Complexidade Assintótica

Norton T. Roman

Apostila baseada nos trabalhos de Marcos Chaim, Cid de Souza,
Cândida da Silva e Delano M. Beder

Crescimento Assintótico de Funções

- Custo da solução aumenta com o tamanho n do problema
  - O tamanho n fornece uma medida da dificuldade para resolver o problema
    - Tempo necessário para resolver o problema aumenta quando n cresce
  - Exemplo: número de comparações para achar o maior elemento de um arranjo (array) ou para ordená-lo aumenta com o tamanho da entrada n.

Crescimento Assintótico de Funções

- Escolha do algoritmo não é um problema crítico quando n é pequeno.
  - O problema é quando n cresce.
- Por isso, é usual analisar o comportamento das funções de custo quando n é bastante grande.
  - Analisa-se o comportamento assintótico das funções de custo
  - Representa o limite do comportamento do custo quando n cresce

Comportamento Assintótico

- Vamos comparar funções assintoticamente, ou seja, para valores grandes, desprezando constantes multiplicativas e termos de menor ordem.

	$n = 100$	$n = 1000$	$n = 10^4$	$n = 10^6$	$n = 10^9$
$\log n$	2	3	4	6	9
n	100	1000	10^4	10^6	10^9
$n \log n$	200	3000	$4 \cdot 10^4$	$6 \cdot 10^6$	$9 \cdot 10^9$
n^2	10^4	10^6	10^8	10^{12}	10^{18}
$100n^2 + 15n$	$1,0015 \cdot 10^6$	$1,00015 \cdot 10^8$	$\approx 10^{10}$	$\approx 10^{14}$	$\approx 10^{20}$
2^n	$\approx 1,26 \cdot 10^{30}$	$\approx 1,07 \cdot 10^{301}$	?	?	?

Comportamento Assintótico

- 1 milhão (10^6) de operações por segundo

Função de custo	10	20	30	40	50	60
n	0,00001s	0,00002s	0,00003s	0,00004s	0,00005s	0,00006s
n^2	0,0001s	0,0004s	0,0009s	0,0016s	0,0025s	0,0036s
n^3	0,001s	0,008s	0,027s	0,064s	0,125s	0,216s
n^5	0,1s	3,2s	24,3s	1,7min	5,2min	12,96min
2^n	0,001s	1,04s	17,9min	12,7dias	35,7 anos	366 séc.
3^n	0,059s	58min	6,5anos	3855séc.	10^8séc.	10^{13}séc.

Comportamento Assintótico

- Influência do aumento de velocidade dos computadores no tamanho x do problema
  - Maior problema possível de se resolver em 1 hora:

Função de custo	Computador Atual (C)	Computador 100C	Computador 1000C
n	x	$100x$	$1000x$
n^2	x	$10x$	$31.6x$
n^3	x	$4,6x$	$10x$
2^n	x	$x + 6,6$	$x + 10$

Comportamento Assintótico

- Influência do aumento de velocidade dos computadores no tamanho x do problema
  - Um aumento de 1.000 vezes resolve um problema 10 vezes maior para algoritmos n^3
  - Para 2^n, adiciona apenas 10 ao tamanho do problema

