Albert-Ludwigs-Universität Freiburg

#### Prof. Dr. Rolf Backofen

Bioinformatics Group / Department of Computer Science Algorithmns and Datastructures, November 2017

## Structure



## *O*-Notation

Motivation / Definition Examples

#### **Ω-Notation**

#### Θ-Notation

#### Runtime

Summary Limit / Convergence L'Hôpital / l'Hospital Practical use

## Structure



## *O*-Notation

Motivation / Definition Examples

**Ω-Notation** 

Θ-Notation

#### Runtime

Summary
Limit / Convergence
L'Hôpital / l'Hospital
Practical use

Motivation

#### We are interested in:

- Example: sorting
  - Runtime of Minsort "is growing as"  $n^2$
  - Runtime of HeapSort "is growing as"  $n \log n$

# Motivation

### We are interested in:

- Example: sorting
  - Runtime of Minsort "is growing as"  $n^2$
  - Runtime of HeapSort "is growing as"  $n \log n$
- $\blacksquare$  Growth of a function in runtime T(n)
  - The role of constants (e.g. 1ns) is minor
  - it is enough if relation holds for some  $n \ge ...$

Motivation

#### We are interested in:

- Example: sorting
  - Runtime of Minsort "is growing as"  $n^2$
  - Runtime of HeapSort "is growing as"  $n \log n$
- $\blacksquare$  Growth of a function in runtime T(n)
  - The role of constants (e.g. 1ns) is minor
  - it is enough if relation holds for some  $n \ge ...$
- Describe the growth of the function more formally
  - By the means of Landau-Symbols [Wik]):
    - $\blacksquare$   $\mathcal{O}(n)$  (Big O of n),
    - $\square$   $\Omega(n)$  (Omega of n),
    - $\blacksquare$   $\Theta(n)$  (Theta of n)



- Consider the function:  $f: \mathbb{N} \to \mathbb{R}, n \mapsto f(n)$ 
  - $\blacksquare$  N: Natural numbers  $\rightarrow$  input size
  - $\blacksquare$   $\mathbb{R}$ : Real numbers  $\rightarrow$  runtime

- Consider the function:  $f: \mathbb{N} \to \mathbb{R}, n \mapsto f(n)$ 
  - $\blacksquare$  N: Natural numbers  $\rightarrow$  input size
  - $\blacksquare$   $\mathbb{R}$ : Real numbers  $\rightarrow$  runtime

## Example:

- f(n) = 3n
- $f(n) = 2n \log n$
- $\overrightarrow{f(n)} = \frac{1}{10}n^2$

- Consider the function:  $f: \mathbb{N} \to \mathbb{R}, n \mapsto f(n)$ 
  - $\blacksquare$  N: Natural numbers  $\rightarrow$  input size
  - $\blacksquare$   $\mathbb{R}$ : Real numbers  $\rightarrow$  runtime

## Example:

- $\blacksquare f(n) = 3n$
- $f(n) = 2n \log n$
- $f(n) = \frac{1}{10}n^2$
- $f(n) = n^2 + 3n \log n 4n$



Big  $\mathscr{O}$ -Notation:

 $\blacksquare$  Given two functions f and g:

 $f,g:\mathbb{N}\to\mathbb{R}$ 

## Big $\mathcal{O}$ -Notation:

- $\blacksquare$  Given two functions f and g:
  - $f,g:\mathbb{N}\to\mathbb{R}$
- **Intuitive:** f is Big-O of g (f is  $\mathcal{O}(g)$ )
  - ... if f relative to g does not grow faster than g
  - the growth rate matters, not the absolute values

- Informal:  $f = \mathcal{O}(g)$ 
  - "=" corresponds to is not isequal

# - IBURG



## Big $\mathcal{O}$ -Notation:

- Informal:  $f = \mathcal{O}(g)$ 
  - "=" corresponds to is not isequal
  - lacksquare ... if for some value  $n_0$  for all  $n \geq n_0$

Definition

## Big *∅*-Notation:

- Informal:  $f = \mathcal{O}(g)$ 
  - "=" corresponds to is not isequal
  - ... if for some value  $n_0$  for all  $n \ge n_0$
  - $f(n) \le C \cdot g(n)$  for a constant C

Definition

## Big *⊘*-Notation:

- Informal:  $f = \mathcal{O}(g)$ 
  - "=" corresponds to is not isequal
  - ... if for some value  $n_0$  for all  $n \ge n_0$
  - $f(n) \le C \cdot g(n)$  for a constant C
  - $(f = \mathcal{O}(g))$ : From a value  $n_0$  for all  $n \ge n_0 \to f(n) \le C \cdot g(n)$

