

Order of magnitude analysis requires a number of mathematical definitions and theorems. The most basic concept is commonly termed big-O.

Definition: Suppose that $f(n)$ and $g(n)$ are nonnegative functions of n . Then we say that $f(n)$ is $O(g(n))$ provided that there are constants $C > 0$ and $N > 0$ such that for all $n > N$, $f(n) \leq Cg(n)$.

By the definition above, demonstrating that a function f is big-O of a function g requires that we find specific constants C and N for which the inequality holds (and show that the inequality does, in fact, hold).

Big-O expresses an upper bound on the growth rate of a function, for sufficiently large values of n .

Take the function obtained in the algorithm analysis example earlier:

$$T(n) = \frac{3}{2}n^2 + \frac{5}{2}n - 3$$

Intuitively, one should expect that this function grows similarly to n^2 .

To show that, we will shortly prove that:

$$\frac{5}{2}n < 5n^2 \text{ for all } n \geq 1 \quad \text{and that} \quad -3 < n^2 \text{ for all } n \geq 1$$

Why? Because then we can argue by substitution (of non-equal quantities):

$$T(n) = \frac{3}{2}n^2 + \frac{5}{2}n - 3 \leq \frac{3}{2}n^2 + n^2 + n^2 = \frac{7}{2}n^2 \text{ for all } n \geq 1$$

Thus, applying the definition with $C = 7/2$ and $N = 1$, $T(n)$ is $O(n^2)$.

Theorem: if $a = b$ and $0 \leq c \leq d$ then $a*c \leq b*d$.

Claim: $\frac{5}{2}n < 5n^2$ for all $n \geq 1$

proof: Obviously $5/2 < 5$, so $(5/2)n < 5n$. Also, if $n \geq 1$ then by the Theorem above, $5n \leq 5n^2$. Hence, since \leq is transitive, $(5/2)n \leq 5n^2$.

Claim: $-3 < n^2$ for all $n \geq 1$

proof: For all n , $n^2 \geq 0$. Obviously $0 \geq -3$. So by transitivity ...

For all the following theorems, assume that $f(n)$ is a function of n and that K is an arbitrary constant.

Theorem 1: K is $O(1)$

Theorem 2: A polynomial is $O(\text{the term containing the highest power of } n)$

$$f(n) = 7n^4 + 3n^2 + 5n + 1000 \text{ is } O(7n^4)$$

Theorem 3: $K \cdot f(n)$ is $O(f(n))$ [i.e., constant coefficients can be dropped]

$$g(n) = 7n^4 \text{ is } O(n^4)$$

Theorem 4: If $f(n)$ is $O(g(n))$ and $g(n)$ is $O(h(n))$ then $f(n)$ is $O(h(n))$. [transitivity]

$$f(n) = 7n^4 + 3n^2 + 5n + 1000 \text{ is } O(n^4)$$

Theorem 5: Each of the following functions is big-O of its successors:

K [constant]

$\log_b(n)$ [always log base 2 if no base is shown]

n

$n \log_b(n)$

n^2

n to higher powers

2^n

3^n

larger constants to the n-th power

$n!$ [n factorial]

n^n

smaller



larger

$$f(n) = 3n \log(n) \text{ is } O(n \log(n)) \text{ and } O(n^2) \text{ and } O(2^n)$$

Theorem 6: In general, $f(n)$ is big-O of the dominant term of $f(n)$, where “dominant” may usually be determined from Theorem 5.

$$f(n) = 7n^2 + 3n \log(n) + 5n + 1000 \text{ is } O(n^2)$$

$$g(n) = 7n^4 + 3^n + 1000000 \text{ is } O(3^n)$$

$$h(n) = 7n(n + \log(n)) \text{ is } O(n^2)$$

Theorem 7: For any base b , $\log_b(n)$ is $O(\log(n))$.

In addition to big-O, we may seek a lower bound on the growth of a function:

Definition: Suppose that $f(n)$ and $g(n)$ are nonnegative functions of n . Then we say that $f(n)$ is $\Omega(g(n))$ provided that there are constants $C > 0$ and $N > 0$ such that for all $n > N$, $f(n) \geq Cg(n)$.

Big- Ω expresses a lower bound on the growth rate of a function, for sufficiently large values of n .

Finally, we may have two functions that grow at essentially the same rate:

Definition: Suppose that $f(n)$ and $g(n)$ are nonnegative functions of n . Then we say that $f(n)$ is $\Theta(g(n))$ provided that $f(n)$ is $O(g(n))$ and also that $f(n)$ is $\Omega(g(n))$.

The task of determining the order of a function is simplified considerably by the following result:

Theorem 8: $f(n)$ is $\Theta(g(n))$ if

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c \text{ where } 0 < c < \infty$$

Recall Theorem 7... we may easily prove it (and a bit more) by applying Theorem 8:

$$\lim_{n \rightarrow \infty} \frac{\log_b(n)}{\log(n)} = \lim_{n \rightarrow \infty} \frac{\frac{1}{n \ln(b)}}{\frac{1}{n \ln(2)}} = \lim_{n \rightarrow \infty} \frac{\ln(2)}{\ln(b)} = \frac{\ln(2)}{\ln(b)}$$

The last term is finite and positive, so $\log_b(n)$ is $\Theta(\log(n))$ by Theorem 8.

Some of the big-O theorems may be strengthened now:

Theorem 9: If $K > 0$ is a constant, then K is $\Theta(1)$.

Theorem 10: A polynomial is Θ (the highest power of n).

proof: Suppose a polynomial of degree k . Then we have:

$$\lim_{n \rightarrow \infty} \frac{a_0 + a_1 n + \dots + a_k n^k}{n^k} = \lim_{n \rightarrow \infty} \left(\frac{a_0}{n^k} + \frac{a_1}{n^{k-1}} + \dots + \frac{a_{k-1}}{n} + a_k \right) = a_k$$

Now $a_k > 0$ since we assume the function is nonnegative. So by Theorem 8, the polynomial is $\Theta(n^k)$.

QED

Theorem 11: If $f(n)$ is $\Theta(g(n))$ and $g(n)$ is $\Theta(h(n))$ then $f(n)$ is $\Theta(h(n))$. [transitivity]

This follows from Theorem 4 and the observation that big- Ω is also transitive.

Theorem 12: If $f(n)$ is $\Theta(g(n))$ then $g(n)$ is $\Theta(f(n))$. [symmetry]

Theorem 13: If $f(n)$ is $\Theta(f(n))$. [reflexivity]

By Theorems 11–13, Θ is an equivalence relation on the set of nonnegative functions.

The equivalence classes represent fundamentally different growth rates.

Ex 1: An algorithm (e.g, see slide 2.9) with complexity function

$$T(n) = \frac{3}{2}n^2 + \frac{5}{2}n - 3 \quad \text{is } \Theta(n^2) \text{ by Theorem 10.}$$

Ex 2: An algorithm (e.g, see slide 2.10) with complexity function

$$T(n) = 3n \log n + 4 \log n + 2 \quad \text{is } O(n \log(n)) \text{ by Theorem 5.}$$

Furthermore, the algorithm is also $\Theta(n \log(n))$ by Theorem 8 since:

$$\lim_{n \rightarrow \infty} \frac{T(n)}{n \log n} = \lim_{n \rightarrow \infty} \left(3 + \frac{4}{n} + \frac{2}{n \log n} \right) = 3$$

For most common complexity functions, it's this easy to determine the big-O and/or big- Θ complexity using the given theorems.