Introduction to
Data Structures and Algorithms

Chapter: **Growth of functions**

- Asymptotic Notation

Friedrich-Alexander-Universität
Erlangen-Nürnberg

Lehrstuhl Informatik 7 (Prof. Dr.-Ing. Reinhard German)
Martensstraße 3, 91058 Erlangen

Growth of Functions

- The order of growth of running time of algorithms is a simple characterization of algorithm´s efficiency

  ⇨ comparison of relative performance of alternative algorithms

- For most algorithms, the running time depends on the input size

- In the simplest case the input size is given by an integer, i.e.

- Running times are defined in terms of functions with natural numbers as their domains

Growth of Functions

- Asymptotic efficiency: How does the running time increase as the input size approaches infinity

- Example 1

  a) The running time $C_{iter}(i)$ of algorithm *fibiter*
  - measured by the number of arithmetic operations executed -
  is

$$C_{iter}(i) = i-1 \qquad \text{for } i \geq 2$$

  b) if we include the "increase index" of a for loop
  as an additional arithmetic operation

$$C'_{iter}(i) = 2(i-1) \qquad \text{for } i \geq 2$$

Growth of Functions

- Example 2:
  The running time $C_{rec}(i)$ of algorithm *fibrec* –
  measured by the number of arithmetic operations executed -
  is bounded as follows

$$3 \cdot 2^{(i-1)/2} - 3 \leq C_{rec}(i) \leq 3 \cdot 2^{i-1} - 3$$

(hint: we know that $2^{(i-2)/2} \leq f_i \leq 2^{(i-2)}$)

- Example 3:
  The running time $C_{isq}(i)$ of algorithm *fibisq* –
  measured by the number of arithmetic operations executed -
  is bounded as follows

$$C_{isq}(i) \leq 26 \cdot (\lfloor \log_2(i-1) \rfloor + 1) + 1$$

Growth of Functions

- Observation: Information about the runtime of an algorithm may be given in various ways, e.g.
  - exactly (*fibiter*)
  - by giving an upper bound (*fibisq*) or
  - by giving upper and lower bounds (*fibrec*)

- By comparing the behavior of the algorithms for increasing input size (⇨ increasing values of i), we recognize that
  - neither constant factors
  - nor terms added

  are of relevance, if related to the order of growth

Asymptotic notation: "Θ"

Definition

- For a given function g we define the set **Θ(g(n))** of functions (pronounced "theta" of "g of n")

  $\Theta(g(n))$ =

$$\{\ f(n) \mid \text{there exist } c_1, c_2 \in \mathbb{R}^+ \text{ and } n_0 \in \mathbb{N} \text{ such that } c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n) \text{ for all } n > n_0\ \}$$

Interpretation

- $\Theta(g(n))$ is the set of functions that can be "sandwiched" between $c_1 g(n)$ and $c_2 g(n)$ for sufficiently large values of n
- For all $n \geq n_0$ the function $f(n)$ is equal to $g(n)$ to within a constant factor
  We say: $g(n)$ is an **asymptotically <u>tight</u> bound** for $f(n)$

