

## 5 Asymptotic Notation

We are usually not interested in exact running times, but only in an asymptotic classification of the running time, that ignores constant factors and constant additive offsets.

## 5 Asymptotic Notation

We are usually not interested in exact running times, but only in an asymptotic classification of the running time, that ignores constant factors and constant additive offsets.

- ▶ We are usually interested in the running times for large values of  $n$ . Then constant additive terms do not play an important role.

## 5 Asymptotic Notation

We are usually not interested in exact running times, but only in an asymptotic classification of the running time, that ignores constant factors and constant additive offsets.

- ▶ We are usually interested in the running times for large values of  $n$ . Then constant additive terms do not play an important role.
- ▶ An exact analysis (e.g. *exactly* counting the number of operations in a RAM) may be hard, but wouldn't lead to more precise results as the computational model is already quite a distance from reality.

## 5 Asymptotic Notation

We are usually not interested in exact running times, but only in an asymptotic classification of the running time, that ignores constant factors and constant additive offsets.

- ▶ We are usually interested in the running times for large values of  $n$ . Then constant additive terms do not play an important role.
- ▶ An exact analysis (e.g. *exactly* counting the number of operations in a RAM) may be hard, but wouldn't lead to more precise results as the computational model is already quite a distance from reality.
- ▶ A linear speed-up (i.e., by a constant factor) is always possible by e.g. implementing the algorithm on a faster machine.

## 5 Asymptotic Notation

We are usually not interested in exact running times, but only in an asymptotic classification of the running time, that ignores constant factors and constant additive offsets.

- ▶ We are usually interested in the running times for large values of  $n$ . Then constant additive terms do not play an important role.
- ▶ An exact analysis (e.g. *exactly* counting the number of operations in a RAM) may be hard, but wouldn't lead to more precise results as the computational model is already quite a distance from reality.
- ▶ A linear speed-up (i.e., by a constant factor) is always possible by e.g. implementing the algorithm on a faster machine.
- ▶ Running time should be expressed by simple functions.

# Asymptotic Notation

## Formal Definition

Let  $f, g$  denote functions from  $\mathbb{N}$  to  $\mathbb{R}^+$ .

- ▶  $\mathcal{O}(f) = \{g \mid \exists c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \leq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **not faster** than  $f$ )

# Asymptotic Notation

## Formal Definition

Let  $f, g$  denote functions from  $\mathbb{N}$  to  $\mathbb{R}^+$ .

- ▶  $\mathcal{O}(f) = \{g \mid \exists c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \leq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **not faster** than  $f$ )
- ▶  $\Omega(f) = \{g \mid \exists c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \geq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **not slower** than  $f$ )

# Asymptotic Notation

## Formal Definition

Let  $f, g$  denote functions from  $\mathbb{N}$  to  $\mathbb{R}^+$ .

- ▶  $\mathcal{O}(f) = \{g \mid \exists c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \leq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **not faster** than  $f$ )
- ▶  $\Omega(f) = \{g \mid \exists c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \geq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **not slower** than  $f$ )
- ▶  $\Theta(f) = \Omega(f) \cap \mathcal{O}(f)$   
(functions that asymptotically have **the same growth** as  $f$ )



# Asymptotic Notation

## Formal Definition

Let  $f, g$  denote functions from  $\mathbb{N}$  to  $\mathbb{R}^+$ .

- ▶  $\mathcal{O}(f) = \{g \mid \exists c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \leq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **not faster** than  $f$ )
- ▶  $\Omega(f) = \{g \mid \exists c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \geq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **not slower** than  $f$ )
- ▶  $\Theta(f) = \Omega(f) \cap \mathcal{O}(f)$   
(functions that asymptotically have **the same growth** as  $f$ )
- ▶  $o(f) = \{g \mid \forall c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \leq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **slower** than  $f$ )

# Asymptotic Notation

## Formal Definition

Let  $f, g$  denote functions from  $\mathbb{N}$  to  $\mathbb{R}^+$ .

