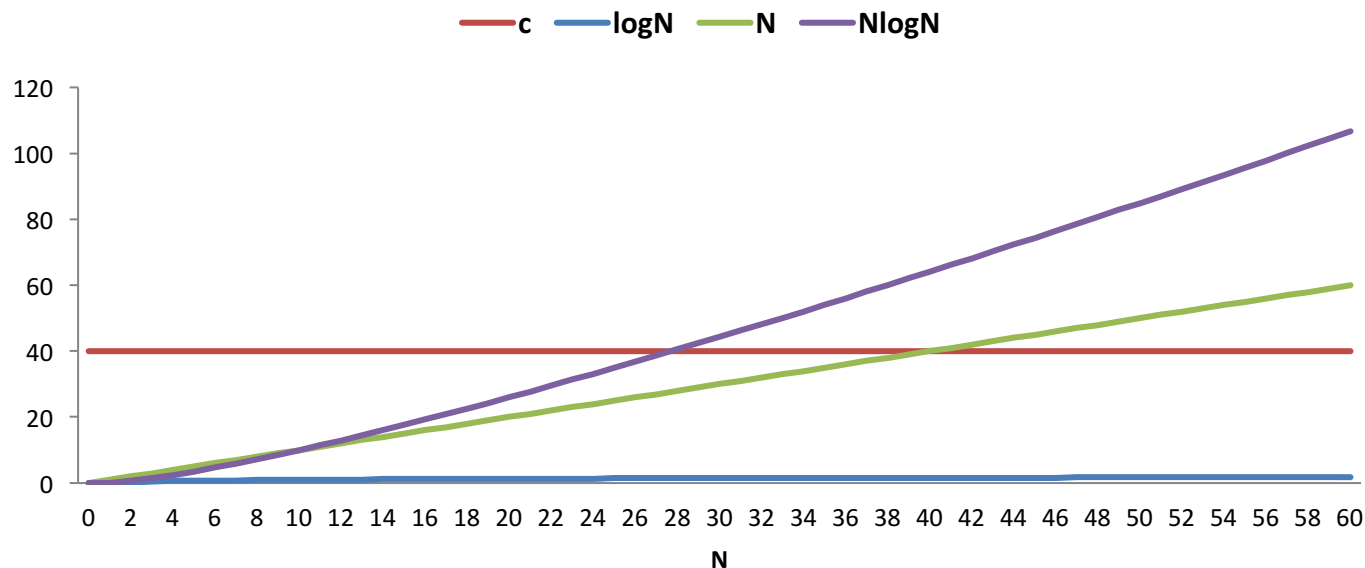


# Estrutura de Dados e Algoritmos

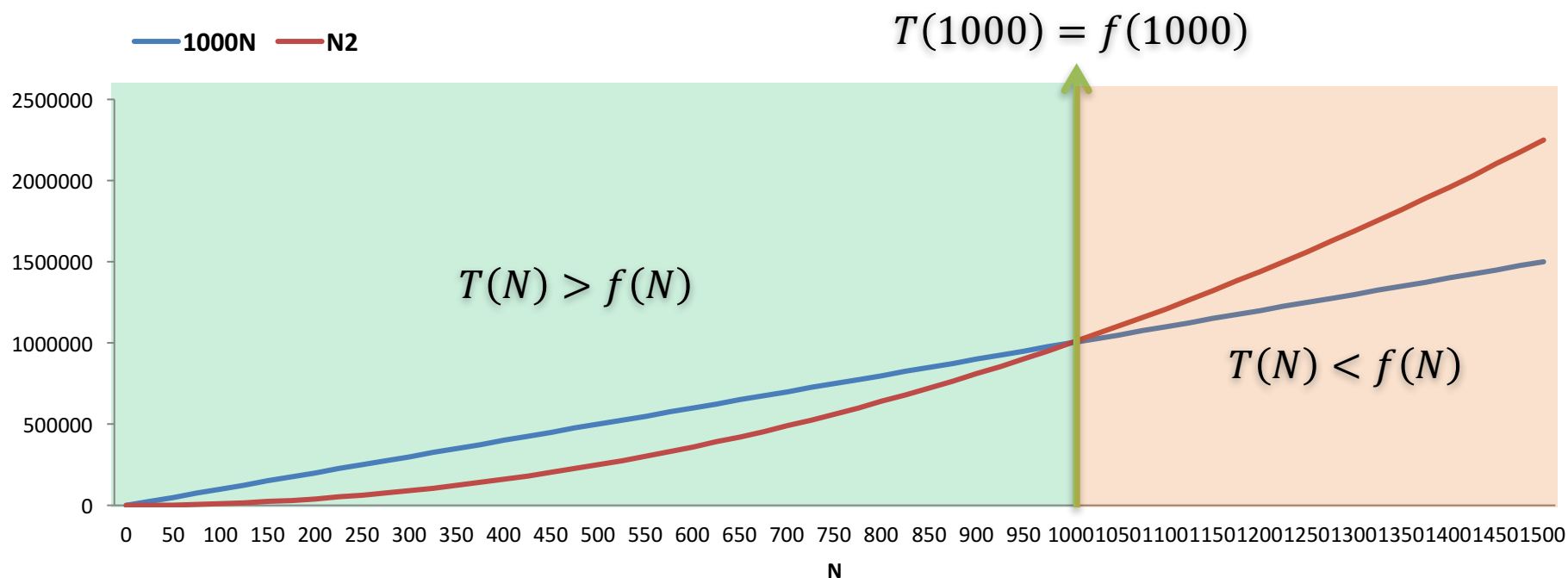
## Notações Assintóticas

- Motivação: Definir ordem entre funções.
