Data structures and Algorithms

INDUCTION & ASYMPTOTICS

Logistics

- All the materials are in English
- Technical questions
  - Q&A forum – use English
  - You are encouraged to help each other
- Administrative questions - Havana
- Office hours: Fridays 13:00-14:00
- Moodle
  - Course material - lecture notes, homework assignments, solutions

Today

Basic concepts:

- Induction
- Asymptotics:
  - Big-O
  - Big-Ω and Big-Θ
  - Little-o and Little-ω

Proof By Induction - definition

Claim: $S(n)$ is true for all $n \geq c$

Proof By Induction - definition

Claim: $S(n)$ is true for all $n \geq c$

1. **Base case**: Show $S(n)$ is true for $n = c$ (often c = 0 or 1)

Proof By Induction - definition

Claim: $S(n)$ is true for all $n \geq c$

1. **Base case**: Show $S(n)$ is true for $n = c$ (often c = 0 or 1)
2. **Inductive hypothesis**: Assume $S(k)$ is true for all $c \leq k < n$

Proof By Induction - definition

Claim: $S(n)$ is true for all $n \geq c$

1. **Base case**: Show $S(n)$ is true for $n = c$ (often c = 0 or 1)
2. **Inductive hypothesis**: Assume $S(k)$ is true for all $c \leq k < n$
3. **Inductive step**: Show that $S(n)$ is then true for n

Note that many times we only assume the claim holds for $k = n - 1$.

Induction Example 1: Arithmetic Sequence Sum:

Claim: $\forall n \geq 1: \sum_{i=1}^{n} i = 1 + 2 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$

Induction Example 1: Arithmetic Sequence Sum:

Claim: $\forall n \geq 1: \sum_{i=1}^{n} i = 1 + 2 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$

Proof: By induction over n:

- Basis – For n=1: $\sum_{i=1}^{1} i = 1 = \frac{2}{2} = \frac{1(1+1)}{2}$

Induction Example 1: Arithmetic Sequence Sum:

Claim: $\forall n \geq 1: \sum_{i=1}^{n} i = 1 + 2 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$

Proof: By induction over n:

- Basis – For n=1: $\sum_{i=1}^{1} i = 1 = \frac{2}{2} = \frac{1(1+1)}{2}$
- Induction Hypothesis – Assume for n: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.

Induction Example 1: Arithmetic Sequence Sum:

Claim: $\forall n \geq 1: \sum_{i=1}^{n} i = 1 + 2 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$

Proof: By induction over n:

- Basis – For n=1: $\sum_{i=1}^{1} i = 1 = \frac{2}{2} = \frac{1(1+1)}{2}$
- Induction Hypothesis – Assume for n: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.
- Induction Step – Under the induction hypothesis prove that the statement is correct for $n + 1$:

Induction Example 1: Arithmetic Sequence Sum:

Claim: $\forall n \geq 1: \sum_{i=1}^{n} i = 1 + 2 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$

Proof: By induction over n:

- Basis – For n=1: $\sum_{i=1}^{1} i = 1 = \frac{2}{2} = \frac{1(1+1)}{2}$
- Induction Hypothesis – Assume for n: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.
- Induction Step – Under the induction hypothesis prove that the statement is correct for $n+1$:

$$\sum_{i=1}^{n+1} i$$

Induction Example 1: Arithmetic Sequence Sum:

Claim: $\forall n \geq 1: \sum_{i=1}^{n} i = 1 + 2 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$

Proof: By induction over n:

- Basis – For n=1: $\sum_{i=1}^{1} i = 1 = \frac{2}{2} = \frac{1(1+1)}{2}$
- Induction Hypothesis – Assume for n: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.
- Induction Step – Under the induction hypothesis prove that the statement is correct for $n+1$:

$$\sum_{i=1}^{n+1} i = \left(\sum_{i=1}^{n} i\right) + (n+1)$$

Induction Example 1: Arithmetic Sequence Sum:

Claim: $\forall n \geq 1: \sum_{i=1}^{n} i = 1 + 2 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$

Proof: By induction over n:

- Basis – For n=1: $\sum_{i=1}^{1} i = 1 = \frac{2}{2} = \frac{1(1+1)}{2}$
- Induction Hypothesis – Assume for n: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.
- Induction Step – Under the induction hypothesis prove that the statement is correct for $n+1$:

$$\sum_{i=1}^{n+1} i = \left(\sum_{i=1}^{n} i\right) + (n+1) =_{I.H} \frac{n(n+1)}{2} + (n+1)$$

Induction Example 1: Arithmetic Sequence Sum:

Claim: $\forall n \geq 1: \sum_{i=1}^{n} i = 1 + 2 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$

Proof: By induction over n:

- Basis – For n=1: $\sum_{i=1}^{1} i = 1 = \frac{2}{2} = \frac{1(1+1)}{2}$
- Induction Hypothesis – Assume for n: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.
- Induction Step – Under the induction hypothesis prove that the statement is correct for $n+1$:

$$\sum_{i=1}^{n+1} i = \left(\sum_{i=1}^{n} i\right) + (n+1) =_{I.H} \frac{n(n+1)}{2} + (n+1) = \frac{n(n+1) + 2(n+1)}{2}$$

Induction Example 1: Arithmetic Sequence Sum:

Claim: $\forall n \geq 1: \sum_{i=1}^{n} i = 1 + 2 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$

Proof: By induction over n:

- Basis – For n=1: $\sum_{i=1}^{1} i = 1 = \frac{2}{2} = \frac{1(1+1)}{2}$
- Induction Hypothesis – Assume for n: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.
- Induction Step – Under the induction hypothesis prove that the statement is correct for $n+1$:

$$\sum_{i=1}^{n+1} i = \left(\sum_{i=1}^{n} i\right) + (n+1) =_{I.H} \frac{n(n+1)}{2} + (n+1) = \frac{n(n+1) + 2(n+1)}{2} = \frac{(n+1)(n+2)}{2}$$

■

Induction - All horses are of the same color

The proof consists of three steps:

The **base case**: If there is only one horse in the group, then clearly all horses in that group have the same color.

The **inductive hypothesis**: Assume that all the horses in groups of n horses always have the same color.

The **inductive Step:** Let us consider a group consisting of n+1 horses.

Induction - All horses are the same color

The **inductive Step:** Let us consider a group consisting of n+1 horses.

- Exclude the last horse and look only at the remaining n horses, we know from the last iteration that all the groups of horses in size n are the same color (this is our hypothesis).
- Exclude the first horse, again we have a group of n horses, which we proved in the last step they are the same color.
- Therefore, the first horse in the group is of the same color as the horses in the middle, who in turn are of the same color as the last horse. Hence the first horse, middle horses, and last horse are all of the same color, and we have proven that: If n horses have the same color, then n+1 horses will also have the same color.


WHAT SEEMS TO BE THE OFFICER PROBLEM?
quickmeme.com

Introduction to Complexity Analysis

Different algorithms can solve the same problem in different manners.

We need a way to estimate which one is better, how can we compare them?

The two most frequent complexity indices are:

- Time Complexity (run time)
- Space/Memory Complexity

We will focus mainly on worst-case time complexity:

- Worst case – worst possible input for an algorithm

Asymptotics

Approximation of the “growth ratio” of a function with respect to its input size.

