**ANNY CAROLINE WALKER SILVA, 1201324404**

# LIMITE DE THETA Θ

- **O que é Θ?**

A notação Θ descreve o limite assintótico firme, é utilizado para analisar o limite inferior e limite superior do crescimento do processamento de um algoritmo, pois o Θ fica entre esses dois limites. Por definição, Θ é:

Θ(g(n)) = f(n), se existem constantes positivas $c_1$, $c_2$ e $n_0$

tais que 0 ≤ $c_1$ g(n) ≤ f(n) ≤ $c_2$ g(n)

para todo n ≥ n0.

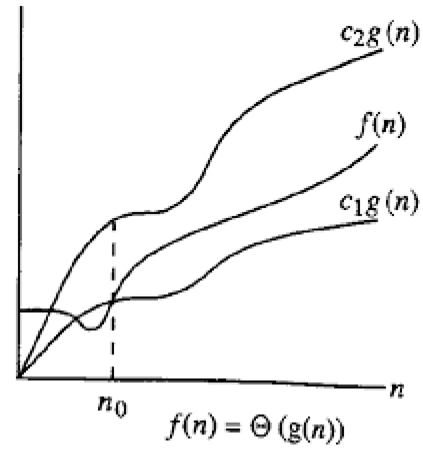


Figura 1. Gráfico representando notações assintóticas.
Fonte: Prof. Moacir Ponti Jr

Percebemos então que Θ fica entre $c_1$g(n) e $c_2$g(n), que representam respectivamente o limite inferior e o limite superior.

- **O que é Ω?**

A notação Ω represente o limite assintótico inferior do crescimento do algoritmo. Por definição:

Ω(g(n)) = f(n), se existem constantes positivas c e $n_0$

tais que 0 ≤ c g(n) ≤ f(n)

para todo n ≥ $n_0$

Para encontrar Ω através do limite, temos que:

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

O resultado do limite de $\frac{f(n)}{g(n)}$, com n tendendo a ∞, em relação a Ω resulta em ∞, pois f(n) cresce infinitamente com relação a g(n), tornando g(n) desprezível em relação a grandeza de f(n), pois algo muito grande dividido por algo muito pequeno, ainda resulta em algo muito grande.

**ANNY CAROLINE WALKER SILVA, 1201324404**

- **O que é O?**

A notação O represente o limite assintótico superior do crescimento do algoritmo. Por definição:

O(g(n)) = f(n), se existem constantes positivas c e $n_0$

tais que 0 ≤ f(n) ≤ c g(n)

para todo n ≥ $n_0$

Para encontrar O através do limite, temos que:

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

O resultado do limite de $\frac{f(n)}{g(n)}$, com n tendendo a ∞, em relação a O resulta em 0, pois g(n) cresce infinitamente com relação a f(n), tornando f(n) desprezível em relação a grandeza de g(n), pois algo muito pequeno dividido por algo muito grande, ainda resulta em algo muito pequena.

- **Como definir Θ usando limite?**

Levando em consideração os limites inferior e superior e a definição de Θ, que fica entre esses dois limites, podemos assumir que:

$$0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty, \text{ f=}\Theta\text{(g)}$$

O que nos permite concluir que o conjunto de valores de $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ que se aplicam em f=Θ(g) é dos números inteiros finitos, sendo maiores que 0 e menor que ∞.

Conjunto de valores de $\lim_{n\to\infty} \frac{f(n)}{g(n)}$, f=Θ(g): {n | n é um número real tal que n > 0}

## LIMITE INFERIOR DO ALGORITMO DE CHECAGEM DE NÚMEROS PRIMOS

AKS primality test ou Agrawal–Kayal–Saxena primality test é atualmente o algoritmo detentor do limite inferior possível para a checagem de números primos, sendo Ω:

$$\boldsymbol{\Omega\,(\log n^6)}$$

**ANNY CAROLINE WALKER SILVA, 1201324404**

## MOSTRE O QUE É PEDIDO

**a)** $n \log n + 5n = \Theta(n \log n)$

$$c_1\, n \log n \le n \log n + 5n \le c_2\, n \log n, \text{ para todo } n \ge n_0$$