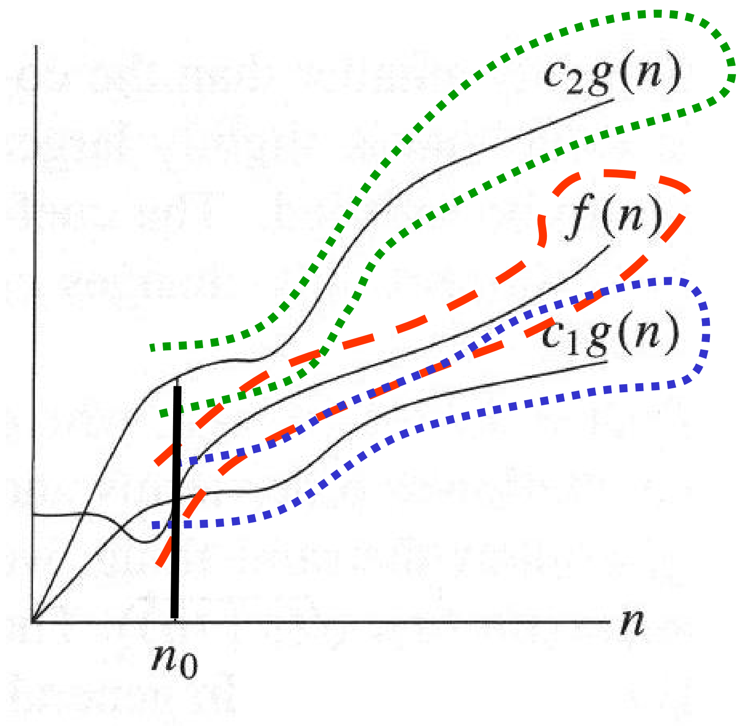
Growth of Functions

Asymptotic notation: "Θ"

Illustration of Θ(g(n)):

f(n) is "sandwiched" between c_1g(n) and c_2g(n) for n>n_0**:**

f(n) $\in$ Θ(g(n))



<u>Example:</u>

$$\tfrac{1}{2}n^2 - 3n \in \Theta(n^2)$$

Growth of Functions

Asymptotic notation: “Θ”

Some simple examples

- Be T(n) the runtime of a given algorithm and input size n
  - If T(n) is a linear function of n, we write $T(n) \in \Theta(n)$
  - If T(n) is a quadratic function of n, we write $T(n) \in \Theta(n^2)$
  - and so on

Growth of Functions

Asymptotic notation: “Θ”

- For given Θ(g(n)) we assume that the limiting function g(n) is **asymptotically <u>nonnegative:</u>** g(n) is nonnegative whenever n is sufficiently large
(⇔ there is a $n_0 \in \mathbb{N}$, so that $g(n) \geq 0$ for all $n > n_0$)

- Otherwise Θ(g(n)) is the empty set
(Consequently, $f(n) \in \Theta(g(n))$ are asymptotically nonnegative)

- Of course the cost functions we deal with are asymptotically nonnegative functions

Asymptotic notation: “Θ”

- Alternative (and usual) notation
  - Instead of writing $f(n) \in \Theta(g(n))$ we often write $f(n) = \Theta(g(n))$
  - E.g. we could write $T(n) = \Theta(n^2)$ instead of $T(n) \in \Theta(n^2)$
  - But: Be aware that this is a convention
    (not to be confused with the common meaning of equality!)
- This allows to write expressions as
  - $3n^2 + 45n - 35 = 3n^2 + \Theta(n)$

  meaning:

  There is a function $f(n) \in \Theta(n)$, so that:

  $$3n^2 + f(n) = 3n^2 + 45n - 35$$

Asymptotic notation: "Θ"

- Example 1:

  Show that $f(n) = 3n^2 + 2n - ½ \in \Theta(n^2)$

  - We must find $c_1, c_2 \in \mathbb{R}+$, $n_0 \in \mathbb{N}$ such that for all $n > n_0$

$$c_1 n^2 \leq 3n^2 + 2n - ½ \leq c_2 n^2$$

- Example 2:

  Show that $f(n) = 3 \cdot \log_2(n) \notin \Theta(n)$

  - We must show that it is not possible to find $c_1, c_2 \in \mathbb{R}+$, $n_0 \in \mathbb{N}$ such that for all $n > n_0$

$$c_1 n \leq 3 \cdot \log_2(n) \leq c_2 n$$

Asymptotic notation: “Θ”

- What can be said about the asymptotic growth of the complexity of our “Fibonacci algorithms”?
  - iterativ (*fibiter*) (for $i \geq 2$)
    - a) $C_{iter}(i) = i-1$
    - b) $C'_{iter}(i) = 2 \cdot (i-1)$
  - recursive (*fibrec*) (for $i \geq 2$)

    $$3 \cdot 2^{(i-1)/2} - 3 \leq C_{rec}(i) \leq 3 \cdot 2^{i-1} - 3$$

  - iterative squaring (*fibisq*) (for $i \geq 2$)

$$C_{isq}(i) \leq 26 \cdot (\lfloor \log_2(i-1) \rfloor + 1) + 1$$

Asymptotic notation: “Θ”

- Solution for the iterative algorithm (*fibiter*)

  a) $C_{iter}(i) = \Theta(i)$

  b) $C'_{iter}(i) = \Theta(i)$

- Although $C'_{iter}(i) > C_{iter}(i)$ holds for all arguments,
  the different cost functions show the same asymptotic growth!