- Avaliação pontual  $\rightarrow$  Não tem sentido:
  - $f(N) < g(N)$  ?



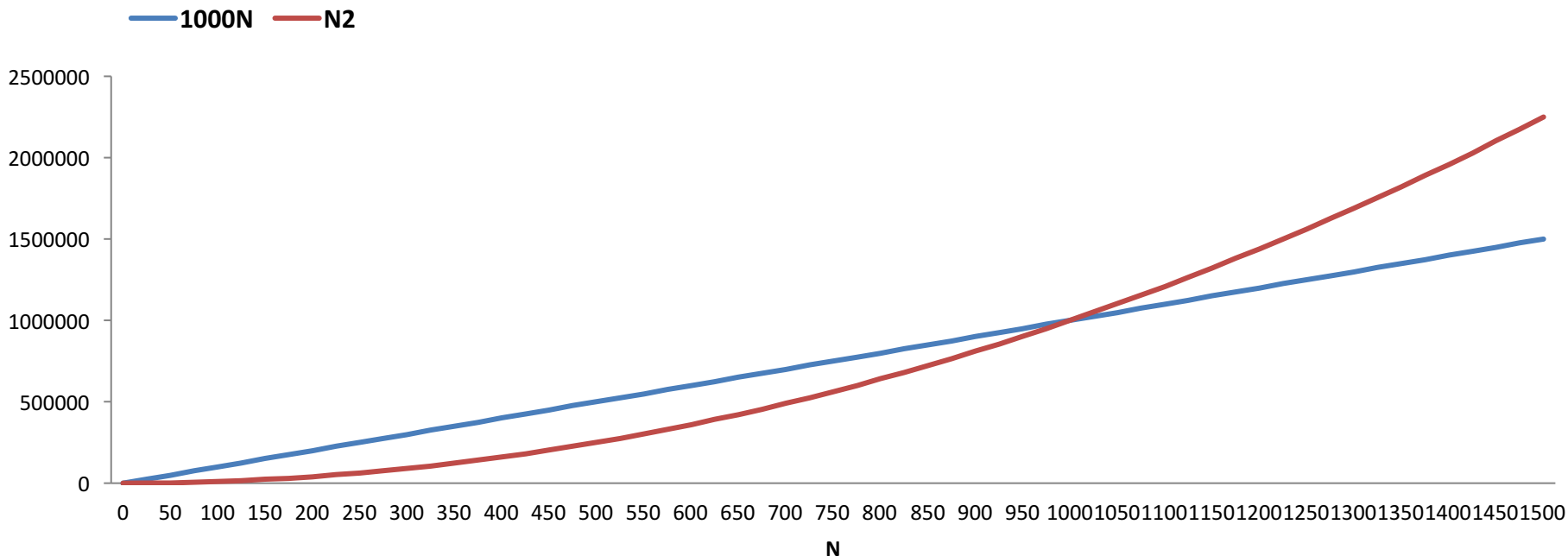
- Forma de Análise: Taxa de crescimento.