Dividindo por $n \log n$:

$$c_1 \le 1 + \frac{5}{\log n} \le c_2$$

A desigualdade do lado direito, pode ser considerada válida para qualquer valor de n ≥ 18, escolhendo n=18, temos $c_2 \approx$ 2,7. Enquanto a desigualdade lado esquerdo, pode ser considerada válida para qualquer valor de n ≥ -18, escolhendo n=-18, temos $c_1 \approx$ -1,8. Ou seja, existe pelo menos um valor de $c_1$ e $c_2$que se satisfaçam para $\mathbf{n \log n + 5n}$ quando **$\Theta(n \log n)$.**

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

$$0 \le 2^{n+1} \le c\, 2^n$$

Dividindo por $2^n$:

$$0 \le 2 \le c$$

A desigualdade do lado direito, pode ser considerada válida para qualquer valor de n ≥ 2, escolhendo n=3, temos $c \approx 3$. Ou seja, existe pelo menos um valor de c se satisfaça para $2^{n+1}$ quando **$O(2^n)$.**

**c)** $2^n - 1 = \Omega(n^2)$

$$0 \le c\, n^2 \le 2^n - 1$$

Complexo resolver através da definição, usaremos então a definição por limite de Ω:

$\lim_{n \to \infty} \frac{2^n - 1}{n^2}$, resolvendo, temos que f(n) ou $2^n - 1$ cresce infinitamente comparado a g(n) ou $n^2$, resultando em ∞. Existe satisfação, porém ela tende ao infinito.

**d)** $n^2 = o(n^3)$

Para qualquer constante c > 0, existe uma constante $n_0$ > 0, tal que $0 \le n^2 \le c\, n^3$, para todo $n \ge n_0$.

Dividindo por $n^2$, temos:

$$0 \le 1 \le c\, n$$

Escolhendo c>0, podemos satisfazer o limite superior o para qualquer n>0, de acordo como a definição.

**ANNY CAROLINE WALKER SILVA, 1201324404**

**e)** $\boldsymbol{n^2}$ **= ω ($\boldsymbol{n}$)**

Para qualquer constante c > 0, existe uma constante $n_0$ > 0, tal que 0 ≤ c $n$ ≤ $\boldsymbol{n^2}$, para todo n ≥ $n_0$.

Dividindo por $\boldsymbol{n}$, temos:

0 ≤ c ≤ n

Escolhendo c=1 e n=1 já satisfazemos as propriedades c>0 e $n_0$ > 0, cumprindo 0 ≤ c ≤ n.

**f)** $\boldsymbol{n \log n}$ **= o ($\boldsymbol{n^2}$)**

Para qualquer constante c > 0, existe uma constante $n_0$ > 0, tal que 0 ≤ $\boldsymbol{n \log n}$ ≤ c $\boldsymbol{n^2}$, para todo n ≥ $n_0$.

Dividindo por $\boldsymbol{n^2}$, temos:

0 ≤ $\frac{\log n}{n}$ ≤ c

Escolhendo c>0, podemos satisfazer o limite superior o para qualquer n>0, pois $\frac{\log n}{n}$ com n>0 é sempre menor que 1.

**g)** $\boldsymbol{2n^2}$ **≠ o ($\boldsymbol{n^2}$)**

Para qualquer constante c > 0, existe uma constante $n_0$ > 0, tal que 0 ≤ $\boldsymbol{2n^2}$ ≤ c $\boldsymbol{n^2}$, para todo n ≥ $n_0$.

Dividindo por $\boldsymbol{n^2}$, temos:

0 ≤ 2 ≤ c

Temos a quebra da propriedade c > 0, pois no caso 0 ≤ 2 ≤ c, c deveria ser sempre maior ou igual a 2.

**REFERÊNCIAIS**

MOACIR PONTI JR. 03 – Análise de Algoritmos (parte 3). Disponível em: < https://edisciplinas.usp.br/pluginfile.php/2027803/mod_resource/content/1/ICC2_03.AnalisedeAlgoritmos_parte3.pdf>. Acesso em 26 de agosto de 2019.