- Informal:  $f = \mathcal{O}(g)$ 
  - "=" corresponds to *is* not *isequal*
  - ... if for some value  $n_0$  for all  $n \ge n_0$
  - $f(n) \le C \cdot g(n)$  for a constant C
  - $(f = \mathcal{O}(g))$ : From a value  $n_0$  for all  $n \ge n_0 \to f(n) \le C \cdot g(n)$
- Formal:  $f \in \mathscr{O}(g)$

# Formal: $f \in \mathcal{O}(g)$

$$\mathscr{O}(g) = \{ \ \mathsf{f} \ : \mathbb{N} \to \mathbb{R} \ \mid \ \exists \, n_0 \in \mathbb{N}, \ \exists \, C > 0, \ \forall \, n > n_0 \colon f(n) \leq C \cdot g(n) \}$$

## Definition

## Big *𝑉*-Notation:

- Informal:  $f = \mathcal{O}(g)$ 
  - "=" corresponds to *is* not *isequal*
  - ... if for some value  $n_0$  for all  $n \ge n_0$
  - $f(n) \le C \cdot g(n)$  for a constant C
  - $(f = \mathcal{O}(g))$ : From a value  $n_0$  for all  $n \ge n_0 \to f(n) \le C \cdot g(n)$
- Formal:  $f \in \mathcal{O}(g)$

# Formal: $f \in \mathcal{O}(g)$

$$\mathscr{O}(g) = \{ f : \mathbb{N} \to \mathbb{R} \mid \exists n_0 \in \mathbb{N}, \exists C > 0, \forall n > n_0 : f(n) \leq C \cdot g(n) \}$$

"set of all functions"

Definition

# Big *∅*-Notation:

- Informal:  $f = \mathcal{O}(g)$ 
  - "=" corresponds to is not isequal
  - ... if for some value  $n_0$  for all  $n \ge n_0$
  - $f(n) \le C \cdot g(n)$  for a constant C
  - $(f = \mathcal{O}(g))$ : From a value  $n_0$  for all  $n \ge n_0 \to f(n) \le C \cdot g(n)$
- Formal:  $f \in \mathcal{O}(g)$

# Formal: $f \in \mathscr{O}(g)$

$$\mathscr{O}(g) = \{ f : \mathbb{N} \to \mathbb{R} \mid \exists n_0 \in \mathbb{N}, \exists C > 0, \forall n > n_0 : f(n) \leq C \cdot g(n) \}$$

"set of all functions"

"for which"

- Informal:  $f = \mathcal{O}(g)$ 
  - "=" corresponds to *is* not *isequal*
  - ... if for some value  $n_0$  for all  $n \ge n_0$
  - $f(n) \le C \cdot g(n)$  for a constant C
  - $(f = \mathcal{O}(g))$ : From a value  $n_0$  for all  $n \ge n_0 \to f(n) \le C \cdot g(n)$
- Formal:  $f \in \mathcal{O}(g)$

# Formal: $f \in \mathcal{O}(g)$

$$\mathscr{O}(g) = \{ f : \mathbb{N} \to \mathbb{R} \mid \exists n_0 \in \mathbb{N}, \exists C > 0, \forall n > n_0 \colon f(n) \leq C \cdot g(n) \}$$
"set of "for which" "it exists"

all functions"

- Informal:  $f = \mathcal{O}(g)$ 
  - "=" corresponds to is not isequal
  - ... if for some value  $n_0$  for all  $n \ge n_0$
  - $f(n) \le C \cdot g(n)$  for a constant C
  - $(f = \mathcal{O}(g))$ : From a value  $n_0$  for all  $n \ge n_0 \to f(n) \le C \cdot g(n)$
- Formal:  $f \in \mathcal{O}(g)$

# Formal: $f \in \mathcal{O}(g)$

$$\mathscr{O}(g) = \{ \text{ f } : \mathbb{N} \to \mathbb{R} \mid \exists n_0 \in \mathbb{N}, \ \exists C > 0, \ \forall n > n_0 \colon f(n) \leq C \cdot g(n) \}$$
 "set of "for which" "it exists" "for all" all functions"