- ▶  $\mathcal{O}(f) = \{g \mid \exists c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \leq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **not faster** than  $f$ )
- ▶  $\Omega(f) = \{g \mid \exists c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \geq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **not slower** than  $f$ )
- ▶  $\Theta(f) = \Omega(f) \cap \mathcal{O}(f)$   
(functions that asymptotically have **the same growth** as  $f$ )
- ▶  $o(f) = \{g \mid \forall c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \leq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **slower** than  $f$ )
- ▶  $\omega(f) = \{g \mid \forall c > 0 \exists n_0 \in \mathbb{N}_0 \forall n \geq n_0 : [g(n) \geq c \cdot f(n)]\}$   
(set of functions that asymptotically grow **faster** than  $f$ )

# Asymptotic Notation

There is an equivalent definition using limes notation (**assuming that the respective limes exists**).  $f$  and  $g$  are functions from  $\mathbb{N}_0$  to  $\mathbb{R}_0^+$ .

$$\blacktriangleright g \in \mathcal{O}(f): 0 \leq \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} < \infty$$

# Asymptotic Notation

There is an equivalent definition using limes notation (**assuming that the respective limes exists**).  $f$  and  $g$  are functions from  $\mathbb{N}_0$  to  $\mathbb{R}_0^+$ .

$$\blacktriangleright g \in \mathcal{O}(f): 0 \leq \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} < \infty$$

$$\blacktriangleright g \in \Omega(f): 0 < \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} \leq \infty$$

# Asymptotic Notation

There is an equivalent definition using limes notation (**assuming that the respective limes exists**).  $f$  and  $g$  are functions from  $\mathbb{N}_0$  to  $\mathbb{R}_0^+$ .

$$\blacktriangleright g \in \mathcal{O}(f): 0 \leq \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} < \infty$$

$$\blacktriangleright g \in \Omega(f): 0 < \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} \leq \infty$$

$$\blacktriangleright g \in \Theta(f): 0 < \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} < \infty$$

# Asymptotic Notation

There is an equivalent definition using limes notation (**assuming that the respective limes exists**).  $f$  and  $g$  are functions from  $\mathbb{N}_0$  to  $\mathbb{R}_0^+$ .

$$\blacktriangleright g \in \mathcal{O}(f): 0 \leq \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} < \infty$$

$$\blacktriangleright g \in \Omega(f): 0 < \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} \leq \infty$$

$$\blacktriangleright g \in \Theta(f): 0 < \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} < \infty$$

$$\blacktriangleright g \in o(f): \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = 0$$

# Asymptotic Notation

There is an equivalent definition using limes notation (**assuming that the respective limes exists**).  $f$  and  $g$  are functions from  $\mathbb{N}_0$  to  $\mathbb{R}_0^+$ .

$$\blacktriangleright g \in \mathcal{O}(f): 0 \leq \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} < \infty$$

$$\blacktriangleright g \in \Omega(f): 0 < \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} \leq \infty$$

$$\blacktriangleright g \in \Theta(f): 0 < \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} < \infty$$

$$\blacktriangleright g \in o(f): \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = 0$$

$$\blacktriangleright g \in \omega(f): \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = \infty$$

# Asymptotic Notation

## Abuse of notation

1. People write  $f = \mathcal{O}(g)$ , when they mean  $f \in \mathcal{O}(g)$ . This is **not** an equality (how could a function be equal to a set of functions).



# Asymptotic Notation

## Abuse of notation

1. People write  $f = \mathcal{O}(g)$ , when they mean  $f \in \mathcal{O}(g)$ . This is **not** an equality (how could a function be equal to a set of functions).
2. People write  $f(n) = \mathcal{O}(g(n))$ , when they mean  $f \in \mathcal{O}(g)$ , with  $f : \mathbb{N} \rightarrow \mathbb{R}^+, n \mapsto f(n)$ , and  $g : \mathbb{N} \rightarrow \mathbb{R}^+, n \mapsto g(n)$ .