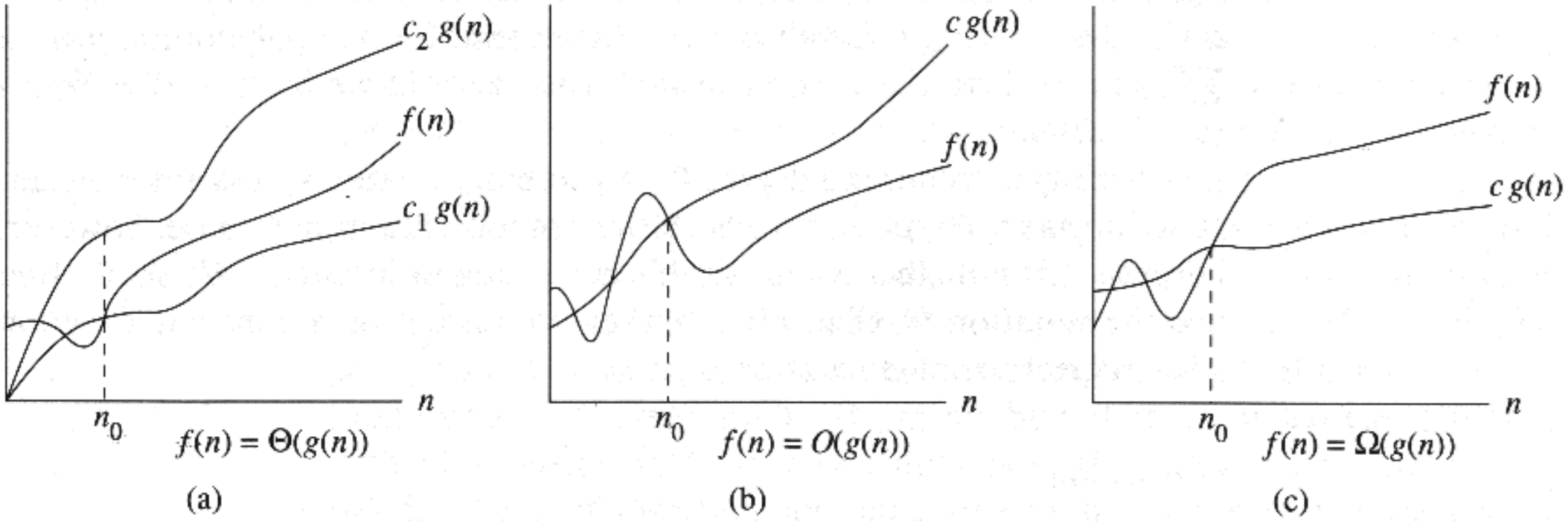
For two functions f(n) and g(n), we would like to determine if f(n) is bigger, smaller or equal to g(n) asymptotically.

Preliminary assumptions:

- A single machine with only one processor
- Simple operations are done in run time of 1, such as:
  - Array cell access
  - Assignment
  - Arithmetical operations (+, -, *, /)
  - Boolean operations
  - Comparison

Asymptotics

For the following definitions we will assume we have an algorithm A that takes f(n) operations to finish a task over input of size n.



Exponents and Logarithms

Let a and b be real numbers and m and n be integers. Then the following properties of exponents hold, provided that all of the expressions appearing in a particular equation are defined.

1. $a^m a^n = a^{m+n}$
2. $(a^m)^n = a^{mn}$
3. $(ab)^m = a^m b^m$
4. $\dfrac{a^m}{a^n} = a^{m-n}$, $a \neq 0$
5. $\left(\dfrac{a}{b}\right)^m = \dfrac{a^m}{b^m}$, $b \neq 0$
6. $a^{-m} = \dfrac{1}{a^m}$, $a \neq 0$
7. $a^{\frac{1}{n}} = \sqrt[n]{a}$
8. $a^0 = 1$, $a \neq 0$
9. $a^{\frac{m}{n}} = \sqrt[n]{a^m} = \left(\sqrt[n]{a}\right)^m$

1. $\log_a xy = \log_a x + \log_a y$
2. $\log_a \dfrac{x}{y} = \log_a x - \log_a y$
3. $\log_a x^y = y \cdot \log_a x$
4. $\log_a a^x = x$
5. $a^{\log_a x} = x$

Geometric and Arithmetic series

Geometric

x > 1

$$\sum_{k=0}^{n} x^k = \frac{x^{n+1} - 1}{x - 1}$$

x < 1

$$\sum_{k=0}^{\infty} x^k = \frac{1}{1 - x}$$

Telescoping

$$\sum_{k=1}^{n} (a_k - a_{k-1}) = a_n - a_0$$

Log products

$$\lg\left(\prod_{k=1}^{n} a_k\right) = \sum_{k=1}^{n} \lg a_k$$

Arithmetic

$$\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$$

$$\sum_{k=0}^{n} k^2 = \frac{n(n+1)(2n+1)}{6}$$

$$\sum_{i=0}^{k} ar^i = \frac{a(1 - r^{k+1})}{1 - r} \quad , for\ r \neq 1$$

Big-O notation

Big-O is an **upper** bound.