- Informal:  $f = \mathcal{O}(g)$ 
  - "=" corresponds to is not isequal
  - ... if for some value  $n_0$  for all  $n \ge n_0$
  - $f(n) \le C \cdot g(n)$  for a constant C
  - $(f = \mathcal{O}(g))$ : From a value  $n_0$  for all  $n \ge n_0 \to f(n) \le C \cdot g(n)$
- Formal:  $f \in \mathcal{O}(g)$

# Formal: $f \in \mathcal{O}(g)$

$$\mathscr{O}(g) = \{ \text{ f } : \mathbb{N} \to \mathbb{R} \mid \exists n_0 \in \mathbb{N}, \exists C > 0, \forall n > n_0 : f(n) \leq C \cdot g(n) \}$$
"set of "for which" "it exists" "for all" "such that" all functions"

## Structure



## *O*-Notation

Motivation / Definition

Examples

**Ω**-Notation

Θ-Notation

#### Runtime

Summary
Limit / Convergence
L'Hôpital / l'Hospital
Practical use

# Illustration of the Big O-Notation:

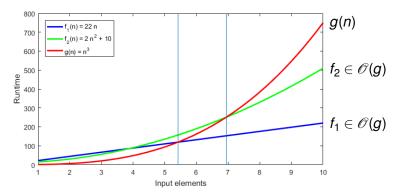


Figure: Runtime of two algorithms  $f_1, f_2$ 

# **Example:**

■ 
$$f(n) = 5n + 7$$
,  $g(n) = n$   
⇒  $5n + 7 \in \mathcal{O}(g)$   
⇒  $f \in \mathcal{O}(g)$ 

#### Intuitive:

$$f(n) = 5n + 7 \rightarrow \text{linear growth}$$

## Attention

 $f(n) \le g(n)$  is not guaranteed, better is  $f(n) \le C \cdot g(n) \ \forall n > n_0$ .



We have to proof:  $\exists n_0, \exists C, \forall n \geq n_0$ :  $5n + 7 \leq C \cdot n$ .

We have to proof:  $\exists n_0, \exists C, \forall n \geq n_0$ :  $5n + 7 \leq C \cdot n$ .

$$5n+7 \leq 5n+n \quad (\text{for } n \geq 7)$$

$$5n+7 \leq 5n+n \quad (\text{for } n \geq 7)$$
  
=  $6n$ 

We have to proof:  $\exists n_0, \exists C, \forall n \geq n_0$ :  $5n + 7 \leq C \cdot n$ .

$$5n+7 \leq 5n+n \text{ (for } n \geq 7)$$
  
=  $6n$ 

$$\Rightarrow$$
  $n_0 = 7$ ,  $C = 6$ 



$$5n+7 \le 5n+7n \text{ (for } n \ge 1)$$

$$5n+7 \le 5n+7n \text{ (for } n \ge 1)$$
  
= 12n

$$5n+7 \le 5n+7n \text{ (for } n \ge 1)$$
  
= 12n

$$\Rightarrow n_0 = 1, C = 12$$

- We are only interested in the term with the highest-order, the fastest growing summand, the others will be ignored
- $\blacksquare$  f(n) is limited from above by  $C \cdot g(n)$

## Examples:

$$2n^{2} + 7n - 20 \in \mathscr{O}(n^{2})$$

$$2n^{2} + 7n \log n - 20 \in$$

$$7n \log n - 20 \in$$

$$5 \in$$

$$2n^{2} + 7n \log n + n^{3} \in$$

## **Harder Example:**

- Polynomes are simple
- More problematic: combination of complex functions

$$2\sqrt{x} + 3\ln x \in \mathscr{O}(\ref{eq:condition})$$

# Structure



#### **O-Notation**

Motivation / Definition Examples

#### **Ω-Notation**

#### Θ-Notation

#### Runtime

Summary Limit / Convergence L'Hôpital / l'Hospital Practical use

## **Omega-Notation:**

- Intuitive:
  - $f \in \Omega(g)$ , f is growing at least as fast as g
  - So the same as Big-O but with at-least and not at-most

# Formal: $f \in \Omega(g)$

$$\Omega(g) = \{f: \mathbb{N} \rightarrow \mathbb{R} \mid \exists n_0 \in \mathbb{N}, \exists C > 0, \forall n > n_0 : f(n) \geq C \cdot g(n)\}$$