# Asymptotic Notation

## Abuse of notation

1. People write  $f = \mathcal{O}(g)$ , when they mean  $f \in \mathcal{O}(g)$ . This is **not** an equality (how could a function be equal to a set of functions).
2. People write  $f(n) = \mathcal{O}(g(n))$ , when they mean  $f \in \mathcal{O}(g)$ , with  $f : \mathbb{N} \rightarrow \mathbb{R}^+, n \mapsto f(n)$ , and  $g : \mathbb{N} \rightarrow \mathbb{R}^+, n \mapsto g(n)$ .
3. People write e.g.  $h(n) = f(n) + o(g(n))$  when they mean that there exists a function  $z : \mathbb{N} \rightarrow \mathbb{R}^+, n \mapsto z(n), z \in o(g)$  such that  $h(n) = f(n) + z(n)$ .

# Asymptotic Notation

## Abuse of notation

1. People write  $f = \mathcal{O}(g)$ , when they mean  $f \in \mathcal{O}(g)$ . This is **not** an equality (how could a function be equal to a set of functions).
2. People write  $f(n) = \mathcal{O}(g(n))$ , when they mean  $f \in \mathcal{O}(g)$ , with  $f : \mathbb{N} \rightarrow \mathbb{R}^+, n \mapsto f(n)$ , and  $g : \mathbb{N} \rightarrow \mathbb{R}^+, n \mapsto g(n)$ .
3. People write e.g.  $h(n) = f(n) + o(g(n))$  when they mean that there exists a function  $z : \mathbb{N} \rightarrow \mathbb{R}^+, n \mapsto z(n), z \in o(g)$  such that  $h(n) = f(n) + z(n)$ .
4. People write  $\mathcal{O}(f(n)) = \mathcal{O}(g(n))$ , when they mean  $\mathcal{O}(f(n)) \subseteq \mathcal{O}(g(n))$ . Again this is not an equality.

# Asymptotic Notation in Equations

How do we interpret an expression like:

$$2n^2 + 3n + 1 = 2n^2 + \Theta(n)$$

# Asymptotic Notation in Equations

How do we interpret an expression like:

$$2n^2 + 3n + 1 = 2n^2 + \Theta(n)$$

Here,  $\Theta(n)$  stands for an **anonymous function** in the set  $\Theta(n)$  that makes the expression true.

# Asymptotic Notation in Equations

How do we interpret an expression like:

$$2n^2 + 3n + 1 = 2n^2 + \Theta(n)$$

Here,  $\Theta(n)$  stands for an **anonymous function** in the set  $\Theta(n)$  that makes the expression true.

Note that  $\Theta(n)$  is on the right hand side, otw. this interpretation is wrong.

# Asymptotic Notation in Equations

How do we interpret an expression like:

$$2n^2 + \mathcal{O}(n) = \Theta(n^2)$$

# Asymptotic Notation in Equations

How do we interpret an expression like:

$$2n^2 + \mathcal{O}(n) = \Theta(n^2)$$

Regardless of how we choose the anonymous function  $f(n) \in \mathcal{O}(n)$  there is an anonymous function  $g(n) \in \Theta(n^2)$  that makes the expression true.



# Asymptotic Notation in Equations

How do we interpret an expression like:

$$\sum_{i=1}^n \Theta(i) = \Theta(n^2)$$

# Asymptotic Notation in Equations

How do we interpret an expression like:

$$\sum_{i=1}^n \Theta(i) = \Theta(n^2)$$

**Careful!**

# Asymptotic Notation in Equations

How do we interpret an expression like:

$$\sum_{i=1}^n \Theta(i) = \Theta(n^2)$$

**Careful!**

“It is understood” that every occurrence of an  $\Theta$ -symbol (or  $\Theta, \Omega, o, \omega$ ) on the left represents **one anonymous function**.

