Algorithms and Data Structures

A8. Runtime Analysis: Asymptotic Notation

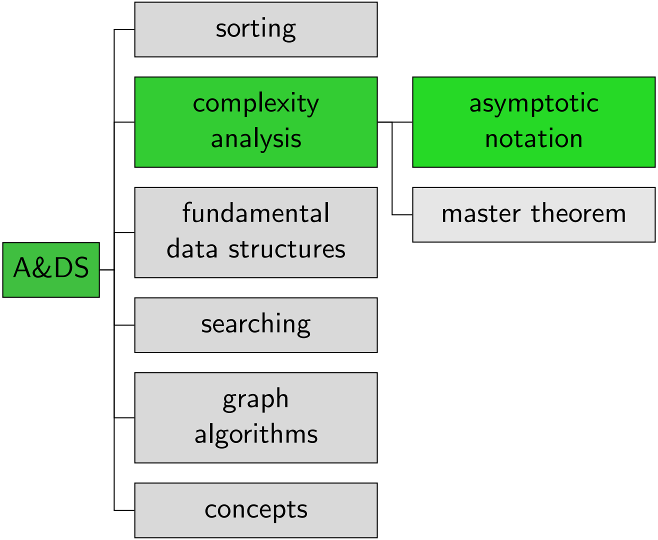
Gabriele Röger and Patrick Schnider

University of Basel

March 5, 2025

Asymptotic Notation

Content of the Course



Result for Merge Sort

"The running time of merge sort grows asymptotically as fast as $n \log_2 n$."

Result for Merge Sort

"The running time of merge sort grows asymptotically as fast as $n \log_2 n$."

Theorem

Bottom-up merge sort has linearithmic running time, i.e. there are constants $c, c', n_0 > 0$, such that for all $n \geq n_0$: $cn \log_2 n \leq T(n) \leq c' n \log_2 n$.

Result for Merge Sort

"The running time of merge sort grows asymptotically as fast as $n \log_2 n$."

Theorem

Bottom-up merge sort has linearithmic running time, i.e. there are constants $c, c', n_0 > 0$, such that for all $n \geq n_0$: $cn \log_2 n \leq T(n) \leq c' n \log_2 n$.

- When determining the bounds, we ignored lower-order terms (constant and n) or let them disappear.

Result for Merge Sort

"The running time of merge sort grows asymptotically as fast as $n \log_2 n$."

Theorem

Bottom-up merge sort has linearithmic running time, i.e. there are constants $c, c', n_0 > 0$, such that for all $n \geq n_0$: $cn \log_2 n \leq T(n) \leq c' n \log_2 n$.

- When determining the bounds, we ignored lower-order terms (constant and n) or let them disappear.
- We were not interested in the exact values of the constants but were satisfied if there exist some suitable constants.

Result for Merge Sort

"The running time of merge sort grows asymptotically as fast as $n \log_2 n$."

Theorem

Bottom-up merge sort has linearithmic running time, i.e. there are constants $c, c', n_0 > 0$, such that for all $n \geq n_0$:
$cn \log_2 n \leq T(n) \leq c' n \log_2 n$.

- When determining the bounds, we ignored lower-order terms (constant and n) or let them disappear.
- We were not interested in the exact values of the constants but were satisfied if there exist some suitable constants.
- The running time for small n is not that important.

Previous Results

Theorem

The merge step has linear running time, i.e., there are constants $c, c', n_0 > 0$ such that for all $n \geq n_0$: $cn \leq T(n) \leq c'n$.

Theorem

Merge sort has linearithmic running time, i.e.
there are constants $c, c', n_0 > 0$, such that for all $n \geq n_0$:
$cn \log_2 n \leq T(n) \leq c'n \log_2 n$.

Theorem

Selection sort has quadratic running time, i.e., there are constants $c > 0, c' > 0, n_0 > 0$ such that for $n \geq n_0$: $cn^2 \leq T(n) \leq c'n^2$.