# Example:

Proof of 
$$f(n) = 5n + 7 \in \Omega(n)$$
:

$$\underbrace{5n+7}_{f(n)} \geq \underbrace{1 \cdot n}_{g(n)} \quad (\text{for } n \geq 1)$$

$$\Rightarrow$$
  $n_0 = 1$ ,  $C = 1$ 

# Illustration of the Omega-Notation:

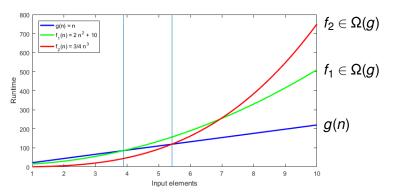


Figure: Runtime of two algorithms  $f_1, f_2$ 

# **Big Omega-Notation:**

- We are only interested in the term with the highest-order, the fastest growing summand, the others will be ignored
- $\blacksquare$  f(n) is limited from underneath by  $c \cdot g(n)$

## Examples:

$$2n^{2}+7n-20 \in \Omega(n^{2})$$

$$2n^{2}+7n\log n-20 \in$$

$$7n\log n-20 \in$$

$$5 \in$$

$$2n^{2}+7n\log n+n^{3} \in$$

# Structure



#### **O-Notation**

Motivation / Definition Examples

#### **Ω-Notation**

#### Θ-Notation

#### Runtime

Summary
Limit / Convergence
L'Hôpital / l'Hospital
Practical use

- **Intuitive**: f is Theta of g ...
  - $\blacksquare$  ... if f is growing as much as g
  - $f \in \Theta(g)$ , f is growing at the same speed as g

# Formal: $f \in \Theta(g)$

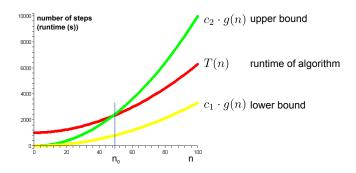
$$\Theta(g) = \underbrace{\mathscr{O}(g) \cap \Omega(g)}_{}$$

Intersection

## **Example:**

$$f(n) = 5n + 7, f(n) \in \mathcal{O}(n), f(n) \in \Omega(n)$$
  
$$\Rightarrow f(n) \in \Theta(n)$$

Proof for  $\mathcal{O}(q)$  and  $\Omega(q)$  look at slides 11 and 17



■ f and g have the same "growth"

# Structure



NotationMotivation / DefinitionExamples

**Ω**-Notation

Θ-Notation

#### Runtime

Summary Limit / Convergence L'Hôpital / l'Hospital Practical use

# Big O-Notation $\mathcal{O}(n)$ :

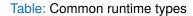
- $\blacksquare$  *f* is growing at most as fast as *g*
- $\blacksquare$   $C \cdot g(n)$  is the upper bound

## Big Omega-Notation $\Omega(n)$ :

- $\blacksquare$  f is growing at least as fast as g
- $C \cdot g(n)$  is the lower bound

# Big Theta-Notation $\Theta(n)$ :

- $\blacksquare$  *f* is growing at the same speed as *g* 
  - Arr  $C_1 \cdot g(n)$  is the lower bound
  - $C_2 \cdot g(n)$  is the upper bound



Runtime	Growth
$f \in \Theta(1)$	constant time
$f \in \Theta(\log n) = \Theta(\log_k n)$	logarithmic time
$f \in \Theta(n)$	linear time
$f \in \Theta(n \log n)$	n-log-n time (nearly linear)
$f \in \Theta(n^2)$	squared time
$f \in \Theta(n^3)$	cubic time
$f \in \Theta(n^k)$	polynomial time
$f \in \Theta(k^n), f \in \Theta(2^n)$	exponential time

# Structure



**O-Notation** 

Motivation / Definition Examples

**Ω-Notation** 

Θ-Notation

#### Runtime

Summary

Limit / Convergence

L'Hôpital / l'Hospital Practical use

- So far discussed:
  - Membership in O(...) proofed by hand: Explicit calculation of  $n_0$  and C
  - **However:** Both hint at limits in calculus

### **Definition of "Limit"**

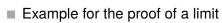
- The limit L exists for an infinite sequence  $f_1, f_2, f_3, ...$  if for all  $\varepsilon > 0$  one  $n_0 \in \mathbb{N}$  exists, such that for all  $n \ge n_0$  the following holds true:  $|f_n L| \le \varepsilon$
- A function  $f: \mathbb{N} \to \mathbb{R}$  can be written as a sequence  $\Rightarrow \lim_{n \to \infty} f_n = L$