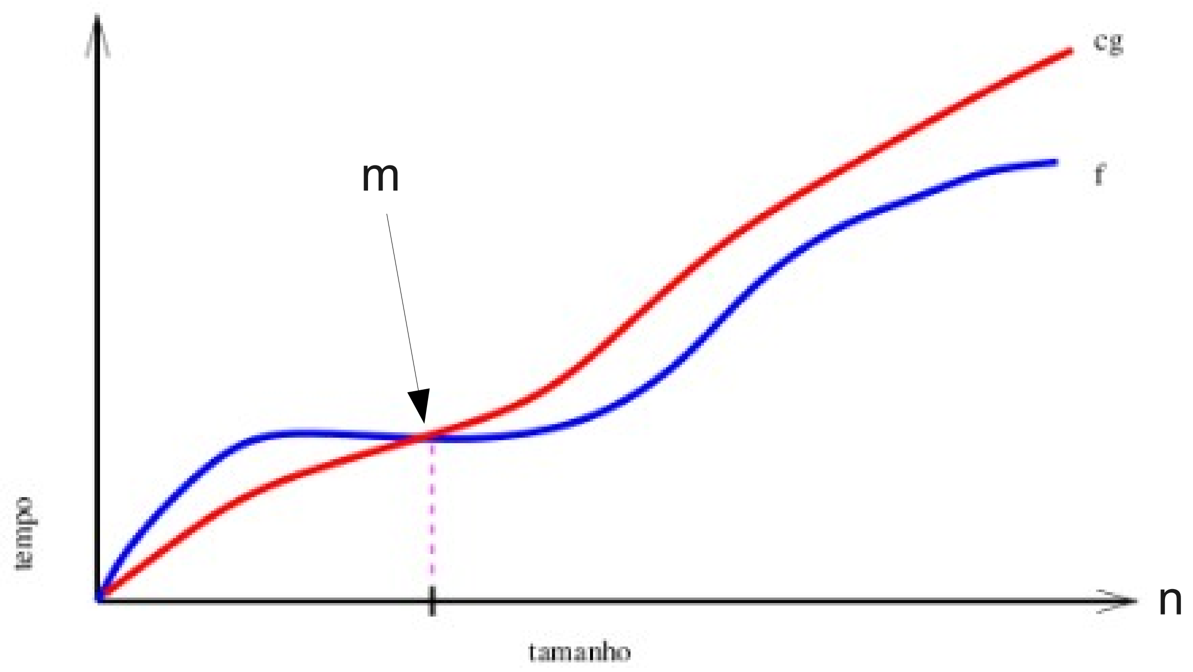
Função de custo	Computador Atual (C)	Computador 100C	Computador 1000C
n	x	$100x$	$1000x$
n^2	x	$10x$	$31.6x$
n^3	x	$4,6x$	$10x$
2^n	x	$x + 6,6$	$x + 10$

Comportamento Assintótico

- Se f(n) é a função de complexidade de um algoritmo A
  - O comportamento assintótico de f (n) representa o limite do comportamento do custo (complexidade) de A quando n cresce.
- A análise de um algoritmo (função de complexidade)
  - Geralmente considera apenas algumas operações elementares ou mesmo uma operação elementar (e.g., o número de comparações).
- A complexidade assintótica relata crescimento assintótico das operações elementares.

Relacionamento Assintótico

- Definição:
  - Uma função g(n) domina assintoticamente outra função f (n) se existem duas constantes positivas c e m tais que, para $n \geq m$, tem-se $|f(n)| \leq c \times |g(n)|$.



Relacionamento Assintótico

- $g(n) = n$ e $f(n) = n^2$
- Alguém domina alguém?
  - ???

Relacionamento Assintótico

- $g(n) = n$ e $f(n) = n^2$
- Alguém domina alguém?
  - $|n| \leq |n^2|$ para todo $n \in \mathbb{N}$.

Relacionamento Assintótico

- $g(n) = n$ e $f(n) = n^2$
- Alguém domina alguém?
  - $|n| \leq |n^2|$ para todo $n \in N$.
  - Para $c = 1$ e $m = 0 \Rightarrow |g(n)| \leq |f(n)|$.
- Portanto, $f(n)$ domina assintoticamente $g(n)$.

Relacionamento Assintótico

- $g(n) = n$ e $f(n) = -n^2$
- Alguém domina alguém?
  - ???