$f(n) = \mathrm{O}(\mathrm{g}(\mathrm{n}))$ meaning $f(n)$ does not increase faster than $g(n)$.

we are only interested in positive-valued functions

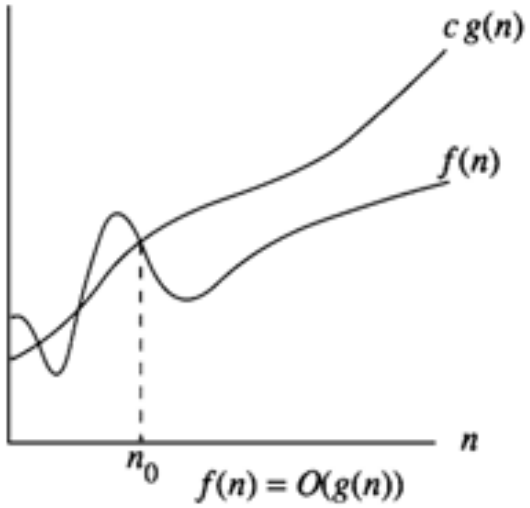
Definition 1:

$f(n) \in \mathrm{O}(\mathrm{g}(\mathrm{n})) \Leftrightarrow \exists \mathrm{c} > 0,\ n_0 \geq 0\ s.t. \forall n \geq n_0\ it\ holds\ that:$

$f(n) \leq cg(n)$

Definition 2:

$f(n) \in \mathrm{O}(\mathrm{g}(\mathrm{n})) \Leftrightarrow \exists \mathrm{k} > 0\ \mathrm{s.t.}\ \lim_{n\to\infty} \frac{f(n)}{g(n)} \leq k$



Big-O notation

Exercise 1:

- Prove that if $f(n) = 10n^2 + 5n$, then $f(n) = O(n^2)$.

Big-O notation

Exercise 1:

- Prove that if $f(n) = 10n^2 + 5n$, then $f(n) = O(n^2)$.

Solution:

- We'll find $n_0, c > 0$, such that for any $n > n_0$, $f(n) \leq cn^2$.
- Note that, for any $n \geq 1$, $f(n) = 10n^2 + 5n \leq 10n^2 + 5n^2 = 15n^2$.
- So, if $n_0 = 1, c = 15$, for any $n > n_0$, $f(n) \leq cn^2$.

Big-O notation

Exercise 2:

- Prove that if $f(n) = n\log(n^5) + 6n$, then:
  a) $f(n) = O(n \log n)$
  b) $f(n) \neq O(n)$

Big-O notation

Exercise 2:

- Prove that if $f(n) = n\log(n^5) + 6n$, then:
  a) $f(n) = O(n\log n)$
  b) $f(n) \neq O(n)$

Solution 2.a:

- We'll find $n_0, c > 0$, such that for any $n > n_0$, $f(n) \leq cn\log(n)$.
- For any $n \geq 2$:

$$f(n) = n\log(n^5) + 6n = 5n\log(n) + 6n \leq 5n\log(n) + 6n\log(n) = 11n\log(n).$$

- So, if $n_0 = 2, c = 11$, for any $n > n_0$, $f(n) \leq cn\log(n)$

Big-O notation

Exercise 2:

- Prove that if $f(n) = n\log(n^5) + 6n$, then:
  a) $f(n) = O(n \log n)$
  b) $f(n) \neq O(n)$

Big-O notation

Exercise 2:

- Prove that if $f(n) = n\log(n^5) + 6n$, then:
  a) $f(n) = O(n\log n)$
  b) $f(n) \neq O(n)$

Solution – 2.b:

- To see that $\mathbf{f(n) \neq O(n)}$, suppose towards a contradiction there exists $\mathbf{n_0, c > 0}$, such that for any $\mathbf{n > n_0}$, $\mathbf{f(n) \leq cn}$.
- In this case, we will show there exists $\mathbf{N \geq n_0}$ such that $\mathbf{f(N) > cN}$.
- Indeed, if $\mathbf{f(n) \leq cn}$, and $\mathbf{n \geq 2}$, this implies in particular: $\mathbf{n\log(n) \leq cn}$, which is equivalent to $\mathbf{\log(n) \leq c.}$
- But, clearly, if $\mathbf{n > 2^c}$, $\mathbf{\log(n) > c}$. So, if $\mathbf{N = \max(n_0, 2^c) + 1}$, then $\mathbf{N > n_0}$ and $\mathbf{f(N) > cN}$.
- It is important to note here, that the function $\mathbf{\log(n)}$ is unbounded, so it can never be true that for some constant $\mathbf{c > 0}$, the following inequality will hold $\mathbf{\log(n) \leq c}$.

