BLG336E - Analysis of Algorithms II

Recitation 2

8.03.2021

Alperen Kantarcı

Asymptotic Notation

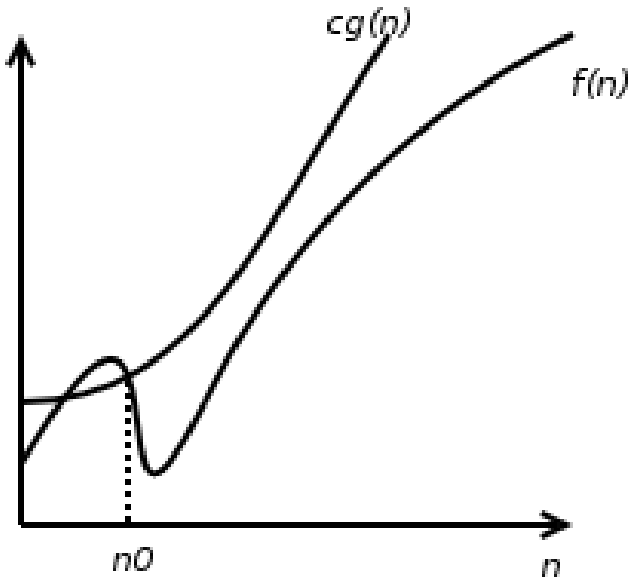


Figure 1: "There exist positive constants c and n_0 such that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0$." $f(n)$ is thus $O(g(n))$.

Asymptotic Notation

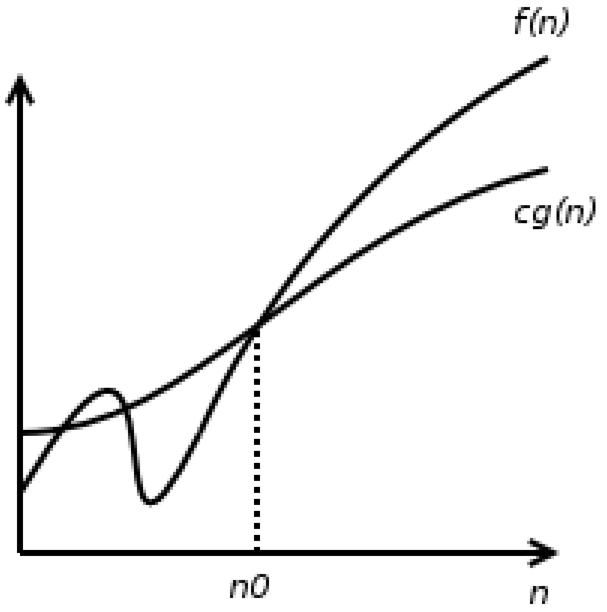


Figure 2: "There exist positive constants c and n_0 such that $0 \leq cg(n) \leq f(n)$ for all $n \geq n_0$." $f(n)$ is thus $\Omega(g(n))$.

