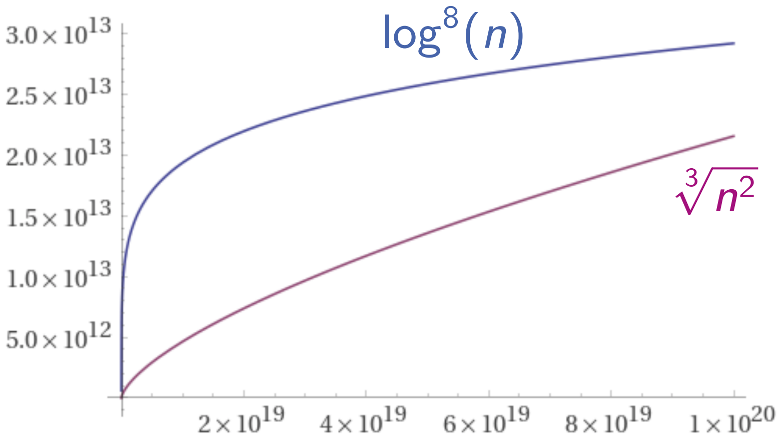


Asymptotik – Tipps

- Plots helfen, um ein Gefühl zu bekommen, können aber auch trügerisch sein

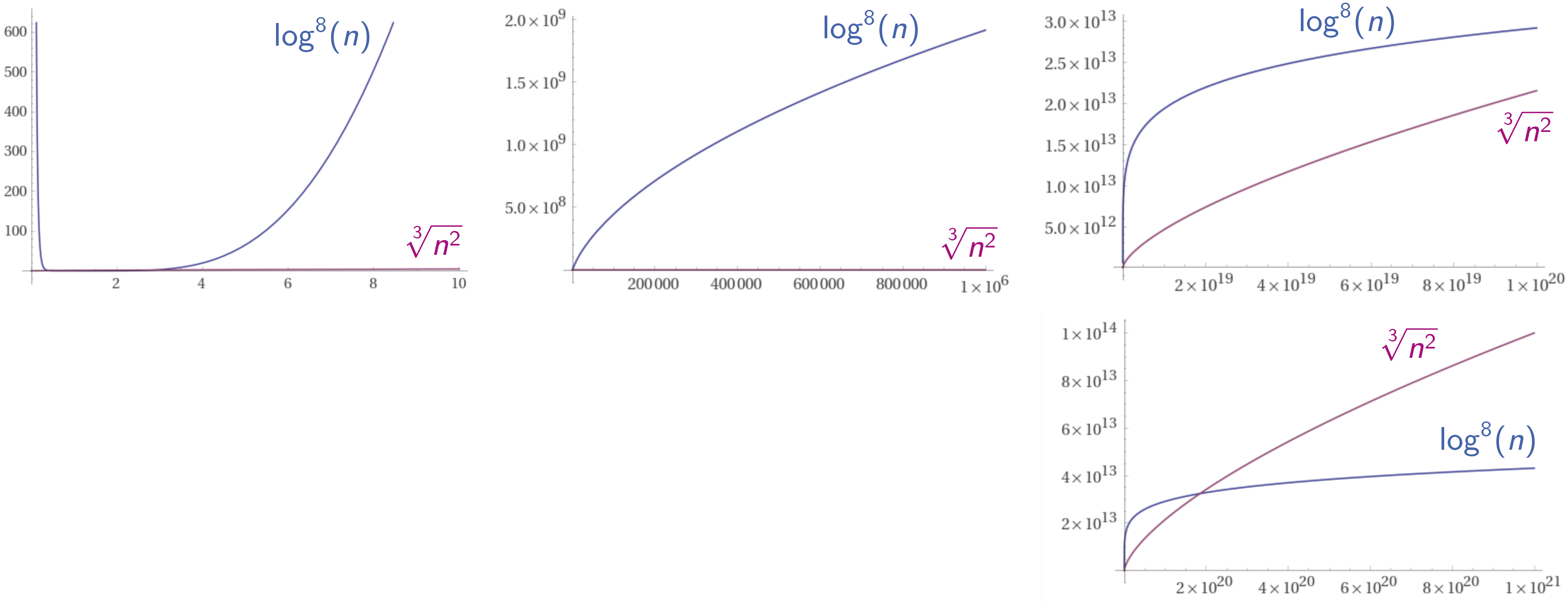






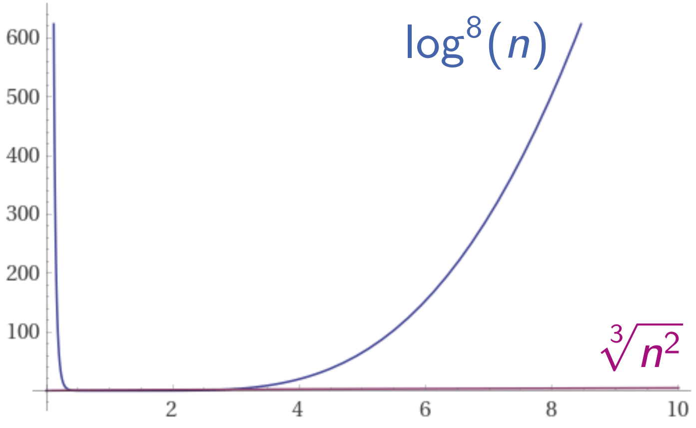
Asymptotik – Tipps

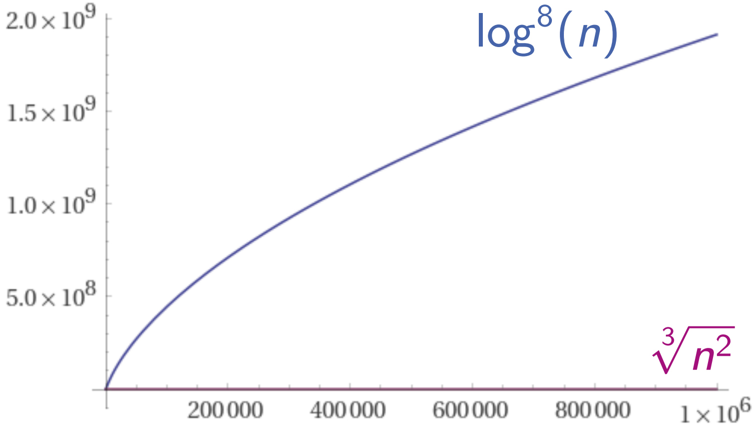
- Plots helfen, um ein Gefühl zu bekommen, können aber auch trügerisch sein

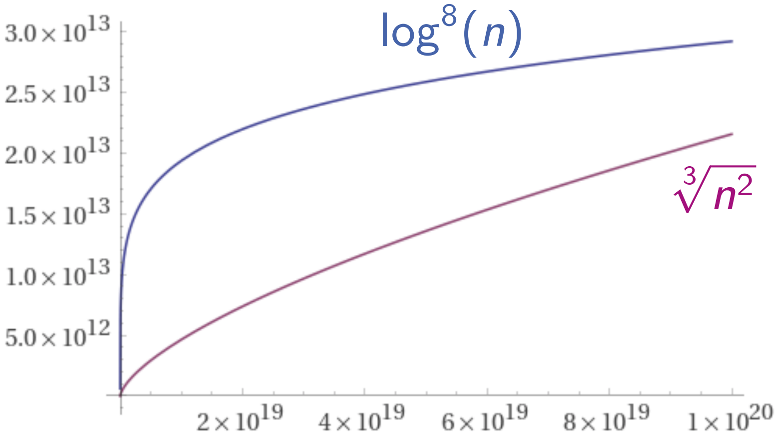


Asymptotik – Tipps

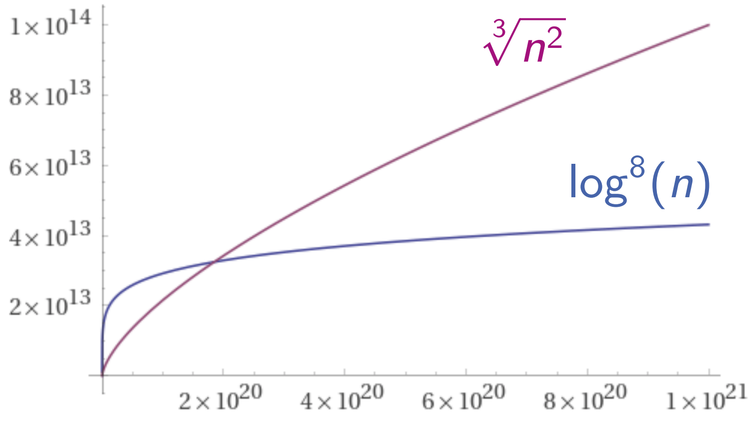
- Plots helfen, um ein Gefühl zu bekommen, können aber auch trügerisch sein







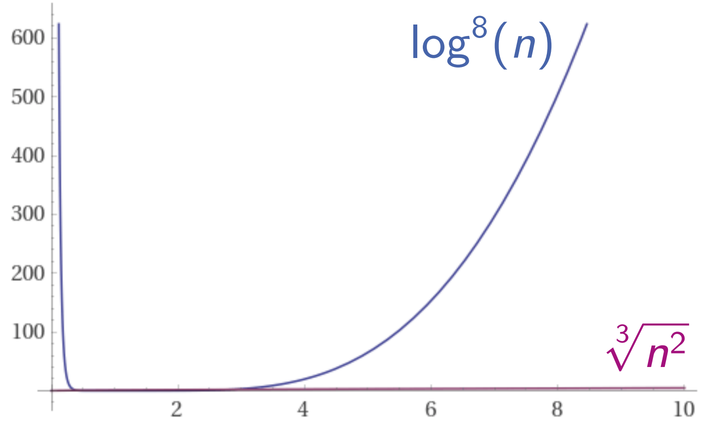
- Mentaler Shortcut: Alle Logarithmen wachsen langsamer als alle Polynome

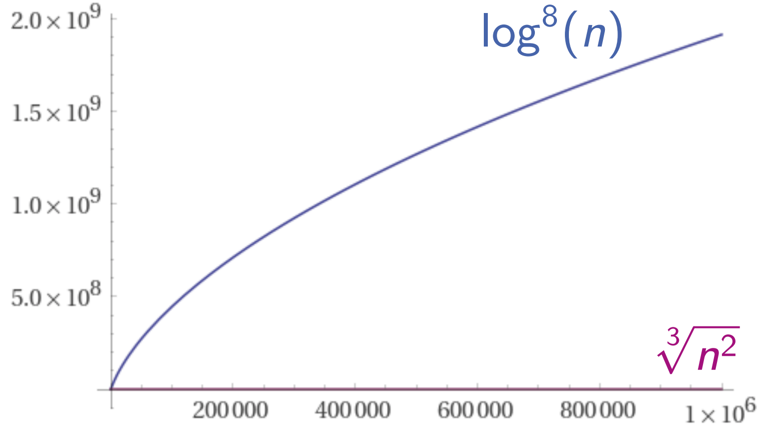


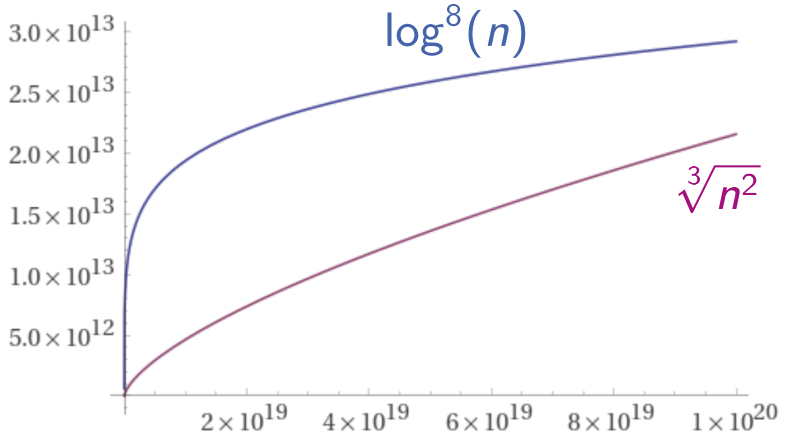
Asymptotik – Tipps

Karlsruher Institut für Technologie

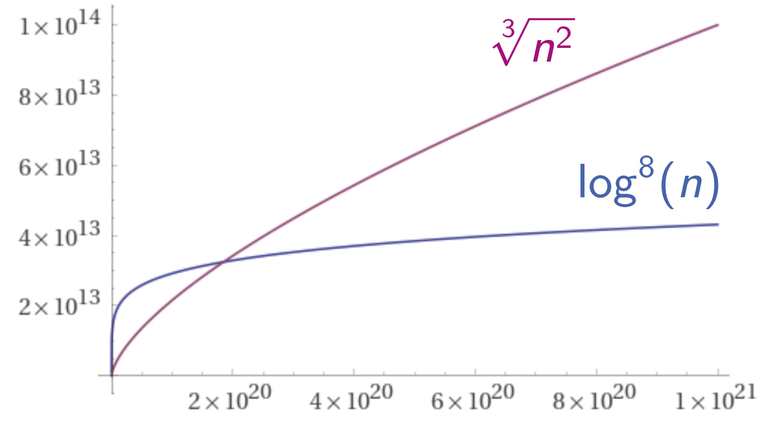
- Plots helfen, um ein Gefühl zu bekommen, können aber auch trügerisch sein





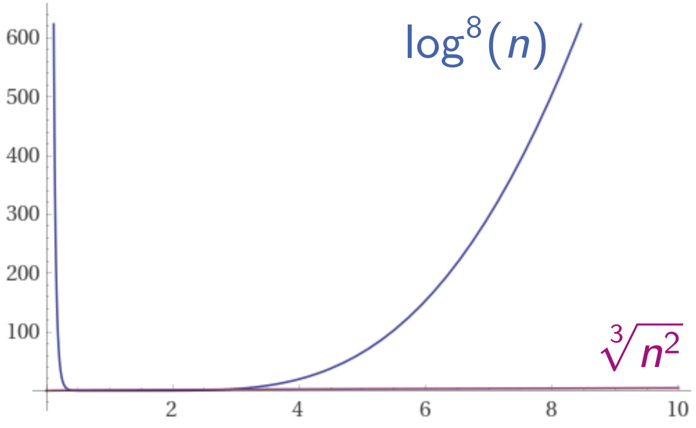


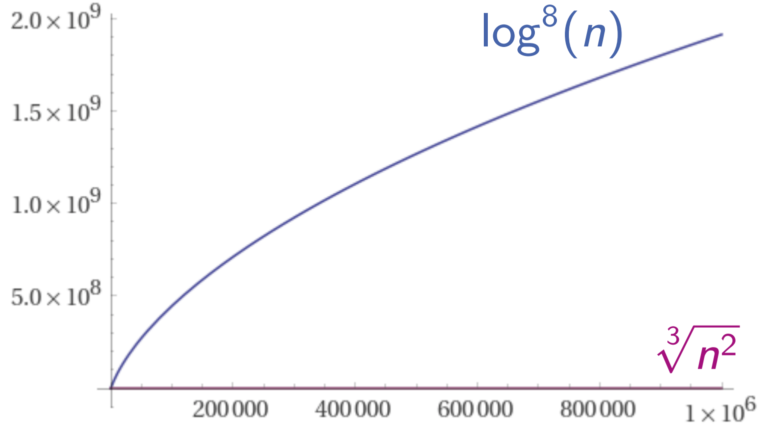
- Mentaler Shortcut: Alle Logarithmen wachsen langsamer als alle Polynome
- Der Logarithmus ist unheimlich klein

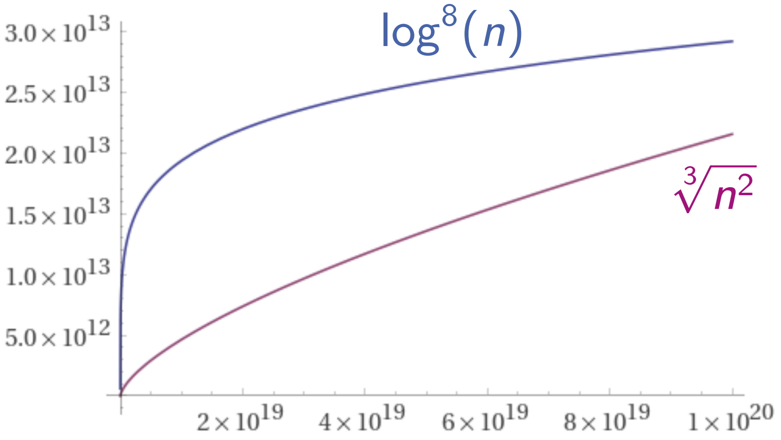


Asymptotik – Tipps

- Plots helfen, um ein Gefühl zu bekommen, können aber auch trügerisch sein

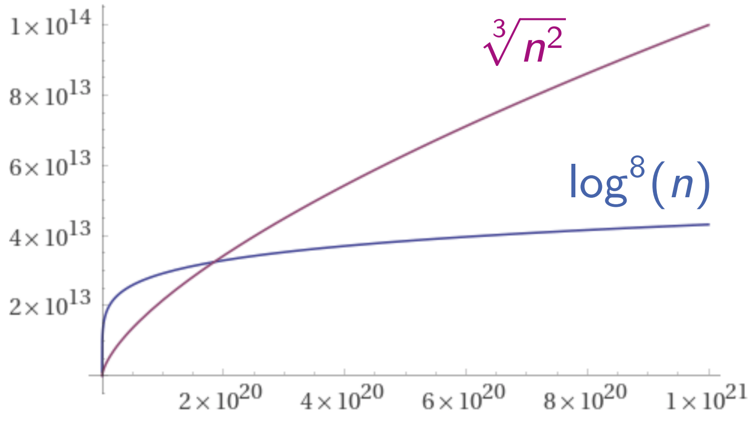






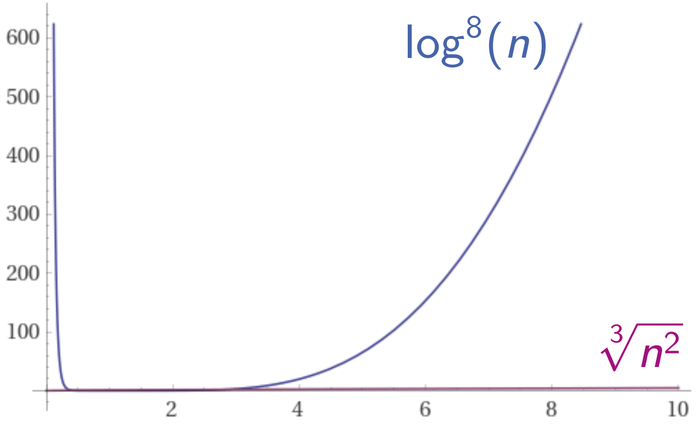
- Mentaler Shortcut: Alle Logarithmen wachsen langsamer als alle Polynome
- Der Logarithmus ist unheimlich klein

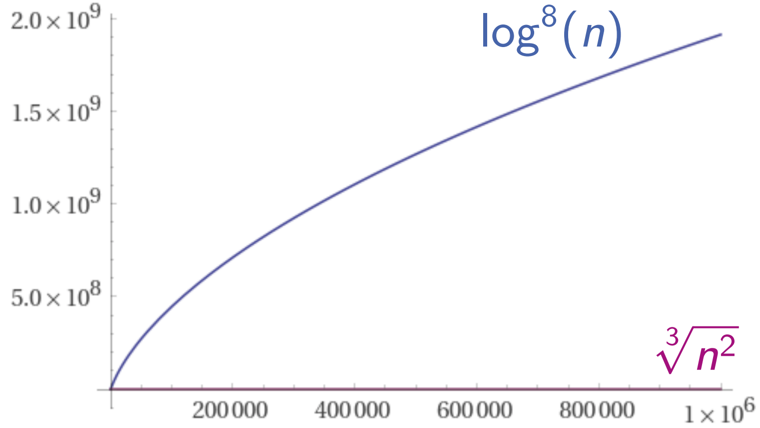
  ca. $6 \cdot 10^{79}$ Atome im Universum $\to \log(6 \cdot 10^{79}) \approx 265$

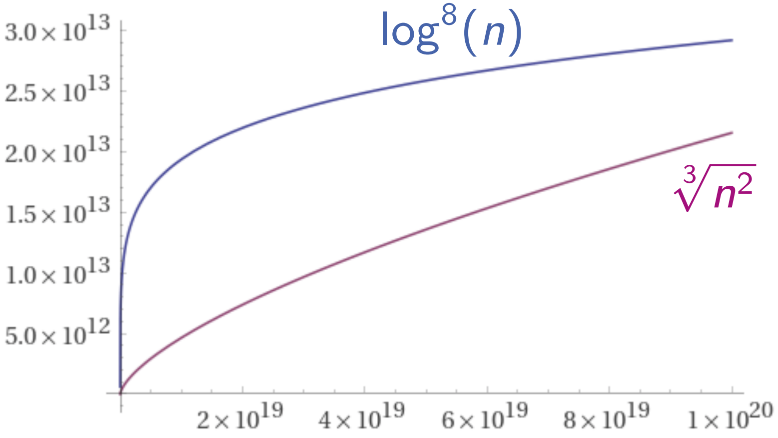


Asymptotik – Tipps

- Plots helfen, um ein Gefühl zu bekommen, können aber auch trügerisch sein



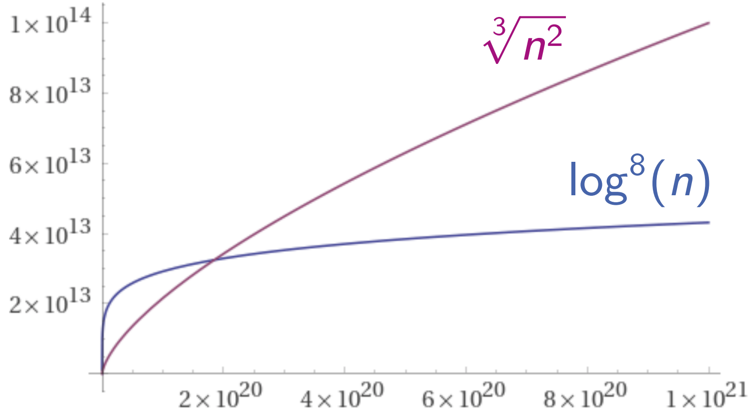




- Mentaler Shortcut: Alle Logarithmen wachsen langsamer als alle Polynome
- Der Logarithmus ist unheimlich klein

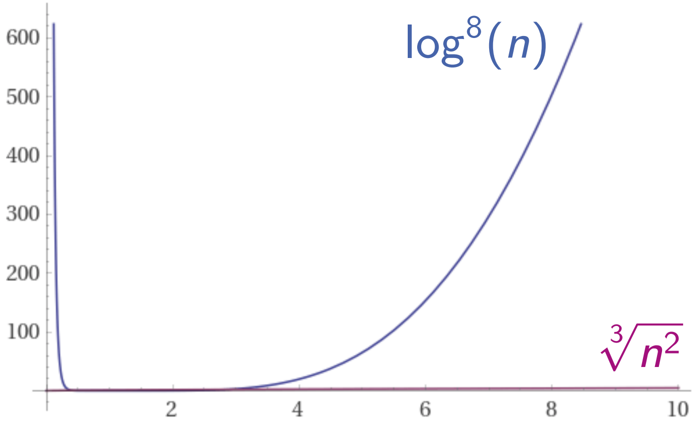
  ca. $6 \cdot 10^{79}$ Atome im Universum $\rightarrow \log(6 \cdot 10^{79}) \approx 265$
- Eine Laufzeit in $O(n \log(n))$ nennt man **quasi-linear**