#### **Definition of "Limit"**

- The limit L exists for an infinite sequence  $f_1, f_2, f_3, ...$  if for all  $\varepsilon > 0$  one  $n_0 \in \mathbb{N}$  exists, such that for all  $n \ge n_0$  the following holds true:  $|f_n L| \le \varepsilon$
- A function  $f: \mathbb{N} \to \mathbb{R}$  can be written as a sequence  $\Rightarrow \lim_{n \to \infty} f_n = L$

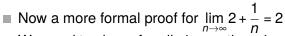
# The limit is converging:

 $\forall \varepsilon > 0 \; \exists n_0 \in \mathbb{N} \; \; \forall n \geq n_0 \colon \; |f_n - L| \leq \varepsilon$ 

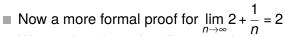


- Function  $f(n) = 2 + \frac{1}{n}$  with limes  $\lim_{n\to\infty} f(n) = 2$
- "Engineering" solution: use  $n = \infty$

$$\frac{1}{\infty} = 0 \Rightarrow \lim_{n \to \infty} f(n) = \lim_{n \to \infty} 2 + \frac{1}{n} = 2$$

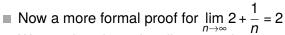


$$\left|2+\frac{1}{n}-2\right|=\left|\frac{1}{n}\right|\leq\varepsilon$$



$$\left|2+\frac{1}{n}-2\right|=\left|\frac{1}{n}\right|\leq\varepsilon$$

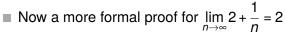
■ E.g.: for  $\varepsilon$  = 0.01 we get  $\frac{1}{n} \le \varepsilon$  for  $n \ge 100$ 



$$\left|2+\frac{1}{n}-2\right|=\left|\frac{1}{n}\right|\leq\varepsilon$$

- E.g.: for  $\varepsilon$  = 0.01 we get  $\frac{1}{n} \le \varepsilon$  for  $n \ge 100$
- In general

$$n_0 = \left\lceil \frac{1}{\varepsilon} \right\rceil$$



$$\left|2+\frac{1}{n}-2\right|=\left|\frac{1}{n}\right|\leq\varepsilon$$

- E.g.: for  $\varepsilon$  = 0.01 we get  $\frac{1}{n} \le \varepsilon$  for  $n \ge 100$
- In general

$$n_0 = \left\lceil \frac{1}{\varepsilon} \right\rceil$$

Then we get:

$$\left|\frac{1}{n}\right| = \frac{1}{n} \le \frac{1}{n_0} = \frac{1}{\left\lceil \frac{1}{\varepsilon} \right\rceil} \le \frac{1}{\frac{1}{\varepsilon}} = \varepsilon \quad \Box$$

$$\lim_{n\to\infty}\frac{f(n)}{g(n)}=L$$

Hence the following holds:

$$f \in \mathscr{O}(g)$$
  $\Leftrightarrow$   $\lim_{n \to \infty} \frac{f(n)}{g(n)} < \infty$  (1)

$$f \in \Omega(g)$$
  $\Leftrightarrow$   $\lim_{n \to \infty} \frac{f(n)}{g(n)} > 0$  (2)

$$f \in \Theta(g)$$
  $\Leftrightarrow$   $0 < \lim_{n \to \infty} \frac{f(n)}{g(n)} < \infty$  (3)



$$f \in \mathscr{O}(g) \Leftrightarrow \lim_{n \to \infty} \frac{f(n)}{g(n)} < \infty$$

# Forward proof $(\Rightarrow)$ :

$$f \in \mathscr{O}(g) \overset{\mathsf{def. of } \mathscr{O}(n)}{\Rightarrow} \exists n_0, C \ \forall n \geq n_0 : \ f(n) \leq C \cdot g(n)$$

$$\Rightarrow \exists n_0, C \ \forall n \geq n_0 : \frac{f(n)}{g(n)} \leq C$$

$$\Rightarrow \qquad \lim_{n \to \infty} \frac{f(n)}{g(n)} \leq C \quad \Box$$

# Backward proof $(\Leftarrow)$ :

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$$

$$\Rightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = C \qquad \text{For some } C \in \mathbb{R} \text{ (Limit)}$$

$$\stackrel{\text{def. limes}}{\Rightarrow} \exists n_0, \forall n \geq n_0 : \qquad \frac{f(n)}{g(n)} \leq C + \varepsilon \quad (e.g. \ \varepsilon = 1)$$

$$\Rightarrow \exists n_0, \forall n \geq n_0 : \qquad f(n) \leq \underbrace{(C+1)}_{O-notation \ constant} \cdot g(n)$$