Growth of Functions

Asymptotic upper and lower bounds

- One result of the analysis of algorithms
  for computing Fibonacci numbers is:
  - Obviously there is a need for asymptotic upper bounds
    and asymptotic lower bounds of functions!

- Similar to the definition Θ (a set of “sandwiching” functions)
  we will define sets of functions
  providing asymptotic lower or asymptotic upper bounds

Asymptotic notation: “O”

Definition

- For a given function g we define the set O(g(n)) of functions (pronounced “big-oh” of “g of n”)

  O(g(n)) =

$$\{\ f(n) \mid \text{there exist } c \in \mathbb{R}^{+} \text{ and } n_0 \in \mathbb{N} \text{ such that } f(n) \leq c \cdot g(n) \text{ for all } n > n_0\ \}$$

Interpretation

- O(g(n)) is the set of functions
  for that c·g(n) is an **upper bound** for **large** values of **n**

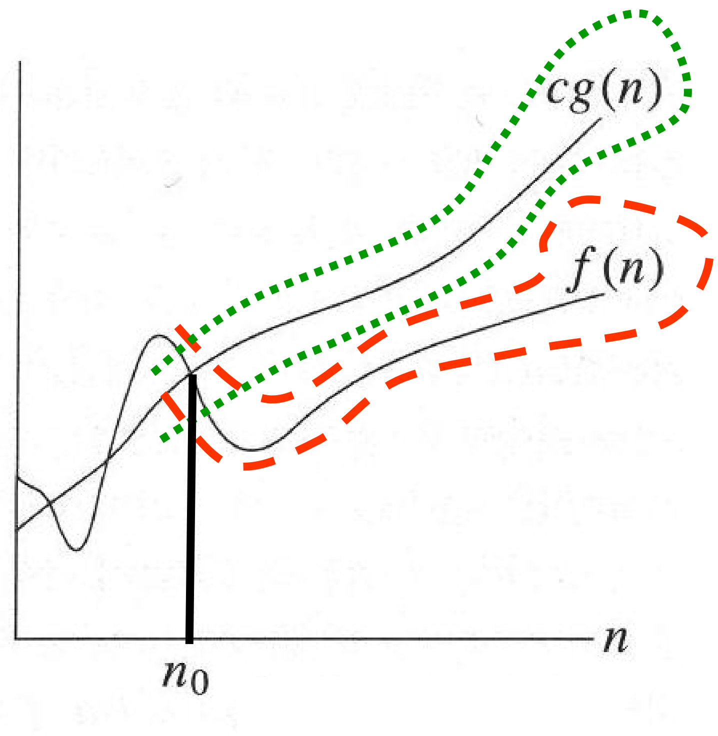
  (⇨ an asymptotic upper bound)

Asymptotic notation: "O"

Illustration of O(g(n)):

f(n) is bounded above by c·g(n) for $n>n_0$**:**

$f(n) \in O(g(n))$



Asymptotic notation: "O"

- What can be said about the asymptotic growth of the recursive and iterative squaring "Fibonacci algorithms"?
  - recursive (*fibrec*) (for i ≥ 2)

    $$3 \cdot 2^{(i-1)/2} - 3 \le C_{rec}(i) \le 3 \cdot 2^{i-1} - 3$$

    ⇨ $C_{rec}(i) = O(2^i)$

  - iterative squaring (*fibisq*) (for i ≥ 2)

    $$C_{isq}(i) \le 26 \cdot (\lfloor \log_2(i-1) \rfloor + 1) + 1$$

    ⇨ $C_{isq}(i) = O(\log_2(i))$

Asymptotic notation: "Ω"

Definition

- For a given function g we define the set Ω(g(n)) of functions (pronounced "big omega" of "g of n")

  Ω(g(n)) =

$$\{\ f(n) \mid \text{there exist } c \in \mathbb{R}^+ \text{ and } n_0 \in \mathbb{N} \text{ such that } c \cdot g(n) \leq f(n) \text{ for all } n > n_0\ \}$$

Interpretation

- Ω(g(n)) is the set of functions for that c·g(n) is a **lower bound** for large values of n

  (⇨ an asymptotic lower bound)