Big-Ω notation

Big- Ω is a **lower** bound.

$f(n) = \Omega(\mathrm{g(n)})$ meaning $f(n)$ is bigger or equal to $g(n)$.

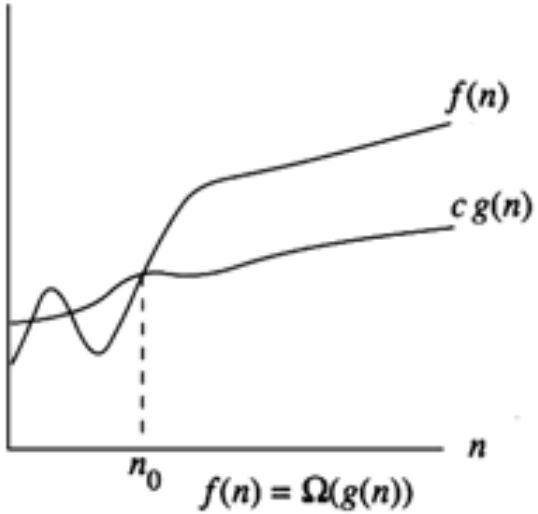
<u>Definition 1:</u>

$f(n) \in \Omega\big(\mathrm{g(n)}\big) \Leftrightarrow \exists \mathrm{c} > 0,\ n_0\ s.t. \forall n \geq n_0\ it\ holds\ that:$

$0 \leq cg(n) \leq f(n)$

<u>Equivalent definition:</u>

$f(n) \in \Omega\big(\mathrm{g(n)}\big) \Leftrightarrow \exists \mathrm{c} > 0 \text{ s.t. } \lim_{n\to\infty} \frac{f(n)}{g(n)} \geq c$



Big-Ω notation

Exercise 3:

- Prove that if $f(n) = 2^n$, then, $f(n) \in \Omega\left(3^{\sqrt{n}}\right)$.

Big-Ω notation

Exercise 3:

- Prove that if $f(n) = 2^n$, then, $f(n) \in \Omega\left(3^{\sqrt{n}}\right)$.

Solution:

- We'll find $n_0, c > 0$, such that for any $n > n_0$, $f(n) \geq 3^{\sqrt{n}}$.
- It will more convenient to consider the logarithms of the functions.
- For any $n \geq 4 > \log(3)^2$:

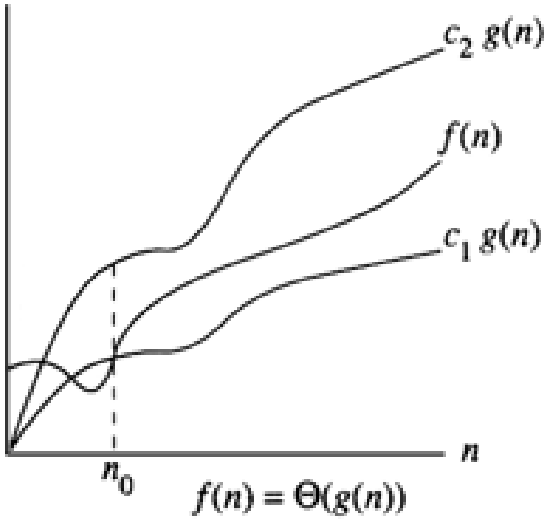
$$\log(f(n)) = n = \sqrt{n}\sqrt{n} \geq \sqrt{n}\log(3) = \log\left(3^{\sqrt{n}}\right).$$

- Since, the logarithm its monotone, this implies $f(n) \geq 3^{\sqrt{n}}$.
- So, if $n_0 = 4, c = 1$, for any $n > n_0$, $f(n) \geq 3^{\sqrt{n}}$.

Big-θ notation

Big-θ holds when both $f(n) = \mathrm{O}(\mathrm{g}(\mathrm{n}))$ and $f(n) = \Omega(\mathrm{g}(\mathrm{n}))$ holds.

$f(n) = \theta(\mathrm{g}(\mathrm{n}))$ meaning $f(n)$ is equal to $g(n)$.