Hence, the left side is **not** equal to

$$\Theta(1) + \Theta(2) + \dots + \Theta(n-1) + \Theta(n)$$

# Asymptotic Notation in Equations

We can view an expression containing asymptotic notation as generating a set:

$$n^2 \cdot \mathcal{O}(n) + \mathcal{O}(\log n)$$

represents

$$\{f : \mathbb{N} \rightarrow \mathbb{R}^+ \mid f(n) = n^2 \cdot g(n) + h(n)$$

$$\text{with } g(n) \in \mathcal{O}(n) \text{ and } h(n) \in \mathcal{O}(\log n)\}$$

# Asymptotic Notation in Equations

Then an asymptotic equation can be interpreted as containment btw. two sets:

$$n^2 \cdot \mathcal{O}(n) + \mathcal{O}(\log n) = \Theta(n^2)$$

represents

$$n^2 \cdot \mathcal{O}(n) + \mathcal{O}(\log n) \subseteq \Theta(n^2)$$

# Asymptotic Notation

## Lemma 1

Let  $f, g$  be functions with the property

$\exists n_0 > 0 \forall n \geq n_0 : f(n) > 0$  (the same for  $g$ ). Then

- ▶  $c \cdot f(n) \in \Theta(f(n))$  for any constant  $c$

# Asymptotic Notation

## Lemma 1

Let  $f, g$  be functions with the property

$\exists n_0 > 0 \forall n \geq n_0 : f(n) > 0$  (the same for  $g$ ). Then

- ▶  $c \cdot f(n) \in \Theta(f(n))$  for any constant  $c$
- ▶  $\mathcal{O}(f(n)) + \mathcal{O}(g(n)) = \mathcal{O}(f(n) + g(n))$

# Asymptotic Notation

## Lemma 1

Let  $f, g$  be functions with the property

$\exists n_0 > 0 \forall n \geq n_0 : f(n) > 0$  (the same for  $g$ ). Then

- ▶  $c \cdot f(n) \in \Theta(f(n))$  for any constant  $c$
- ▶  $\mathcal{O}(f(n)) + \mathcal{O}(g(n)) = \mathcal{O}(f(n) + g(n))$
- ▶  $\mathcal{O}(f(n)) \cdot \mathcal{O}(g(n)) = \mathcal{O}(f(n) \cdot g(n))$



# Asymptotic Notation

## Lemma 1

Let  $f, g$  be functions with the property

$\exists n_0 > 0 \forall n \geq n_0 : f(n) > 0$  (the same for  $g$ ). Then

- ▶  $c \cdot f(n) \in \Theta(f(n))$  for any constant  $c$
- ▶  $\mathcal{O}(f(n)) + \mathcal{O}(g(n)) = \mathcal{O}(f(n) + g(n))$
- ▶  $\mathcal{O}(f(n)) \cdot \mathcal{O}(g(n)) = \mathcal{O}(f(n) \cdot g(n))$
- ▶  $\mathcal{O}(f(n)) + \mathcal{O}(g(n)) = \mathcal{O}(\max\{f(n), g(n)\})$

# Asymptotic Notation

## Lemma 1

Let  $f, g$  be functions with the property

$\exists n_0 > 0 \forall n \geq n_0 : f(n) > 0$  (the same for  $g$ ). Then

- ▶  $c \cdot f(n) \in \Theta(f(n))$  for any constant  $c$
- ▶  $\mathcal{O}(f(n)) + \mathcal{O}(g(n)) = \mathcal{O}(f(n) + g(n))$
- ▶  $\mathcal{O}(f(n)) \cdot \mathcal{O}(g(n)) = \mathcal{O}(f(n) \cdot g(n))$
- ▶  $\mathcal{O}(f(n)) + \mathcal{O}(g(n)) = \mathcal{O}(\max\{f(n), g(n)\})$

The expressions also hold for  $\Omega$ . Note that this means that  $f(n) + g(n) \in \Theta(\max\{f(n), g(n)\})$ .