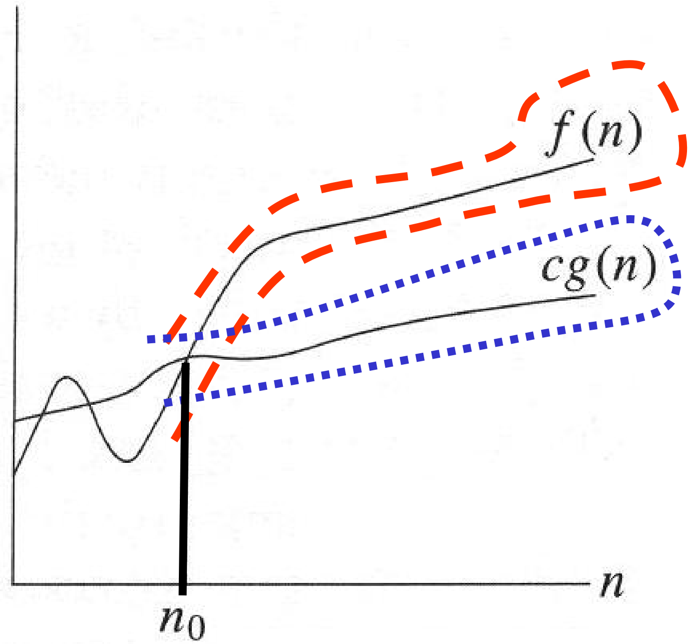
Growth of Functions

Asymptotic notation: “Ω”

Illustration of Ω(g(n)):

f(n) is bounded below by c·g(n) for $n > n_0$**:**

$f(n) \in \Omega(g(n))$



Asymptotic notation: "Ω"

- What can be said about lower bounds of the asymptotic growth of the recursive "Fibonacci algorithm"?
  - recursive (*fibrec*) (for $i \geq 2$)

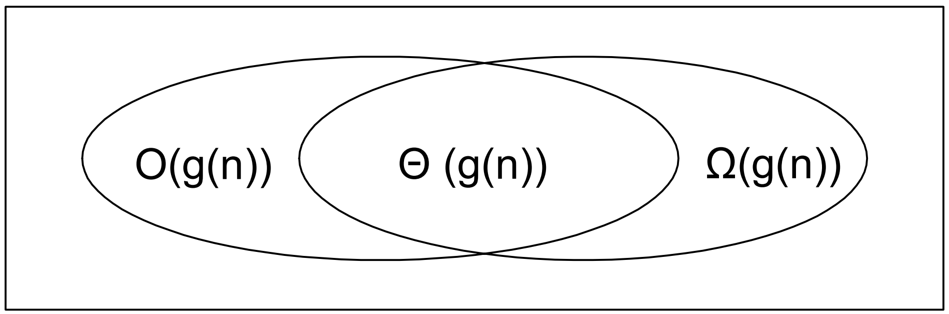
$$3 \cdot 2^{(i-1)/2} - 3 \leq C_{rec}(i) \leq 3 \cdot 2^{i-1} - 3$$

$$\Rightarrow C_{rec}(i) = \Omega(2^{i/2})$$

Growth of Functions

Basic relations between Θ, Ω and O

- It follows from the definitions
  that for each asymptotic nonnegative function g(n)
  - $\Theta(g(n)) \subseteq O(g(n))$
  - $\Theta(g(n)) \subseteq \Omega(g(n))$
- It follows from the definitions
  that for all asymptotic nonnegative functions f(n) and g(n)
  - **$f(n) = O(g(n))$ and $f(n) = \Omega(g(n)) \Leftrightarrow f(n) = \Theta(g(n))$**



Asymptotic upper and lower bounds

- The asymptotic upper and lower bounds defined by O and Ω are not necessarily tight bounds
- Example:

  Be $f(n) = n^2+5n-17$

  - $O(n^3)$ is an asymptotic upper bound for f $(f \in O(n^3))$
  - $\Omega(n)$ is an asymptotic lower bound for f $(f \in \Omega(n))$
  - But these bounds are not tight
  - Note:
    - $O(n^2)$ and $\Omega(n^2)$ are tight bounds for f
    - As $f \in O(n^2)$ and $f \in \Omega(n^2)$ ⇨ $f \in \Theta(n^2)$

Non-tight asymptotic upper bounds

- The function $g(n) = n^3$ is an upper bound for $f(n) = n^2+5n-17$ that grows significantly faster than f(n)
  (or f(n) grows significantly slower than g(n)).