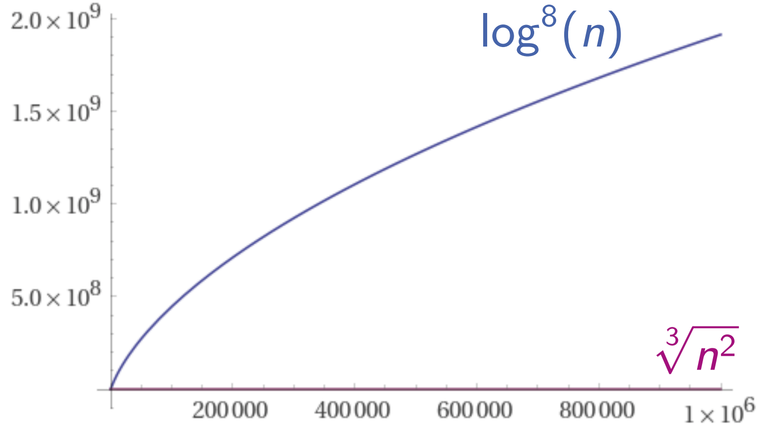
  manchmal auch $O(n \log^k(n))$

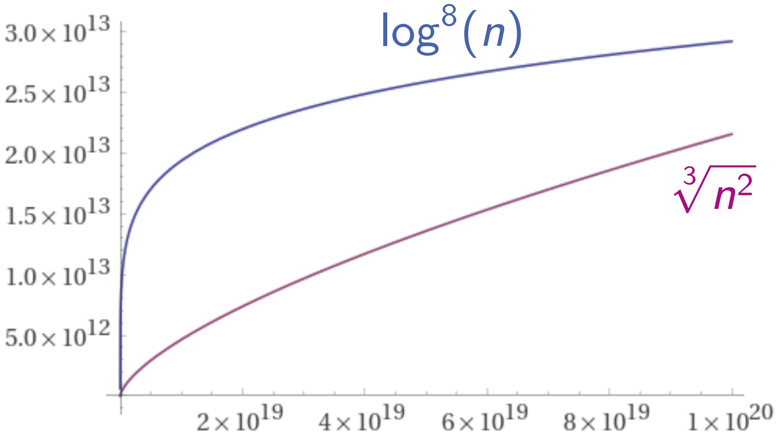


Asymptotik – Tipps

- Plots helfen, um ein Gefühl zu bekommen, können aber auch trügerisch sein



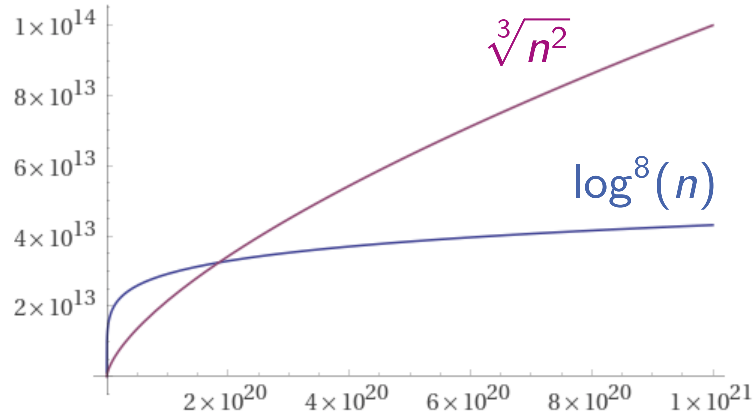




- Mentaler Shortcut: Alle Logarithmen wachsen langsamer als alle Polynome
- Der Logarithmus ist unheimlich klein

  ca. $6 \cdot 10^{79}$ Atome im Universum $\rightarrow \log(6 \cdot 10^{79}) \approx 265$
- Eine Laufzeit in $O(n \log(n))$ nennt man **quasi-linear**

  manchmal auch $O(n \log^k(n))$



- In der Praxis ist ab quadratisch nur bedingt brauchbar, exponentiell geht gar nicht

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c \, \exists n_0 \, \forall n > n_0 \; f(n) \leq c \cdot g(n)$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $\boldsymbol{10n \in O(n^2)}$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$10n \leq c \cdot n^2$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

$$\exists c \, \exists n_0 \, \forall n > n_0 \; f(n) \leq c \cdot g(n)$$

Zeige: $10n \in O(n^2)$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$ $\quad \exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$

$$10n \leq c \cdot n^2$$

$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$

$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$

$$n \leq n^2$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\;\, f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$

$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$

$$n \leq n^2$$

$$1 \leq n$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

$$\exists c \, \exists n_0 \, \forall n > n_0 \; f(n) \leq c \cdot g(n)$$

Zeige: $10n \in O(n^2)$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\;\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$

Asymptotik – Beispiele

$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$

Zeige: $10n \in O(n^2)$

$\exists c\, \exists n_0\, \forall n > n_0\;\; f(n) \leq c \cdot g(n)$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10 \;\; n_0 = 1$$

Asymptotik – Beispiele

Karlsruher Institut für Technologie

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\ f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\ n_0 = 1 \qquad c = 5\ n_0 = 2$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$ $\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\;\; n_0 = 1 \qquad c = 5\;\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n\log(n)\log\log(n)), \varepsilon > 0$ $\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$\exists c \, \exists n_0 \, \forall n > n_0 \; f(n) \leq c \cdot g(n)$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10 \; n_0 = 1 \qquad c = 5 \; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$\exists c \, \exists n_0 \, \forall n > n_0 \; c \cdot f(n) \geq g(n)$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c \, \exists n_0 \, \forall n > n_0 \; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10 \;\; n_0 = 1 \qquad c = 5 \;\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$$\exists c \, \exists n_0 \, \forall n > n_0 \; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\; n_0 = 1 \qquad c = 5\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n\log(n)\log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n\log(n)\log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log(n)\log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\;\; n_0 = 1 \qquad c = 5\;\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n\log(n)\log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n\log(n)\log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log(n)\log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\log(c) + \varepsilon\log(n) \geq 2\log\log(n)$$

Asymptotik – Beispiele

Karlsruher Institut für Technologie

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\;\; n_0 = 1 \qquad c = 5\;\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

$$\log(c) + \varepsilon \log(n) \geq 2 \log\log(n) \quad c = 1$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$ $\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$

$$\begin{aligned} 10n &\leq c \cdot n^2 \\ n &\leq \tfrac{c}{10} \cdot n^2 \quad c = 10 \\ n &\leq n^2 \\ 1 &\leq n \qquad n_0 = 1 \end{aligned}$$

$$\begin{aligned} \log(10) + \log(n) &\leq \log(c) + 2\log(n) \\ \log(10) - \log(c) &\leq \log(n) \\ \tfrac{10}{c} &\leq n \quad c = 10\; n_0 = 1 \qquad c = 5\; n_0 = 2 \end{aligned}$$

Zeige: $n^{1+\varepsilon} \in \Omega(n\log(n)\log\log(n)), \varepsilon > 0$ $\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$

$$c \cdot n^{1+\varepsilon} \geq n\log(n)\log\log(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$c \cdot n^{\varepsilon} \geq \log(n)\log\log(n)$$

$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

$$\log(c) + \varepsilon\log(n) \geq 2\log\log(n) \quad c = 1$$

$$\varepsilon\log(n) \geq 2\log\log(n)$$

Asymptotik – Beispiele

Karlsruher Institut für Technologie

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\;\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\;\; n_0 = 1 \qquad c = 5\;\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\;\; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\log(c) + \varepsilon \log(n) \geq 2 \log\log(n) \quad c = 1$$
$$\varepsilon \log(n) \geq 2 \log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)}$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\; n_0 = 1 \qquad c = 5\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

$$\log(c) + \varepsilon\log(n) \geq 2\log\log(n) \quad c = 1$$

$$\varepsilon\log(n) \geq 2\log\log(n)$$

$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\ f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\ n_0 = 1 \quad c = 5\ n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\ c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\log(c) + \varepsilon \log(n) \geq 2 \log\log(n) \quad c = 1$$
$$\varepsilon \log(n) \geq 2 \log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\ f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\ n_0 = 1 \quad c = 5\ n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\ c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\log(c) + \varepsilon \log(n) \geq 2 \log\log(n) \quad c = 1$$
$$\varepsilon \log(n) \geq 2 \log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$
$$\varepsilon \geq \frac{m}{2^{m-1}}$$

Asymptotik – Beispiele

Karlsruher Institut für Technologie

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\ f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\ n_0 = 1 \qquad c = 5\ n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\ c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n\log(n)\log\log(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$c \cdot n^{\varepsilon} \geq \log(n)\log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$
$$\log(c) + \varepsilon\log(n) \geq 2\log\log(n) \quad c = 1$$
$$\varepsilon\log(n) \geq 2\log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\varepsilon \geq \frac{m}{2^{m-1}}$$
$$\varepsilon \geq \frac{2^{m/2}}{2^{m-1}}$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\ n_0 = 1 \quad c = 5\ n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n\log(n)\log\log(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$c \cdot n^{\varepsilon} \geq \log(n)\log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$
$$\log(c) + \varepsilon\log(n) \geq 2\log\log(n) \quad c = 1$$
$$\varepsilon\log(n) \geq 2\log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$
$$\varepsilon \geq \frac{m}{2^{m-1}}$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\varepsilon \geq \frac{2^{m/2}}{2^{m-1}}$$
$$\varepsilon \geq 2^{-m/2+1}$$

Institut für Theoretische Informatik, Skalierbare Algorithmen

Asymptotik – Beispiele

Karlsruher Institut für Technologie

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\; n_0 = 1 \qquad c = 5\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n\log(n)\log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n\log(n)\log\log(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$c \cdot n^{\varepsilon} \geq \log(n)\log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$
$$\log(c) + \varepsilon\log(n) \geq 2\log\log(n) \quad c = 1$$
$$\varepsilon\log(n) \geq 2\log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad n = 2^{2^m}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\varepsilon \geq \frac{m}{2^{m-1}}$$
$$\varepsilon \geq \frac{2^{m/2}}{2^{m-1}}$$
$$\varepsilon \geq 2^{-m/2+1}$$
$$\log(\varepsilon) \geq -m/2 + 1$$

Asymptotik – Beispiele

$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$

Zeige: $10n \in O(n^2)$

$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\; n_0 = 1 \qquad c = 5\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\log(c) + \varepsilon \log(n) \geq 2 \log\log(n) \quad c = 1$$
$$\varepsilon \log(n) \geq 2 \log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$
$$\varepsilon \geq \frac{m}{2^{m-1}}$$
$$\varepsilon \geq \frac{2^{m/2}}{2^{m-1}}$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\varepsilon \geq 2^{-m/2+1}$$
$$\log(\varepsilon) \geq -m/2 + 1$$
$$2 - 2\log(\varepsilon) \leq m$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\;\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\;\; n_0 = 1 \qquad c = 5\;\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\;\; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$
$$\log(c) + \varepsilon \log(n) \geq 2 \log\log(n) \quad c = 1$$
$$\varepsilon \log(n) \geq 2 \log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\varepsilon \geq \frac{m}{2^{m-1}}$$
$$\varepsilon \geq \frac{2^{m/2}}{2^{m-1}}$$
$$\varepsilon \geq 2^{-m/2+1}$$
$$\log(\varepsilon) \geq -m/2 + 1$$
$$2 - 2\log(\varepsilon) \leq m \quad \boxed{\log\log(n) = m}$$

Asymptotik – Beispiele

Karlsruher Institut für Technologie

$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$

Zeige: $10n \in O(n^2)$

$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\; n_0 = 1 \quad c = 5\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$
$$\log(c) + \varepsilon \log(n) \geq 2 \log\log(n) \quad c = 1$$
$$\varepsilon \log(n) \geq 2 \log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\varepsilon \geq \frac{m}{2^{m-1}}$$
$$\varepsilon \geq \frac{2^{m/2}}{2^{m-1}}$$
$$\varepsilon \geq 2^{-m/2+1}$$
$$\log(\varepsilon) \geq -m/2 + 1$$
$$2 - 2\log(\varepsilon) \leq m \quad \boxed{\log\log(n) = m}$$

$$2 - 2\log(\varepsilon) \leq \log\log(n)$$

Asymptotik – Beispiele

Karlsruher Institut für Technologie

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\;\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \qquad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\;\; n_0 = 1 \qquad c = 5\;\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n\log(n)\log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\;\; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n\log(n)\log\log(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$c \cdot n^{\varepsilon} \geq \log(n)\log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$
$$\log(c) + \varepsilon\log(n) \geq 2\log\log(n) \quad c = 1$$
$$\varepsilon\log(n) \geq 2\log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

$$2 - 2\log(\varepsilon) \leq m \quad \boxed{\log\log(n) = m}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\varepsilon \geq \frac{m}{2^{m-1}}$$
$$\varepsilon \geq \frac{2^{m/2}}{2^{m-1}}$$
$$\varepsilon \geq 2^{-m/2+1}$$
$$\log(\varepsilon) \geq -m/2 + 1$$

$$2 - 2\log(\varepsilon) \leq \log\log(n)$$
$$2^{2-2\log(\varepsilon)} \leq \log(n)$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\;\; n_0 = 1 \qquad c = 5\;\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\log(c) + \varepsilon \log(n) \geq 2 \log\log(n) \quad c = 1$$
$$\varepsilon \log(n) \geq 2 \log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$
$$\varepsilon \geq \frac{m}{2^{m-1}}$$
$$\varepsilon \geq \frac{2^{m/2}}{2^{m-1}}$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\varepsilon \geq 2^{-m/2+1}$$
$$\log(\varepsilon) \geq -m/2 + 1$$
$$2 - 2\log(\varepsilon) \leq m \quad \boxed{\log\log(n) = m}$$

$$2 - 2\log(\varepsilon) \leq \log\log(n)$$
$$2^{2-2\log(\varepsilon)} \leq \log(n)$$
$$2^{2^{2-2\log(\varepsilon)}} \leq n$$

Asymptotik – Beispiele

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $10n \in O(n^2)$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

$$10n \leq c \cdot n^2$$
$$n \leq \frac{c}{10} \cdot n^2 \quad c = 10$$
$$n \leq n^2$$
$$1 \leq n \quad n_0 = 1$$

$$\log(10) + \log(n) \leq \log(c) + 2\log(n)$$
$$\log(10) - \log(c) \leq \log(n)$$
$$\frac{10}{c} \leq n \quad c = 10\;\; n_0 = 1 \qquad c = 5\;\; n_0 = 2$$

Zeige: $n^{1+\varepsilon} \in \Omega(n \log(n) \log\log(n)), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; c \cdot f(n) \geq g(n)$$

$$c \cdot n^{1+\varepsilon} \geq n \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log(n) \log\log(n)$$
$$c \cdot n^{\varepsilon} \geq \log^2(n)$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\log(c) + \varepsilon \log(n) \geq 2 \log\log(n) \quad c = 1$$
$$\varepsilon \log(n) \geq 2 \log\log(n)$$
$$\frac{\varepsilon}{2} \geq \frac{\log\log(n)}{\log(n)} \quad \boxed{n = 2^{2^m}}$$

$$\frac{\varepsilon}{2} \geq \frac{m}{2^m}$$
$$\varepsilon \geq \frac{m}{2^{m-1}}$$
$$\varepsilon \geq \frac{2^{m/2}}{2^{m-1}}$$

Alles was diese Ungleichung erfüllt, erfüllt auch diese.

$$\varepsilon \geq 2^{-m/2+1}$$
$$\log(\varepsilon) \geq -m/2 + 1$$
$$2 - 2\log(\varepsilon) \leq m \quad \boxed{\log\log(n) = m}$$

$$2 - 2\log(\varepsilon) \leq \log\log(n)$$
$$2^{2-2\log(\varepsilon)} \leq \log(n)$$
$$2^{2^{2-2\log(\varepsilon)}} \leq n$$
$$n_0 = 2^{2^{2-2\log(\varepsilon)}}$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\ f(n) \leq c \cdot g(n)$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

Asymptotik – Beispiele (2)

Karlsruher Institut für Technologie

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\;\; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon)$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\;\; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\;\; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \quad c = 1$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\ f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\ f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \quad c = 1$$

$$\log(n) \leq \frac{\log(1+\varepsilon)}{c'} n$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$ $\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \quad c = 1$$

$$\log(n) \leq \frac{\log(1+\varepsilon)}{c'} n \qquad n = 2^m$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\ f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\ f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \qquad c = 1$$

$$\log(n) \leq \frac{\log(1+\varepsilon)}{c'} n \qquad \boxed{n = 2^m}$$

$$\frac{m}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \qquad c = 1$$

$$\log(n) \leq \frac{\log(1+\varepsilon)}{c'} n \qquad \boxed{n = 2^m}$$

$$\frac{m}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$\frac{2^{m/2}}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subsetneq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c \, \exists n_0 \, \forall n > n_0 \; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c' \, \exists n_0' \, \forall n > n_0' \; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \qquad c = 1$$

$$\log(n) \leq \frac{\log(1+\varepsilon)}{c'} n \qquad \boxed{n = 2^m}$$

$$\frac{m}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$\frac{2^{m/2}}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$2^{-m/2} \leq \frac{\log(1+\varepsilon)}{c'}$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \qquad c = 1$$

$$\log(n) \leq \frac{\log(1+\varepsilon)}{c'} n \qquad \boxed{n = 2^m}$$

$$\frac{m}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$\frac{2^{m/2}}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$2^{-m/2} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$m \geq 2 \log\left(\frac{c'}{\log(1+\varepsilon)}\right)$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $\mathbf{2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0}$

$$\exists c \, \exists n_0 \, \forall n > n_0 \; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c' \, \exists n_0' \, \forall n > n_0' \; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \qquad c = 1$$

$$\log(n) \leq \frac{\log(1+\varepsilon)}{c'} n \qquad \boxed{n = 2^m}$$

$$\frac{m}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$\frac{2^{m/2}}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$2^{-m/2} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$m \geq 2 \log\left(\frac{c'}{\log(1+\varepsilon)}\right) \qquad \boxed{m = \log(n)}$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $\mathbf{2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0}$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \qquad c = 1$$

$$\log(n) \leq \frac{\log(1+\varepsilon)}{c'} n \qquad \boxed{n = 2^m}$$

$$\frac{m}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$\frac{2^{m/2}}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$2^{-m/2} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$m \geq 2\log\left(\frac{c'}{\log(1+\varepsilon)}\right) \qquad \boxed{m = \log(n)}$$

$$\log(n) \geq \log\left(\left(\frac{c'}{\log(1+\varepsilon)}\right)^2\right)$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \qquad c = 1$$

$$\log(n) \leq \frac{\log(1+\varepsilon)}{c'} n \qquad \boxed{n = 2^m}$$

$$\frac{m}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$\frac{2^{m/2}}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$2^{-m/2} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$m \geq 2 \log\left(\frac{c'}{\log(1+\varepsilon)}\right) \qquad \boxed{m = \log(n)}$$

$$\log(n) \geq \log\left(\left(\frac{c'}{\log(1+\varepsilon)}\right)^2\right)$$

$$n \geq \left(\frac{c'}{\log(1+\varepsilon)}\right)^2$$

Asymptotik – Beispiele (2)

$$O(\log\log(n)) \subseteq O(\log(n)) \subseteq O(n^{1/x}), x \geq 1 \subseteq O(n) \subseteq O(n^x), x \geq 1, \subseteq O(x^n), x > 1$$

Zeige: $\mathbf{2^{O(\log(n))} \subseteq O((1+\varepsilon)^n), \varepsilon > 0}$

$$\exists c\, \exists n_0\, \forall n > n_0\; f(n) \leq c \cdot g(n)$$

- Zu zeigen: Für jedes $f(n) \in 2^{O(\log(n))}$ gilt: $f(n) \in O((1+\varepsilon)^n)$

$$\exists c'\, \exists n_0'\, \forall n > n_0'\; f(n) \leq 2^{c' \log(n)} = n^{c'}$$

$$n^{c'} \leq c \cdot (1+\varepsilon)^n$$

$$c' \log(n) \leq \log(c) + n \log(1+\varepsilon) \qquad c = 1$$

$$\log(n) \leq \frac{\log(1+\varepsilon)}{c'} n \qquad \boxed{n = 2^m}$$

$$\frac{m}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$\frac{2^{m/2}}{2^m} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$2^{-m/2} \leq \frac{\log(1+\varepsilon)}{c'}$$

$$m \geq 2 \log\left(\frac{c'}{\log(1+\varepsilon)}\right) \qquad \boxed{m = \log(n)}$$

$$\log(n) \geq \log\left(\left(\frac{c'}{\log(1+\varepsilon)}\right)^2\right)$$

$$n \geq \left(\frac{c'}{\log(1+\varepsilon)}\right)^2 =: n_0$$

Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty$$

Asymptotik – Beispiele (3)

Zeige: $\boldsymbol{\frac{n}{\log(n)} \in \omega(\sqrt{n})}$

$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}}$$

Asymptotik – Beispiele (3)

Zeige: $\boldsymbol{\frac{n}{\log(n)} \in \omega(\sqrt{n})}$

$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}}$$

Asymptotik – Beispiele (3)

Karlsruher Institut für Technologie

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)}$$

Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)}$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)

Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}}$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)

Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4}$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)

Asymptotik – Beispiele (3)

Zeige: $\boldsymbol{\frac{n}{\log(n)} \in \omega(\sqrt{n})}$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)

Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)

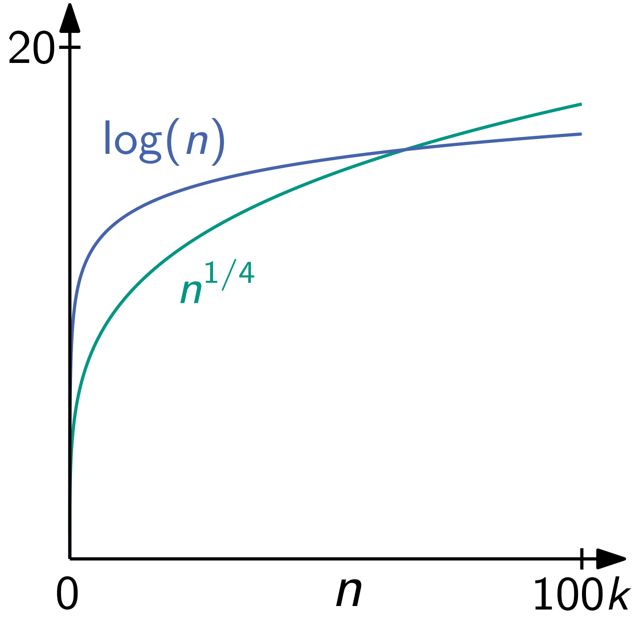
Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)



Asymptotik – Beispiele (3)