Asymptotic Notation

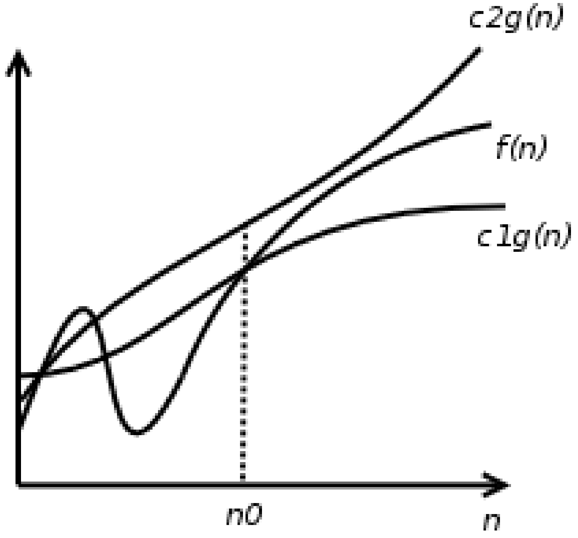
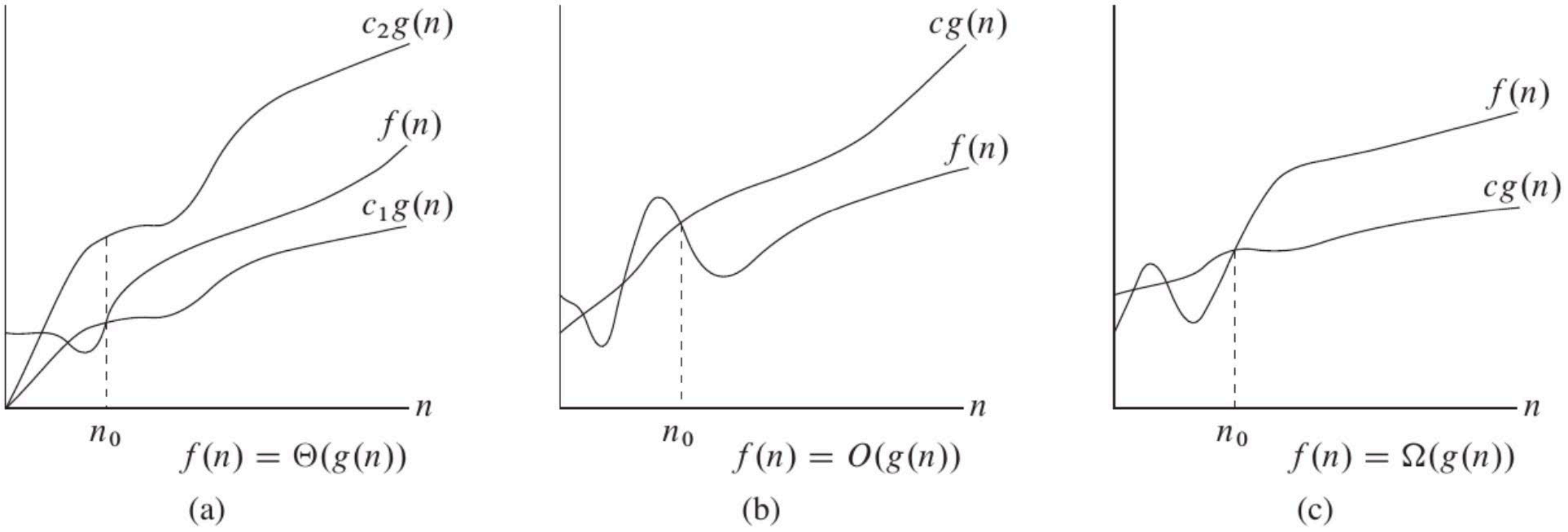


Figure 3: "There exist positive constants c_1, c_2, and n_0 such that $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$ for all $n \geq n_0$". $f(n)$ is thus $\Theta(g(n))$.

Asymptotic Notation



Question 1: True or false

$a)$ $2^{n+1} = O(2^n)$

$b)$ $2^{2n} = O(2^n)$

Solution 1a

If we can find constant n_0 and c values that satisfies the inequality of $2^{n+1} \leq c.2^n$ we can say that 2^n is an upper bound for 2^{n+1}.

$$2^{n+1} = 2.2^n$$

$$2.2^n \leq c.2^n$$

For $c = 2$ and $n \geq 1$ the inequality is satisfied.

So $2^{n+1} = O(2^n)$ is true.

Solution 1b

If we can find constant n_0 and c values that satisfies the inequality of $2^{2n} \leq c.2^n$ we can say that 2^n is an upper bound for 2^{2n}.

$$2^{2n} = 2^n 2^n$$

$$2^n.2^n \leq c.2^n$$

$$2^n \leq c$$

There is no constant c value that implies that this inequality since $2^n \leq c$ and c must be a value that depends on n.

So $2^{2n} = O(2^n)$ is false.

Question 2

- Suppose you have algorithms with five running times listed below. (Assume these are the exact running times.) How much slower do each of these algorithms get when you a) double the input size, or b) increase the input size by one?