Definition 1:

$$f(n) \in \theta\big(\mathrm{g}(\mathrm{n})\big) \Leftrightarrow \exists c_1, c_2 > 0,\ n_0\ s.t. \forall n \geq n_0\ it\ holds\ that:$$

$$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$$

Definition 2:

$$f(n) \in \theta\big(\mathrm{g}(\mathrm{n})\big) \Leftrightarrow \exists \mathrm{c} > 0 \text{ s.t. } \lim_{n\to\infty} \frac{f(n)}{g(n)} = c$$

In this case we can also say that $f(n) = \theta(\mathrm{g}(\mathrm{n}))$ iff $f(n) = \mathrm{O}(\mathrm{g}(\mathrm{n}))$ and $f(n) = \Omega\big(\mathrm{g}(\mathrm{n})\big)$.

Big-Θ notation

Exercise 4:

Let $f(n) = log(n!)$, prove that $f(n) \in \theta(n \log n)$.

Big-Θ notation

Exercise 4:

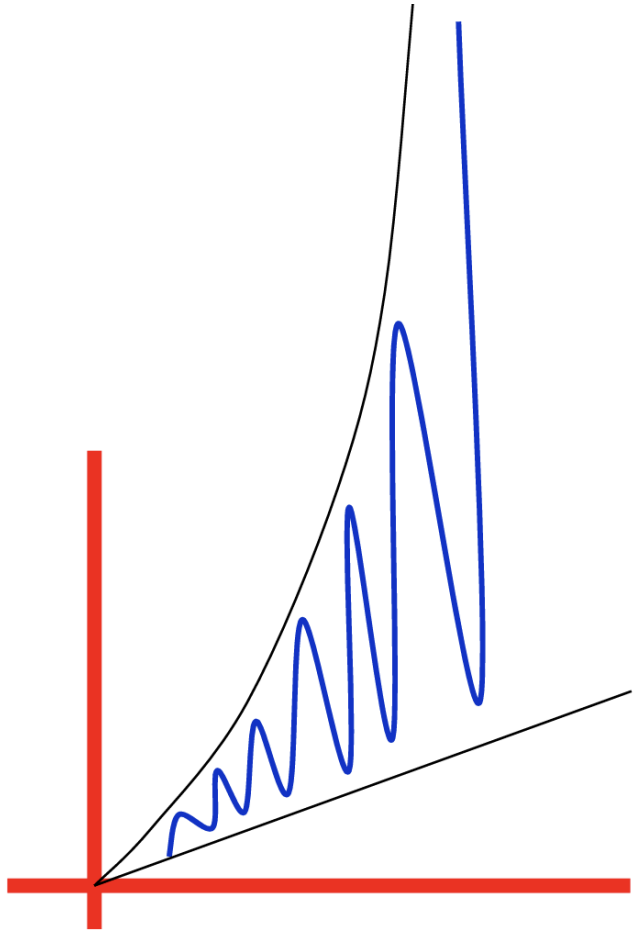
Let $f(n) = log(n!)$, prove that $f(n) \in \theta(n \log n)$.

Solution:

- To show that $\log(n!) = O(n \log n)$ we note that
  - $\log(n!) = \sum_{i=1}^{n} \log(i) \leq n\log(n)$, so $n_0 = 2$, and $c = 1$ works.
- To show that $\log n! = \Omega(n \log n)$ we note that
  - $\log(n!) = \sum_{i=1}^{n} \log(i) \geq \sum_{i=n/2}^{n} \log(i) \geq \frac{n}{2}\log\left(\frac{n}{2}\right) = \frac{1}{2}\text{nlog(n)} - \frac{1}{2}\text{nlog(2)} \geq \frac{1}{4}n\log(n)$, where the last inequality is true for every $n > 4$.
  - so $n_0 = 4$, and $c = 0.25$ works.

Not always that easy

f(n)= $n^3 + n^3 sin\, n + n^2$



Additional notations

Little-o notation:

- Strictly upper bound: $f(n) = o(g(n))$ iff:

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

Little-ω notation:

- Strictly lower bound: $f(n) = \omega(g(n))$ iff:

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

Short Summary

$$f(n) = \mathrm{O}\big(\mathrm{g(n)}\big) \approx f(n) \text{ “} \leq \text{ ”} g(n)$$
$$f(n) = \Omega\big(\mathrm{g(n)}\big) \approx f(n) \text{ “} \geq \text{ ”} g(n)$$
$$f(n) = \theta\big(\mathrm{g(n)}\big) \approx f(n) \text{ “} = \text{ “} g(n)$$

NOTE: the signs $\leq, \geq, =, <, >$ are asymptotic, and **not** real equations.

