GA-026: Algoritmos I

Prof. Luiz Gadelha

Programa de Pós-Graduação em Modelagem Computacional, P4/2019
Laboratório Nacional de Computação Científica

3 de outubro de 2019

Crescimento de Funções: Notação Θ

- Seja $g(n) : \mathbb{R} \to \mathbb{R}$.

$$\Theta(g(n)) = \{f(n) : \text{existem constantes positivas } c_1, c_2, \text{ e } n_0$$
$$\text{tais que } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ para todo } n \geq n_0\}$$

- Embora $\Theta(g(n))$ seja um conjunto, é comum usar a notação $f(n) = \Theta(g(n))$ ao invés de $f(n) \in \Theta(g(n))$.
- Exemplo: $2n^2 - n = \Theta(n^2)$.
  - Precisamos encontrar c_1, c_2 e n_0 tais que $c_1 n^2 \leq 2n^2 - n \leq c_2 n^2$.
  - Para $n > 0$ temos $c_1 \leq 2 - \frac{1}{n} \leq c_2$.
  - Tomando $c_1 = 1$, $c_2 = 2$ e $n_0 = 1$ temos uma combinação que satisfaz a desigualdade para $n \geq n_0$.

Crescimento de Funções: Notação Θ

- Seja $g(n) : \mathbb{R} \rightarrow \mathbb{R}$.

$$\Theta(g(n)) = \{f(n) : \text{existem constantes positivas } c_1, c_2, \text{ e } n_0$$

$$\text{tais que } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ para todo } n \geq n_0\}$$

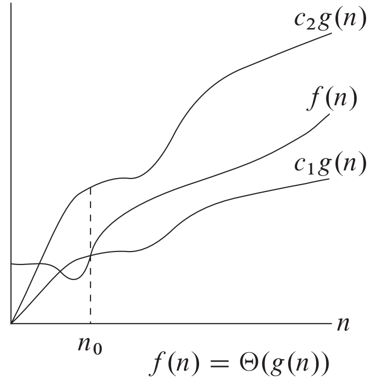
- Exemplo: $2n^3 \neq \Theta(n^2)$.
  - Precisamos encontrar c_1, c_2 e n_0 tais que $c_1 n^2 \leq 2n^3 \leq c_2 n^2$.
  - Para $n > 0$ temos $c_1 \leq 2n \leq c_2$.
  - $2n \leq c_2$ implica que $n \leq \frac{c_2}{2}$ e essa desigualdade não será verdadeira para n suficientemente grande.

Laboratório Nacional de Computação Científica

Crescimento de Funções: Notação Θ

- Em geral podemos ignorar termos de menor ordem em uma função função positiva assintoticamente.
- P.ex., para $f(n) = an^2 + bn + c$ $(a > 0)$ podemos verificar $f(n) = \Theta(n^2)$ escolhendo $c_1 = \frac{a}{2}$ e $c_2 = \frac{3a}{2}$:
  - $c_1 n^2 = \frac{a}{2} n^2 \leq an^2 + bn + c$
    $\Rightarrow \frac{a}{2} n^2 + bn + c \geq 0$ (1)
  - $an^2 + bn + c \leq c_2 n^2 = \frac{3a}{2} n^2$
    $\Rightarrow -\frac{a}{2} n^2 + bn + c \leq 0$ (2)
  - Sabemos que existe n_0 a partir do qual (1) e (2) são verdadeiras. (**Exercício**)
  - Uma constante $c = \Theta(n^0)$, que representaremos como $\Theta(1)$.

Crescimento de Funções: Notação Θ



- Fonte: Cormen, T., Leiserson, C., Rivest, R., Stein, C. (2009). Introduction to Algorithms. MIT Press.