$a)\ n^2$

$b)\ n^3$

$c)\ 100n^2$

$d)\ nlog(n)$

$e)\ 2^n$

Solution 2a: double the input size

- If $n \rightarrow 2n$ the algorithms get slower by

$a)\ n^2 \rightarrow 4n^2$: **the factor of 4**

$b)\ n^3 \rightarrow 8n^3$: **the factor of 8**

$c)\ 100n^2 \rightarrow 400n^2$: **the factor of 4**

$d)\ nlog(n) \rightarrow 2nlog(2n)$: **the factor of 2 plus an additive 2n**

$e)\ 2^n \rightarrow 2^{2n}$: **the factor of 2^n**

Solution 2b: increase the input size by one

- If $n \to n + 1$ the algorithms get slower by

$a)\ n^2 \to n^2 + 2n + 1 :$ **an additive $\mathbf{2n + 1}$**

$b)\ n^3 \to n^3 + 3n^2 + 3n + 1 :$ **an additive $\mathbf{3n^2 + 3n + 1}$**

$c)\ 100n^2 \to 100n^2 + 200n + 100 :$ **an additive $\mathbf{200n + 100}$**

$d)\ nlog(n) \to (n + 1)log(n + 1) :$ **additive $\boldsymbol{log(n + 1) + n[log(n + 1) - logn]}$**

$e)\ 2^n \to 2^{n+1}:$ **the factor of 2**

Question 3

a) Show that $\frac{1}{2}n^2 - 3n = O(n^2)$ and determine positive constants c and n_0

b) Show that $\frac{1}{2}n^2 - 3n = \theta(n^2)$ and determine positive constants c_1, c_2, and n_0

Solution 3a

$$\frac{1}{2}n^2 - 3n = O(n^2)$$

$$0 \leq \frac{1}{2}n^2 - 3n \leq cn^2$$

For all $n \geq n_0$, dividing by n^2 yields:

$$0 \leq \frac{1}{2} - \frac{3}{n} \leq c$$

The left handside inequality can be made to hold for any value of $n \geq 6$.

Thus by choosing $c = \frac{1}{2}$, $n_0 = 6$, we can verify that $\frac{1}{2}n^2 - 3n = O(n^2)$.

Solution 3b

$$c_1 n^2 \leq \frac{1}{2} n^2 - 3n \leq c_2 n^2$$

For all $n \geq n_0$, dividing by n^2 yields:

$$c_1 \leq \frac{1}{2} - \frac{3}{n} \leq c_2$$

By solving this inequality, a sample choice of solution would be:

$$c_1 = \frac{1}{14}, \mathrm{c}_2 = \frac{1}{2}, \mathrm{n}_0 = 7$$

Question 4

- Order the following functions by asymptotic growth rate:

a) n

b) $n^{1000}\log n$

c) $\log(n + n^2)$

d) n^{logn}

e) 9^n

f) 8^n

g) n^{5n}

Solution 4

Logarithms < Polynomials < Exponential with constant base < Exponential with base n

Therefore:

$$\log(n+n^2) < n < n^{1000}logn, n^{logn}, 8^n, 9^n < n^{5n}$$

We can use limit:

$$\lim_{n\to\infty}\frac{n^{1000}logn}{n^{logn}} < \lim_{n\to\infty}\frac{n^{1001}}{n^{logn}} < \lim_{n\to\infty}\frac{1}{n^{logn-1001}} = \frac{1}{\infty} = 0$$

Therefore: $n^{1000}logn < \mathrm{n}^{logn}$

Solution 4

- We can use log:

$$\log\left(n^{logn}\right) = log^2 n$$

$$\log(log^2 n) = \log(logn)^2 = 2\log(logn) = O(\log(logn))$$

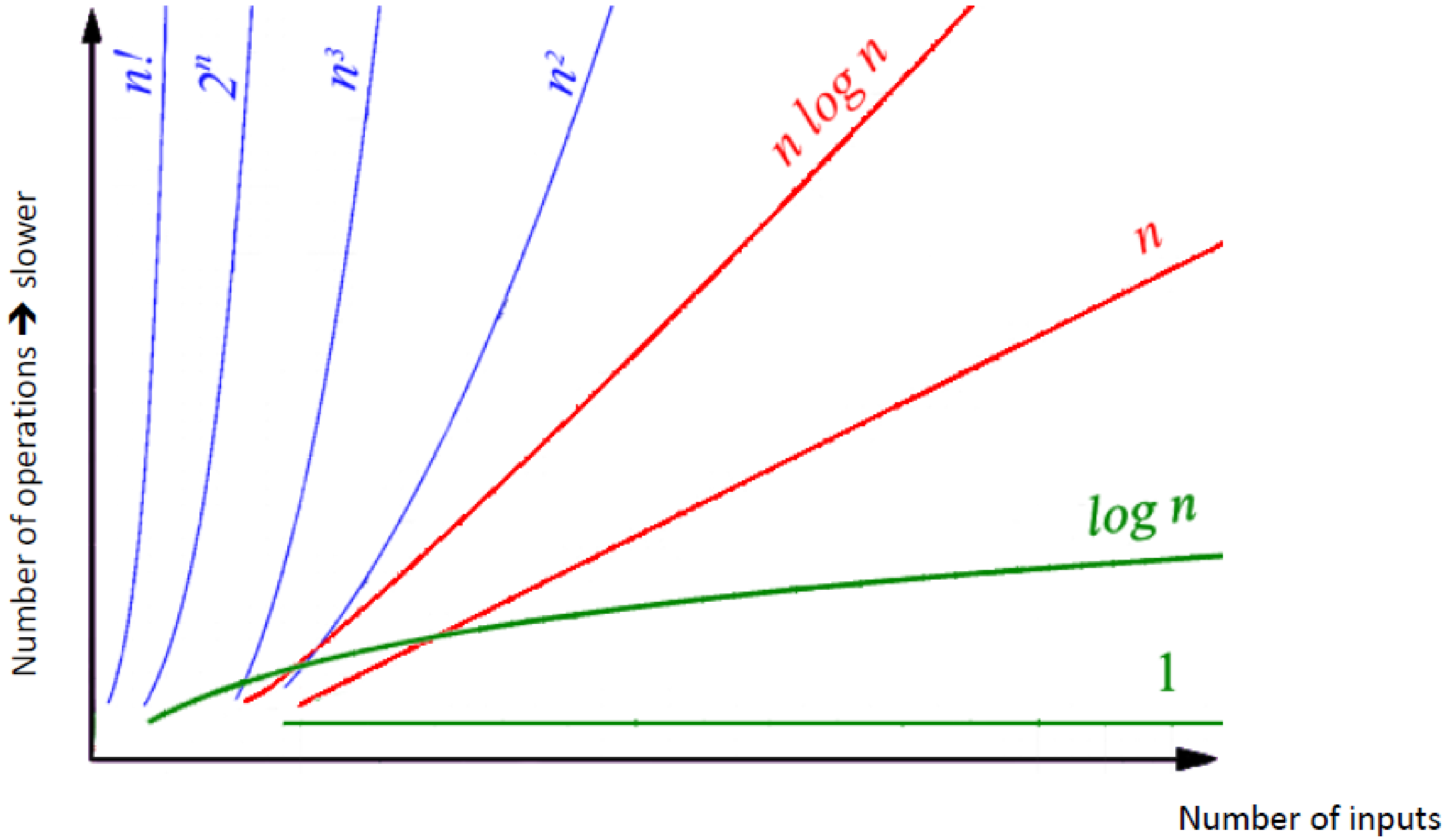
$$\log(8^n) = nlog8$$

$$\log(nlog8) = \log(n)$$

Therefore:

$$n^{logn} < 8^n$$

$$\log(n + n^2) < n < n^{1000} logn < n^{logn} < 8^n < 9^n < n^{5n}$$



$$n! \gg 2^n \gg n^3 \gg n^2$$

$$n^{5n} > 9^n > 8^n > n^2 > n^{logn} > n^{1000} logn > n > \log(n + n^2)$$

Question 5

Indicate, for each pair of expression (A, B) in the table below, whether A is O, Θ or Ω of B. Assume that $n_0 > 0$, $k \geq 1$, $\varepsilon > 0$, and $c > 1$ are constants. Your answer should be in the form of the table with "yes" or "no" written in each box.

A	B	O	θ	Ω
$lg^2 n$	n^ϵ			
n^k	c^n			
$\sqrt{n}$	$n^{\sin n}$			
2^n	$2^{n/2}$			
n^{lgc}	c^{lgn}			
$lg(n!)$	$lg(n^n)$			

Answers 5

k ≥ 1, $\varepsilon > 0$, and c > 1

A	B	O	θ	Ω
$lg^2 n$	n^{ϵ}	yes	no	no
n^k	c^n	yes	no	no
$\sqrt{n}$	$n^{\sin n}$	no	no	no
2^n	$2^{n/2}$	no	no	no
n^{lgc}	c^{lgn}	yes	yes	yes
$lg(n!)$	$lg(n^n)$	yes	yes	yes