Cheatsheet - Asymptotic Analysis

Fabio Lama – fabio.lama@pm.me

1. About

Asymptotic analysis is an alternative way of describing the time or memory requirements of an algorithm.

2. Big O Notation

Big O notation $O(x)$ defines a set of functions that act as an **upper bound** $g(N)$ for $T(N)$. Formally defined as:

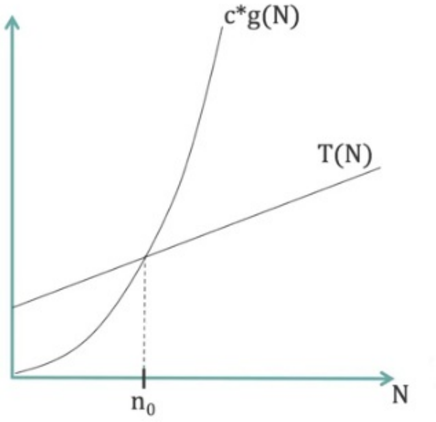
$T(N)$ is $O(g(N))$ if there exist positive constants c and n_0 such that:

$$T(N) \leq c \times g(N) \quad \text{for all} \quad N > n_0$$

Alternatively:

$$T(N) \in O(g(N)) \Leftrightarrow \exists c > 0 \exists n_0 > 0 \forall N (N \leq n_0 \rightarrow c \times g(N) \geq T(N))$$

Note that there can be **multiple** functions $g_x(N)$ that act as an upper bound for $T(N)$. Additionally, do notice that it's **not necessary** that $c \times g(N)$ is equal to or greater than $T(N)$ **for all values** of N.



For example, consider:

$$T(N) = 10N^2 + 15N + 5$$
$$g(N) = N^2$$
$$c = 1$$

Here, $c \times g(N)$ is never greater than $T(N)$, because there is no solution for:

$$10N^2 + 15N + 5 \leq 1 \times N^2$$

However, consider:

$$c = 25$$

In case of $N = 1$ we get:

$$10 \times 1^2 + 15 \times 1 + 5 \leq 25 \times 1^2$$
$$= 10 + 15 + 5 \leq 25$$
$$= 30 \leq 25$$

Which is false. However, for $N = 2$ we get:

$$10 \times 2^2 + 15 \times 2 + 5 \leq 25 \times 2^2$$
$$= 40 + 30 + 5 \leq 100$$
$$= 75 \leq 100$$

Which is true. Therefore:

$$T(N) \quad \text{is} \quad O(N^2) \quad \text{because}$$
$$T(N) \leq 25 \times g(N) \quad \text{for all} \quad N \geq 2$$

There choice for c **is arbitrary**, as long as it satisfies the conditions.

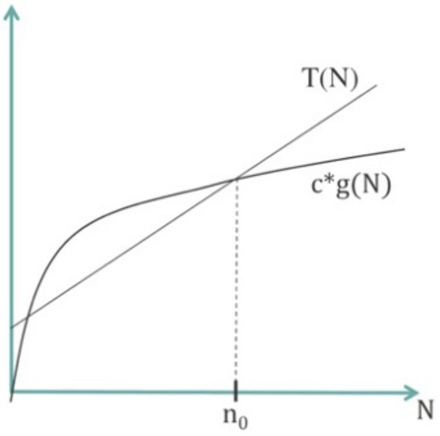
3. Omega Notation

The Omega notation $\Omega(x)$ defines a set of functions that act as a **lower bound** $g(N)$ for ($T(N)$). Formally defined as:

$T(N)$ is $\Omega(g(N))$ if there exist positive constants c and n_0 such that:

$$T(N) \geq c \times g(N) \quad \text{for all} \quad N > n_0$$

Similarly to the Big O notation, there can be multiple functions $g_x(N)$ that act as a lower bound for $T(N)$ and it's **not necessary** that $c \times g(N)$ is equal to or less than $T(N)$ **for all values** of N, but only for the **larger values**.



4. Theta Notation

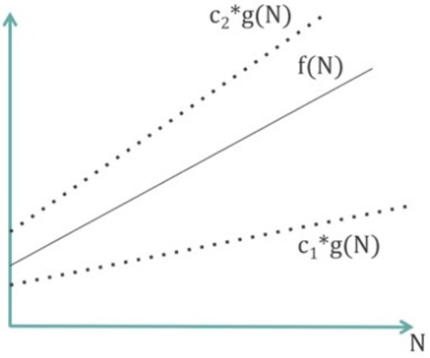
The Theta notation $\Theta(x)$ defines a **single function** that acts as both an **upper and lower bound** for ($T(N)$). Formally defined as:

$T(N)$ is $\Theta(g(N))$ if there exist positive constants c_1, c_2 and n_o such that both those conditions hold true:

$$T(N) \geq c_1 \times g(N) \quad \text{for all} \quad N > n_0$$
$$T(N) \leq c_2 \times g(N) \quad \text{for all} \quad N > n_0$$

Alternatively:

$$c_1 \times g(N) \leq T(N) \leq c_2 \times g(N) \quad \text{for all} \quad N > n_0$$



As already noted, Theta notation has **only one function**.

Last updated 2024-05-21 19:56:20 UTC