$$T(N) = 1000N$$



- Apesar de  $1000N$  ser maior que  $N^2$  para  $N$  pequenos, a taxa de crescimento de  $N^2$  é maior e ultrapassa  $1000N$  para  $N > 1000$ . Logo,  $N^2$  é maior que  $1000N$ .
- Há um valor de  $N$  ( $n_0$ ) a partir do qual  $c.f(N)$  será sempre, no mínimo, tão grande quanto  $T(N)$ . Neste caso:  $n_0=1000$  e  $c=1$ . Outra possibilidade seria  $n_0=100$  e  $c=10$ .

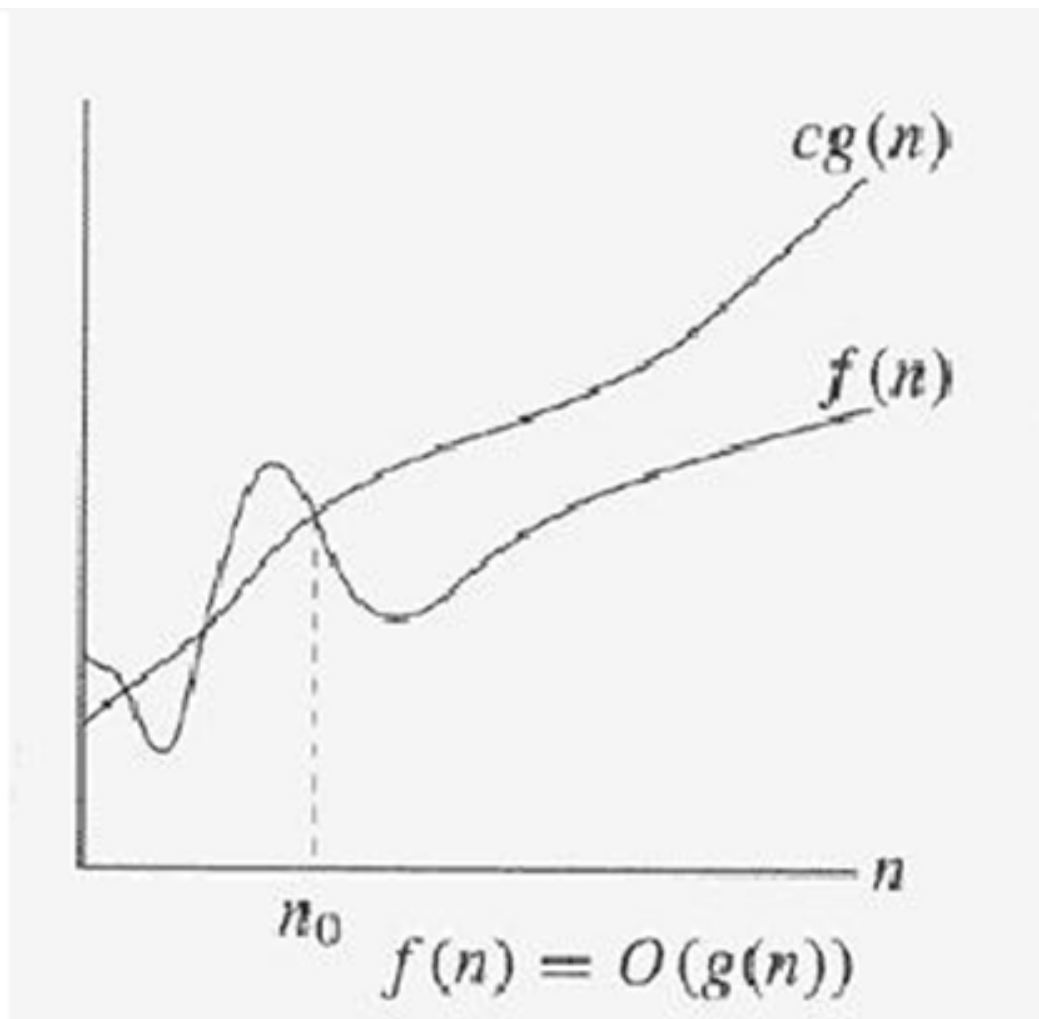
- Concluindo, podemos dizer que a taxa de crescimento de  $T(N)=1000N$  é menor ou igual à taxa de crescimento de  $f(N)=N^2$ .



