Asymptotic Notation

Running time of an algorithm, order of growth
Worst case
Running time of an algorith increases with the size of the input in the limit as the size of the input increases without bound.

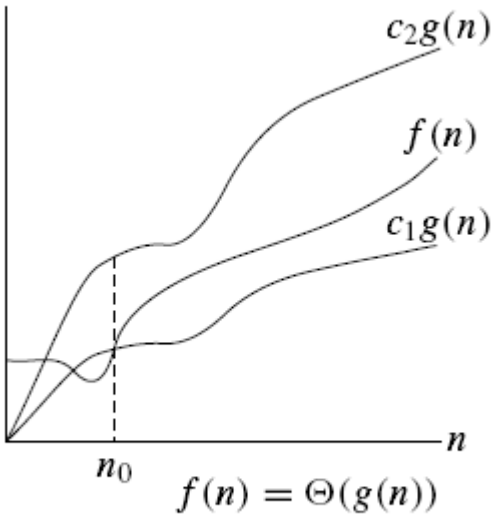
Big-theta notation

$$\Theta(g(n)) = \{f(n) : \text{there exist positive constants } c_1, c_2, \text{ and } n_0 \text{ such that } 0 \le c_1 g(n) \le f(n) \le c_2 g(n) \text{ for all } n \ge n_0\}.^{1}$$

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

"$f(n) = \Theta(g(n))$"

g(n) is an asymptotically tight bound of f(n)



(a)

Example

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

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

for all $n \ge n_0$. Dividing by n^2 yields

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

n >= 1, c2 >= 1/2
n >= 7, c1 <= 1/14

choose c1 = 1/14, c2 = ½, n0 = 7.

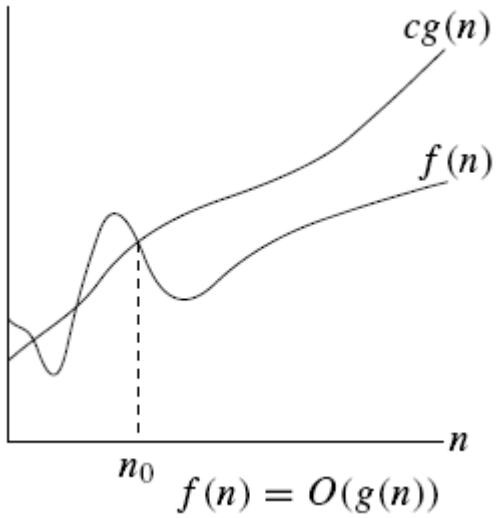
O-notation
Asymptotic upper bound

$$O(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \le f(n) \le cg(n) \text{ for all } n \ge n_0\} .$$

$$f(n) = \Theta(g(n)) \text{ implies } f(n) = O(g(n))$$

$$\Theta(g(n)) \subseteq O(g(n)).$$

f(n) = O(g(n)) some constant multiple of g(n) is an asymptotic upper bound of f(n), no claim about how tight an upper bound is.

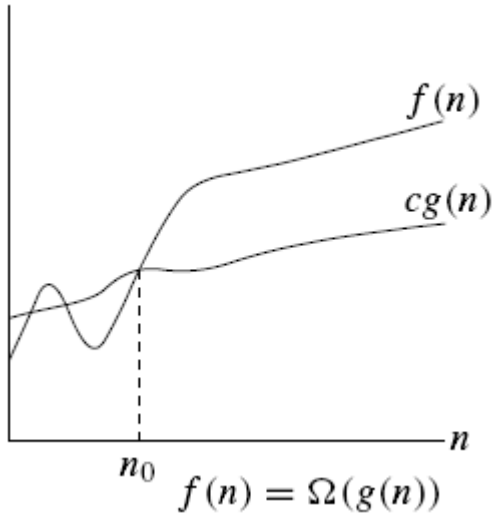


Example
The running time is $O(n^2)$ means there is a function f(n) that is $O(n^2)$ such that for any value of n, no matter what particular input of size n is chosen, the running time of that input is bounded from above by the value f(n).