Previous Results

Theorem

The merge step has linear running time, i.e., there are constants $c, c', n_0 > 0$ such that for all $n \geq n_0$: $cn \leq T(n) \leq c'n$.

Theorem

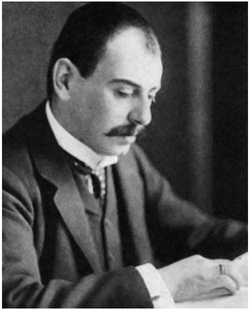
Merge sort has linearithmic running time, i.e.
there are constants $c, c', n_0 > 0$, such that for all $n \geq n_0$:
$cn \log_2 n \leq T(n) \leq c'n \log_2 n$.

Theorem

Selection sort has quadratic running time, i.e., there are constants $c > 0, c' > 0, n_0 > 0$ such that for $n \geq n_0$: $cn^2 \leq T(n) \leq c'n^2$.

Can't we write this more compactly?

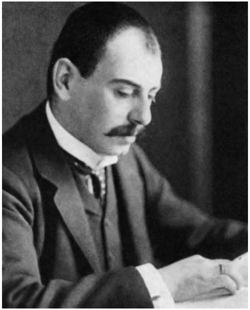
Asymptotic Notation/Landau-Bachmann Notation



Edmund Landau

- German mathematician (1877–1938)
- analytic number theory
- no friend of applied mathematics

Asymptotic Notation/Landau-Bachmann Notation



Edmund Landau

- German mathematician (1877–1938)
- analytic number theory
- no friend of applied mathematics

Neutral term: Asymptotic notation
German: Landau notation
Internationally: Bachmann–Landau notation also after Paul Gustav Heinrich Bachmann (German mathematician)

Symbol Theta

Definition

For a function $g : \mathbb{N} \to \mathbb{R}$, we denote by $\Theta(g)$ the set of all functions $f : \mathbb{N} \to \mathbb{R}$ that grow asymptotically as fast as g:

$$\Theta(g) = \{f \mid \exists c > 0\ \exists c' > 0\ \exists n_0 > 0\ \forall n \geq n_0 : c \cdot g(n) \leq f(n) \leq c' \cdot g(n)\}$$

Symbol Theta

Definition

For a function $g : \mathbb{N} \to \mathbb{R}$, we denote by $\Theta(g)$ the set of all functions $f : \mathbb{N} \to \mathbb{R}$ that grow asymptotically as fast as g:

$$\Theta(g) = \{f \mid \exists c > 0\ \exists c' > 0\ \exists n_0 > 0\ \forall n \geq n_0 : c \cdot g(n) \leq f(n) \leq c' \cdot g(n)\}$$

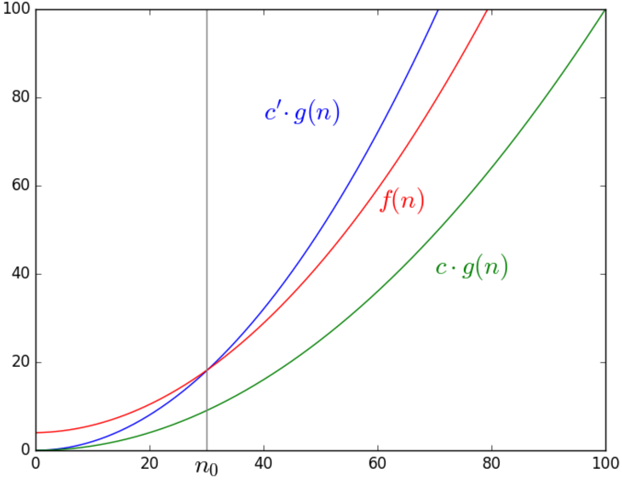
"The running time of merge sort is in $\Theta(n \log_2 n)$."
"$f \in \Theta(n^2)$ with $f(n) = 3n^2 + 5n + 39$"

or by convention (abusing notation/terminology) also

"The running time of merge sort is $\Theta(n \log_2 n)$."
"$3n^2 + 5n + 39 = \Theta(n^2)$"

Symbol Theta: Illustration

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



Jupyter Notebook (with Exercises)

Jupyter notebook: `asymptotic_notation.ipynb`

More Symbols for Asymptotic Growth

- "f grows no faster than g."

$$O(g) = \{f \mid \exists c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$

More Symbols for Asymptotic Growth

- "f grows no faster than g."

$$O(g) = \{f \mid \exists c > 0 \; \exists n_0 > 0 \; \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$

- O for "Ordnung" (order) of the function.