Karlsruher Institut für Technologie

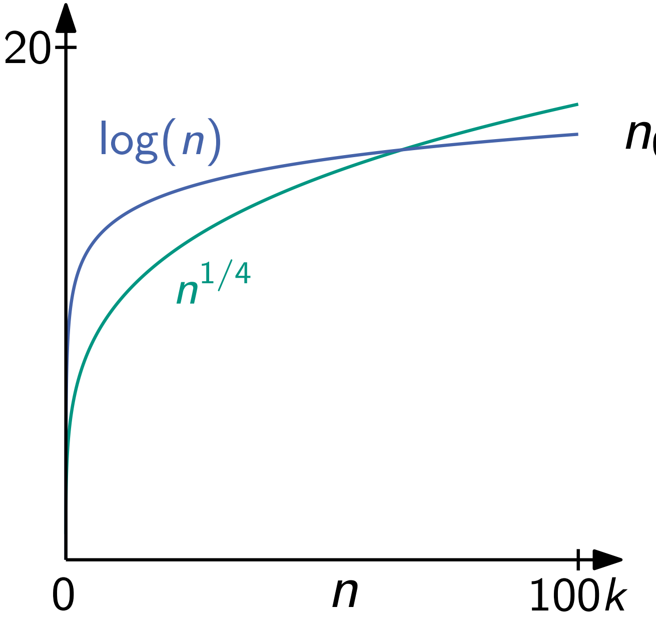
Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)

$(n \geq n_0)$



$n_0 = 2^{16}$

Asymptotik – Beispiele (3)

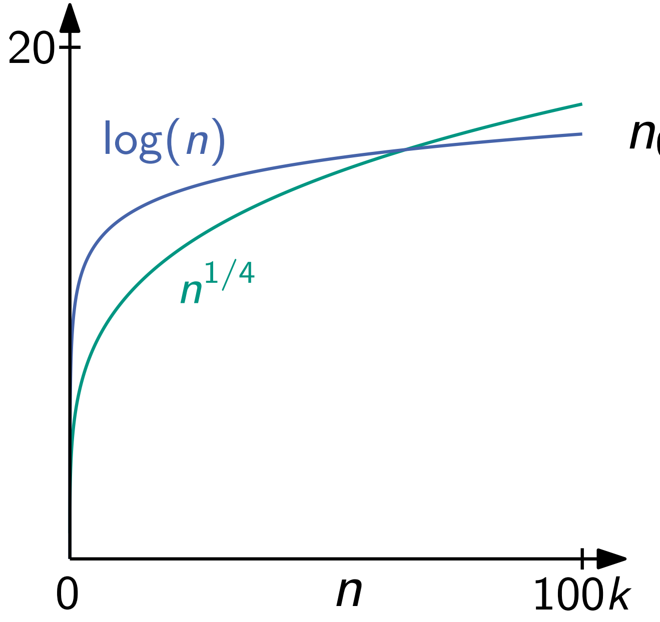
Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)

$(n \geq n_0)$



$n_0 = 2^{16}$ $\log(n_0) = 16$

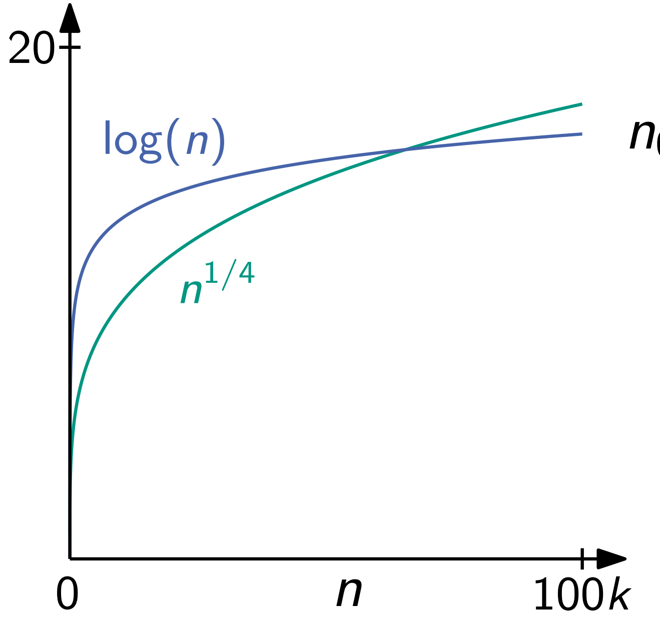
Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)



$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$ $(n \geq n_0)$

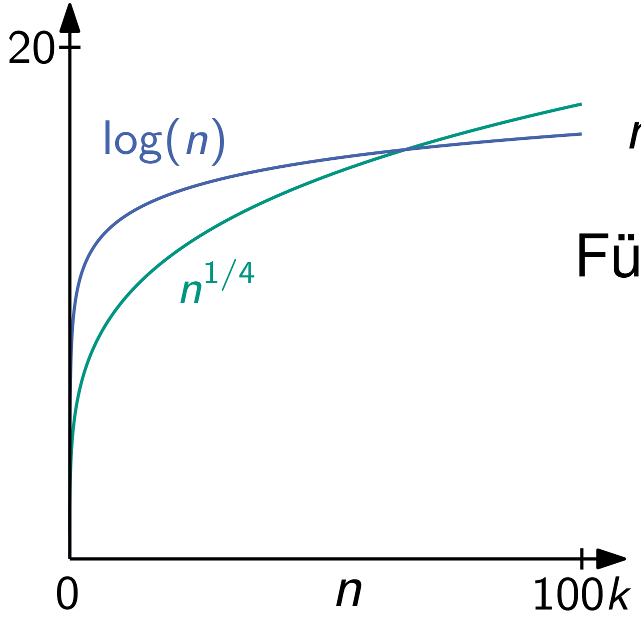
Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)



$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$ $(n \geq n_0)$

Für $n_0 = 2^{16}$ gilt $\log(n_0) = n_0^{1/4}$.

Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

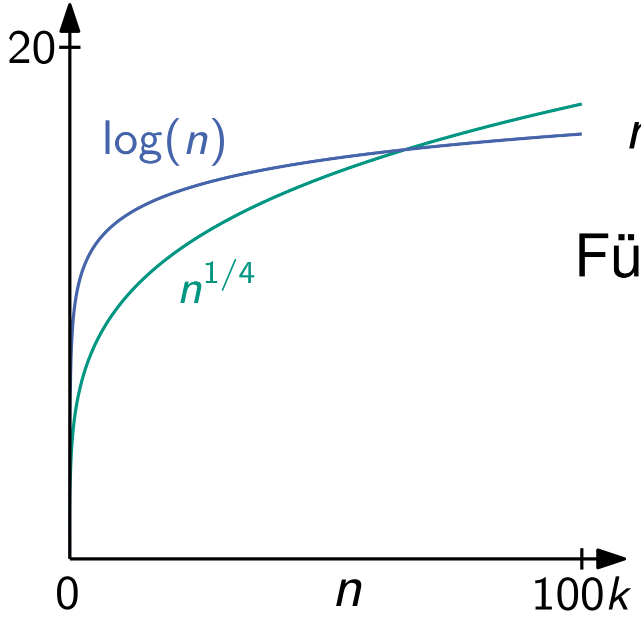
$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)

$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$ $(n \geq n_0)$

Für $n_0 = 2^{16}$ gilt $\log(n_0) = n_0^{1/4}$.



Wie können wir zeigen, dass eine Funktion schneller wächst als eine andere?

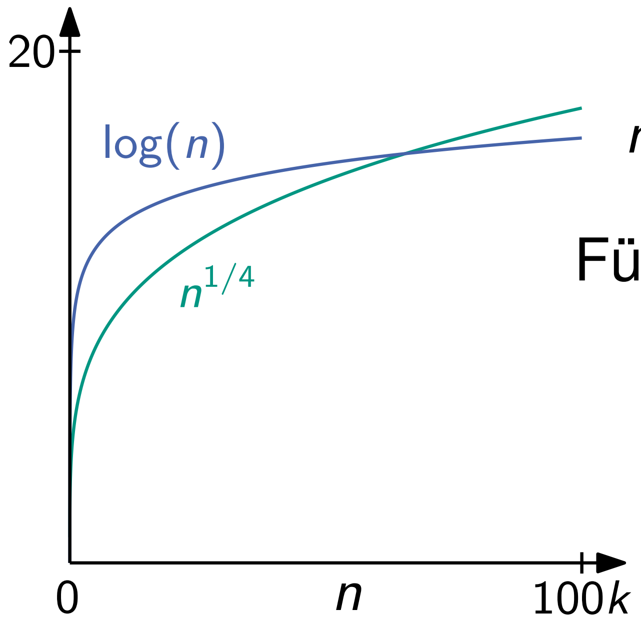
Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)



$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$ $(n \geq n_0)$

Für $n_0 = 2^{16}$ gilt $\log(n_0) = n_0^{1/4}$.

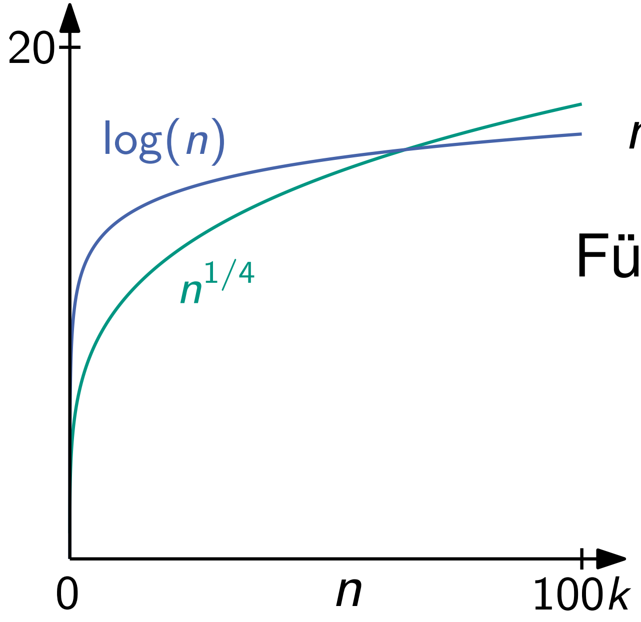
Wie können wir zeigen, dass eine Funktion schneller wächst als eine andere? → Anstieg

Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$



$\log(n) \leq n^{1/4}$ (für ausreichend großes n)

$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$ $(n \geq n_0)$

Für $n_0 = 2^{16}$ gilt $\log(n_0) = n_0^{1/4}$.

Wie können wir zeigen, dass eine Funktion schneller wächst als eine andere? → Anstieg → Ableitung

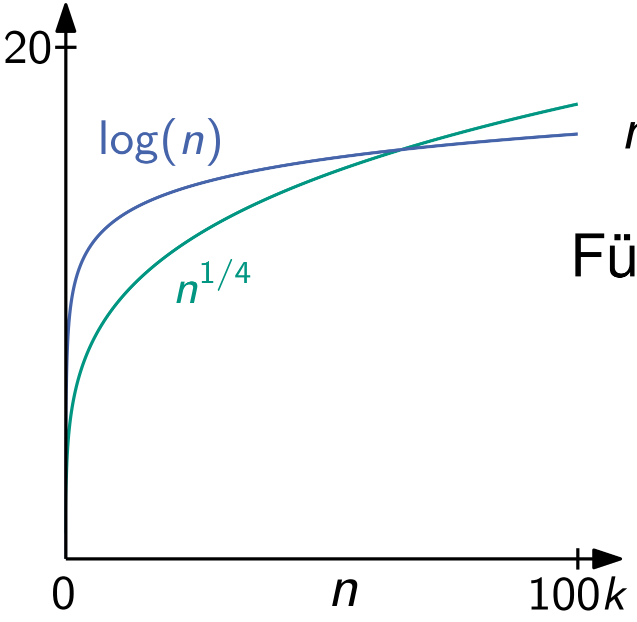
Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)



$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$ $(n \geq n_0)$

Für $n_0 = 2^{16}$ gilt $\log(n_0) = n_0^{1/4}$.

Wie können wir zeigen, dass eine Funktion schneller wächst als eine andere? → Anstieg → Ableitung

$(\log(n))\frac{d}{dn} = n^{-1}$

Asymptotik – Beispiele (3)

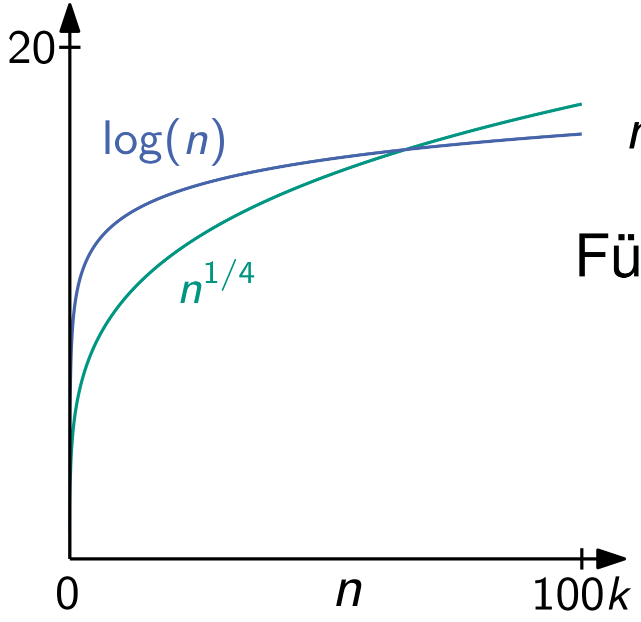
Karlsruher Institut für Technologie

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)



$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$ $(n \geq n_0)$

Für $n_0 = 2^{16}$ gilt $\log(n_0) = n_0^{1/4}$.

Wie können wir zeigen, dass eine Funktion schneller wächst als eine andere? → Anstieg → Ableitung

$(\log(n))\frac{d}{dn} = n^{-1}$ $(n^{1/4})\frac{d}{dn} = \frac{1}{4}n^{-3/4}$

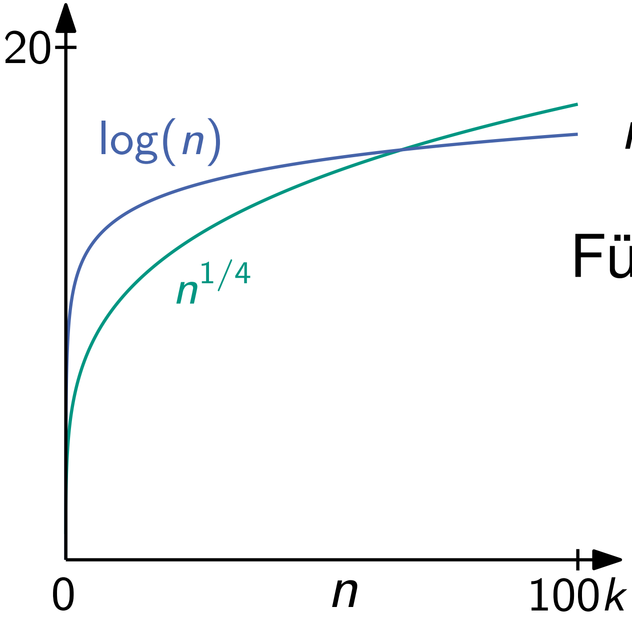
Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)



$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$ $(n \geq n_0)$

Für $n_0 = 2^{16}$ gilt $\log(n_0) = n_0^{1/4}$.

Wie können wir zeigen, dass eine Funktion schneller wächst als eine andere? → Anstieg → Ableitung

$(\log(n))\frac{d}{dn} = n^{-1}$ $(n^{1/4})\frac{d}{dn} = \frac{1}{4}n^{-3/4}$

$$\ell = \lim_{n\to\infty} \frac{\frac{1}{4}n^{-3/4}}{n^{-1}}$$

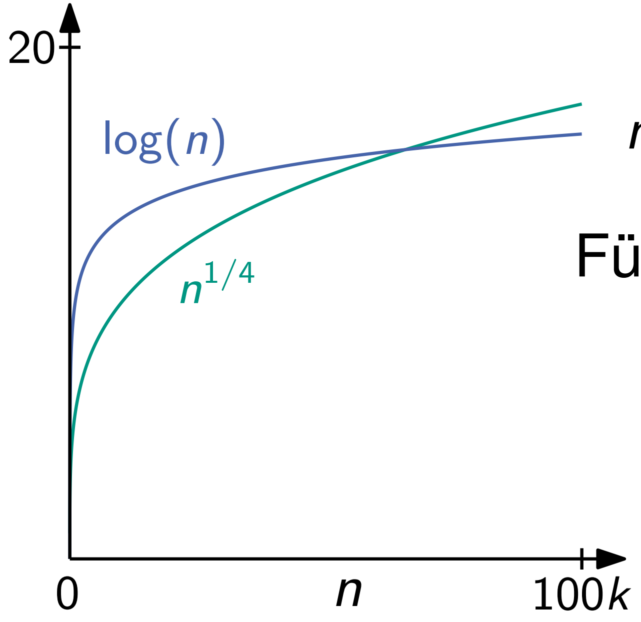
Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)



$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$ $(n \geq n_0)$

Für $n_0 = 2^{16}$ gilt $\log(n_0) = n_0^{1/4}$.

Wie können wir zeigen, dass eine Funktion schneller wächst als eine andere? → Anstieg → Ableitung

$(\log(n))\frac{d}{dn} = n^{-1}$ $(n^{1/4})\frac{d}{dn} = \frac{1}{4}n^{-3/4}$

$$\ell = \lim_{n\to\infty} \frac{\frac{1}{4}n^{-3/4}}{n^{-1}} = \lim_{n\to\infty} \frac{1}{4}n^{1-3/4}$$

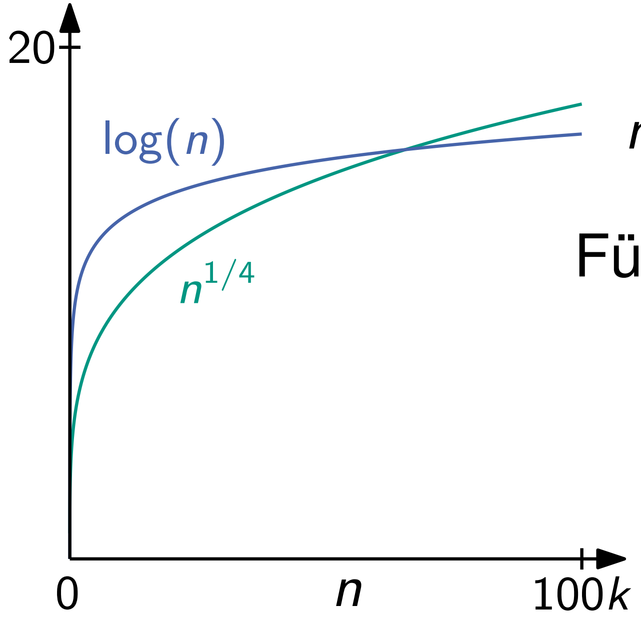
Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)



$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$ $(n \geq n_0)$

Für $n_0 = 2^{16}$ gilt $\log(n_0) = n_0^{1/4}$.

Wie können wir zeigen, dass eine Funktion schneller wächst als eine andere? → Anstieg → Ableitung

$(\log(n))\frac{d}{dn} = n^{-1}$ $(n^{1/4})\frac{d}{dn} = \frac{1}{4}n^{-3/4}$

$$\ell = \lim_{n\to\infty} \frac{\frac{1}{4}n^{-3/4}}{n^{-1}} = \lim_{n\to\infty} \frac{1}{4}n^{1-3/4} = \lim_{n\to\infty} \frac{1}{4}n^{1/4}$$

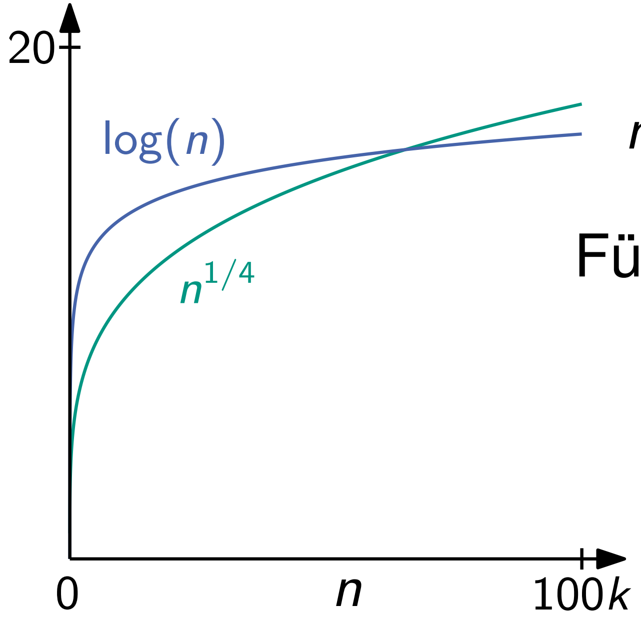
Asymptotik – Beispiele (3)

Zeige: $\frac{n}{\log(n)} \in \omega(\sqrt{n})$

$$f(n) \in \omega(g(n)) \Rightarrow \ell = \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$\ell = \lim_{n\to\infty} \frac{\frac{n}{\log(n)}}{\sqrt{n}} = \lim_{n\to\infty} \frac{n}{\log(n)\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \geq \lim_{n\to\infty} \frac{\sqrt{n}}{n^{1/4}} = \lim_{n\to\infty} n^{1/4} = \infty$$

$\log(n) \leq n^{1/4}$ (für ausreichend großes n)



$(n \geq n_0)$

$n_0 = 2^{16}$ $\log(n_0) = 16$ $n_0^{1/4} = (2^{16})^{1/4} = 2^4 = 16$

Für $n_0 = 2^{16}$ gilt $\log(n_0) = n_0^{1/4}$.

Wie können wir zeigen, dass eine Funktion schneller wächst als eine andere? → Anstieg → Ableitung

$(\log(n))\frac{d}{dn} = n^{-1}$ $(n^{1/4})\frac{d}{dn} = \frac{1}{4}n^{-3/4}$

$$\ell = \lim_{n\to\infty} \frac{\frac{1}{4}n^{-3/4}}{n^{-1}} = \lim_{n\to\infty} \frac{1}{4}n^{1-3/4} = \lim_{n\to\infty} \frac{1}{4}n^{1/4}$$

Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

https://scale.iti.kit.edu/teaching/2022ss/algo1/start

Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

`https://scale.iti.kit.edu/teaching/2022ss/algo1/start`

- für Menschen gut lesbar

Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

https://scale.iti.kit.edu/teaching/2022ss/algo1/start

- für Menschen gut lesbar
- richtige Abstraktionsebene

Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

  Algorithmusidee — Wir iterieren alle Ziffern und ...

  Pseudocode — **for** i in $\{0, 1, \ldots, n-1\}$ **do**

  https://scale.iti.kit.edu/teaching/2022ss/algo1/start

  Code — `for (int i = 0; i < n; ++i) { ... }`

- für Menschen gut lesbar
- richtige Abstraktionsebene

wrongAlgo(*input*)
  return correct output

Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...	
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**	`https://scale.iti.kit.edu/teaching/2022ss/algo1/start`
Code	`for (int i = 0; i < n; ++i) { ... }`	

- für Menschen gut lesbar
- richtige Abstraktionsebene

~~**wrongAlgo**(*input*)~~
~~**return** correct output~~

Pseudocode

Karlsruher Institut für Technologie

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

https://scale.iti.kit.edu/teaching/2022ss/algo1/start

- für Menschen gut lesbar
- richtige Abstraktionsebene

~~**wrongAlgo**(*input*)~~
~~**return** correct output~~

- Beispiel: Was macht der von diesem Pseudocode beschriebene Algorithmus?

Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

https://scale.iti.kit.edu/teaching/2022ss/algo1/start

- für Menschen gut lesbar
- richtige Abstraktionsebene

~~**wrongAlgo**(*input*)~~
~~**return** correct output~~

- Beispiel: Was macht der von diesem Pseudocode beschriebene Algorithmus?

secret(x)

  if $x \leq 10$ **return** 1
  return $1 +$ **secret**$(x/10)$

Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

https://scale.iti.kit.edu/teaching/2022ss/algo1/start

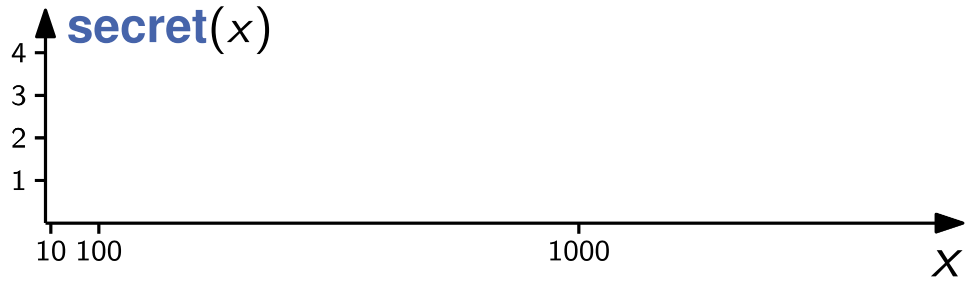
- für Menschen gut lesbar
- richtige Abstraktionsebene

~~**wrongAlgo**(*input*)~~
~~**return** correct output~~

- Beispiel: Was macht der von diesem Pseudocode beschriebene Algorithmus?

secret(x)
- **if** $x \leq 10$ **return** 1
- **return** $1 + \textbf{secret}(x/10)$

Beispiel Eingaben:



Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

`https://scale.iti.kit.edu/teaching/2022ss/algo1/start`

- für Menschen gut lesbar
- richtige Abstraktionsebene

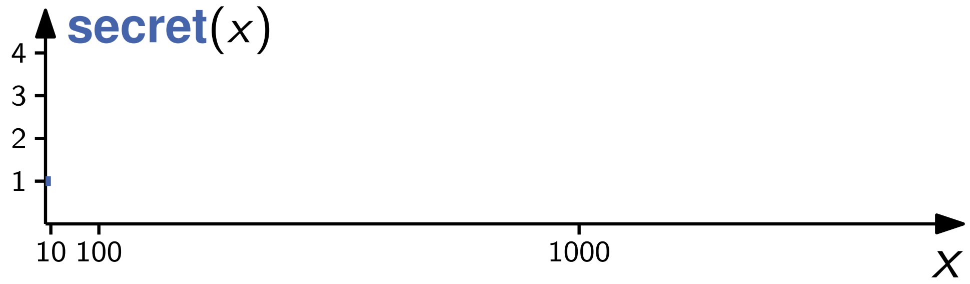
~~**wrongAlgo**(*input*)~~
~~**return** correct output~~

- Beispiel: Was macht der von diesem Pseudocode beschriebene Algorithmus?

secret(x)
  if $x \leq 10$ **return** 1
  return $1 +$ **secret**$(x/10)$

Beispiel Eingaben:

$x \in [1, 10] \rightarrow 1$



Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

https://scale.iti.kit.edu/teaching/2022ss/algo1/start

- für Menschen gut lesbar
- richtige Abstraktionsebene

~~**wrongAlgo**(*input*)~~
~~**return** correct output~~

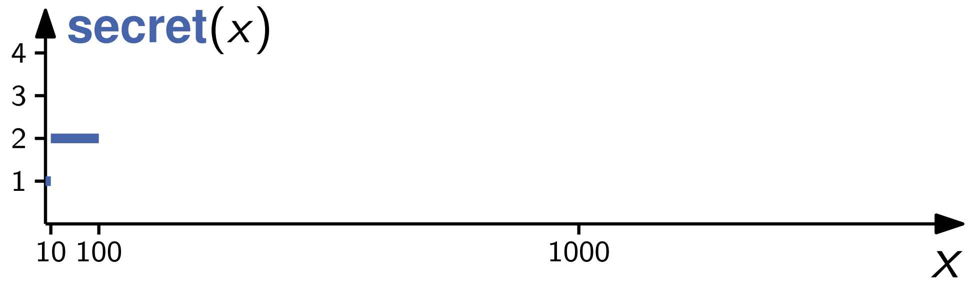
- Beispiel: Was macht der von diesem Pseudocode beschriebene Algorithmus?

secret(x)
  if $x \leq 10$ **return** 1
  return $1 +$ **secret**$(x/10)$

Beispiel Eingaben:

$x \in [1, 10] \rightarrow 1$

$x \in [11, 100] \rightarrow 2$



Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

https://scale.iti.kit.edu/teaching/2022ss/algo1/start

- für Menschen gut lesbar
- richtige Abstraktionsebene

~~**wrongAlgo**(*input*)~~
~~**return** correct output~~

- Beispiel: Was macht der von diesem Pseudocode beschriebene Algorithmus?

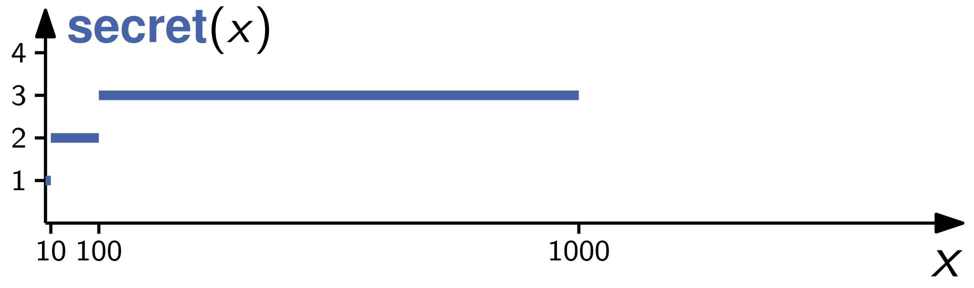
secret(x)
  if $x \leq 10$ **return** 1
  return $1 +$ **secret**$(x/10)$

Beispiel Eingaben:

$x \in [1, 10] \rightarrow 1$

$x \in [11, 100] \rightarrow 2$

$x \in [101, 1000] \rightarrow 3$



Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

https://scale.iti.kit.edu/teaching/2022ss/algo1/start

- für Menschen gut lesbar
- richtige Abstraktionsebene

~~**wrongAlgo**(*input*)~~
~~**return** correct output~~

- Beispiel: Was macht der von diesem Pseudocode beschriebene Algorithmus?

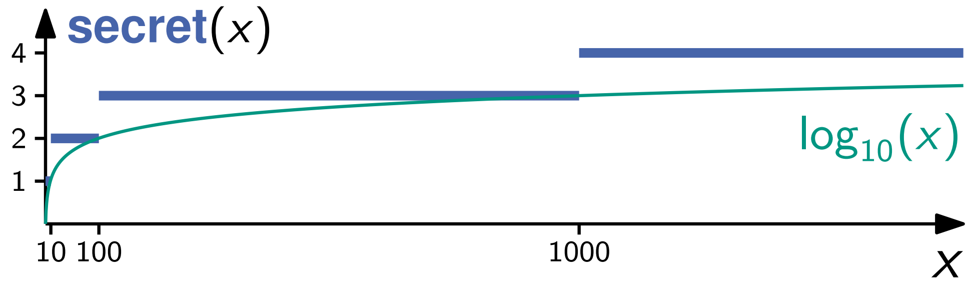
secret(x)
  if $x \leq 10$ **return** 1
  return $1 +$ **secret**$(x/10)$

Beispiel Eingaben:

$x \in [1, 10] \rightarrow 1$

$x \in [11, 100] \rightarrow 2$

$x \in [101, 1000] \rightarrow 3$



Pseudocode

- Ein Zwischenschritt auf dem Weg von der Algorithmusidee zur Implementierung

Algorithmusidee	Wir iterieren alle Ziffern und ...
Pseudocode	**for** i in $\{0, 1, \ldots, n-1\}$ **do**
Code	`for (int i = 0; i < n; ++i) { ... }`

https://scale.iti.kit.edu/teaching/2022ss/algo1/start

- für Menschen gut lesbar
- richtige Abstraktionsebene

~~**wrongAlgo**(*input*)~~
~~**return** correct output~~

- Beispiel: Was macht der von diesem Pseudocode beschriebene Algorithmus?

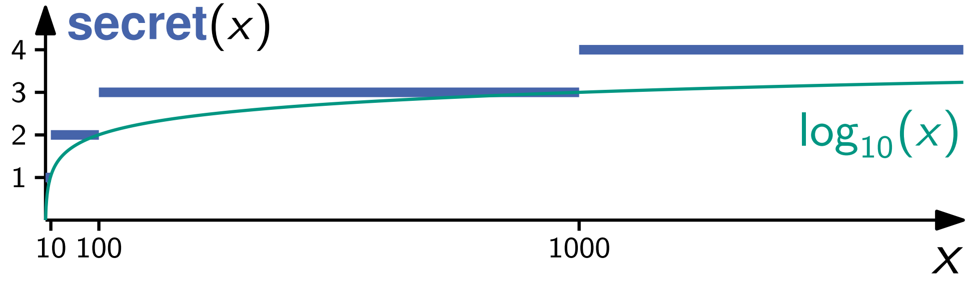
secret(x)
if $x \leq 10$ **return** 1
return $1 +$ **secret**$(x/10)$

Beispiel Eingaben:

$x \in [1, 10] \rightarrow 1$

$x \in [11, 100] \rightarrow 2$

$x \in [101, 1000] \rightarrow 3$



Gegeben eine Zahl $x \geq 1$, berechnet der Algorithmus $\lceil \log_{10}(x) \rceil$

Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

log10(x)

if $x \leq 10$ **return** 1
return $1 +$ **log10**$(x/10)$

Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

log10(x)

> **if** $x \leq 10$ **return** 1
> **return** $1 +$ **log10**$(x/10)$

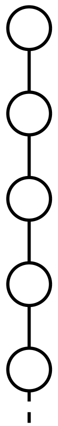
Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

$\textbf{log10}(x)$
  $\textbf{if } x \leq 10 \textbf{ return } 1$
  $\textbf{return } 1 + \textbf{log10}(x/10)$



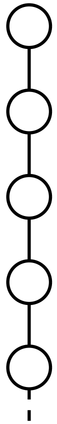
Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

```
log10(x)
  if x ≤ 10 return 1
  return 1 + log10(x/10)
```



$T(1) \in \Theta(1)$

Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

Master–Theorem

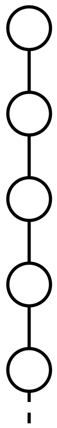
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

log10(x)

  if $x \leq 10$ **return** 1

  return $1 +$ **log10**$(x/10)$



$T(1) \in \Theta(1)$

$T(n) = f(n) + 1 \cdot T(n/10)$

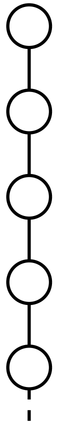
Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

log10(x)
  if $x \leq 10$ **return** 1
  return $1 +$ **log10**$(x/10)$



$T(1) \in \Theta(1)$

$$T(n) = f(n) + \underset{a}{1} \cdot T(n/10)$$

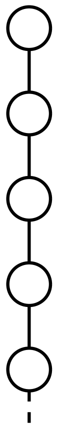
Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

log10(x)
  if $x \leq 10$ **return** 1
  return $1 +$ **log10**$(x/10)$



$T(1) \in \Theta(1)$ $\quad a \quad b$

$$T(n) = f(n) + \underset{a}{1} \cdot T(n/\underset{b}{10})$$

Master–Theorem

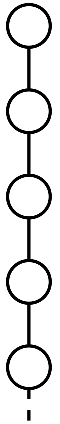
- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

log10(x)

    if $x \leq 10$ **return** 1
    return $1 +$ **log10**$(x/10)$



$$T(1) \in \Theta(1) \qquad a \qquad b$$

$$T(n) = f(n) + 1 \cdot T(n/10)$$

$$f(n) \in \Theta(1) = \Theta(n^0)$$

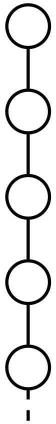
Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

log10(x)
  if $x \leq 10$ **return** 1
  return $1 +$ **log10**$(x/10)$



$$T(1) \in \Theta(1) \qquad a \qquad b \qquad c$$

$$T(n) = f(n) + 1 \cdot T(n/10)$$

$$f(n) \in \Theta(1) = \Theta(n^0)$$

Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

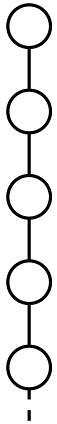
Master–Theorem

Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

log10(x)

- **if** $x \leq 10$ **return** 1
- **return** $1 +$ **log10**$(x/10)$



$T(1) \in \Theta(1)$ $\quad a \quad b \quad c \qquad b^c = 10^0 = 1 = a$

$$T(n) = f(n) + 1 \cdot T(n/10)$$

$$f(n) \in \Theta(1) = \Theta(n^0)$$

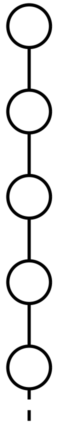
Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

log10(x)
  if $x \leq 10$ **return** 1
  return $1 +$ **log10**$(x/10)$



$T(1) \in \Theta(1)$ $\quad a \quad b \quad c$

$T(n) = f(n) + 1 \cdot T(n/10)$

$f(n) \in \Theta(1) = \Theta(n^0)$

$b^c = 10^0 = 1 = a$

$T(n) = \Theta(n^0 \log(n))$

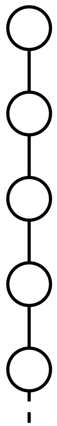
Master–Theorem

- Werkzeug zur Berechnung der Laufzeit von rekursiven Algorithmen

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

log10(x)
 if $x \leq 10$ **return** 1
 return $1 +$ **log10**$(x/10)$



$T(1) \in \Theta(1)$ a b c

$$T(n) = f(n) + 1 \cdot T(n/10)$$

$$f(n) \in \Theta(1) = \Theta(n^0)$$

$$b^c = 10^0 = 1 = a$$

$$T(n) = \Theta(n^0 \log(n)) = \Theta(\log(n))$$

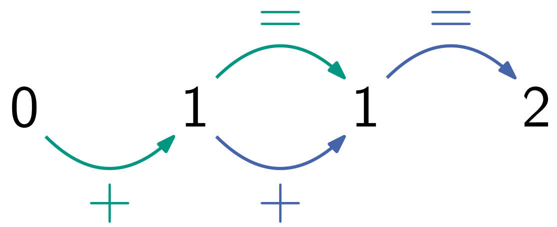
Fibonacci

$0 \qquad 1$

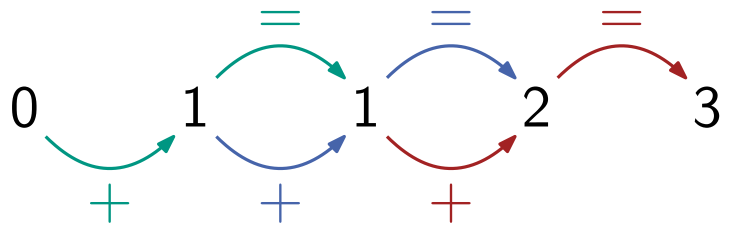
Fibonacci

$$\begin{tikzcd} 0 \arrow[r, bend right, "+"'] & 1 \arrow[r, bend left, "="] & 1 \end{tikzcd}$$

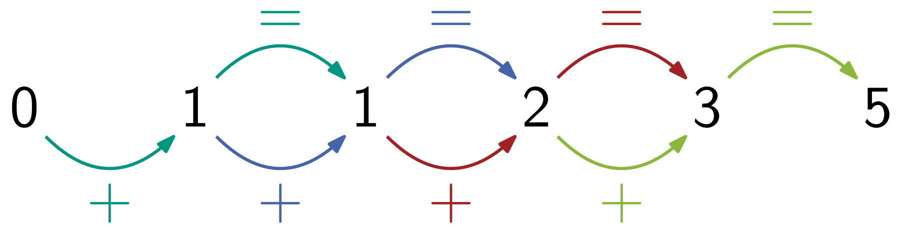
Fibonacci



Fibonacci



Fibonacci



Fibonacci

$$0 \quad 1 \quad 1 \quad 2 \quad 3 \quad 5 \quad 8 \quad 13 \quad 21 \quad 34 \quad 55 \quad 89 \quad \cdots$$

Fibonacci

	0	1	1	2	3	5	8	13	21	34	55	89	$\cdots$
$n =$	0	1	2	3	4	5	6	7	8	9	10	11	$\cdots$

Fibonacci

$$\begin{array}{lccccccccccccc}
 & 0 & 1 & 1 & 2 & 3 & 5 & 8 & 13 & 21 & 34 & 55 & 89 & \cdots \\
n = & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & \cdots
\end{array}$$

fib(n)

> **if** $n \leq 1$ **return** n
> **return fib**$(n-1)+$
> **fib**$(n-2)$

Fibonacci

$0 \quad 1 \quad 1 \quad 2 \quad 3 \quad 5 \quad 8 \quad 13 \quad 21 \quad 34 \quad 55 \quad 89 \quad \cdots$

$n = \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 9 \quad 10 \quad 11 \quad \cdots$

fib(4)

fib(n)

> **if** $n \leq 1$ **return** n
> **return** **fib**$(n-1) +$
> **fib**$(n-2)$

Fibonacci

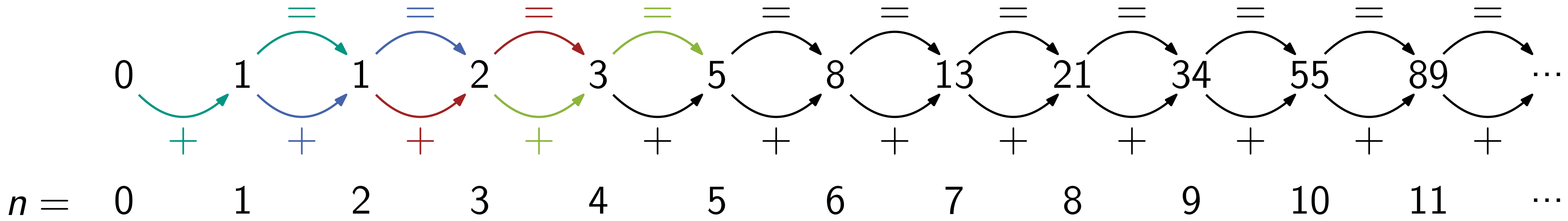
	0	1	1	2	3	5	8	13	21	34	55	89	$\cdots$
$n =$	0	1	2	3	4	5	6	7	8	9	10	11	$\cdots$

fib(n)

> **if** $n \leq 1$ **return** n
> **return fib**$(n-1)+$
> **fib**$(n-2)$

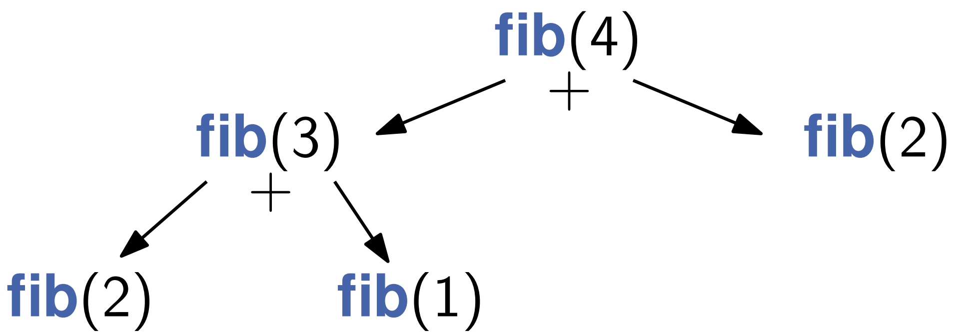
$$\mathbf{fib}(4) \to \mathbf{fib}(3) + \mathbf{fib}(2)$$

Fibonacci

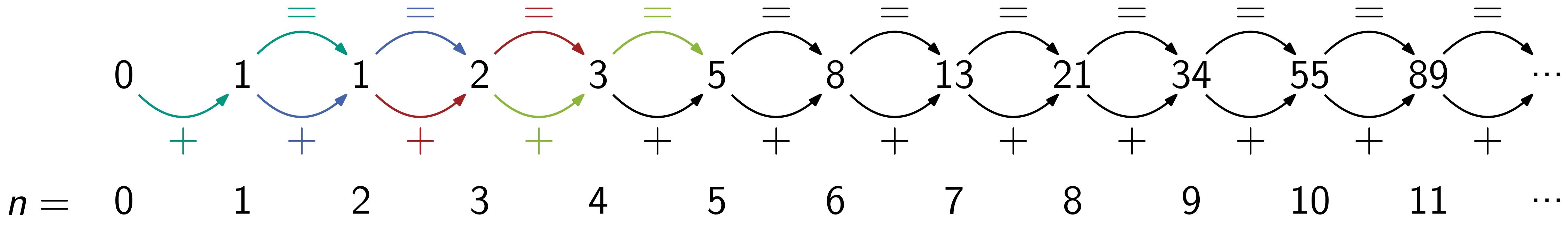


fib(n)

