

Projeto e Análise de Algoritmos

José Ricardo Mello Viana

Aula 05
Análise de Algoritmos

2010.2

1 Crescimento de Funções

2 Notações Assintóticas

- Ordem de crescimento permite comparar algoritmos
- Para n tão grande quanto você queira
- Às vezes é possível determinar o tempo exato (ordenação por inserção)
 - Não é necessário
 - Termos pequenos são dominados pelos grandes

- Eficiência **assintótica**
- Aumento do tempo de execução para entradas muito grandes (sem limitação)
- Um algoritmo assintoticamente mais eficiente será a melhor escolha para todas as entradas (exceto as muito pequenas)

- Notação Θ

- Para uma função $g(n)$, denotamos $\Theta(g(n))$ as funções:
 $\Theta(g(n)) = \{f(n) : \text{existem constantes positivas } c_1, c_2 \text{ e } n_0 \text{ tais}$
 $\text{que } 0 \leq c_1g(n) \leq f(n) \leq c_2(g(n)) \text{ para todo } n \geq n_0\}$
- $f(n) \in \Theta(g(n))$
- Dizemos que $g(n)$ é um limite assintótico restrito para $f(n)$
- Ex: $\frac{1}{2}n^2 - 3n = \Theta(n^2)$ e $6n^3 \neq \Theta(n^2)$

- Notação O

- Para uma função $g(n)$, denotamos $O(g(n))$ as funções:
 $O(g(n)) = \{f(n) : \text{existem constantes positivas } c \text{ e } n_0 \text{ tais que } 0 \leq f(n) \leq cg(n) \text{ para todo } n \geq n_0\}$
- $f(n) \in O(g(n))$ implica $f(n) \in \Theta(g(n))$
- $\Theta(g(n)) \subseteq O(g(n))$
- Dizemos que $g(n)$ é um limite assintótico superior para $f(n)$
- Ex: $an^2 + bn + c = O(n^2)$

- Notação Ω

- Para uma função $g(n)$, denotamos $\Omega(g(n))$ as funções:
$$\Omega(g(n)) = \{f(n) : \text{existem constantes positivas } c \text{ e } n_0 \text{ tais que}$$
$$0 \leq cg(n) \leq f(n) \text{ para todo } n \geq n_0\}$$
- $f(n) \in \Omega(g(n))$
- Dizemos que $g(n)$ é um limite assintótico inferior para $f(n)$
- $f(n) = \Theta(g(n))$ se, e somente se, $f(n) = O(g(n))$ e $f(n) = \Omega(g(n))$

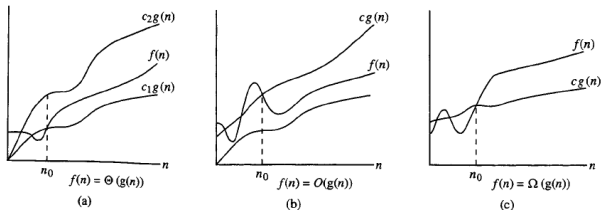


Figure: Notações Assintóticas