More Symbols for Asymptotic Growth

- "f grows no faster than g."

$$O(g) = \{f \mid \exists c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$

- O for "Ordnung" (order) of the function.
- "f grows at least as fast as g."

$$\Omega(g) = \{f \mid \exists c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : c \cdot g(n) \leq f(n)\}$$

More Symbols for Asymptotic Growth

- "f grows no faster than g."

$$O(g) = \{f \mid \exists c > 0 \; \exists n_0 > 0 \; \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$

- O for "Ordnung" (order) of the function.
- "f grows at least as fast as g."

$$\Omega(g) = \{f \mid \exists c > 0 \; \exists n_0 > 0 \; \forall n \geq n_0 : c \cdot g(n) \leq f(n)\}$$

- $\Theta(g) = O(g) \cap \Omega(g)$

More Symbols for Asymptotic Growth

- "f grows no faster than g."

$$O(g) = \{f \mid \exists c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$

- O for "Ordnung" (order) of the function.
- "f grows at least as fast as g."

$$\Omega(g) = \{f \mid \exists c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : c \cdot g(n) \leq f(n)\}$$

- $\Theta(g) = O(g) \cap \Omega(g)$
- $f \in \Omega(g)$ if and only if $g \in O(f)$.

More Symbols for Asymptotic Growth

- "f grows no faster than g."

$$O(g) = \{f \mid \exists c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$

- O for "Ordnung" (order) of the function.
- "f grows at least as fast as g."

$$\Omega(g) = \{f \mid \exists c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : c \cdot g(n) \leq f(n)\}$$

- $\Theta(g) = O(g) \cap \Omega(g)$
- $f \in \Omega(g)$ if and only if $g \in O(f)$.
- In computer science, we are often only interested in an upper bound on the growth of the running time: O instead of Θ

More Symbols for Asymptotic Growth

- "f grows no faster than g."

$$O(g) = \{f \mid \exists c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$

- O for "Ordnung" (order) of the function.
- "f grows at least as fast as g."

$$\Omega(g) = \{f \mid \exists c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : c \cdot g(n) \leq f(n)\}$$

- $\Theta(g) = O(g) \cap \Omega(g)$
- $f \in \Omega(g)$ if and only if $g \in O(f)$.
- In computer science, we are often only interested in an upper bound on the growth of the running time: O instead of Θ

Pronunciation: Θ: Theta, Ω: Omega, O: Oh

Less Frequently needed Symbols

- "f grows slower than g."

$$o(g) = \{f \mid \forall c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$

Less Frequently needed Symbols

- "f grows slower than g."

$$o(g) = \{f \mid \forall c > 0 \; \exists n_0 > 0 \; \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$

- "f grows faster than g."

$$\omega(g) = \{f \mid \forall c > 0 \; \exists n_0 > 0 \; \forall n \geq n_0 : c \cdot g(n) \leq f(n)\}$$

Less Frequently needed Symbols

- "f grows slower than g."

$$o(g) = \{f \mid \forall c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$

- "f grows faster than g."