- **if** $n \leq 1$ **return** n
- **return** **fib**$(n-1) +$ **fib**$(n-2)$

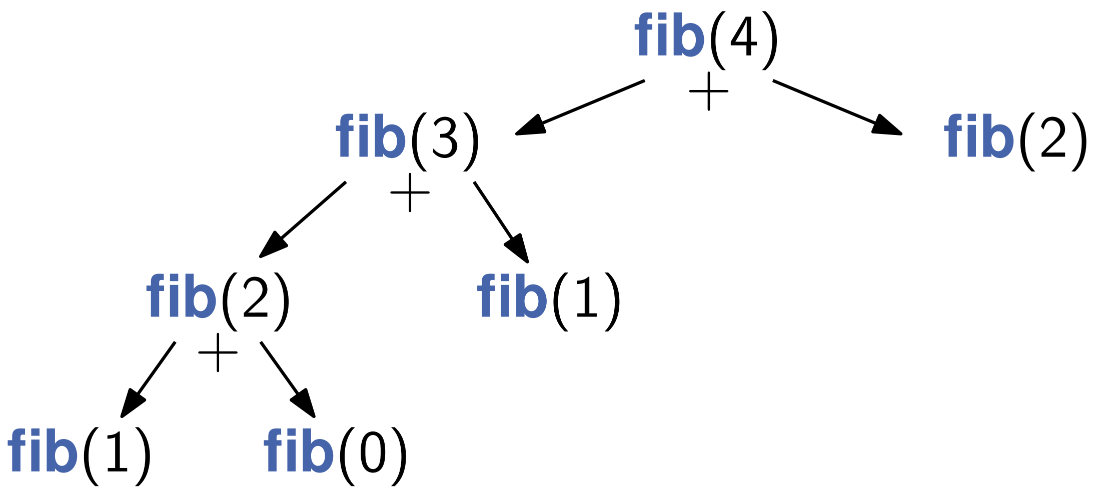


Fibonacci

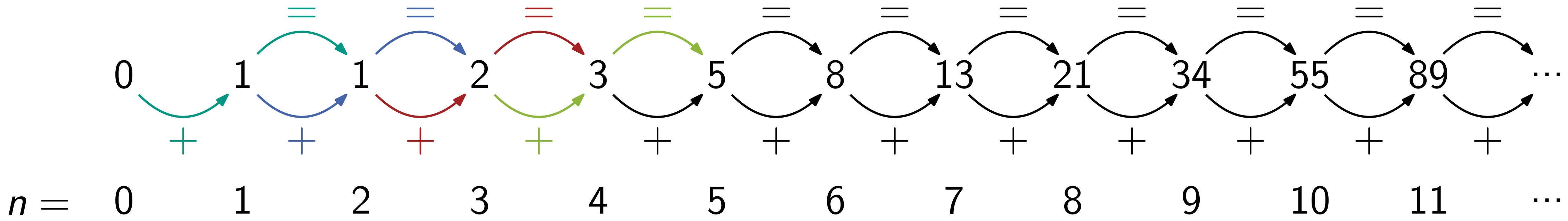


fib(n)

if $n \leq 1$ **return** n
return **fib**$(n-1)+$
fib$(n-2)$



Fibonacci

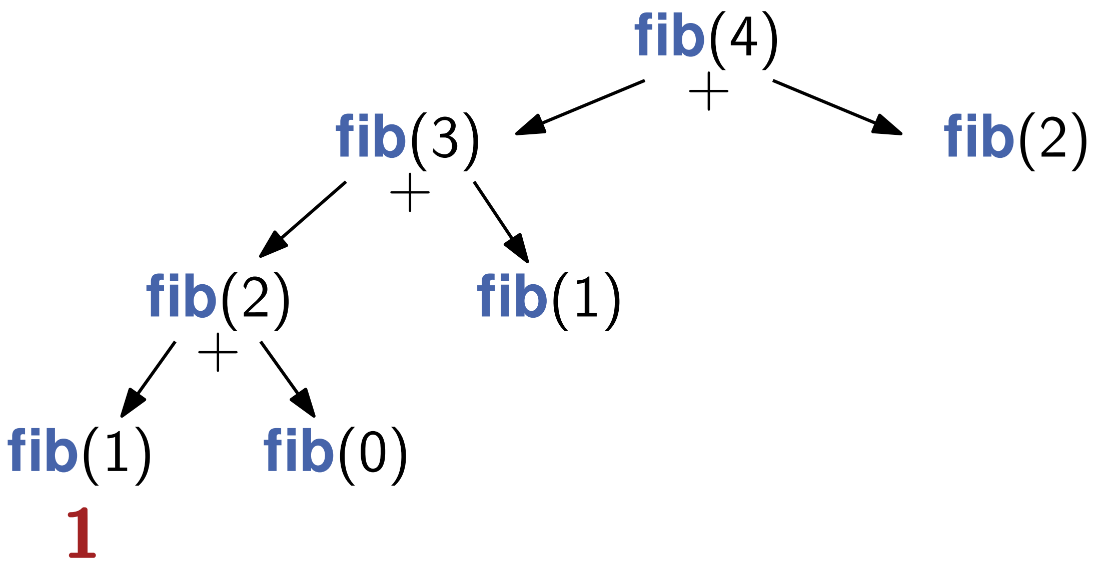


fib(n)

if $n \leq 1$ **return** n

return **fib**$(n-1)+$

fib$(n-2)$



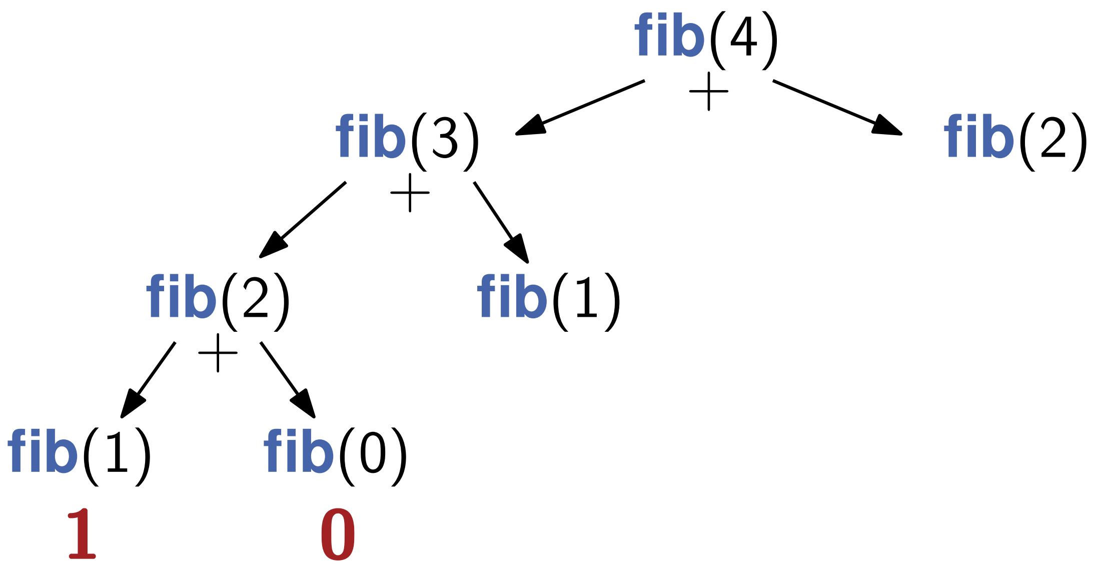
Fibonacci

$$0 \quad 1 \quad 1 \quad 2 \quad 3 \quad 5 \quad 8 \quad 13 \quad 21 \quad 34 \quad 55 \quad 89 \quad \cdots$$

$$n = \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 9 \quad 10 \quad 11 \quad \cdots$$

fib(n)

> **if** $n \leq 1$ **return** n
> **return fib**$(n-1)+$
> **fib**$(n-2)$

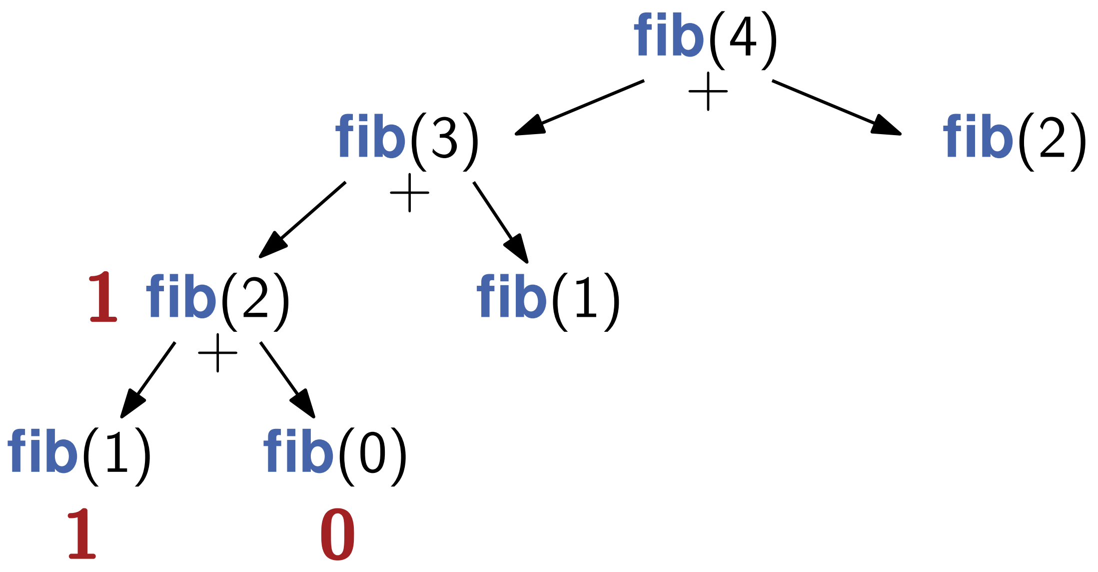


Fibonacci

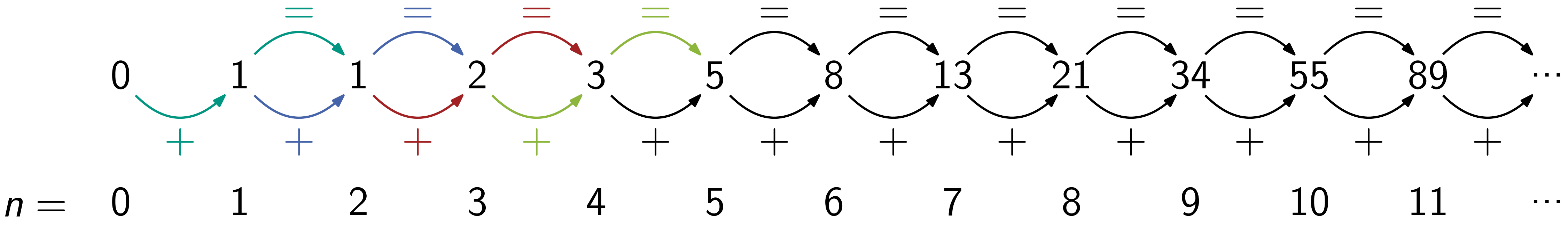
$$
\begin{array}{ccccccccccccc}
 & 0 & 1 & 1 & 2 & 3 & 5 & 8 & 13 & 21 & 34 & 55 & 89 & \cdots \\
n = & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & \cdots
\end{array}
$$

fib(n)

> **if** $n \le 1$ **return** n
> **return fib**$(n-1)+$
> **fib**$(n-2)$

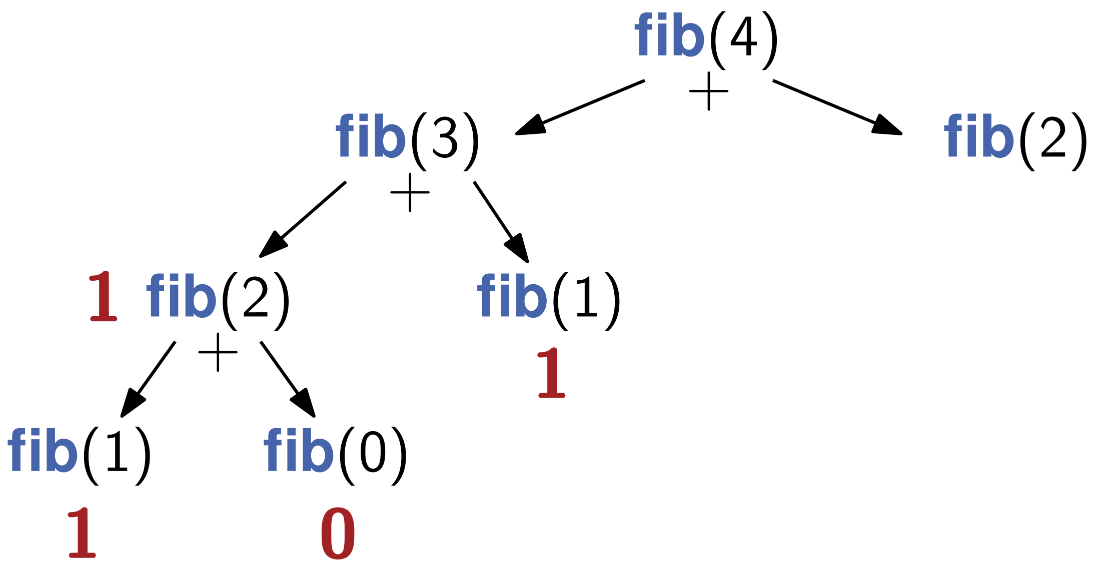


Fibonacci



fib(n)

> **if** $n \leq 1$ **return** n
> **return fib**$(n-1)+$
> **fib**$(n-2)$



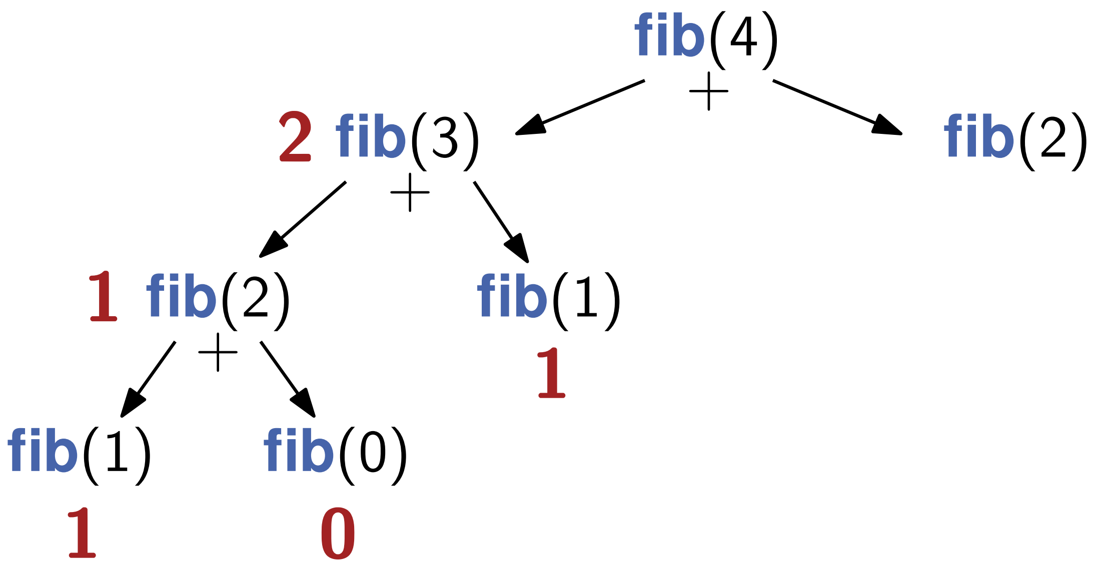
Fibonacci

$$0 \quad 1 \quad 1 \quad 2 \quad 3 \quad 5 \quad 8 \quad 13 \quad 21 \quad 34 \quad 55 \quad 89 \quad \cdots$$

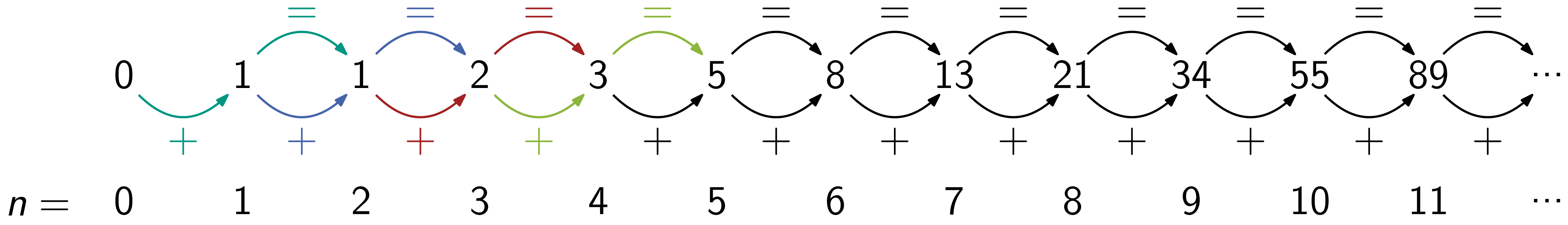
$$n = \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 9 \quad 10 \quad 11 \quad \cdots$$

fib(n)

> **if** $n \leq 1$ **return** n
> **return fib**$(n-1)+$
> **fib**$(n-2)$

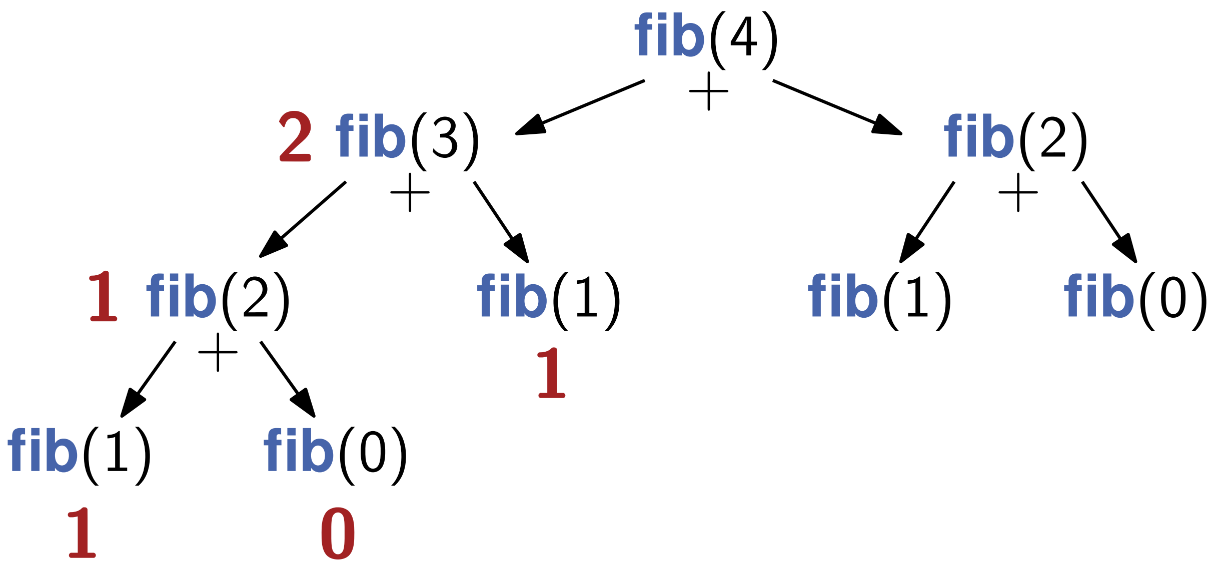


Fibonacci

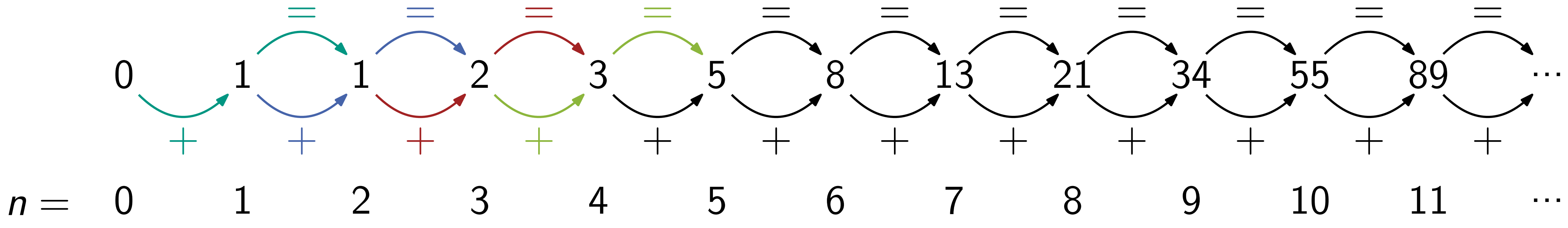


$\mathsf{fib}(n)$

> **if** $n \leq 1$ **return** n
> **return** $\mathsf{fib}(n-1) +$
> $\mathsf{fib}(n-2)$

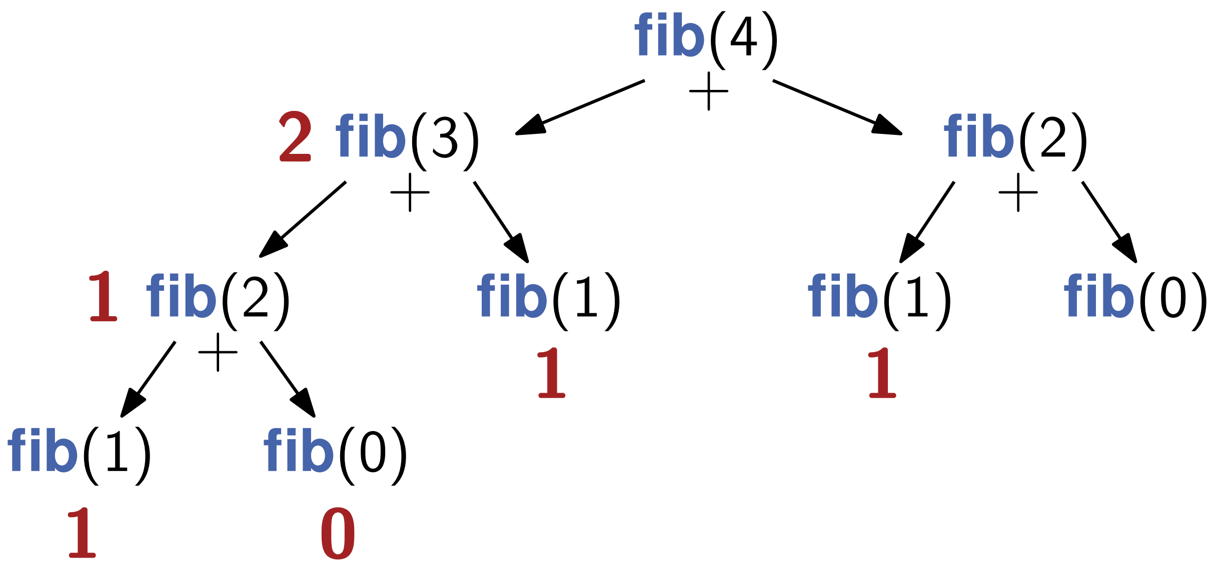


Fibonacci

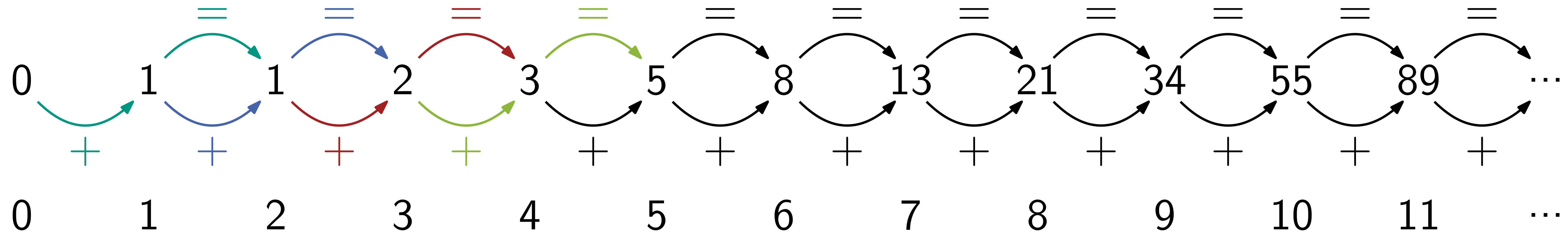


fib(n)

if $n \leq 1$ **return** n
return fib$(n-1)+$
fib$(n-2)$

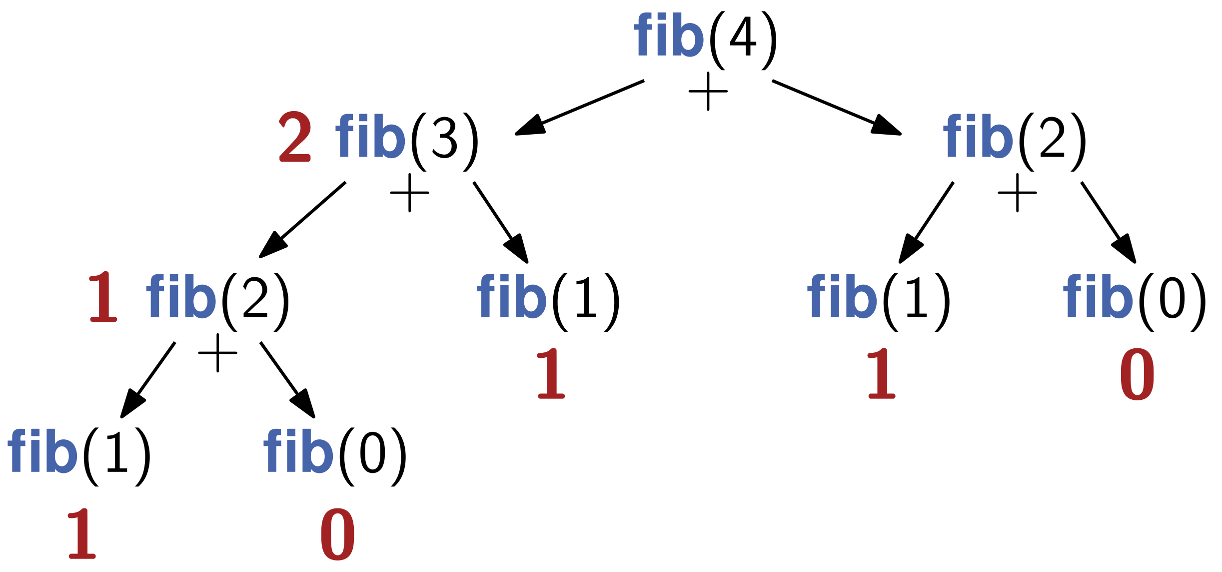


Fibonacci

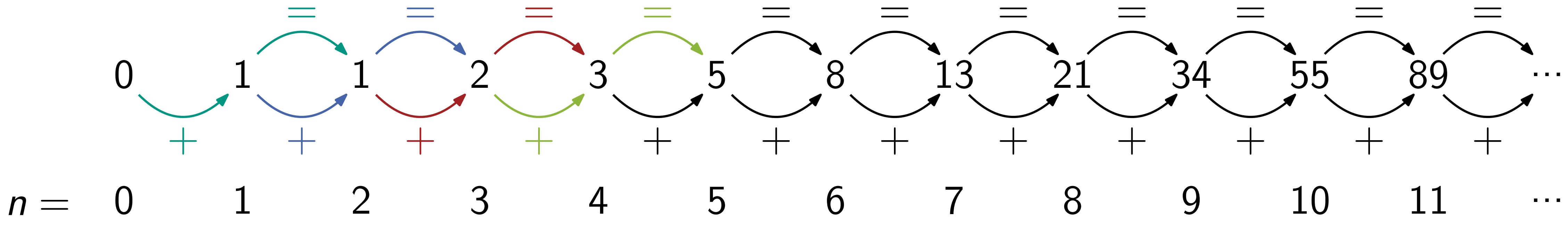


fib(n)