Relacionamento Assintótico

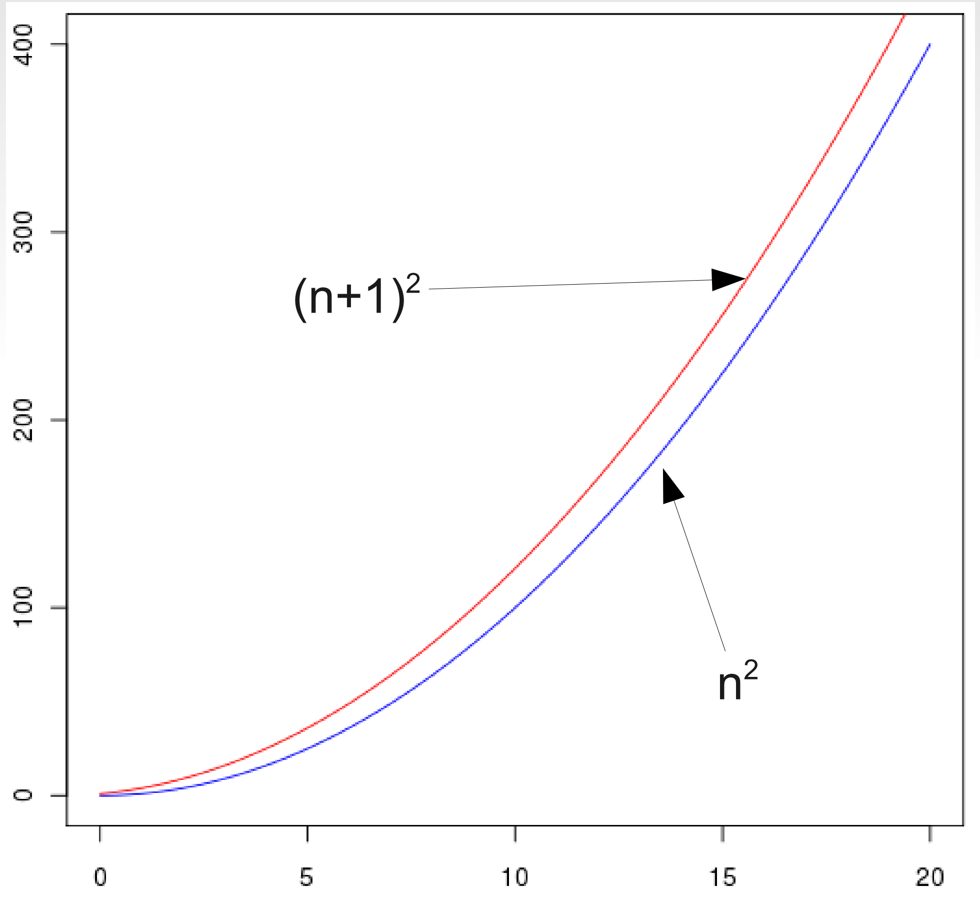
- $g(n) = n$ e $f(n) = -n^2$
- Alguém domina alguém?
  - $|n| \leq |-n^2|$ para todo $n \in N$.
    - Por ser módulo, o sinal não importa
  - Para $c = 1$ e $m = 0 \Rightarrow |g(n)| \leq |f(n)|$.
- Portanto, $f(n)$ domina assintoticamente $g(n)$.

Relacionamento Assintótico

- $g(n) = (n+1)^2$ e $f(n) = n^2$
- Alguém domina alguém?
  - ???