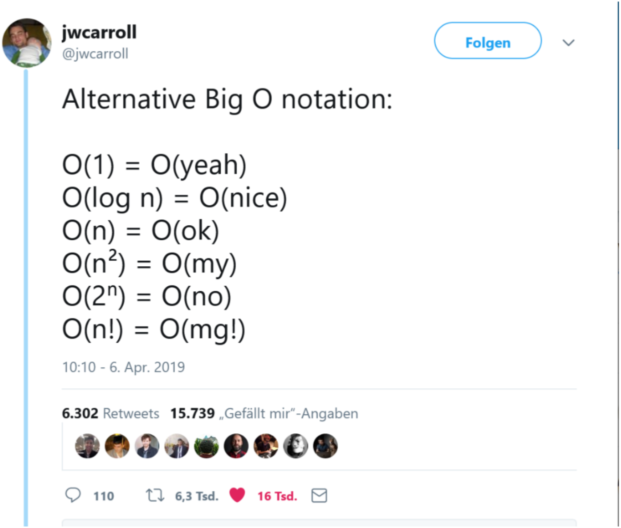
$$\omega(g) = \{f \mid \forall c > 0\ \exists n_0 > 0\ \forall n \geq n_0 : c \cdot g(n) \leq f(n)\}$$

Pronunciation: ω: little-omega

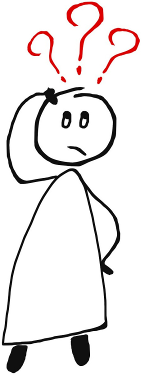
Some Relevant Classes of Functions

In increasing order (except for the general n^k):

g	growth
1	constant
$\log n$	logarithmic
n	linear
$n \log n$	linearithmic
n^2	quadratic
n^3	cubic
n^k	polynomial (constant k)
2^n	exponential


jwcarroll
@jwcarroll
Folgen
Alternative Big O notation:
O(1) = O(yeah)
O(log n) = O(nice)
O(n) = O(ok)
O(n²) = O(my)
O(2ⁿ) = O(no)
O(n!) = O(mg!)
10:10 - 6. Apr. 2019
6.302 Retweets 15.739 „Gefällt mir"-Angaben
110
6,3 Tsd.
16 Tsd.

Questions



Questions?

Rules

Examples for Θ

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.

Examples for Θ

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.
- Examples:
  - $f_1(n) = 5n^2 + 3n - 9$
  - $f_2(n) = 3n \log_2 n + 2n^2$
  - $f_3(n) = 9n \log_2 n + n + 17$
  - $f_4(n) = 8$

Examples for Θ

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.
- Examples:
  - $f_1(n) = 5n^2 + 3n - 9 \ \in \Theta(n^2)$
  - $f_2(n) = 3n \log_2 n + 2n^2$
  - $f_3(n) = 9n \log_2 n + n + 17$
  - $f_4(n) = 8$

Examples for Θ

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.
- Examples:
  - $f_1(n) = 5n^2 + 3n - 9 \ \in \Theta(n^2)$
  - $f_2(n) = 3n \log_2 n + 2n^2 \ \in \Theta(n^2)$
  - $f_3(n) = 9n \log_2 n + n + 17$
  - $f_4(n) = 8$

Examples for Θ

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.
- Examples:
  - $f_1(n) = 5n^2 + 3n - 9 \;\in \Theta(n^2)$
  - $f_2(n) = 3n\log_2 n + 2n^2 \;\in \Theta(n^2)$
  - $f_3(n) = 9n\log_2 n + n + 17 \;\in \Theta(n \log n)$
  - $f_4(n) = 8$

Examples for Θ

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.
- Examples:
  - $f_1(n) = 5n^2 + 3n - 9 \;\in \Theta(n^2)$
  - $f_2(n) = 3n \log_2 n + 2n^2 \;\in \Theta(n^2)$
  - $f_3(n) = 9n \log_2 n + n + 17 \;\in \Theta(n \log n)$
  - $f_4(n) = 8 \;\in \Theta(1)$

Examples for Big-O

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.