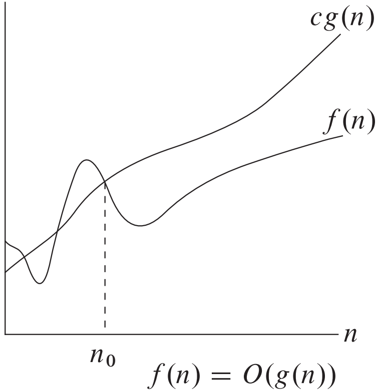
Crescimento de Funções: Notação O

- Seja $g(n) : \mathbb{R} \to \mathbb{R}$.

$$O(g(n)) = \{f(n) : \text{existe constante positiva } c \text{ e } n_0$$
$$\text{tal que } 0 \leq f(n) \leq cg(n) \text{ para todo } n \geq n_0\}$$

- A notação O é usada para definir um limite superior assintótico.
- $f(n) = \Theta(g(n)) \Rightarrow f(n) = O(g(n))$.
- A recíproca não é verdadeira:
  - $an + b = O(n^2)$.
  - $an + b \neq \Theta(n^2)$.
- A notação O pode ser usada para descrever o pior caso de complexidade de tempo de execução de um algoritmo.
  - Nesse caso, o limite serve para qualquer entrada do algoritmo (além do pior caso).

Crescimento de Funções: Notação O



- Fonte: Cormen, T., Leiserson, C., Rivest, R., Stein, C. (2009). Introduction to Algorithms. MIT Press.

Laboratório Nacional de Computação Científica

Crescimento de Funções: Notação Ω

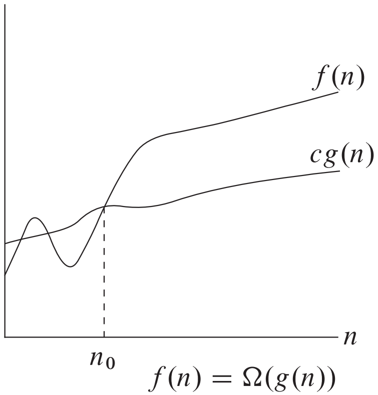
- Seja $g(n) : \mathbb{R} \rightarrow \mathbb{R}$.

$$\Omega(g(n)) = \{f(n) : \text{existe constante positiva } c \text{ e } n_0$$

$$\text{tal que } 0 \leq cg(n) \leq f(n) \text{ para todo } n \geq n_0\}$$

- A notação Ω é usada para definir um limite inferior assintótico.
- $f(n) = \Theta(g(n)) \Rightarrow f(n) = \Omega(g(n))$.
- A recíproca não é verdadeira:
  - $n^3 = \Omega(n^2)$.
  - $n^3 \neq \Theta(n^2)$.
- A notação Ω pode ser usada para descrever o melhor caso de complexidade de tempo de execução de um algoritmo.
  - Nesse caso, o limite serve para qualquer entrada do algoritmo (além do melhor caso).

Crescimento de Funções: Notação Ω



- Fonte: Cormen, T., Leiserson, C., Rivest, R., Stein, C. (2009). Introduction to Algorithms. MIT Press.

- **Teorema**. Para quaisquer funções $f(n)$ e $g(n)$, temos $f(n) = \Theta(g(n)) \Leftrightarrow f(n) = O(g(n))$ e $f(n) = \Omega(g(n))$.
- **Exercício**. Demonstrar o teorema.
- Outras propriedades:
  - Transitividade:
    $f(n) = \Theta(g(n))$ e $g(n) = \Theta(h(n)) \Rightarrow f(n) = \Theta(h(n))$
    $f(n) = O(g(n))$ e $g(n) = O(h(n)) \Rightarrow f(n) = O(h(n))$
    $f(n) = \Omega(g(n))$ e $g(n) = \Omega(h(n)) \Rightarrow f(n) = \Omega(h(n))$
  - Reflexividade:
    $f(n) = \Theta(f(n))$, $f(n) = O(f(n))$, $f(n) = \Omega(f(n))$
  - Simetria: $f(n) = \Theta(g(n))$ se e somente se $g(n) = \Theta(f(n))$
  - Simetria transposta:
    $f(n) = O(g(n))$ se e somente se $g(n) = \Omega(f(n))$

Obrigado!

E-mail: `lgadelha@lncc.br`