$$\Rightarrow f \in \mathscr{O}(g) \quad \square$$

# Structure



Notation
Motivation / DefinitionExamples

**Ω-Notation** 

Θ-Notation

#### Runtime

Summary Limit / Convergence

L'Hôpital / l'Hospital

Practical use

#### Intuitive:

$$\lim_{n\to\infty}2+\frac{1}{n}=2+\frac{1}{\infty}=2$$

#### Intuitive:

$$\lim_{n\to\infty}2+\frac{1}{n}=2+\frac{1}{\infty}=2$$

## ■ With L'Hôpital:

■ Let 
$$f, g : \mathbb{N} \to \mathbb{R}$$
■ If  $\lim_{n \to \infty} f(n) = \lim_{n \to \infty} g(n) = \infty/0$ 

$$\Rightarrow \lim_{n \to \infty} \frac{f(n)}{g(n)} = \lim_{n \to \infty} \frac{f'(n)}{g'(n)}$$

#### Intuitive:

$$\lim_{n \to \infty} 2 + \frac{1}{n} = 2 + \frac{1}{\infty} = 2$$

## ■ With L'Hôpital:

Let 
$$f, g : \mathbb{N} \to \mathbb{R}$$
If  $\lim_{n \to \infty} f(n) = \lim_{n \to \infty} g(n) = \infty/0$ 

$$\Rightarrow \lim_{n \to \infty} \frac{f(n)}{g(n)} = \lim_{n \to \infty} \frac{f'(n)}{g'(n)}$$

## Holy inspiration

you need a doctoral degree for that

## The limit can not be determined in the way of an Engineer:

$$\lim_{n\to\infty}\frac{\ln(n)}{n}=\frac{\lim_{n\to\infty}\ln(n)}{\lim_{n\to\infty}n}\qquad \stackrel{\text{plugging in}}{\longrightarrow}\qquad \stackrel{\infty}{\longrightarrow}$$

## Determine the limit with using L'Hôpital:

$$\lim_{n\to\infty}\frac{f(n)}{g(n)}=\lim_{n\to\infty}\frac{f'(n)}{g'(n)}$$

Numerator:  $f(n): n \mapsto ln(n)$ 

Denominator:  $q(n): n \mapsto n$ 

$$\Rightarrow f'(n) = \frac{1}{n}$$
 (derivation from Numerator)  
\Rightarrow g'(n) = 1 (derivation from Denominator)

$$\lim_{n\to\infty}\frac{f'(n)}{g'(n)}=\lim_{n\to\infty}\frac{1}{n}=0 \quad \Rightarrow \quad \lim_{n\to\infty}\frac{f(n)}{g(n)}=\lim_{n\to\infty}\frac{\ln(n)}{n}=0$$

# Limits with L'Hôpital



## What can we take for granted without proofing?

- Only things that are trivial
- It is always better to proof it

## **Examples:**

$$\lim_{n \to \infty} \frac{1}{n} = 0 \qquad \text{is trivial}$$

$$\lim_{n \to \infty} \frac{1}{n^2} = 0 \qquad \text{is trivial}$$

$$\lim_{n \to \infty} \frac{\log(n)}{n} = 0 \qquad \text{use L'Hopital}$$

## Structure



**O-Notation** 

Motivation / Definition Examples

**Ω-Notation** 

Θ-Notation

#### Runtime

Summary Limit / Convergence L'Hôpital / l'Hospital

Practical use

- It is much easier to determine the runtime of an algorithm by using the *O*-Notation
  - Computing rules
  - 2 Practical use

### ■ Transitivity:

$$f \in \Theta(g) \land g \in \Theta(h)$$

$$f \in \mathscr{O}(g) \land g \in \mathscr{O}(h)$$

$$f \in \Omega(g) \land g \in \Omega(h)$$

### Transitivity:

$$f \in \Theta(g) \land g \in \Theta(h) \rightarrow f \in \Theta(h)$$
  
 $f \in \mathcal{O}(g) \land g \in \mathcal{O}(h)$   
 $f \in \Omega(g) \land g \in \Omega(h)$ 

### Transitivity:

$$\begin{split} &f \in \Theta(g) \, \wedge \, g \in \Theta(h) \quad \rightarrow \quad f \in \Theta(h) \\ &f \in \mathscr{O}(g) \, \wedge \, g \in \mathscr{O}(h) \quad \rightarrow \quad f \in \mathscr{O}(h) \\ &f \in \Omega(g) \, \wedge \, g \in \Omega(h) \end{split}$$