Examples for Big-O

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.
- Examples:
  - $f_1(n) = 8n^2 - 3n - 9$
  - $f_2(n) = n^3 - 3n \log_2 n$
  - $f_3(n) = 3n \log_2 n + 1000n + 10^{200}$

Examples for Big-O

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.
- Examples:
  - $f_1(n) = 8n^2 - 3n - 9 \; \in O(n^2)$
  - $f_2(n) = n^3 - 3n \log_2 n$
  - $f_3(n) = 3n \log_2 n + 1000n + 10^{200}$

Examples for Big-O

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.
- Examples:
  - $f_1(n) = 8n^2 - 3n - 9 \ \in O(n^2)$
  - $f_2(n) = n^3 - 3n \log_2 n \ \in O(n^3)$
  - $f_3(n) = 3n \log_2 n + 1000n + 10^{200}$

Examples for Big-O

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.
- Examples:
  - $f_1(n) = 8n^2 - 3n - 9 \ \in O(n^2)$
  - $f_2(n) = n^3 - 3n\log_2 n \ \in O(n^3)$
  - $f_3(n) = 3n\log_2 n + 1000n + 10^{200} \ \in O(n\log n)$

Examples for Big-O

- In the analysis, only the highest-order term (= fastest-growing summand) of a function is relevant.
- Examples:
  - $f_1(n) = 8n^2 - 3n - 9 \;\in O(n^2)$
  - $f_2(n) = n^3 - 3n \log_2 n \;\in O(n^3)$
  - $f_3(n) = 3n \log_2 n + 1000n + 10^{200} \;\in O(n \log n)$
- Why is this the case?

Connections

It holds that:

- $O(1) \subset O(\log n) \subset O(n) \subset O(n \log n) \subset O(n^k) \subset O(2^n)$ (for $k \geq 2$)

Connections

It holds that:

- $O(1) \subset O(\log n) \subset O(n) \subset O(n \log n) \subset O(n^k) \subset O(2^n)$ (for $k \geq 2$)
- $O(n^{k_1}) \subset O(n^{k_2})$ for $k_1 < k_2$
  e.g. $O(n^2) \subset O(n^3)$

Calculation Rules

- Product
  $f_1 \in O(g_1)$ and $f_2 \in O(g_2) \Rightarrow f_1 f_2 \in O(g_1 g_2)$

Calculation Rules

- Product
  $f_1 \in O(g_1)$ and $f_2 \in O(g_2) \Rightarrow f_1 f_2 \in O(g_1 g_2)$
- Sum
  $f_1 \in O(g_1)$ and $f_2 \in O(g_2) \Rightarrow f_1 + f_2 \in O(g_1 + g_2)$

Calculation Rules

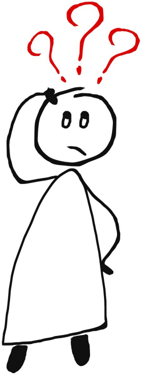
- Product
  $f_1 \in O(g_1)$ and $f_2 \in O(g_2) \Rightarrow f_1 f_2 \in O(g_1 g_2)$
- Sum
  $f_1 \in O(g_1)$ and $f_2 \in O(g_2) \Rightarrow f_1 + f_2 \in O(g_1 + g_2)$
- Multiplication with a constant
  $k > 0$ and $f \in O(g) \Rightarrow kf \in O(g)$
  $k > 0 \Rightarrow O(kg) = O(g)$

Reason for Sufficiency of Highest-order Term

Example: $5n^3 + 2n \in O(n^3)$

- Due to rule for multiplication with a constant:
  - $5n^3 \in O(n^3)$
  - $2n \in O(n)$
- Because of $2n \in O(n)$ and $O(n) \subset O(n^3)$:
  - $2n \in O(n^3)$
- Sum rule:
  - $5n^3 + 2n \in O(n^3 + n^3)$
- Multiplication with a constant (for a class):
  - $5n^3 + 2n \in O(n^3)$

Questions



Questions?

Summary

Summary

- With asymptotic notation, we refer to classes of functions that grow no faster/no slower/... than a function g.
  - $O(g)$: Growth no faster than g.
  - $\Theta(g)$: Growth asymptotically as fast as g.