Big-Omega notation

Asymptotic lowerbound

$$\Omega(g(n)) = \{f(n) : \text{ there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \le cg(n) \le f(n) \text{ for all } n \ge n_0\} .$$



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

Theorem

For any two functions $f(n)$ and $g(n)$, we have $f(n) = \Theta(g(n))$ if and only if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$. ■

When we say that the running time (no modifier) of an algorithm is Ω(g(n)), we mean that no matter what particular input of size n is chosen for each value of n, the running time on that input is at least a constant times g(n), for sufficiently large n

Interpretation

$$2n^2 + 3n + 1 = 2n^2 + \Theta(n)$$

$$2n^2 + 3n + 1 = 2n^2 + f(n),$$

not specifying all lower-terms exactly

$$T(n) = 2T(n/2) + \Theta(n)$$

$$\sum_{i=1}^{n} O(i)$$

not the same as $O(1) + O(2) + \cdots + O(n)$

$$2n^2 + \Theta(n) = \Theta(n^2)$$

"No matter how the anonymous functions are chosen on the left of the equal sign, there is a way to choose the anonymous functions on the right of the equal sign to make the equation valid."

for any function f (n) $\in$ Θ(n), there is some function g(n) $\in$Θ (n^2)
such that $2n^2$ + f (n) = g(n) for all n.

In other words, the right-hand side of an equation provides a coarser level of detail than the left-hand side.

$$\begin{aligned} 2n^2 + 3n + 1 &= 2n^2 + \Theta(n) \\ &= \Theta(n^2) \ . \end{aligned}$$

o-notation

$$o(g(n)) = \{f(n) : \text{ for any positive constant } c > 0, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \le f(n) < cg(n) \text{ for all } n \ge n_0\} \ .$$

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

w-notation

$$\omega(g(n)) = \{f(n) : \text{ for any positive constant } c > 0, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \le cg(n) < f(n) \text{ for all } n \ge n_0\} \ .$$

$$f(n) \in \omega(g(n)) \text{ if and only if } g(n) \in o(f(n)) \ .$$

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

Properties

Transitivity:

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

Reflexivity:

$$
\begin{aligned}
f(n) &= \Theta(f(n)) , \\
f(n) &= O(f(n)) , \\
f(n) &= \Omega(f(n)) .
\end{aligned}
$$

Symmetry:

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

Transpose symmetry:

$$
\begin{aligned}
f(n) &= O(g(n)) \text{ if and only if } g(n) = \Omega(f(n)) , \\
f(n) &= o(g(n)) \text{ if and only if } g(n) = \omega(f(n)) .
\end{aligned}
$$

analogy to comparison of two real numbers, a, b.

$$
\begin{aligned}
f(n) = O(g(n)) &\approx a \leq b , \\
f(n) = \Omega(g(n)) &\approx a \geq b , \\
f(n) = \Theta(g(n)) &\approx a = b , \\
f(n) = o(g(n)) &\approx a < b , \\
f(n) = \omega(g(n)) &\approx a > b .
\end{aligned}
$$

Standard notation

Floor and ceiling,

for any real x

$$x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1 .$$

for any integer n

$$\lceil n/2 \rceil + \lfloor n/2 \rfloor = n ,$$

and for any real number $n \geq 0$ and integers $a, b > 0$,

$$\begin{aligned} \lceil \lceil n/a \rceil / b \rceil &= \lceil n/ab \rceil , \\ \lfloor \lfloor n/a \rfloor / b \rfloor &= \lfloor n/ab \rfloor , \\ \lceil a/b \rceil &\leq (a + (b - 1))/b , \\ \lfloor a/b \rfloor &\geq ((a - (b - 1))/b . \end{aligned}$$

Modular arithmetic

$$a \bmod n = a - \lfloor a/n \rfloor n .$$

if (a mod n) = (b mod n), we write

$$a \equiv b \pmod{n}$$

a is equivalent to b, modulo n. in other words, a and b have the same remainder when divided by n. Or n is a divisor of b – a.

Polynomials

$$p(n) = \sum_{i=0}^{d} a_i n^i$$

$$p(n) = \Theta(n^d)$$

Exponentials

For all real constants a and b such that a > 1

$$\lim_{n\to\infty} \frac{n^b}{a^n} = 0$$

we can conclude that

$$n^b = o(a^n)$$

Thus, any exponential function with a base strictly greater than 1 grows faster than any polynomial function.

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots = \sum_{i=0}^{\infty} \frac{x^i}{i!} ,$$

$$e^x \geq 1 + x ,$$

when

$$|x| \leq 1,$$

we have the approximation

$$1 + x \leq e^x \leq 1 + x + x^2 .$$

When $x \to 0$, the approximation of e^x by $1 + x$ is quite good:

$$e^x = 1 + x + \Theta(x^2) .$$

$$\lim_{n\to\infty} \left(1 + \frac{x}{n}\right)^n = e^x$$

Logarithms

$$\begin{aligned} \lg n &= \log_2 n \quad \text{(binary logarithm)} , \\ \ln n &= \log_e n \quad \text{(natural logarithm)} , \\ \lg^k n &= (\lg n)^k \quad \text{(exponentiation)} , \\ \lg \lg n &= \lg(\lg n) \quad \text{(composition)} . \end{aligned}$$