Do not use these notations.

Special Properties

Transitive

- If $f(n) = \theta(g(n))$ and $g(n) = \theta(h(n))$, than $f(n) = \theta(h(n))$
- If $f(n) = O(g(n))$ and $g(n) = O(h(n))$, than $f(n) = O(h(n))$
- If $f(n) = \Omega(g(n))$ and $g(n) = \Omega(h(n))$, than $f(n) = \Omega(h(n))$
- If $f(n) = o(g(n))$ and $g(n) = o(h(n))$, than $f(n) = o(h(n))$
- If $f(n) = \omega(g(n))$ and $g(n) = \omega(h(n))$, than $f(n) = \omega(h(n))$

Reflexive

- $f(n) = \theta(f(n))$
- $f(n) = O(f(n))$
- $f(n) = \Omega(f(n))$

Symmetric

- $f(n) = \theta(g(n))$ iff $g(n) = \theta(f(n))$

NOTE: O and Ω are not symmetric.

Asymptotic growth rates

Growth	**Terminology**
$\Theta(1)$	constant growth
$\Theta(\log n)$	logarithmic growth
$\Theta(\log^k n)$, for some $k \geq 1$	polylogarithmic growth
$\Theta(n^k)$ for some positve $k < 1$	sublinear growth
$\Theta(n)$	linear growth
$\Theta(n \log n)$	log-linear growth
$\Theta(n \log^k n)$, for some $k \geq 1$	polylog-linear growth
$\mathrm{O}(n^k)$ for some $k \geq 1$	polynomial growth
$\Omega(n^k)$, for every $k \geq 1$	superpolynomial growth
$\Omega(a^n)$ for some $a > 1$	exponential growth

Asymptotics and Induction : Be Cautious...

(False) Proposition: For $f(n) = n$ we get $f(n) = O(1)$

(False) Proof: We will show that $f(n) = O(1)$ by induction on $n \geq 1$:

- Basis – For n=1 we get $f(1) = 1 \leq 1 \cdot 1$
- Induction Hypothesis – For any $k < n$: $f(k) = O(1)$.
- Induction Step – we will show that $f(n)$ is bounded by a constant:

$$f(n) = n = f(n-1) + 1 =_{I.H} O(1) + 1 \leq c \cdot 1 + 1 \leq (c+1) \cdot 1 = c' \cdot 1$$

■

Induction + Big-O – Doing It right

Stick to the definitions:

Assume we want to prove $f(n)$ is $O(g(n))$:

- Prove by induction over n – There are c, n_0 s.t. for any $n \geq n_0$: $f(n) \leq cg(n)$:
- Base case - Prove $f(n_0) \leq cg(n_0)$
- I.H – For any $n_0 \leq k < n$: $f(k) \leq cg(k)$.
- Inductive step – Assming I.H prove that $f(n) \leq cg(n)$.
- **Important – Use the same c for all the steps – mind the order of the quantifiers in the definition of big-O.**

Prove or disprove

Exercise :

- For every two functions f and g, either $f = O(g)$ or $g = O(f)$.

Prove or disprove

Exercise :

- For every two positive valued functions f and g, either $f = O(g)$ or $g = O(f)$.

Solution:

- False. E.g.:
- $f(n) = n$,
- $g(n) = n^2$ for even n and 1 otherwise.

Prove or disprove

Exercise :

- If $f, g \geq 0$, then $f + g = \Theta(\max(f, g))$

Prove or disprove

Exercise :

- If $f, g \geq 0$, then $f + g = \Theta(\max(f, g))$

Solution:

- True. $\max(f, g) \leq f + g \leq 2\max(f, g)$.

Prove or disprove

Exercise :

- true/false? $f(n) = \Theta\left(f\left(\frac{n}{2}\right)\right)$.

Prove or disprove

Exercise :

- true/false? $f(n) = \Theta\left(f\left(\frac{n}{2}\right)\right)$.

Solution:

- False. Take $f(n) = 2^n$.