$T(N) = O(f(N))$ , se houver as constantes positivas  $c$  e  $n_o$ , tal que  $T(N) \leq cf(N)$ , quando  $N \geq n_o$ .

- Se  $T(N) = O(f(N))$ : estamos garantindo que a função  $T(N)$  cresce a uma taxa igual ou inferior à  $f(N)$ .
- Ainda,  $f(N)$  representa o limite superior de  $T(N)$ .

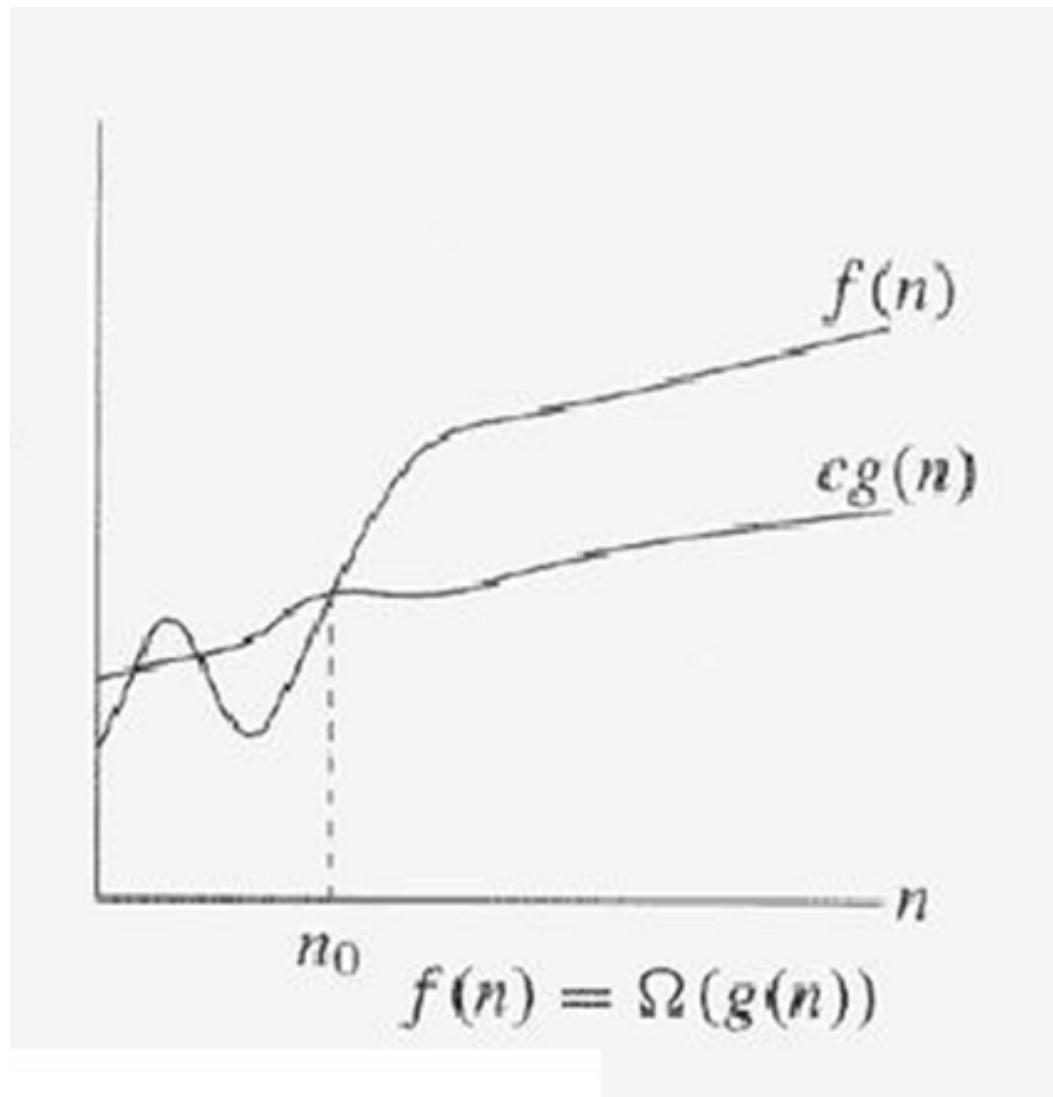