> **if** $n \leq 1$ **return** n
> **return** fib($n-1$) +
> fib($n-2$)

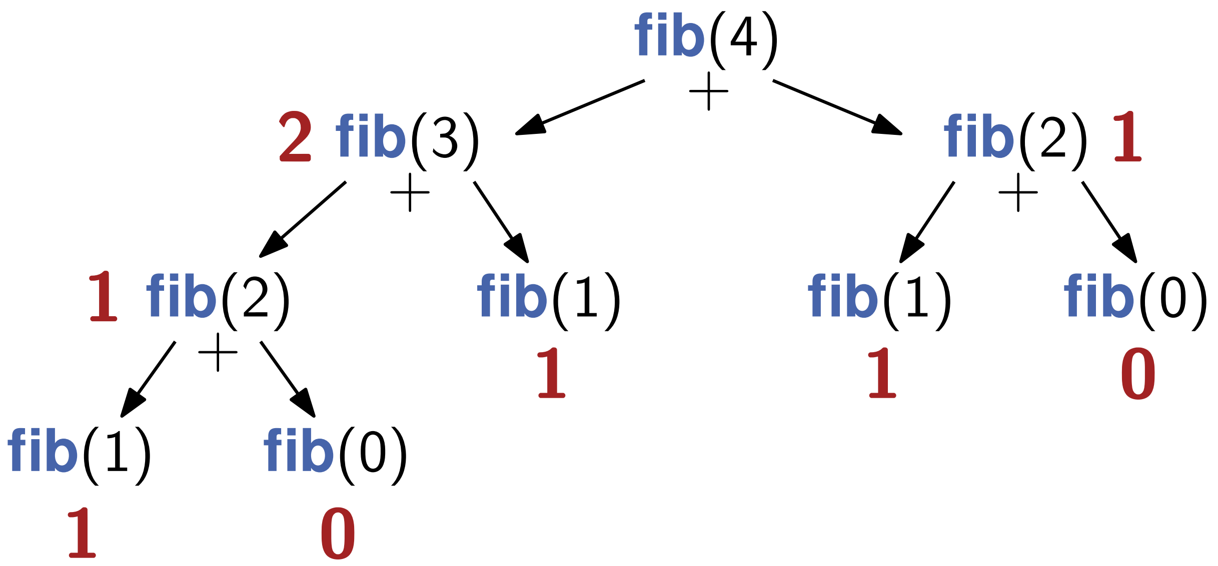


Fibonacci

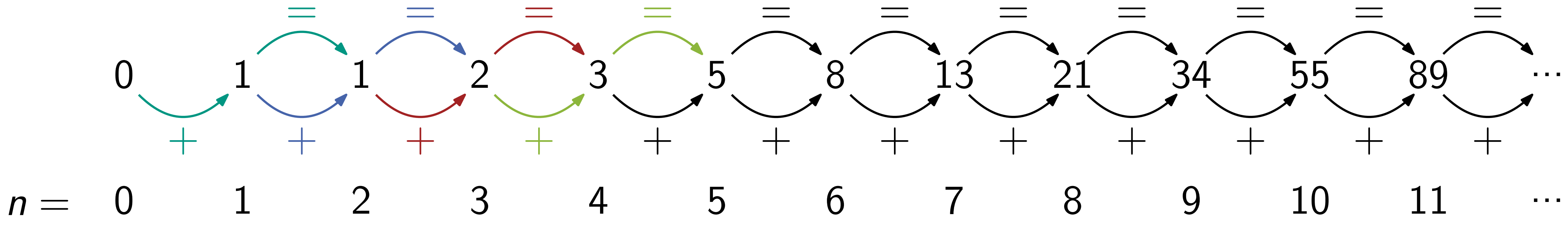


fib(n)

> **if** $n \leq 1$ **return** n
> **return fib**($n-1$) $+$ **fib**($n-2$)

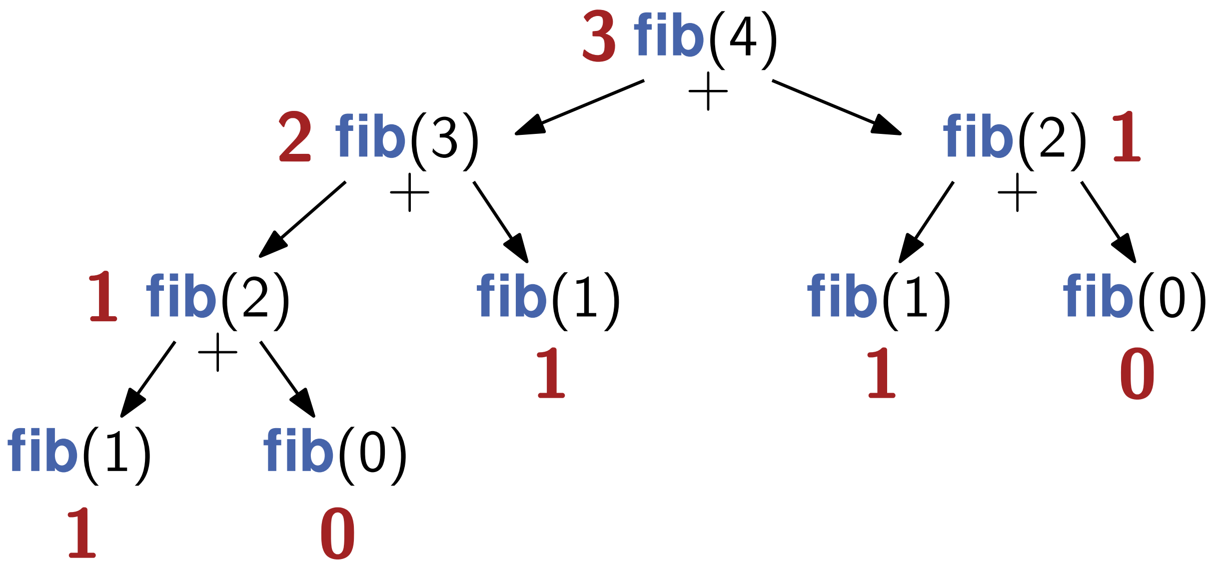


Fibonacci



fib(n)

> **if** $n \leq 1$ **return** n
> **return** **fib**$(n-1) +$ **fib**$(n-2)$



Fibonacci – Laufzeit

$\mathsf{fib}(n)$

$\quad$ **if** $n \leq 1$ **return** n

$\quad$ **return** $\mathsf{fib}(n-1) +$

$\qquad\qquad \mathsf{fib}(n-2)$

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

```
fib(n)
  if n ≤ 1 return n
  return fib(n − 1) +
         fib(n − 2)
```

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)

- **if** $n \leq 1$ **return** n
- **return** fib($n-1$) +
  fib($n-2$)

$$T(n) = T(n-1) + T(n-2) + 1$$

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

$$T(n) = T(n-1) + T(n-2) + 1$$

fib(n)

- **if** $n \leq 1$ **return** n
- **return** fib($n-1$) + fib($n-2$)

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)

  if $n \leq 1$ **return** n
  return fib($n-1$) +
      fib($n-2$)

$$T(n) = T(n-1) + T(n-2) + 1$$

$$T(n-2) \leq T(n-1)$$

Fibonacci – Laufzeit

Master–Theorem

Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)

> **if** $n \leq 1$ **return** n
> **return** fib($n-1$) +
> fib($n-2$)

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \end{aligned}$$

$T(n-2) \leq T(n-1)$

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

```
fib(n)
  if n ≤ 1 return n
  return fib(n − 1) +
         fib(n − 2)
```

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \end{aligned}$$

$T(n-2) \leq T(n-1)$

Fibonacci – Laufzeit

Master–Theorem

Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)

if $n \leq 1$ **return** n
return fib$(n-1) +$
fib$(n-2)$

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \end{aligned}$$

$T(n-2) \leq T(n-1)$

Fibonacci – Laufzeit

Karlsruher Institut für Technologie

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

```
fib(n)
    if n ≤ 1 return n
    return fib(n − 1) +
           fib(n − 2)
```

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \end{aligned}$$

$T(n-2) \leq T(n-1)$

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)

- **if** $n \leq 1$ **return** n
- **return fib**$(n-1) +$ **fib**$(n-2)$

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \\ &\quad \dots \\ &\leq 2^k \cdot T(n-k) + (2^k - 1) \end{aligned}$$

Induktion

$T(n-2) \leq T(n-1)$

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)

> **if** $n \leq 1$ **return** n
> **return fib**$(n-1)+$
> **fib**$(n-2)$

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \\ &\dots \\ \text{Induktion} \quad k = n \quad &\leq 2^k \cdot T(n-k) + (2^k - 1) \end{aligned}$$

$T(n-2) \leq T(n-1)$

Fibonacci – Laufzeit

Master–Theorem

Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)

- **if** $n \leq 1$ **return** n
- **return fib**$(n-1) +$ **fib**$(n-2)$

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \\ &\dots \\ \text{Induktion} \quad k = n \quad &\leq 2^k \cdot T(n-k) + (2^k - 1) \\ &\leq 2^n \cdot 1 + (2^n - 1) \end{aligned}$$

$T(n-2) \leq T(n-1)$

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)

if $n \leq 1$ **return** n
return fib$(n-1)+$
fib$(n-2)$

$T(n-2) \leq T(n-1)$

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \\ &\cdots \\ &\leq 2^k \cdot T(n-k) + (2^k - 1) \\ &\leq 2^n \cdot 1 + (2^n - 1) \in O(2^n) \end{aligned}$$

Induktion $\quad k = n$

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)
- **if** $n \leq 1$ **return** n
- **return fib**$(n-1) +$ **fib**$(n-2)$

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \\ &\quad \dots \\ \text{Induktion} \quad k = n \quad &\leq 2^k \cdot T(n-k) + (2^k - 1) \\ &\leq 2^n \cdot 1 + (2^n - 1) \in O(2^n) \end{aligned}$$

$T(n-2) \leq T(n-1)$

$\geq 2 \cdot T(n-2) + 1$

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

```
fib(n)
  if n ≤ 1 return n
  return fib(n − 1) +
         fib(n − 2)
```

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \\ &\cdots \end{aligned}$$

Induktion $\quad k = n$

$$\begin{aligned} &\leq 2^k \cdot T(n-k) + (2^k - 1) \\ &\leq 2^n \cdot 1 + (2^n - 1) \in O(2^n) \end{aligned}$$

$T(n-2) \leq T(n-1)$

$$\begin{aligned} &\geq 2 \cdot T(n-2) + 1 \\ &\geq 2 \cdot (2 \cdot T(n-4) + 1) + 1 \end{aligned}$$

Fibonacci – Laufzeit

Master–Theorem

Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

```
fib(n)
    if n ≤ 1 return n
    return fib(n − 1) +
           fib(n − 2)
```

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \\ &\dots \end{aligned}$$

Induktion $\quad k = n \quad \leq 2^k \cdot T(n-k) + (2^k - 1)$

$$\leq 2^n \cdot 1 + (2^n - 1) \in O(2^n)$$

$T(n-2) \leq T(n-1)$

$$\begin{aligned} &\geq 2 \cdot T(n-2) + 1 \\ &\geq 2 \cdot (2 \cdot T(n-4) + 1) + 1 \\ &\geq 4T(n-4) + 3 \end{aligned}$$

Fibonacci – Laufzeit

Karlsruher Institut für Technologie

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

```
fib(n)
  if n ≤ 1 return n
  return fib(n − 1) +
         fib(n − 2)
```

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\le 2 \cdot T(n-1) + 1 \\ &\le 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\le 4 \cdot T(n-2) + 3 \\ &\le 8 \cdot T(n-3) + 7 \\ &\dots \end{aligned}$$

Induktion $\quad k = n \quad \le 2^k \cdot T(n-k) + (2^k - 1)$

$$\le 2^n \cdot 1 + (2^n - 1) \in O(2^n)$$

$T(n-2) \le T(n-1)$

$$\begin{aligned} &\ge 2 \cdot T(n-2) + 1 \\ &\ge 2 \cdot (2 \cdot T(n-4) + 1) + 1 \\ &\ge 4T(n-4) + 3 \\ &\ge 8T(n-6) + 7 \end{aligned}$$

Fibonacci – Laufzeit

Master–Theorem

Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)

- **if** $n \leq 1$ **return** n
- **return fib**$(n-1)+$ **fib**$(n-2)$

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \\ &\dots \\ &\leq 2^k \cdot T(n-k) + (2^k - 1) \\ &\leq 2^n \cdot 1 + (2^n - 1) \in O(2^n) \end{aligned}$$

Induktion $k = n$

$T(n-2) \leq T(n-1)$

$$\begin{aligned} &\geq 2 \cdot T(n-2) + 1 \\ &\geq 2 \cdot (2 \cdot T(n-4) + 1) + 1 \\ &\geq 4T(n-4) + 3 \\ &\geq 8T(n-6) + 7 \\ &\dots \\ &\geq 2^k \cdot T(n-2k) + (2^k - 1) \end{aligned}$$

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

fib(n)
- **if** $n \leq 1$ **return** n
- **return fib**$(n-1)+$ **fib**$(n-2)$

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \\ &\quad \ldots \\ \text{Induktion} \quad k = n \quad &\leq 2^k \cdot T(n-k) + (2^k - 1) \\ &\leq 2^n \cdot 1 + (2^n - 1) \in O(2^n) \end{aligned}$$

$T(n-2) \leq T(n-1)$

$$\begin{aligned} &\geq 2 \cdot T(n-2) + 1 \\ &\geq 2 \cdot (2 \cdot T(n-4) + 1) + 1 \\ &\geq 4T(n-4) + 3 \\ &\geq 8T(n-6) + 7 \\ &\quad \ldots \\ &\geq 2^k \cdot T(n-2k) + (2^k - 1) \quad k = \frac{n}{2} \end{aligned}$$

Fibonacci – Laufzeit

Master–Theorem
Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

```
fib(n)
    if n ≤ 1 return n
    return fib(n − 1) +
           fib(n − 2)
```

$$T(n) = T(n-1) + T(n-2) + 1$$

$T(n-2) \leq T(n-1)$

$$\begin{aligned} &\leq 2 \cdot T(n-1) + 1 \\ &\leq 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\leq 4 \cdot T(n-2) + 3 \\ &\leq 8 \cdot T(n-3) + 7 \\ &\dots \\ &\leq 2^k \cdot T(n-k) + (2^k - 1) \quad k = n \\ &\leq 2^n \cdot 1 + (2^n - 1) \in O(2^n) \end{aligned}$$

$$\begin{aligned} &\geq 2 \cdot T(n-2) + 1 \\ &\geq 2 \cdot (2 \cdot T(n-4) + 1) + 1 \\ &\geq 4T(n-4) + 3 \\ &\geq 8T(n-6) + 7 \\ &\dots \\ &\geq 2^k \cdot T(n-2k) + (2^k - 1) \quad k = \frac{n}{2} \\ &\geq 2^{n/2} \cdot 1 + (2^{n/2} - 1) \end{aligned}$$

Induktion

Fibonacci – Laufzeit

Master–Theorem

Sei $T(n) = a \cdot T(\frac{n}{b}) + f(n)$ mit $f(n) \in \Theta(n^c)$ und $T(1) \in \Theta(1)$. Dann gilt

$$T(n) \in \begin{cases} \Theta\left(n^c\right) & \text{wenn } a < b^c, \\ \Theta\left(n^c \log n\right) & \text{wenn } a = b^c, \\ \Theta\left(n^{\log_b(a)}\right) & \text{wenn } a > b^c. \end{cases}$$

```
fib(n)
    if n ≤ 1 return n
    return fib(n − 1) +
           fib(n − 2)
```

$T(n-2) \le T(n-1)$

$$\begin{aligned} T(n) &= T(n-1) + T(n-2) + 1 \\ &\le 2 \cdot T(n-1) + 1 \\ &\le 2 \cdot (2 \cdot T(n-2) + 1) + 1 \\ &\le 4 \cdot T(n-2) + 3 \\ &\le 8 \cdot T(n-3) + 7 \\ &\quad \dots \\ &\le 2^k \cdot T(n-k) + (2^k - 1) \qquad k = n \\ &\le 2^n \cdot 1 + (2^n - 1) \in O(2^n) \end{aligned}$$

$$\begin{aligned} T(n) &\ge 2 \cdot T(n-2) + 1 \\ &\ge 2 \cdot (2 \cdot T(n-4) + 1) + 1 \\ &\ge 4T(n-4) + 3 \\ &\ge 8T(n-6) + 7 \\ &\quad \dots \\ &\ge 2^k \cdot T(n-2k) + (2^k - 1) \qquad k = \frac{n}{2} \\ &\ge 2^{n/2} \cdot 1 + (2^{n/2} - 1) \in \Omega(\sqrt{2}^n) \end{aligned}$$

Induktion

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \le c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \le c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \ge c^{n-1} + c^{n-2} + 1$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$
  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} - 1 + 1$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$
  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{- 1 + 1}$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$
  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n - 1 \geq c^{n-1} - 1 + c^{n-2}$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \leq c^n - 1$
  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \leq c^n - 1$
  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$
  $c^n \geq c^{n-1} + c^{n-2}$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \geq c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \leq c^n - 1$

  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$

  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

  $$c^n \geq c^{n-1} + c^{n-2}$$

  $$c^n - c^{n-1} - c^{n-2} \geq 0$$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \le c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\to T(n) \le c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \ge c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \le c^n - 1$
  $\to T(n) \le c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n \cancel{-1} \ge c^{n-1} \cancel{-1} + c^{n-2}$

$$\begin{aligned} c^n &\ge c^{n-1} + c^{n-2} \\ c^n - c^{n-1} - c^{n-2} &\ge 0 \\ c^2 - c - 1 &\ge 0 \end{aligned}$$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

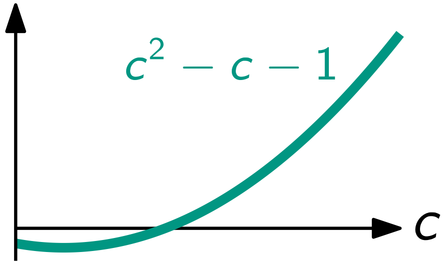
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \geq c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \leq c^n - 1$

  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$

  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

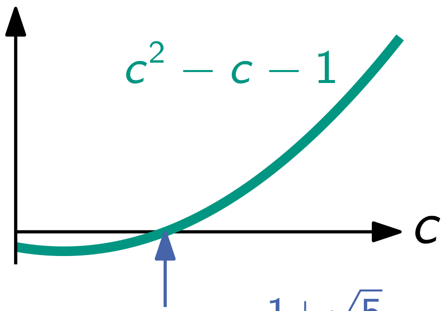
$$
\begin{aligned}
c^n &\geq c^{n-1} + c^{n-2} \\
c^n - c^{n-1} - c^{n-2} &\geq 0 \\
c^2 - c - 1 &\geq 0
\end{aligned}
$$



Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \leq c^n - 1$
  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

$$\begin{aligned} c^n &\geq c^{n-1} + c^{n-2} \\ c^n - c^{n-1} - c^{n-2} &\geq 0 \\ c^2 - c - 1 &\geq 0 \end{aligned}$$



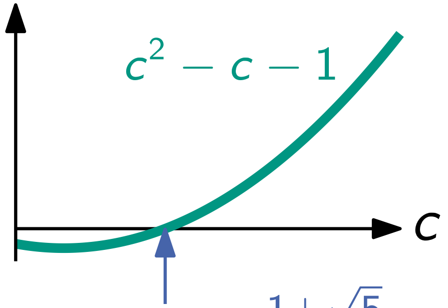
$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$
  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

$$
\begin{aligned}
c^n &\geq c^{n-1} + c^{n-2} \\
c^n - c^{n-1} - c^{n-2} &\geq 0 \\
c^2 - c - 1 &\geq 0
\end{aligned}
$$

Test: $T(0) \leq \phi^0 - 1 = 0, T(1) \leq \phi^1 - 1 \approx 0.618$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

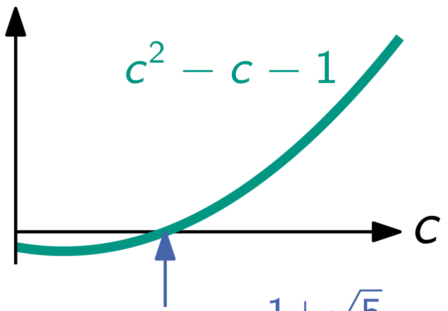
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$

  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$

  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

$$\begin{aligned} c^n &\geq c^{n-1} + c^{n-2} \\ c^n - c^{n-1} - c^{n-2} &\geq 0 \\ c^2 - c - 1 &\geq 0 \end{aligned}$$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Test: $T(0) \leq \phi^0 - 1 = 0$, $T(1) \leq \phi^1 - 1 \approx 0.618$

$\not\geq 1$ $\qquad \not\geq 1$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$