#### Transitivity:

$$f \in \Theta(g) \land g \in \Theta(h) \rightarrow f \in \Theta(h)$$

$$f \in \mathcal{O}(g) \land g \in \mathcal{O}(h) \rightarrow f \in \mathcal{O}(h)$$

$$f \in \Omega(g) \land g \in \Omega(h) \rightarrow f \in \Omega(h)$$

#### Characteristics

Transitivity:

$$f \in \Theta(g) \land g \in \Theta(h) \rightarrow f \in \Theta(h)$$
  
$$f \in \mathcal{O}(g) \land g \in \mathcal{O}(h) \rightarrow f \in \mathcal{O}(h)$$

$$f \in \mathcal{O}(g) \land g \in \mathcal{O}(h) \rightarrow f \in \mathcal{O}(h)$$

$$f \in \Omega(g) \land g \in \Omega(h) \rightarrow f \in \Omega(h)$$

Symmetry:

$$f \in \Theta(g) \leftrightarrow g \in \Theta(f)$$

#### ■ Transitivity:

$$\begin{split} & f \in \Theta(g) \, \wedge \, g \in \Theta(h) \quad \rightarrow \quad f \in \Theta(h) \\ & f \in \mathscr{O}(g) \, \wedge \, g \in \mathscr{O}(h) \quad \rightarrow \quad f \in \mathscr{O}(h) \\ & f \in \Omega(g) \, \wedge \, g \in \Omega(h) \quad \rightarrow \quad f \in \Omega(h) \end{split}$$

## Symmetry:

$$f \in \Theta(g) \leftrightarrow g \in \Theta(f)$$
  
 $f \in \mathscr{O}(g)$ 

#### Transitivity:

$$f \in \Theta(g) \land g \in \Theta(h) \rightarrow f \in \Theta(h)$$

$$f \in \mathscr{O}(g) \land g \in \mathscr{O}(h) \rightarrow f \in \mathscr{O}(h)$$

$$f \in \Omega(g) \land g \in \Omega(h) \rightarrow f \in \Omega(h)$$

## Symmetry:

$$f \in \Theta(g) \leftrightarrow g \in \Theta(f)$$

$$f \in \mathscr{O}(g) \ \leftrightarrow \ g \in \Omega(f)$$

#### Characteristics

Transitivity:

$$f \in \Theta(g) \land g \in \Theta(h) \rightarrow f \in \Theta(h)$$
  
$$f \in \mathcal{O}(g) \land g \in \mathcal{O}(h) \rightarrow f \in \mathcal{O}(h)$$
  
$$f \in \Omega(g) \land g \in \Omega(h) \rightarrow f \in \Omega(h)$$

Symmetry:

$$f \in \Theta(g) \leftrightarrow g \in \Theta(f)$$
  
 $f \in \mathscr{O}(g) \leftrightarrow g \in \Omega(f)$ 

Reflexivity:

$$f \in \Theta(f)$$
  $f \in \Omega(f)$   $f \in \mathcal{O}(f)$ 

#### ■ Trivial:

$$\begin{array}{rcl} f & \in & \mathcal{O}(f) \\ k \cdot \mathcal{O}(f) & = & \mathcal{O}(f) \\ \mathcal{O}(f+k) & = & \mathcal{O}(f) \end{array}$$

Addition:

$$\mathcal{O}(f) + \mathcal{O}(g) = \mathcal{O}(\max\{f, g\})$$

Multiplication:

$$\mathcal{O}(f)\cdot\mathcal{O}(g) = \mathcal{O}(f\cdot g)$$

Runtime Complexity

- The input size for all examples is *n*
- Basic operations

$$i1 = 0$$
  $\mathcal{O}(1)$ 

Sequences of basic operations

## Loops

for i in range(0, n):  