# Notação O



$T(N) = \Omega(f(N))$ , se houver as constantes positivas  $c$  e  $n_o$ , tal que  $T(N) \geq cf(N)$ , quando  $N \geq n_o$ .

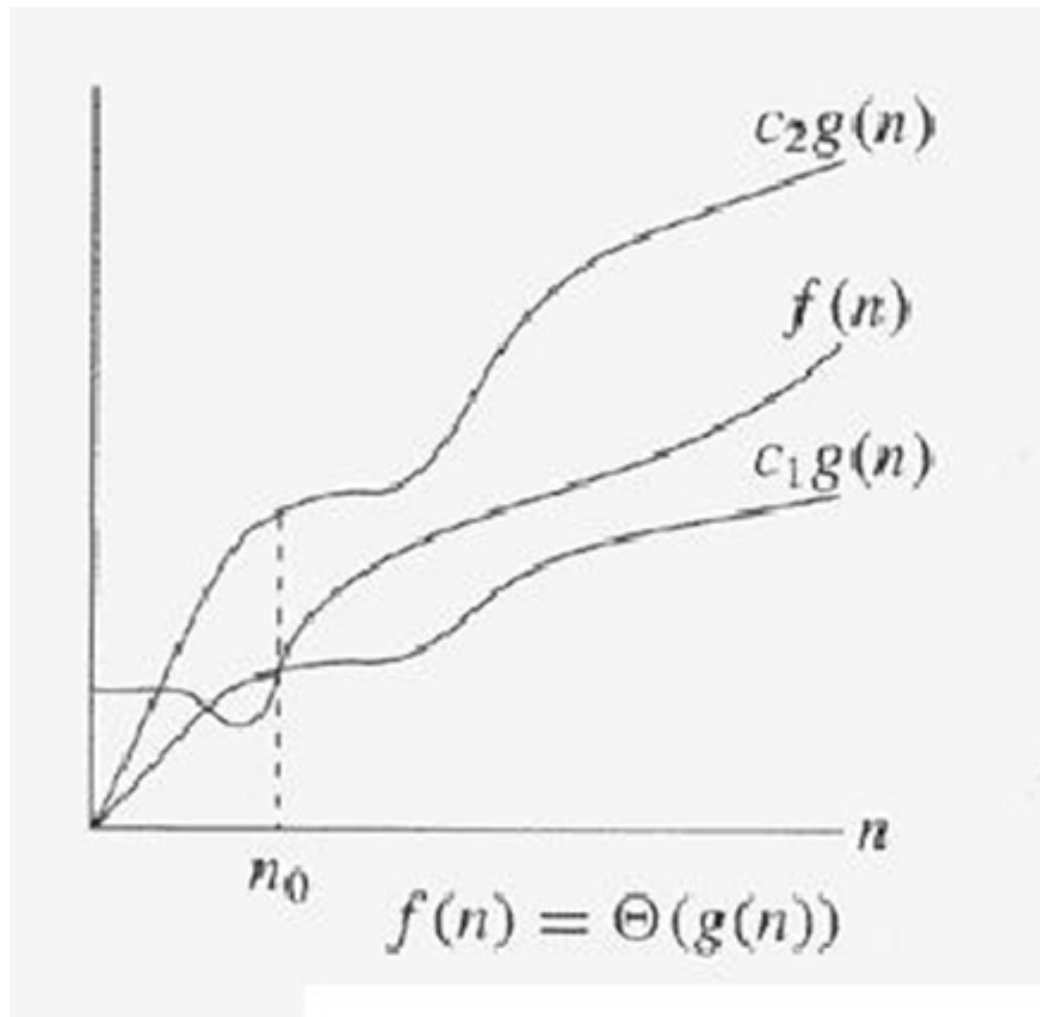
- Se  $T(N) = \Omega(g(N))$ : estamos garantindo que a função  $T(N)$  cresce a uma taxa igual ou superior à  $g(N)$ .
- Ainda,  $g(N)$  representa o limite inferior de  $T(N)$ .