# Asymptotic Notation

## Comments

- ▶ Do not use asymptotic notation within induction proofs.

# Asymptotic Notation

## Comments

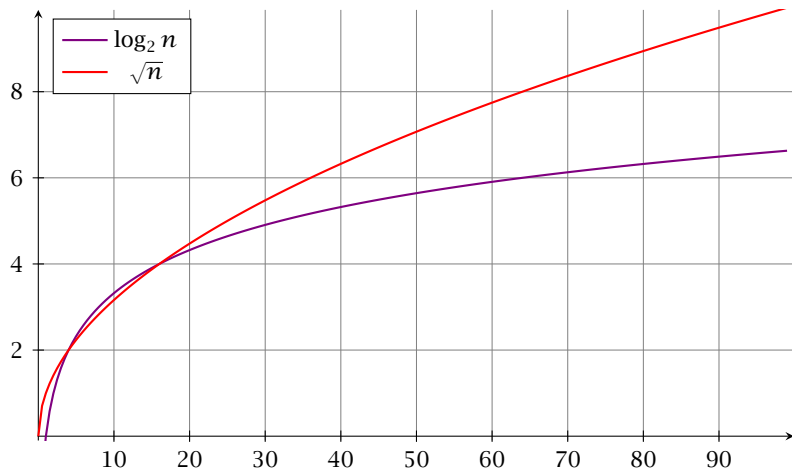
- ▶ Do not use asymptotic notation within induction proofs.
- ▶ For any constants  $a, b$  we have  $\log_a n = \Theta(\log_b n)$ .  
Therefore, we will usually ignore the base of a logarithm within asymptotic notation.

# Asymptotic Notation

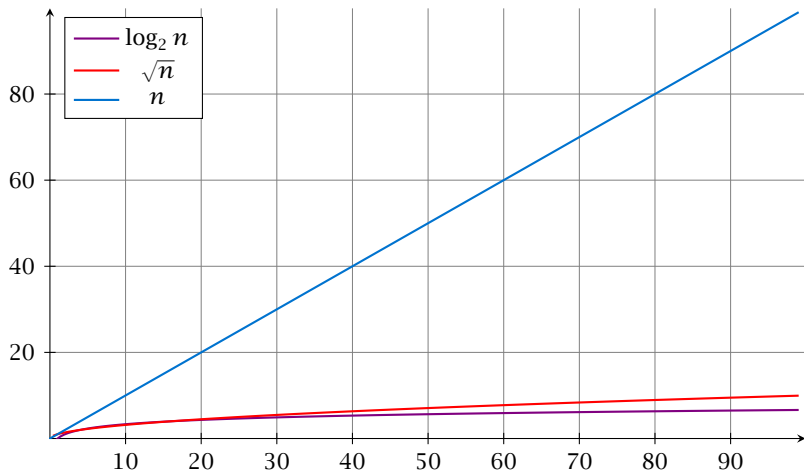
## Comments

- ▶ Do not use asymptotic notation within induction proofs.
- ▶ For any constants  $a, b$  we have  $\log_a n = \Theta(\log_b n)$ .  
Therefore, we will usually ignore the base of a logarithm within asymptotic notation.
- ▶ In general  $\log n = \log_2 n$ , i.e., we use 2 as the default base for the logarithm.