Definition

- For a given function g we define the set o(g(n)) of functions (pronounced “little-oh” of “g of n”)

  o(g(n)) =

$$\{\ f(n) \mid \underline{\text{for all } c \in \mathbb{R}^+} \text{ there exists } n_0 \in \mathbb{N} \text{ such that } \underline{f(n) < c \cdot g(n)} \text{ for all } n > n_0\ \}$$

Non-tight asymptotic upper bounds

- If f(n) $\in$ o(g(n)), then

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

- Example:

  g(n) = n^3 is a non-tight asymptotic upper bound
  for f(n) = $n^2+5n-17$

  Obviously $$\lim_{n \to \infty} \frac{n^2 + 5n - 17}{n^3} = 0$$

Non-tight asymptotic lower bounds

- The function g(n) = n is a lower bound for f(n) = n^2+5n-17 that grows significantly slower than f(n)
  (or f(n) grows significantly faster than g(n)).

Definition

- For a given function g we define the set ω(g(n)) of functions (pronounced "little-omega" of "g of n")

ω(g(n)) =

$$\{ f(n) \mid \underline{\text{for all } c \in \mathbb{R}^+} \text{ there exists } n_0 \in \mathbb{N} \text{ such that } \underline{c \cdot g(n) < f(n)} \text{ for all } n > n_0 \}$$

Non-tight asymptotic lower bounds

- If f(n) $\in$ ω (g(n)), then

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

- Example:

  g(n) = n is a non-tight asymptotic lower bound
  for f(n) = n^2+5n-17

  Obviously

$$\lim_{n\to\infty} \frac{n^2+5n-17}{n} = \infty$$

Growth of Functions

Some rules for the so-called Landau symbols Θ, Ω, ω, O and o

- $$f(n) = o(g(n)) \quad \text{implies} \quad f(n) = O(g(n))$$
  $$f(n) = \omega(g(n)) \quad \text{implies} \quad f(n) = \Omega(g(n))$$

- **Transitivity**

$$f(n) = \Theta(g(n)) \quad \text{and} \quad g(n) = \Theta(h(n)) \quad \text{imply} \quad f(n) = \Theta(h(n))$$
$$f(n) = O(g(n)) \quad \text{and} \quad g(n) = O(h(n)) \quad \text{imply} \quad f(n) = O(h(n))$$
$$f(n) = \Omega(g(n)) \quad \text{and} \quad g(n) = \Omega(h(n)) \quad \text{imply} \quad f(n) = \Omega(h(n))$$
$$f(n) = o(g(n)) \quad \text{and} \quad g(n) = o(h(n)) \quad \text{imply} \quad f(n) = o(h(n))$$
$$f(n) = \omega(g(n)) \quad \text{and} \quad g(n) = \omega(h(n)) \quad \text{imply} \quad f(n) = \omega(h(n))$$

Growth of Functions

Some rules for the so-called Landau symbols Θ, Ω, ω, O and o

- **Reflexivity**

$$f(n) = \Theta(f(n))$$
$$f(n) = O(f(n))$$
$$f(n) = \Omega(f(n))$$

- **Symmetry**

$$f(n) = \Theta(g(n)) \quad \text{if and only if} \quad g(n) = \Theta(f(n))$$

- **Transpose symmetry**

$$f(n) = O(g(n)) \quad \text{if and only if} \quad g(n) = \Omega(f(n))$$
$$f(n) = o(g(n)) \quad \text{if and only if} \quad g(n) = \omega(f(n))$$

Asymptotic notation (Supplement)

- Remember:
  The notation $3n^2 + 45n - 35 = 3n^2 + \Theta(n)$ means:
  - There is a function $f(n) \in \Theta(n)$,
    so that:

    $$3n^2 + f(n) = 3n^2 + 45n - 35$$

- We could also write:
  - $3n^2 + 45n - 35 = 3n^2 + o(n^2)$

  Which stands for:
  - "only the term $3n^2$ is important, the other terms may be neglected, as these do not contribute significantly to its growth"