  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{- 1 + 1}$

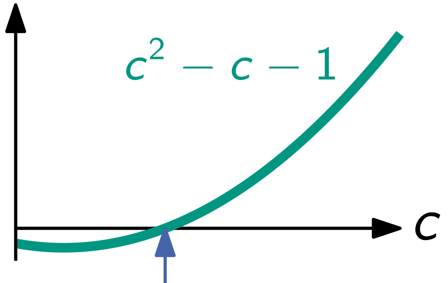
  Bedingung: $c^n \cancel{- 1} \geq c^{n-1} \cancel{- 1} + c^{n-2}$

$$\begin{aligned} c^n &\geq c^{n-1} + c^{n-2} \\ c^n - c^{n-1} - c^{n-2} &\geq 0 \\ c^2 - c - 1 &\geq 0 \end{aligned}$$

Test: $T(0) \leq \phi^0 - 1 = 0$, $T(1) \leq \phi^1 - 1 \approx 0.618$

$\not\geq 1$ $\not\geq 1$

- Idee 3: $T(n) \leq 2 \cdot c^n - 1$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

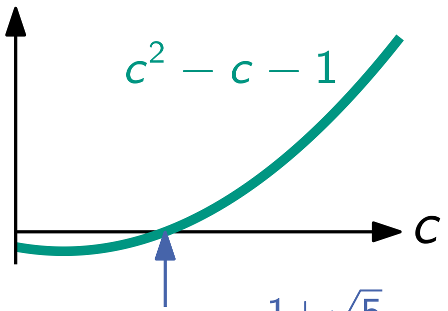
Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$
  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

$$\begin{aligned} c^n &\geq c^{n-1} + c^{n-2} \\ c^n - c^{n-1} - c^{n-2} &\geq 0 \\ c^2 - c - 1 &\geq 0 \end{aligned}$$

Test: $T(0) \leq \phi^0 - 1 = 0$, $T(1) \leq \phi^1 - 1 \approx 0.618$
$\not\geq 1$ $\not\geq 1$

- Idee 3: $T(n) \leq 2 \cdot c^n - 1$
  $\rightarrow T(n) \leq 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

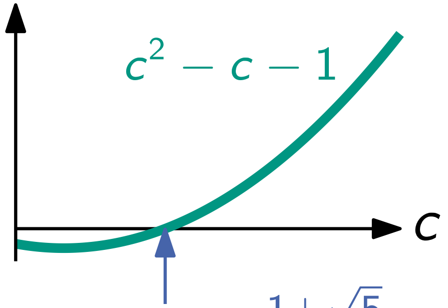
Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \le c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \le c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \ge c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \le c^n - 1$
  $\rightarrow T(n) \le c^{n-1} - 1 + c^{n-2} \cancel{- 1 + 1}$
  Bedingung: $c^n \cancel{-1} \ge c^{n-1} \cancel{-1} + c^{n-2}$

$$
\begin{aligned}
c^n &\ge c^{n-1} + c^{n-2} \\
c^n - c^{n-1} - c^{n-2} &\ge 0 \\
c^2 - c - 1 &\ge 0
\end{aligned}
$$

Test: $T(0) \le \phi^0 - 1 = 0$, $T(1) \le \phi^1 - 1 \approx 0.618$
$\ngeq 1$ $\ngeq 1$

- Idee 3: $T(n) \le 2 \cdot c^n - 1$
  $\rightarrow T(n) \le 2c^{n-1} - 1 + 2c^{n-2} \cancel{- 1 + 1}$
  Bed.: $2c^n - 1 \ge 2c^{n-1} - 1 + 2c^{n-2}$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$

  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{- 1 + 1}$

  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

$$\begin{aligned} c^n &\geq c^{n-1} + c^{n-2} \\ c^n - c^{n-1} - c^{n-2} &\geq 0 \\ c^2 - c - 1 &\geq 0 \end{aligned}$$

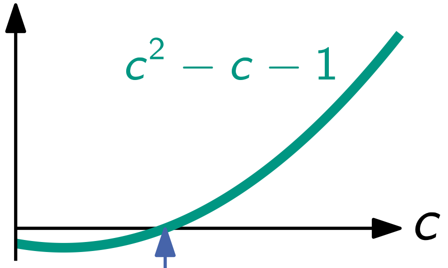
Test: $T(0) \leq \phi^0 - 1 = 0$, $T(1) \leq \phi^1 - 1 \approx 0.618$

$\not\geq 1$ $\qquad$ $\not\geq 1$

- Idee 3: $T(n) \leq 2 \cdot c^n - 1$

  $\rightarrow T(n) \leq 2c^{n-1} - 1 + 2c^{n-2} \cancel{- 1 + 1}$

  Bed.: $2c^n \cancel{-1} \geq 2c^{n-1} \cancel{-1} + 2c^{n-2}$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

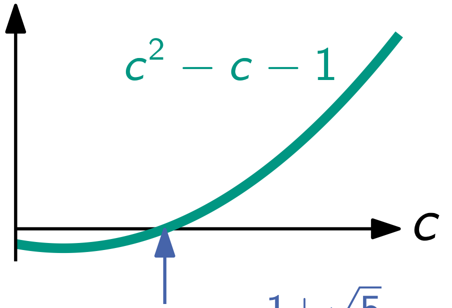
Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\to T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \leq c^n - 1$
  $\to T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

$$\begin{aligned} c^n &\geq c^{n-1} + c^{n-2} \\ c^n - c^{n-1} - c^{n-2} &\geq 0 \\ c^2 - c - 1 &\geq 0 \end{aligned}$$

Test: $T(0) \leq \phi^0 - 1 = \underset{\not\geq 1}{0}$, $T(1) \leq \phi^1 - 1 \approx \underset{\not\geq 1}{0.618}$

- Idee 3: $T(n) \leq 2 \cdot c^n - 1$
  $\to T(n) \leq 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$
  Bed.: $2c^n \cancel{-1} \geq 2c^{n-1} \cancel{-1} + 2c^{n-2}$
  $2c^n \geq 2c^{n-1} + 2c^{n-2}$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$

  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$

  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

$$\begin{aligned} c^n &\geq c^{n-1} + c^{n-2} \\ c^n - c^{n-1} - c^{n-2} &\geq 0 \\ c^2 - c - 1 &\geq 0 \end{aligned}$$

Test: $T(0) \leq \phi^0 - 1 = 0$, $T(1) \leq \phi^1 - 1 \approx 0.618$

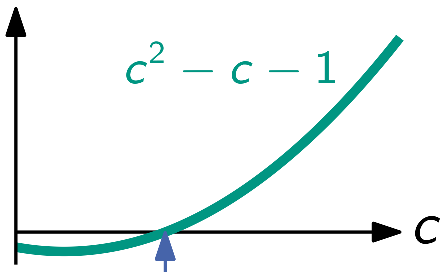
$\underset{\not\geq 1}{0}$ $\quad$ $\underset{\not\geq 1}{0.618}$

- Idee 3: $T(n) \leq 2 \cdot c^n - 1$

  $\rightarrow T(n) \leq 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$

  Bed.: $2c^n \cancel{-1} \geq 2c^{n-1} \cancel{-1} + 2c^{n-2}$

$$\begin{aligned} 2c^n &\geq 2c^{n-1} + 2c^{n-2} \\ c^n &\geq c^{n-1} + c^{n-2} \end{aligned}$$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

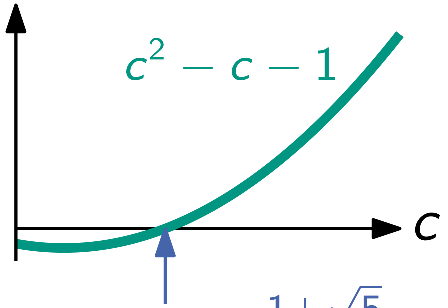
- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \le c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \le c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \ge c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \le c^n - 1$
  $\rightarrow T(n) \le c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n \cancel{-1} \ge c^{n-1} \cancel{-1} + c^{n-2}$

$$c^n \ge c^{n-1} + c^{n-2}$$
$$c^n - c^{n-1} - c^{n-2} \ge 0$$
$$c^2 - c - 1 \ge 0$$

Test: $T(0) \le \phi^0 - 1 = 0$, $T(1) \le \phi^1 - 1 \approx 0.618$
$\ngeq 1$ $\ngeq 1$

- Idee 3: $T(n) \le 2 \cdot c^n - 1$
  $\rightarrow T(n) \le 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$
  Bed.: $2c^n \cancel{-1} \ge 2c^{n-1} \cancel{-1} + 2c^{n-2}$

$$2c^n \ge 2c^{n-1} + 2c^{n-2}$$
$$c^n \ge c^{n-1} + c^{n-2}$$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \le c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \le c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \ge c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \le c^n - 1$
  $\rightarrow T(n) \le c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n \cancel{-1} \ge c^{n-1} \cancel{-1} + c^{n-2}$

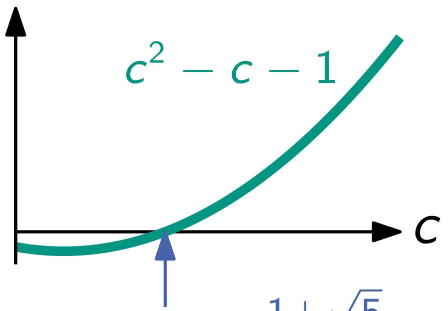
$$c^n \ge c^{n-1} + c^{n-2}$$
$$c^n - c^{n-1} - c^{n-2} \ge 0$$
$$c^2 - c - 1 \ge 0$$

Test: $T(0) \le \phi^0 - 1 = 0$, $T(1) \le \phi^1 - 1 \approx 0.618$

$\not\le 1$ $\not\le 1$

- Idee 3: $T(n) \le 2 \cdot c^n - 1$
  $\rightarrow T(n) \le 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$
  Bed.: $2c^n \cancel{-1} \ge 2c^{n-1} \cancel{-1} + 2c^{n-2}$

$$2c^n \ge 2c^{n-1} + 2c^{n-2}$$
$$c^n \ge c^{n-1} + c^{n-2} \qquad c = \phi$$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

Karlsruher Institut für Technologie

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \geq c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \leq c^n - 1$

  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$

  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

  $$c^n \geq c^{n-1} + c^{n-2}$$
  $$c^n - c^{n-1} - c^{n-2} \geq 0$$
  $$c^2 - c - 1 \geq 0$$

  Test: $T(0) \leq \phi^0 - 1 = 0$, $T(1) \leq \phi^1 - 1 \approx 0.618$

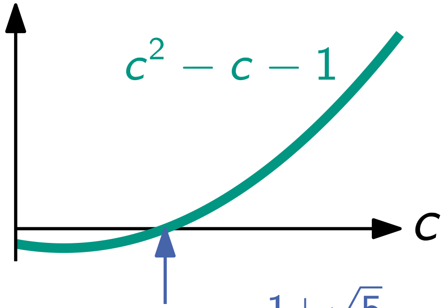
  $\not\geq 1$ $\qquad$ $\not\geq 1$
- Idee 3: $T(n) \leq 2 \cdot c^n - 1$

  $\rightarrow T(n) \leq 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$

  Bed.: $2c^n \cancel{-1} \geq 2c^{n-1} \cancel{-1} + 2c^{n-2}$

  $$2c^n \geq 2c^{n-1} + 2c^{n-2}$$
  $$c^n \geq c^{n-1} + c^{n-2} \qquad c = \phi$$

  Test: $T(0) \leq 2\phi^0 - 1 = 1$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \le c^n$, $c \in [\sqrt{2}, 2]$ konstant

  $\to T(n) \le c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \ge c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \le c^n - 1$

  $\to T(n) \le c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$

  Bedingung: $c^n \cancel{-1} \ge c^{n-1} \cancel{-1} + c^{n-2}$

$$\begin{aligned} c^n &\ge c^{n-1} + c^{n-2} \\ c^n - c^{n-1} - c^{n-2} &\ge 0 \\ c^2 - c - 1 &\ge 0 \end{aligned}$$

Test: $T(0) \le \phi^0 - 1 = 0$, $T(1) \le \phi^1 - 1 \approx 0.618$

$\not\ge 1$ $\not\ge 1$

- Idee 3: $T(n) \le 2 \cdot c^n - 1$

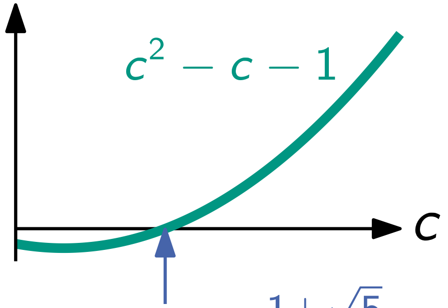
  $\to T(n) \le 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$

  Bed.: $2c^n \cancel{-1} \ge 2c^{n-1} \cancel{-1} + 2c^{n-2}$

$$\begin{aligned} 2c^n &\ge 2c^{n-1} + 2c^{n-2} \\ c^n &\ge c^{n-1} + c^{n-2} \end{aligned}$$

$c = \phi$

Test: $T(0) \le 2\phi^0 - 1 = 1$ ≥ 1



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

Karlsruher Institut für Technologie

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant
  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$
  Bedingung: $c^n \geq c^{n-1} + c^{n-2} + 1$
- Idee 2: $T(n) \leq c^n - 1$
  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$
  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

$$c^n \geq c^{n-1} + c^{n-2}$$
$$c^n - c^{n-1} - c^{n-2} \geq 0$$
$$c^2 - c - 1 \geq 0$$

Test: $T(0) \leq \phi^0 - 1 = 0$, $T(1) \leq \phi^1 - 1 \approx 0.618$

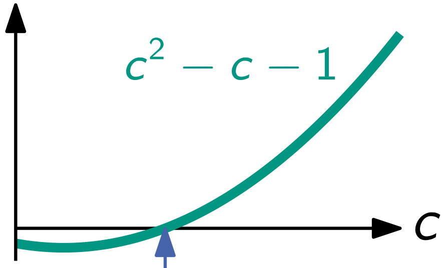
$\not\geq 1$ $\not\geq 1$

- Idee 3: $T(n) \leq 2 \cdot c^n - 1$
  $\rightarrow T(n) \leq 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$
  Bed.: $2c^n \cancel{-1} \geq 2c^{n-1} \cancel{-1} + 2c^{n-2}$

$$2c^n \geq 2c^{n-1} + 2c^{n-2}$$
$$c^n \geq c^{n-1} + c^{n-2} \qquad c = \phi$$

Test: $T(0) \leq 2\phi^0 - 1 = 1$ $\quad \geq 1$

$T(1) \leq 2\phi^1 - 1 \approx 2.236$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \geq c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \leq c^n - 1$

  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$

  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

$$\begin{aligned} c^n &\geq c^{n-1} + c^{n-2} \\ c^n - c^{n-1} - c^{n-2} &\geq 0 \\ c^2 - c - 1 &\geq 0 \end{aligned}$$

Test: $T(0) \leq \phi^0 - 1 = 0$, $T(1) \leq \phi^1 - 1 \approx 0.618$

$\not\geq 1$ $\not\geq 1$

- Idee 3: $T(n) \leq 2 \cdot c^n - 1$

  $\rightarrow T(n) \leq 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$

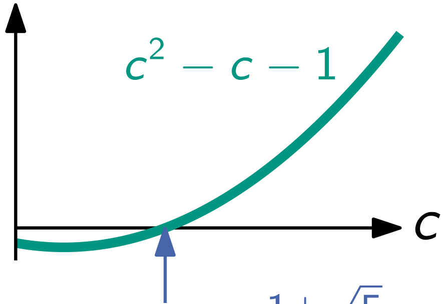
  Bed.: $2c^n \cancel{-1} \geq 2c^{n-1} \cancel{-1} + 2c^{n-2}$

$$\begin{aligned} 2c^n &\geq 2c^{n-1} + 2c^{n-2} \\ c^n &\geq c^{n-1} + c^{n-2} \end{aligned}$$

$c = \phi$

Test: $T(0) \leq 2\phi^0 - 1 = 1$ ≥ 1

$T(1) \leq 2\phi^1 - 1 \approx 2.236$ ≥ 1



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \geq c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \leq c^n - 1$

  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$

  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

  $$c^n \geq c^{n-1} + c^{n-2}$$
  $$c^n - c^{n-1} - c^{n-2} \geq 0$$
  $$c^2 - c - 1 \geq 0$$

  Test: $T(0) \leq \phi^0 - 1 = 0$, $T(1) \leq \phi^1 - 1 \approx 0.618$

  $\not\geq 1$ $\not\geq 1$

- Idee 3: $T(n) \leq 2 \cdot c^n - 1$

  $\rightarrow T(n) \leq 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$

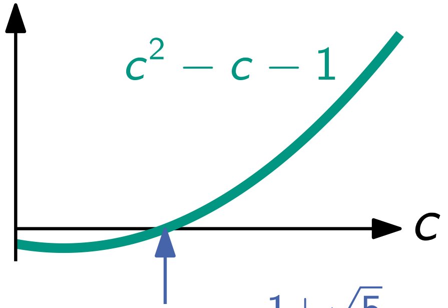
  Bed.: $2c^n \cancel{-1} \geq 2c^{n-1} \cancel{-1} + 2c^{n-2}$

  $$2c^n \geq 2c^{n-1} + 2c^{n-2}$$
  $$c^n \geq c^{n-1} + c^{n-2} \qquad c = \phi$$

  Test: $T(0) \leq 2\phi^0 - 1 = 1$ $\quad \geq 1$

  $T(1) \leq 2\phi^1 - 1 \approx 2.236$ $\quad \geq 1$

  $$T(n) \leq 2 \cdot \phi^n - 1$$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Laufzeit (bessere obere Schranke)

- Was wir bereits wissen
  - $T(n) = T(n-1) + T(n-2) + 1$
  - $T(0) = T(1) = 1$
  - $T(n) \in \Omega(\sqrt{2}^n)$, wobei $\sqrt{2} \approx 1.4142$ und
  - $T(n) \in O(2^n)$
- Idee: $T(n) \leq c^n$, $c \in [\sqrt{2}, 2]$ konstant

  $\rightarrow T(n) \leq c^{n-1} + c^{n-2} + 1$

  Bedingung: $c^n \geq c^{n-1} + c^{n-2} \boxed{+1}$
- Idee 2: $T(n) \leq c^n - 1$

  $\rightarrow T(n) \leq c^{n-1} - 1 + c^{n-2} \cancel{-1+1}$

  Bedingung: $c^n \cancel{-1} \geq c^{n-1} \cancel{-1} + c^{n-2}$

$$c^n \geq c^{n-1} + c^{n-2}$$
$$c^n - c^{n-1} - c^{n-2} \geq 0$$
$$c^2 - c - 1 \geq 0$$

Test: $T(0) \leq \phi^0 - 1 = 0$, $T(1) \leq \phi^1 - 1 \approx 0.618$

$\not\geq 1$ $\quad$ $\not\geq 1$

- Idee 3: $T(n) \leq 2 \cdot c^n - 1$

  $\rightarrow T(n) \leq 2c^{n-1} - 1 + 2c^{n-2} \cancel{-1+1}$

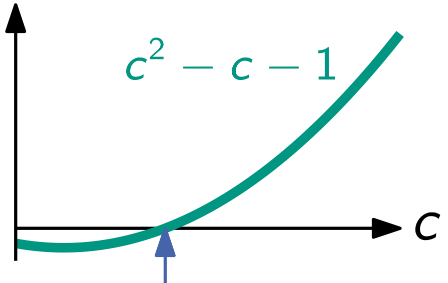
  Bed.: $2c^n \cancel{-1} \geq 2c^{n-1} \cancel{-1} + 2c^{n-2}$

$$2c^n \geq 2c^{n-1} + 2c^{n-2}$$
$$c^n \geq c^{n-1} + c^{n-2} \qquad c = \phi$$

Test: $T(0) \leq 2\phi^0 - 1 = 1 \qquad \geq 1$

$T(1) \leq 2\phi^1 - 1 \approx 2.236 \qquad \geq 1$

$$T(n) \leq 2 \cdot \phi^n - 1$$
$$T(n) \in O(\phi^n) \subseteq O(1.6181^n)$$



$c = \frac{1+\sqrt{5}}{2} =: \phi \approx 1.6180$

Fibonacci – Das Problem

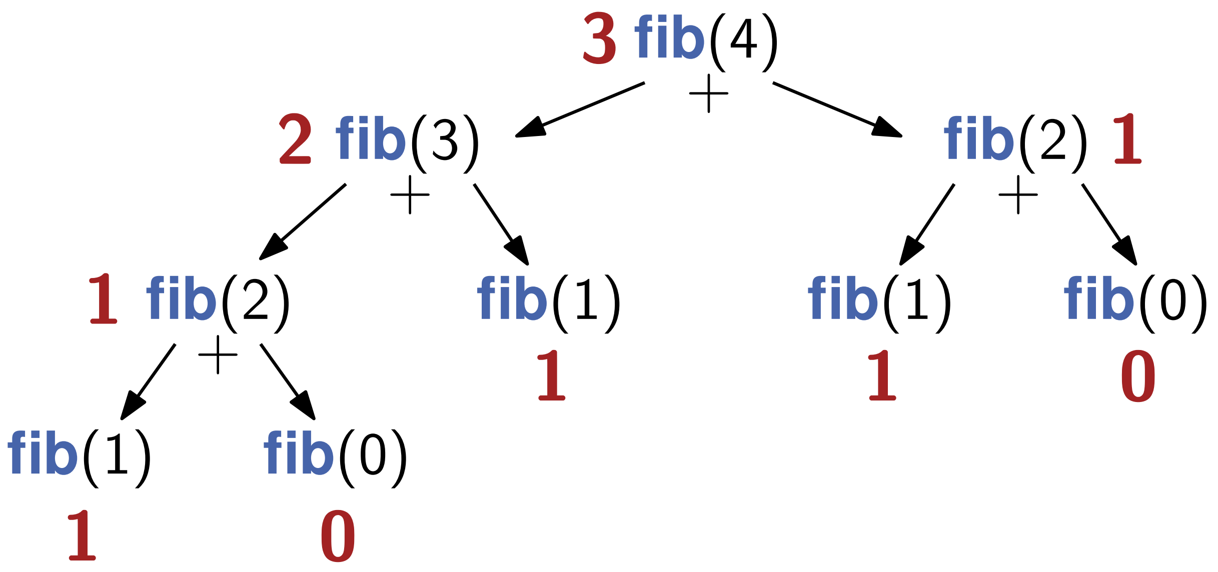
- Im Rekursionsbaum verdoppeln sich die Knoten mit jeder weiteren Lage (ungefähr)

$\textbf{fib}(n)$

$\quad$ **if** $n \leq 1$ **return** n

$\quad$ **return** $\textbf{fib}(n-1) +$

$\qquad\qquad \textbf{fib}(n-2)$



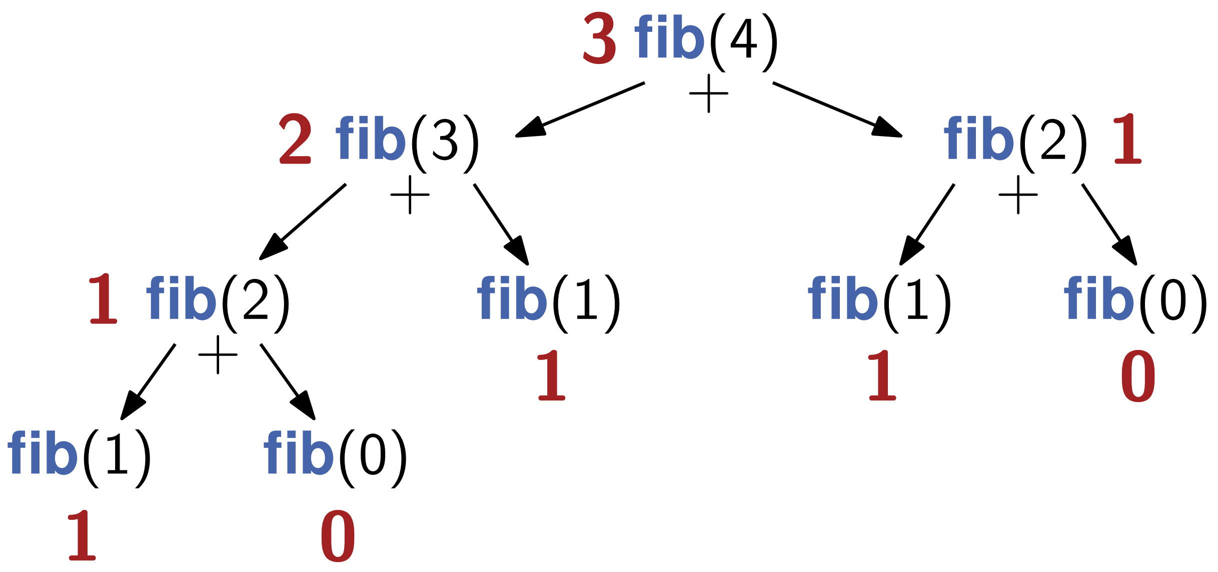
Fibonacci – Das Problem

- Im Rekursionsbaum verdoppeln sich die Knoten mit jeder weiteren Lage (ungefähr)
- Mit wachsender Anzahl von Lagen schrumpft n nicht schnell genug.