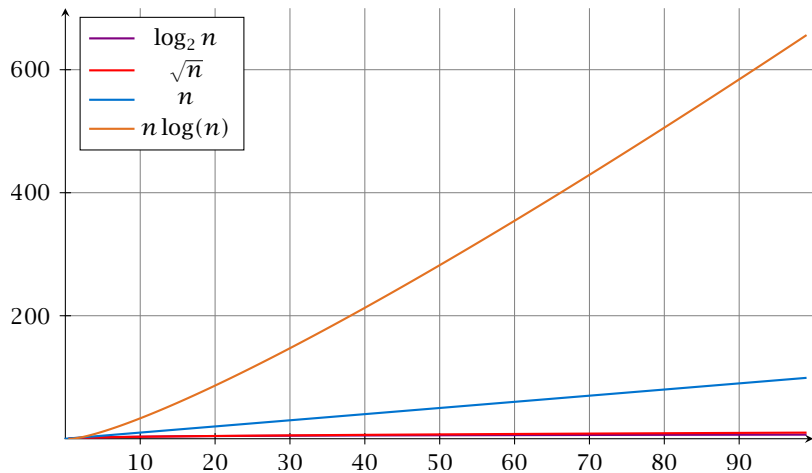
# Funktionen



# Funktionen

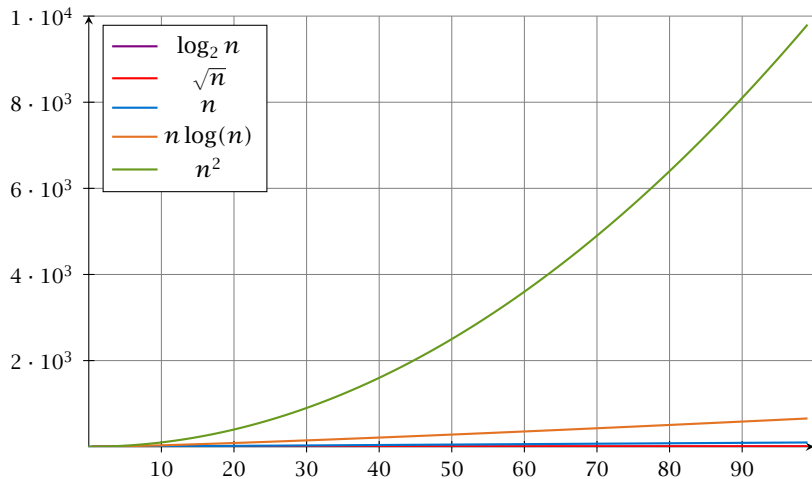


# Funktionen

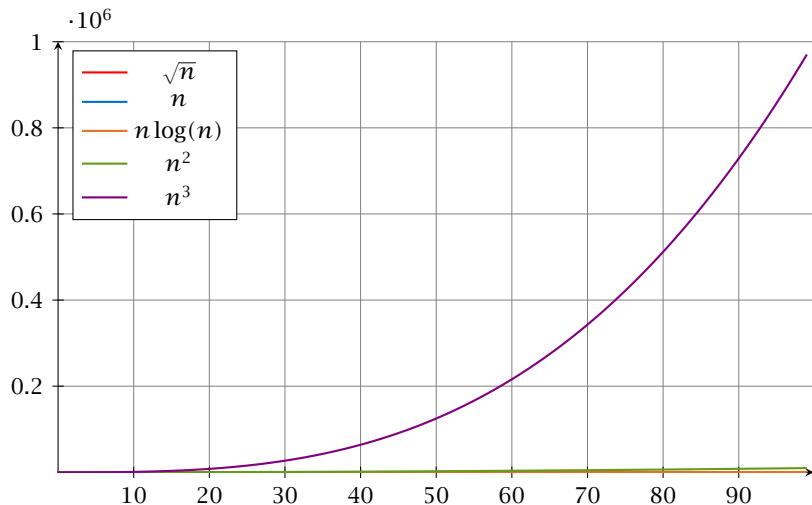




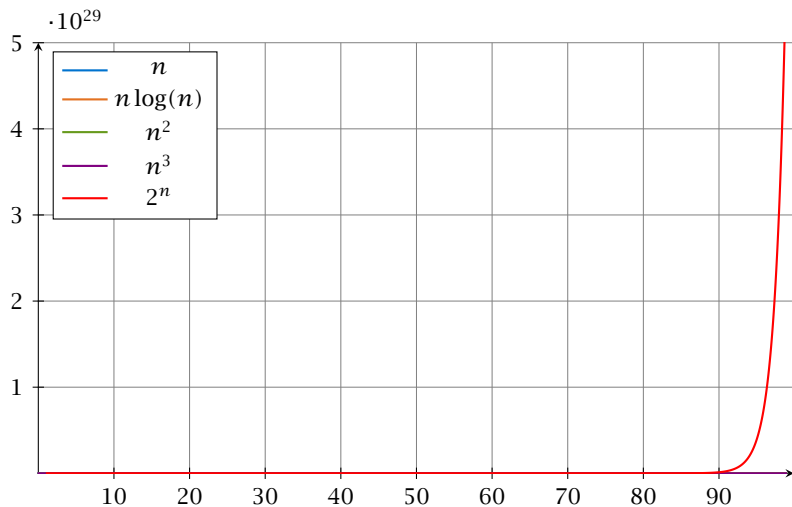
# Funktionen



# Funktionen



# Funktionen



# Laufzeiten

Funktion	Eingabelänge $n$							
	10	$10^2$	$10^3$	$10^4$	$10^5$	$10^6$	$10^7$	$10^8$
$\log n$	33ns	66ns	0.1 $\mu$ s	0.1 $\mu$ s	0.2 $\mu$ s	0.2 $\mu$ s	0.2 $\mu$ s	0.3 $\mu$ s
$\sqrt{n}$	32ns	0.1 $\mu$ s	0.3 $\mu$ s	1 $\mu$ s	3.1 $\mu$ s	10 $\mu$ s	31 $\mu$ s	0.1ms
$n$	100ns	1 $\mu$ s	10 $\mu$ s	0.1ms	1ms	10ms	0.1s	1s
$n \log n$	0.3 $\mu$ s	6.6 $\mu$ s	0.1ms	1.3ms	16ms	0.2s	2.3s	27s
$n^{3/2}$	0.3 $\mu$ s	10 $\mu$ s	0.3ms	10ms	0.3s	10s	5.2min	2.7h
$n^2$	1 $\mu$ s	0.1ms	10ms	1s	1.7min	2.8h	11d	3.2y
$n^3$	10 $\mu$ s	10ms	10s	2.8h	115d	317y	$3.2 \cdot 10^5$ y	
$1.1^n$	26ns	0.1ms	$7.8 \cdot 10^{25}$ y					
$2^n$	10 $\mu$ s	$4 \cdot 10^{14}$ y						
$n!$	36ms	$3 \cdot 10^{142}$ y						

1 Operation = 10ns; 100MHz

Alter des Universums: ca.  $13.8 \cdot 10^9$ y

# Asymptotic Notation

In general asymptotic classification of running times is a good measure for comparing algorithms:

- ▶ If the running time analysis is tight and actually occurs in practise (i.e., the asymptotic bound is not a purely theoretical worst-case bound), then the algorithm that has better asymptotic running time will always outperform a weaker algorithm for large enough values of  $n$ .

# Asymptotic Notation

In general asymptotic classification of running times is a good measure for comparing algorithms:

- ▶ If the running time analysis is tight and actually occurs in practise (i.e., the asymptotic bound is not a purely theoretical worst-case bound), then the algorithm that has better asymptotic running time will always outperform a weaker algorithm for large enough values of  $n$ .
- ▶ However, suppose that I have two algorithms:

# Asymptotic Notation

In general asymptotic classification of running times is a good measure for comparing algorithms:

- ▶ If the running time analysis is tight and actually occurs in practise (i.e., the asymptotic bound is not a purely theoretical worst-case bound), then the algorithm that has better asymptotic running time will always outperform a weaker algorithm for large enough values of  $n$ .
- ▶ However, suppose that I have two algorithms:
  - ▶ Algorithm A. Running time  $f(n) = 1000 \log n = \mathcal{O}(\log n)$ .