for all real a > 0, b > 0, c > 0 and n

$$
\begin{aligned}
a &= b^{\log_b a}\,, \\
\log_c(ab) &= \log_c a + \log_c b\,, \\
\log_b a^n &= n \log_b a\,, \\
\log_b a &= \frac{\log_c a}{\log_c b}\,,
\end{aligned}
$$

$$
\begin{aligned}
\log_b(1/a) &= -\log_b a\,, \\
\log_b a &= \frac{1}{\log_a b}\,, \\
a^{\log_b c} &= c^{\log_b a}\,,
\end{aligned}
$$

where logarithm bases are not 1.

Changing the base of a logarithm from one constant to another only changes the value of the logarithm by a constant factor, and so we shall often use the notation "lg n" when we don't care about constant factors.

There is a simple series expansion for $\ln(1+x)$ when $|x| < 1$:

$$\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \frac{x^5}{5} - \cdots\,.$$

We also have the following inequalities for $x > -1$:

$$\frac{x}{1+x} \le \ln(1+x) \le x\,,$$

where equality holds only for $x = 0$.

polylogarithmic bound

$$f(n) = O(\lg^k n)$$

$$\lim_{n\to\infty} \frac{\lg^b n}{(2^a)^{\lg n}} = \lim_{n\to\infty} \frac{\lg^b n}{n^a} = 0\,.$$

From this limit, we can conclude that

$$\lg^b n = o(n^a)$$

for any constant a > 0. Thus, any positive polynomial function grows faster than any polylogarithmic function.

Factorials

n >= 0

$$n! = \begin{cases} 1 & \text{if } n = 0 , \\ n \cdot (n-1)! & \text{if } n > 0 . \end{cases}$$

Thus, $n! = 1 \cdot 2 \cdot 3 \cdots n$.

A weak upper bound on the factorial function is $n! \leq n^n$, since each of the n terms in the factorial product is at most n. Stirling's approximation,

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right) ,$$

gives a tighter upper and lower bounds.

$$\begin{aligned} n! &= o(n^n) , \\ n! &= \omega(2^n) , \\ \lg(n!) &= \Theta(n \lg n) , \end{aligned}$$

Function iteration

$$f^{(i)}(n) = \begin{cases} n & \text{if } i = 0 , \\ f(f^{(i-1)}(n)) & \text{if } i > 0 . \end{cases}$$

Iterated logarithm function

$$\lg^* n = \min \{i \geq 0 : \lg^{(i)} n \leq 1\}$$

reads "log star of n"

The iterated logarithm is a very slowly growing function:

$$
\begin{aligned}
\lg^* 2 &= 1 , \\
\lg^* 4 &= 2 , \\
\lg^* 16 &= 3 , \\
\lg^* 65536 &= 4 , \\
\lg^*(2^{65536}) &= 5 .
\end{aligned}
$$

Be sure to distinguish $\lg^{(i)} n$ (the logarithm function applied I times in succession, starting with argument n) from $\lg^i n$ (the logarithm of n raised to the ith power).

Fibonacci numbers

$$
\begin{aligned}
F_0 &= 0 , \\
F_1 &= 1 , \\
F_i &= F_{i-1} + F_{i-2} \qquad \text{for } i \geq 2 .
\end{aligned}
$$

$$
\begin{aligned}
\phi &= \frac{1+\sqrt{5}}{2} \\
&= 1.61803\ldots , \\
\widehat{\phi} &= \frac{1-\sqrt{5}}{2} \\
&= -.61803\ldots .
\end{aligned}
$$

Specifically, we have

$$
F_i = \frac{\phi^i - \widehat{\phi}^i}{\sqrt{5}} ,
$$

Homework

Rank the following functions by order of growth

$\lg(\lg^* n)$	$2^{\lg^* n}$	$(\sqrt{2})^{\lg n}$	n^2	$n!$	$(\lg n)!$
$(\frac{3}{2})^n$	n^3	$\lg^2 n$	$\lg(n!)$	2^{2^n}	$n^{1/\lg n}$
$\ln \ln n$	$\lg^* n$	$n \cdot 2^n$	$n^{\lg \lg n}$	$\ln n$	1
$2^{\lg n}$	$(\lg n)^{\lg n}$	e^n	$4^{\lg n}$	$(n+1)!$	$\sqrt{\lg n}$
$\lg^*(\lg n)$	$2^{\sqrt{2 \lg n}}$	n	2^n	$n \lg n$	$2^{2^{n+1}}$