$T(N) = \Theta(h(N))$ , se e somente se  $T(N) = O(h(N))$  e  $T(N) = \Omega(h(N))$ .

- Se  $T(N) = \Theta(h(N))$ : estamos garantindo que a função  $T(N)$  cresce a uma taxa igual à  $h(N)$ .



$$N^2 = O(N^3)$$

$$N^3 = \Omega(N^2)$$

$$f(N) = N^2, \quad g(N) = 2N^2 + N$$

$$f(N) = O(g(N)) \text{ e } f(N) = \Omega(g(N)) \rightarrow f(N) = \Theta(g(N))$$

$$f(N) = 2N^2 + 3N$$

$$f(N) = O(N^4), f(N) = O(N^3), f(N) = \mathbf{O}(N^2)$$

Colocando:

$$f(N) = \mathbf{\Theta}(N^2)$$

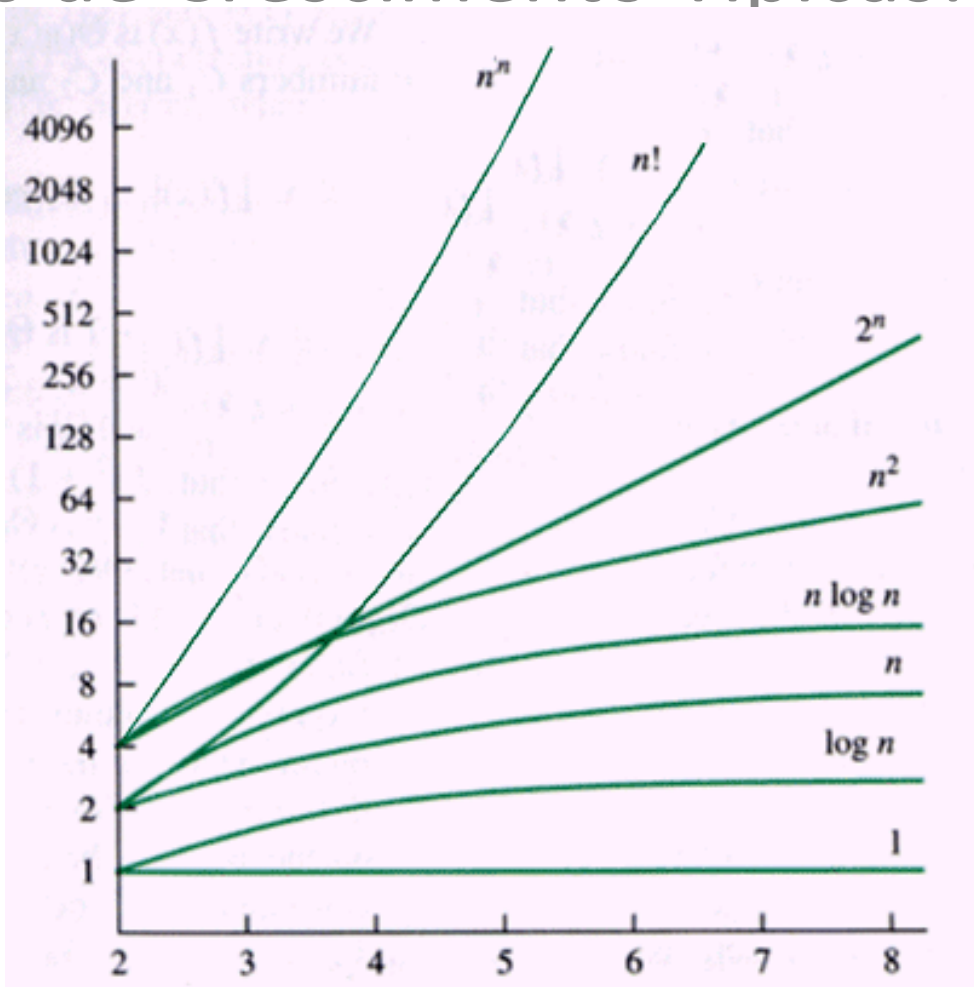
Implica não só que  $f(N) = O(N^2)$ , mas também que  $N^2$  é a ordem de crescimento que melhor se ajusta ao comportamento assintótico de  $f(N)$ .

- Se  $T_1(N) = O(f(N))$  e  $T_2(N) = O(g(N))$ , então:
  - a)  $T_1(N) + T_2(N) = O(f(N) + g(N))$ ;
  - b)  $T_1(N) \times T_2(N) = O(f(N) \times g(N))$ ;
- Se  $T(N)$  é um polinômio de ordem  $k$ , então:
  - a)  $T(N) = \Theta(N^k)$
- $\log^k N = O(N)$  para qualquer constante  $k$ . Isto indica que logaritmos crescem muito lentamente.

- Taxas de Crescimento Típicas:

- $c$  – *Constante;*
- $\log N$  – *Logarítmica;*
- $\log^2 N$  – *Log quadrática;*
- $N$  – *Linear;*
- $N \cdot \log N$
- $N^2$  – *Quadrática;*
- $N^3$  – *Cúbica;*
- $2^N$  – *Exponencial;*

- Taxas de Crescimento Típicas:



- Considere quaisquer funções com, por exemplo, uma variável independente  $n$  (como  $n+10$  ou  $n^2+1$ ):
- A análise de algoritmos ignora os valores pequenos e concentra-se em instâncias grandes para  $n$  ( $n \rightarrow +\infty$ ).
- Para valores enormes de  $n$ , as funções  $n^2$ ,  $(3/2)n^2$ ,  $9999n^2$ ,  $n^2/1000$ ,  $n^2+100n$  crescem todas com a mesma velocidade.
- Esse tipo de análise matemática é chamada *assintótica*. Nessa análise, as funções são classificadas em ordens.
- As cinco funções acima, por exemplo, pertencem à mesma ordem e são, portanto, equivalentes.




- Assim, podemos não considerar constantes ou termos de ordens baixas:

a)  $T(N) = O(2N^2),$

b)  $T(N) = O(N^2 + N),$

c)  $T(N) = O(N^2 + 7),$


$$T(N) = O(N^2)$$

- Não faz sentido:

a)  $f(N) \leq O(g(N)),$

b)  $f(N) \geq O(g(N)),$