  → Viele Lagen!

fib(n)

> **if** $n \leq 1$ **return** n
> **return** **fib**$(n-1)+$
> **fib**$(n-2)$

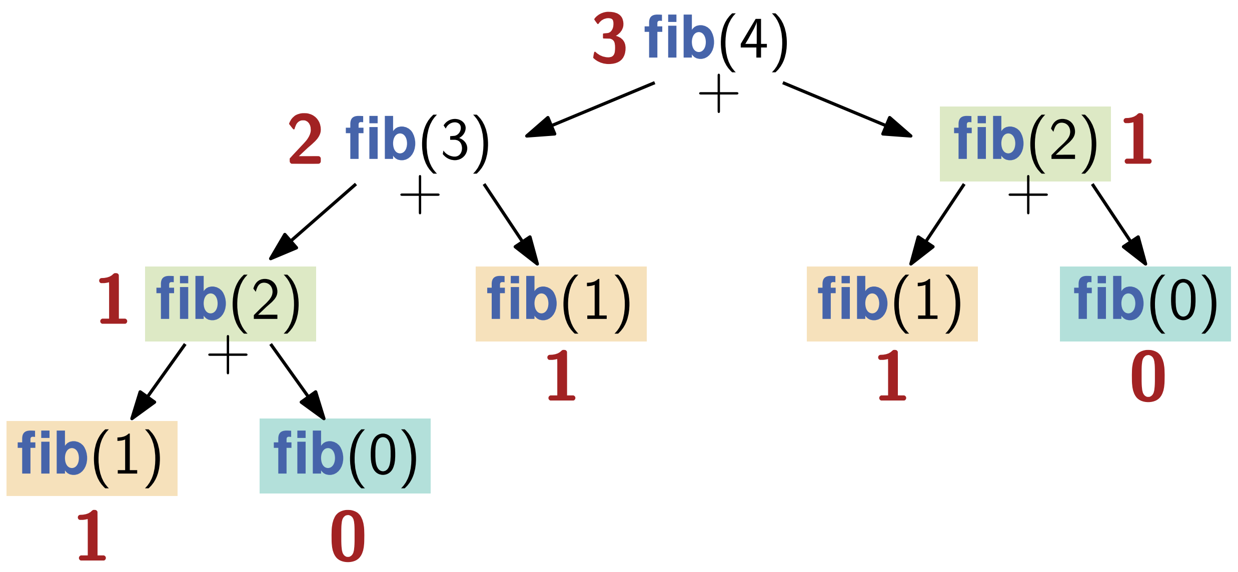


Fibonacci – Das Problem

- Im Rekursionsbaum verdoppeln sich die Knoten mit jeder weiteren Lage (ungefähr)
- Mit wachsender Anzahl von Lagen schrumpft n nicht schnell genug.
  → Viele Lagen!
- Viele Teilergebnisse werden mehrfach berechnet

$\mathsf{fib}(n)$

> **if** $n \leq 1$ **return** n
> **return** $\mathsf{fib}(n-1) +$
> $\qquad\qquad \mathsf{fib}(n-2)$



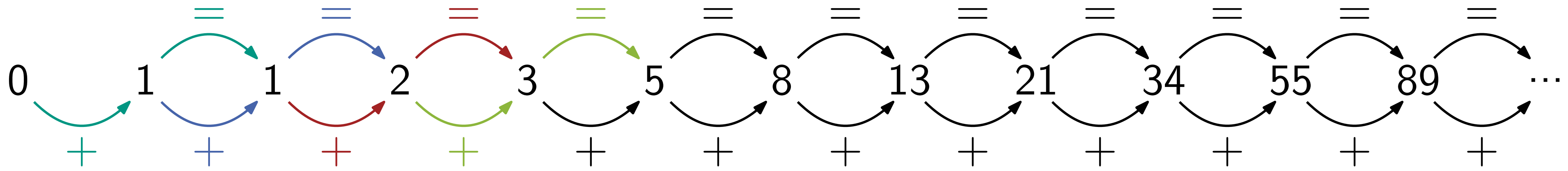
Fibonacci – Schneller

	0	1	1	2	3	5	8	13	21	34	55	89	$\cdots$
$n =$	0	1	2	3	4	5	6	7	8	9	10	11	$\cdots$

fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

Fibonacci – Schneller



$n =$ 0 1 2 3 4 5 6 7 8 9 10 11 ⋯

fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

Fibonacci – Schneller

Karlsruher Institut für Technologie

$$
\begin{array}{cccccccccccccc}
 & & = & = & = & = & = & = & = & = & = & = & = & \\
 & 0 & 1 & 1 & 2 & 3 & 5 & 8 & 13 & 21 & 34 & 55 & 89 & \cdots \\
 & + & + & + & + & + & + & + & + & + & + & + & + & \\
n = & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & \cdots
\end{array}
$$

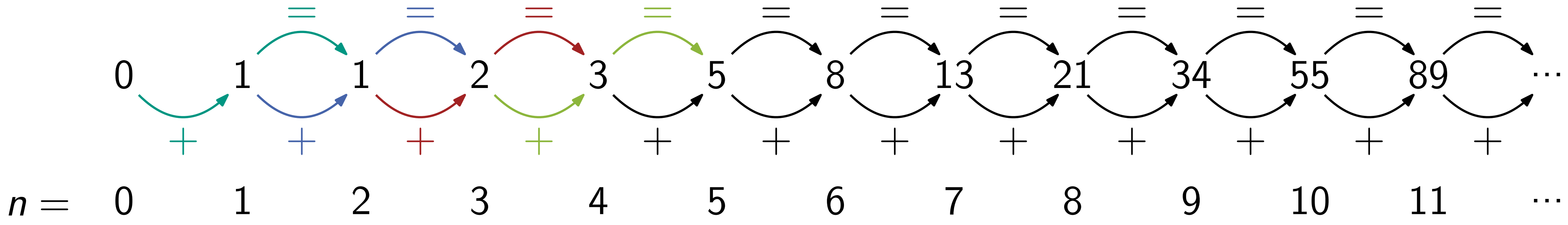
fib(n)

fib := 0
fibNext := 1
for i from 0 to $n-1$ **do**
  temp := fib
  fib := fibNext
  fibNext := fib + temp
return fib

fib(0)

0

Fibonacci – Schneller



fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

0
1

Fibonacci – Schneller

	0	1	1	2	3	5	8	13	21	34	55	89	$\cdots$
$n =$	0	1	2	3	4	5	6	7	8	9	10	11	$\cdots$

fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

0
1
0

Fibonacci – Schneller

	0	1	1	2	3	5	8	13	21	34	55	89	$\cdots$
$n =$	0	1	2	3	4	5	6	7	8	9	10	11	$\cdots$

fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

0
1
0 **-1**

Fibonacci – Schneller

0 1 1 2 3 5 8 13 21 34 55 89 ⋯

$n =$ 0 1 2 3 4 5 6 7 8 9 10 11 ⋯

fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

0
1
0 **-1**

Fibonacci – Schneller

	0	1	1	2	3	5	8	13	21	34	55	89	$\cdots$
$n =$	0	1	2	3	4	5	6	7	8	9	10	11	$\cdots$

fib(n)

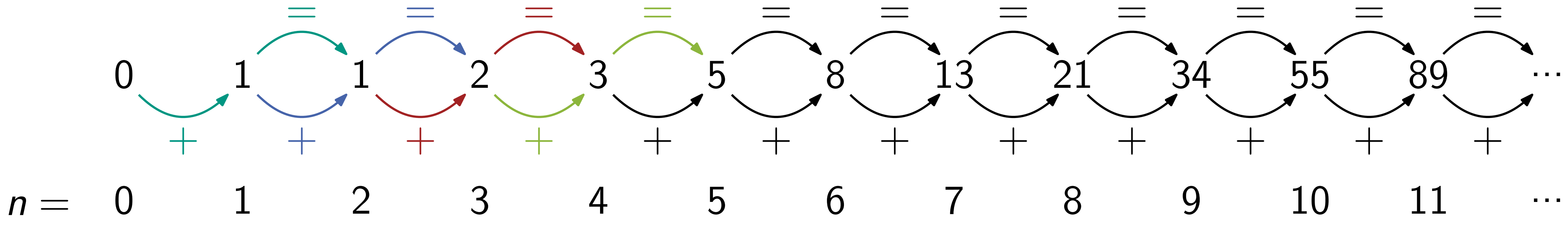
```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

0
1
0 **-1**

fib(1)

Fibonacci – Schneller



fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

0	
1	
0	**-1**

fib(1)

0
1
0

Fibonacci – Schneller

	0	1	1	2	3	5	8	13	21	34	55	89	…
$n =$	0	1	2	3	4	5	6	7	8	9	10	11	…

fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

0	
1	
0	**-1**

fib(1)

0	
1	
0	**0**

Fibonacci – Schneller

0 1 1 2 3 5 8 13 21 34 55 89 ⋯

$n =$ 0 1 2 3 4 5 6 7 8 9 10 11 ⋯

fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

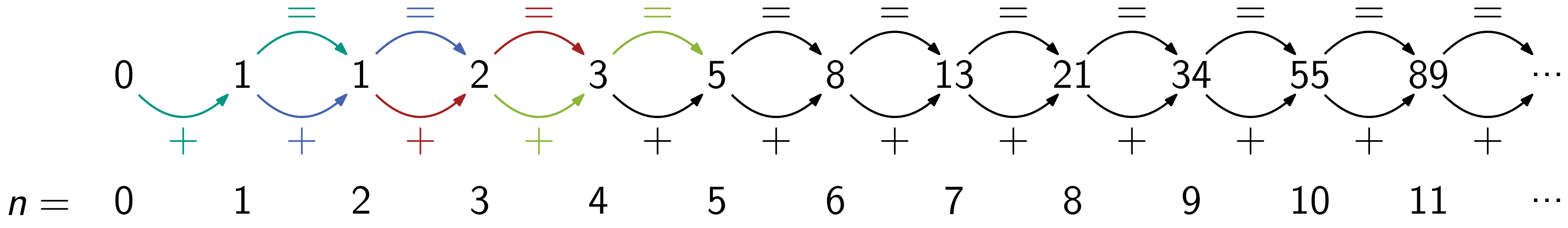
fib(0)

0	
1	
0	**-1**

fib(1)

0	
1	
0	**0**
0	

Fibonacci – Schneller



fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

0
1
0 **-1**

fib(1)

0
1
0 **0**
0
1

Fibonacci – Schneller

	0	1	1	2	3	5	8	13	21	34	55	89	⋯
$n =$	0	1	2	3	4	5	6	7	8	9	10	11	⋯

fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

0
1
0 **-1**

fib(1)

0
1
0 **0**
0
1
1

Fibonacci – Schneller

	0	1	1	2	3	5	8	13	21	34	55	89	···
$n =$	0	1	2	3	4	5	6	7	8	9	10	11	···

fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

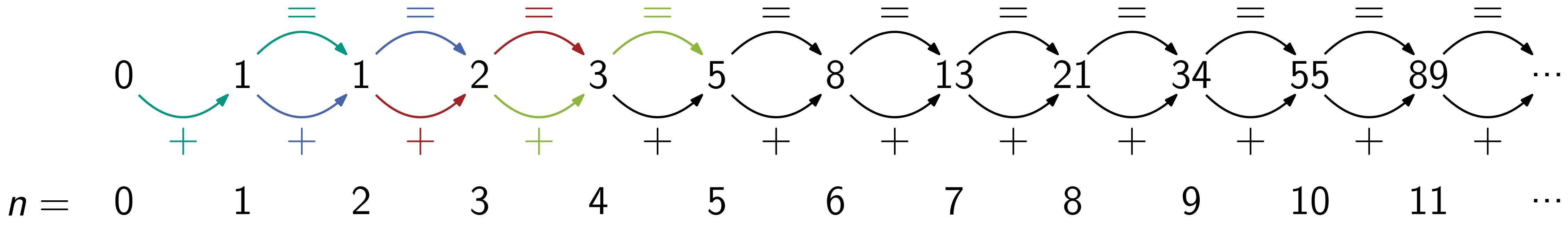
fib(0)

0
1
0 -1

fib(1)

0 1
1 1
0 0
0
1
1

Fibonacci – Schneller



fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

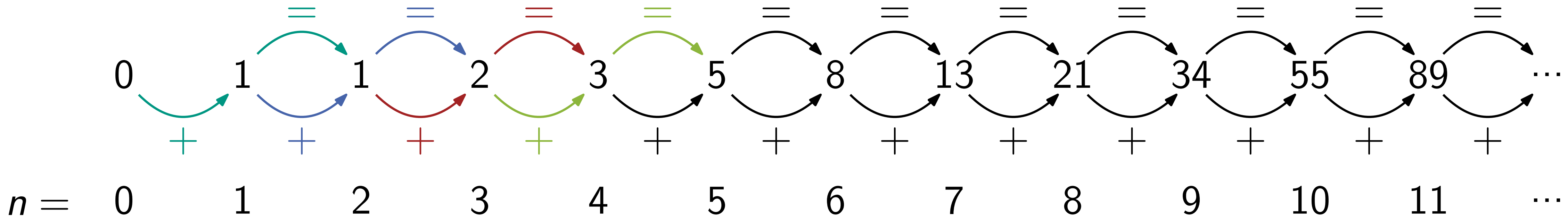
fib(0)

0
1
0 -1

fib(1)

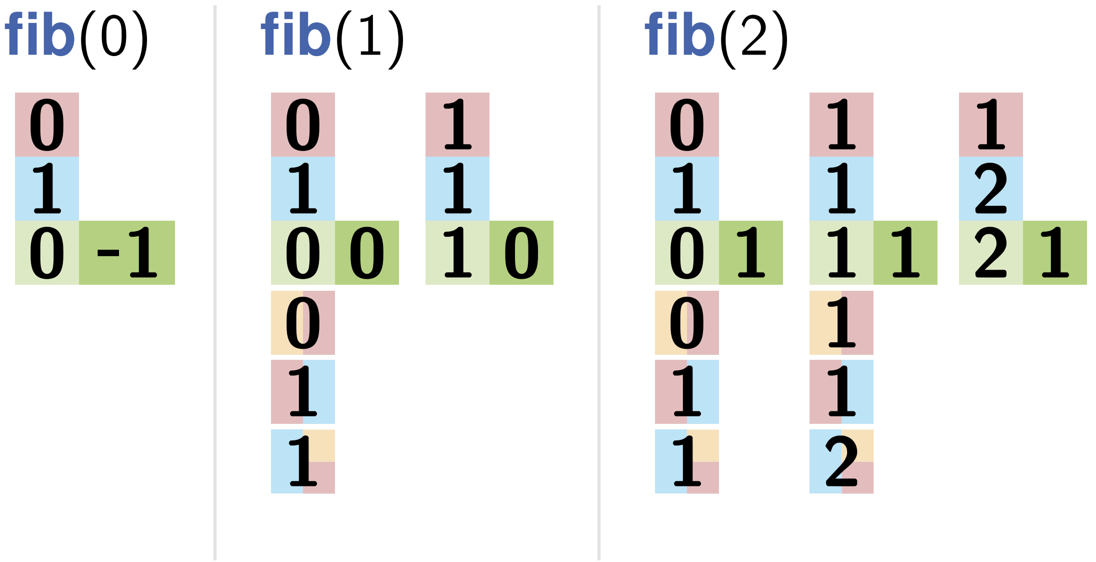
0 1
1 1
0 0 1 0
0
1
1

Fibonacci – Schneller

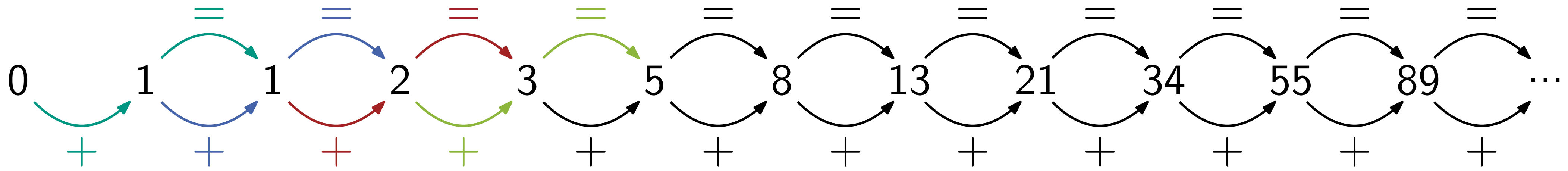


fib(n)

```
fib := 0
fibNext := 1
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```



Fibonacci – Schneller



$n =$ 0 1 2 3 4 5 6 7 8 9 10 11 ···

fib(n)

```
fib := 0                          // O(1)
fibNext := 1                      // O(1)
for i from 0 to n − 1 do
    temp := fib
    fib := fibNext
    fibNext := fib + temp
return fib
```

fib(0)

0	
1	
0	-1

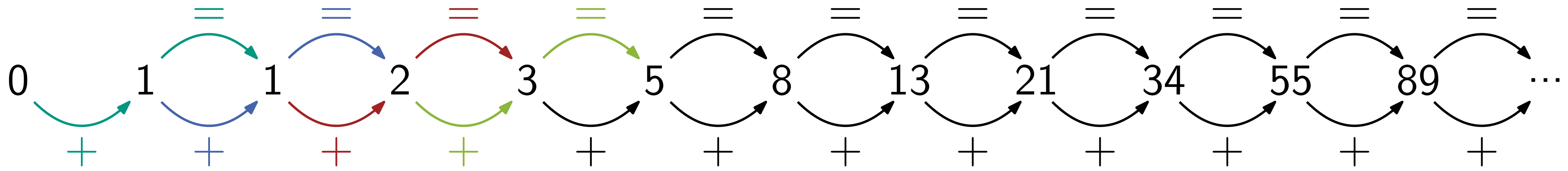
fib(1)

0		1	
1		1	
0	0	1	0
0			
1			
1			

fib(2)

0		1		1	
1		1		2	
0	1	1	1	2	1
0		1			
1		1			
1		2			

Fibonacci – Schneller



$n =$ 0 1 2 3 4 5 6 7 8 9 10 11 ⋯

fib(n)

```
fib := 0                          // O(1)
fibNext := 1                      // O(1)
for i from 0 to n − 1 do
    temp := fib                   // O(1)
    fib := fibNext                // O(1)
    fibNext := fib + temp         // O(1)
return fib
```

fib(0)

0
1
0 -1

fib(1)

0 1
1 1
0 0 1 0
0
1
1

fib(2)

0 1 1
1 1 2
0 1 1 1 2 1
0 1
1 1
1 2

Fibonacci – Schneller

	0	1	1	2	3	5	8	13	21	34	55	89	⋯
$n =$	0	1	2	3	4	5	6	7	8	9	10	11	⋯

fib(n)

```
fib := 0                         // O(1)
fibNext := 1                     // O(1)
for i from 0 to n − 1 do         // O(n)
    temp := fib                  // O(1)
    fib := fibNext               // O(1)
    fibNext := fib + temp        // O(1)
return fib
```

fib(0)

0
1
0 -1

fib(1)

0	1
1	1
0 0	1 0
0	
1	
1	

fib(2)

0	1	1
1	1	2
0 1	1 1	2 1
0	1	
1	1	
1	2	

Fibonacci – Schneller

$n =$	0	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
	0	1	1	2	3	5	8	13	21	34	55	89	$\cdots$

fib(n)

```
fib := 0                          // O(1)
fibNext := 1                      // O(1)
for i from 0 to n − 1 do          // O(n)
    temp := fib                   // O(1)
    fib := fibNext                // O(1)
    fibNext := fib + temp         // O(1)
return fib                        // O(n)
```

fib(0)

0	
1	
0	-1

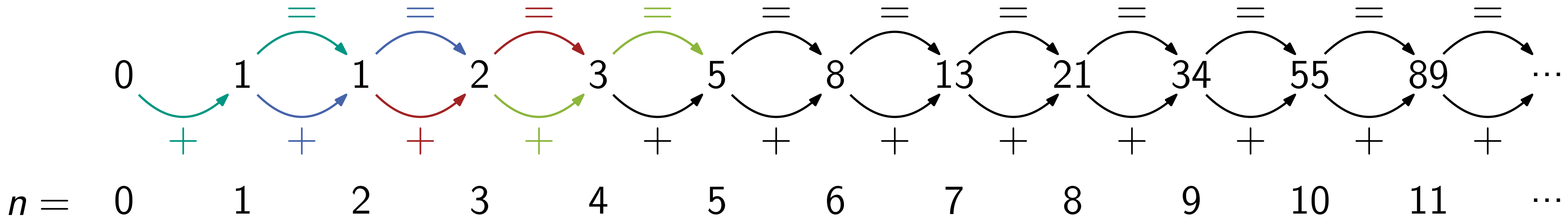
fib(1)

0		1	
1		1	
0	0	1	0
0			
1			
1			

fib(2)

0		1		1	
1		1		2	
0	1	1	1	2	1
0		1			
1		1			
1		2			

Fibonacci – Schneller



fib(n)

```
fib := 0                        // O(1)
fibNext := 1                    // O(1)
for i from 0 to n − 1 do        // O(n)
    temp := fib                 // O(1)
    fib := fibNext              // O(1)
    fibNext := fib + temp       // O(1)
return fib                      // O(n)
```

fib(0)

0
1
0 -1

fib(1)

0 1
1 1
0 0 1 0
0
1
1

fib(2)

0 1 1
1 1 2
0 1 1 1 2 1
0 1
1 1
1 2

- Fibonacci rekursiv: exponentielle Laufzeit. Fibonacci iterativ: lineare Laufzeit.