$$a[i] = 0$$

$$\boxed{\mathcal{O}(n)}$$

$$a1[i] = 0$$

$$\boxed{\mathcal{O}(n)}$$

$$0(1) \cdot \mathcal{O}(n) = \mathcal{O}(n)$$

$$\boxed{\mathcal{O}(n)}$$

$$\boxed{\mathcal{O}(n)}$$

$$\boxed{\mathcal{O}(n)}$$

$$\boxed{\mathcal{O}(1)}$$

$$\boxed{\mathcal{O}(1)}$$

$$\cdots$$

$$\boxed{\mathcal{O}(1)}$$

$$\cdots$$

$$\boxed{\mathcal{O}(1)}$$

$$\boxed{\mathcal{O}(1)}$$

$$\cdots$$

$$\boxed{\mathcal{O}(1)}$$

$$\boxed{\mathcal{O}(1)}$$

$$\cdots$$

$$\boxed{\mathcal{O}(1)}$$

$$\boxed{\mathcal{O}(1)}$$

$$\boxed{\mathcal{O}(1)}$$

$$\boxed{\mathcal{O}(1)}$$

$$\boxed{\mathcal{O}(1)}$$

$$\boxed{\mathcal{O}(1)}$$

$$\boxed{\mathcal{O}(1)}$$

$$\boxed{\mathcal{O}(1)}$$

## Loops

# ■ Conditions

- Input: List *x* with *n* numbers
- Output: a[i] is the arithmetic mean of x[0] to x[i]

```
def arithMean(x):
    a = [0] * len(x)
    for i in range(0, len(x)):
        s = 0
        for j in range(0, i+1):
            s = s + x[j]
        a[i] = s / (i+1)
```

for i in range(0, len(x)):
$$s = 0$$

$$for j in range(0, i+1):$$

$$s = s + x[j]$$

$$a[i] = s / (i+1)$$

$$O(n)$$

$$O(i)$$

$$O(i)$$

$$O(i)$$

$$O(i)$$

$$O(n)$$

$$O(n$$

■ How often will the instructions in the loop be executed, when the problem has size *n*?

$$1+2+\ldots+n=\frac{n\cdot(n+1)}{2}\in\mathscr{O}(n^2)$$

Discussion

## Way of speaking:

- With the Ø-Notation we look at the behavior of a function when  $n \to \infty$
- We only analyze the runtime when  $n \ge n_0$
- We talk about asymptotic analysis, when we discuss cost, runtime, etc. as  $\mathcal{O}(...)$ ,  $\Omega(...)$  or  $\Theta(...)$

#### Attention:

- If you are using **asymptotic analysis**, you can not make any predictions about the runtime of smaller input sizes  $(n < n_0)$
- For small input sizes (mostly n < 10), the runtime is predictably small
- $\blacksquare$   $n_0$  does not necessarily have to be small

Discussion

#### **Examples:**

- Let A and B be algorithms
  - A has the runtime f(n) = 80 n
  - B has the runtime  $f(n) = 2n \log_2 n$
- So  $f = \mathcal{O}(g)$  but **not**  $\Theta(g)$ 
  - $\blacksquare$   $\Rightarrow$  A is asymptotic faster than B
  - ⇒ There is a  $n_0$  for that  $n \ge n_0$ :  $f(n) \le g(n)$

#### When is A faster then B?

We search the minimal  $n_0$ :

$$f(n_0) = g(n_0)$$

$$80 n_0 = 2 n_0 \log_2 n_0$$

$$40 = \log_2 n_0$$

$$n_0 = 2^{40}$$

$$= (2^{10})^4 = (1024)^4$$

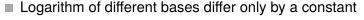
$$\approx (10^3)^4 = 10^{12}$$

$$\approx 1 \text{ trillion}$$

A ist faster than B if  $n_0$  has more than 1 trillion elements

## Runtime Examples

Continued



$$\log_a n = \frac{\log_b n}{\log_b a} = \frac{1}{\log_b a} \cdot \log_b n$$

- Hence:  $\log_a n \in \Theta(\log_b n)$
- For exponent this does not hold

$$3^n\not\in\Theta(2^n)$$

Proof: Use equation (1) from Slide 31

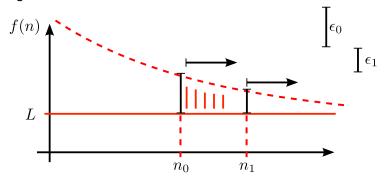
$$3^n \in \mathscr{O}(2^n) \Leftrightarrow \lim_{n \to \infty} \frac{3^n}{2^n} < \infty$$

However:

$$\lim_{n\to\infty}\frac{3^n}{2^n}=\lim_{n\to\infty}\left(\frac{3}{2}\right)^n=\infty$$



■ Figure for slide 28



#### ■ General

[MS08] Kurt Mehlhorn and Peter Sanders.
Algorithms and data structures, 2008.

https://people.mpi-inf.mpg.de/~mehlhorn/ftp/Mehlhorn-Sanders-Toolbox.pdf.

## ■ Big O notation

[Wik] Big O notation

https://en.wikipedia.org/wiki/Big\_O\_notation