# Asymptotic Notation

In general asymptotic classification of running times is a good measure for comparing algorithms:

- ▶ If the running time analysis is tight and actually occurs in practise (i.e., the asymptotic bound is not a purely theoretical worst-case bound), then the algorithm that has better asymptotic running time will always outperform a weaker algorithm for large enough values of  $n$ .
- ▶ However, suppose that I have two algorithms:
  - ▶ Algorithm A. Running time  $f(n) = 1000 \log n = \mathcal{O}(\log n)$ .
  - ▶ Algorithm B. Running time  $g(n) = \log^2 n$ .



# Asymptotic Notation

In general asymptotic classification of running times is a good measure for comparing algorithms:

- ▶ If the running time analysis is tight and actually occurs in practise (i.e., the asymptotic bound is not a purely theoretical worst-case bound), then the algorithm that has better asymptotic running time will always outperform a weaker algorithm for large enough values of  $n$ .
- ▶ However, suppose that I have two algorithms:
  - ▶ Algorithm A. Running time  $f(n) = 1000 \log n = \mathcal{O}(\log n)$ .
  - ▶ Algorithm B. Running time  $g(n) = \log^2 n$ .

Clearly  $f = o(g)$ . However, as long as  $\log n \leq 1000$  Algorithm B will be more efficient.

# Multiple Variables in Asymptotic Notation

Sometimes the input for an algorithm consists of several parameters (e.g., nodes and edges of a graph ( $n$  and  $m$ )).

# Multiple Variables in Asymptotic Notation

Sometimes the input for an algorithm consists of several parameters (e.g., nodes and edges of a graph ( $n$  and  $m$ )).

If we want to make asymptotic statements for  $n \rightarrow \infty$  and  $m \rightarrow \infty$  we have to extend the definition to multiple variables.

# Multiple Variables in Asymptotic Notation

Sometimes the input for an algorithm consists of several parameters (e.g., nodes and edges of a graph ( $n$  and  $m$ )).

If we want to make asymptotic statements for  $n \rightarrow \infty$  and  $m \rightarrow \infty$  we have to extend the definition to multiple variables.

## Formal Definition

Let  $f, g$  denote functions from  $\mathbb{N}^d$  to  $\mathbb{R}_0^+$ .

$$\blacktriangleright \mathcal{O}(f) = \{g \mid \exists c > 0 \exists N \in \mathbb{N}_0 \forall \vec{n} \text{ with } n_i \geq N \text{ for some } i : [g(\vec{n}) \leq c \cdot f(\vec{n})]\}$$

(set of functions that asymptotically grow **not faster** than  $f$ )

# Multiple Variables in Asymptotic Notation

## Example 2

►  $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n - 1$

# Multiple Variables in Asymptotic Notation

## Example 2

- ▶  $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n - 1$   
then  $f = \mathcal{O}(g)$  does not hold

# Multiple Variables in Asymptotic Notation

## Example 2

- ▶  $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n - 1$   
then  $f = \mathcal{O}(g)$  does not hold
- ▶  $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n$

# Multiple Variables in Asymptotic Notation

## Example 2

- ▶  $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n - 1$   
then  $f = \mathcal{O}(g)$  does not hold
- ▶  $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n$   
then:  $f = \mathcal{O}(g)$



# Multiple Variables in Asymptotic Notation

## Example 2

- ▶  $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n - 1$   
then  $f = \mathcal{O}(g)$  does not hold
- ▶  $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n$   
then:  $f = \mathcal{O}(g)$
- ▶  $f : \mathbb{N}_0 \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N}_0 \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n$

# Multiple Variables in Asymptotic Notation

## Example 2

- ▶  $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n - 1$   
then  $f = \mathcal{O}(g)$  does not hold
- ▶  $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N} \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n$   
then:  $f = \mathcal{O}(g)$
- ▶  $f : \mathbb{N}_0 \rightarrow \mathbb{R}_0^+$ ,  $f(n, m) = 1$  und  $g : \mathbb{N}_0 \rightarrow \mathbb{R}_0^+$ ,  $g(n, m) = n$   
then  $f = \mathcal{O}(g)$  does not hold