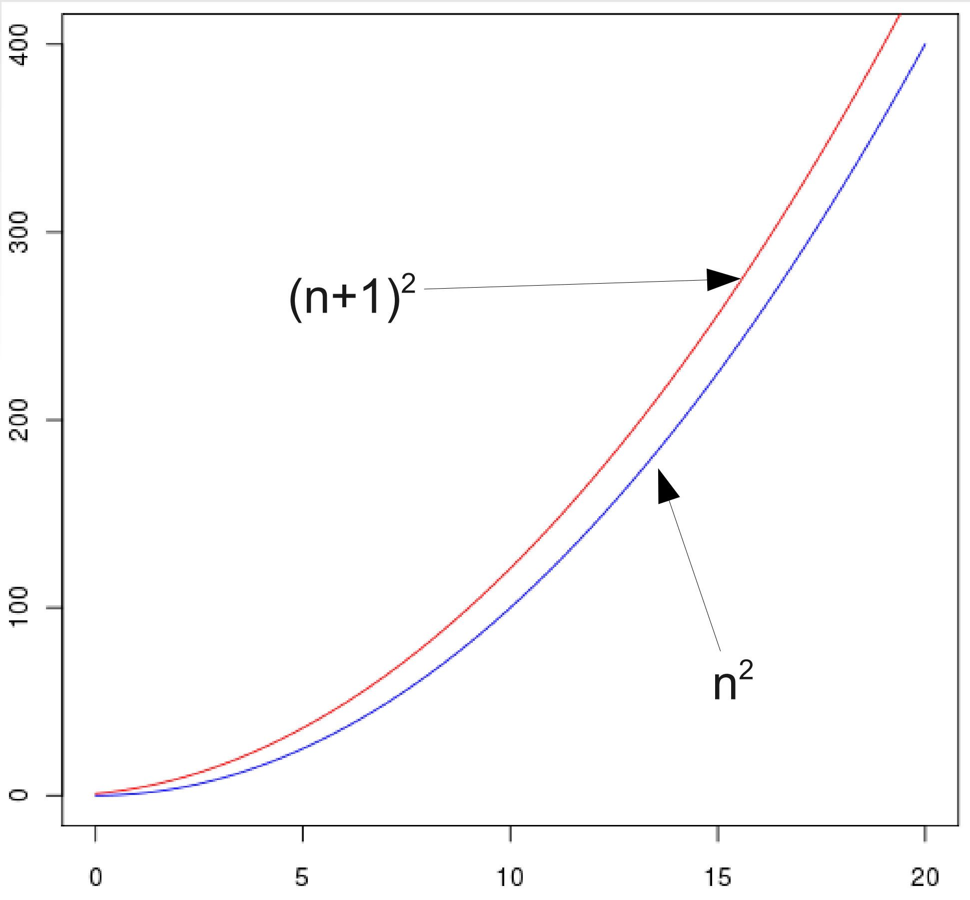
Relacionamento Assintótico

- $g(n) = (n+1)^2$ e $f(n) = n^2$
- Alguém domina alguém?
  - Vamos por em um gráfico



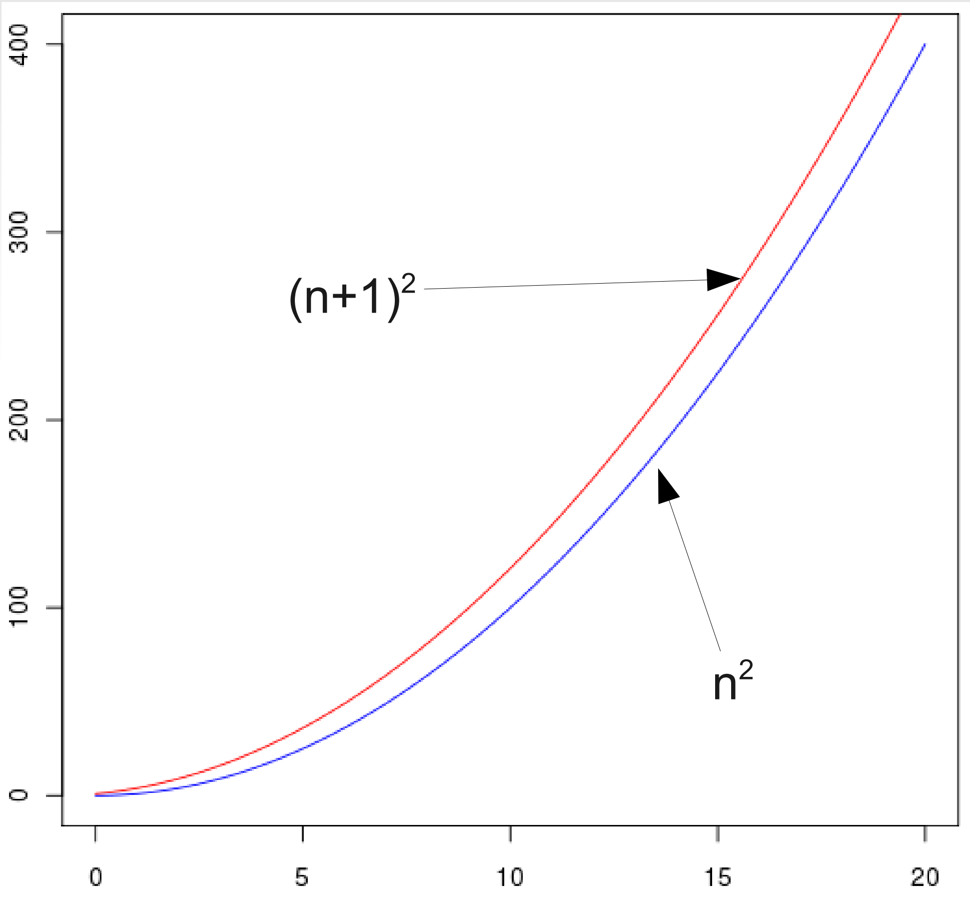
Relacionamento Assintótico

- $g(n) = (n+1)^2$ e $f(n) = n^2$
- Alguém domina alguém?
  - Vamos por em um gráfico
  - Óbvio: $|n^2| \leq |(n+1)^2|$, para $n \geq 0$
  - g(n) domina f(n)



Relacionamento Assintótico

- $g(n) = (n+1)^2$ e $f(n) = n^2$
- Alguém domina alguém?
  - Será somente isso?
  - Não há como f(n) dominar g(n)?
    - ???

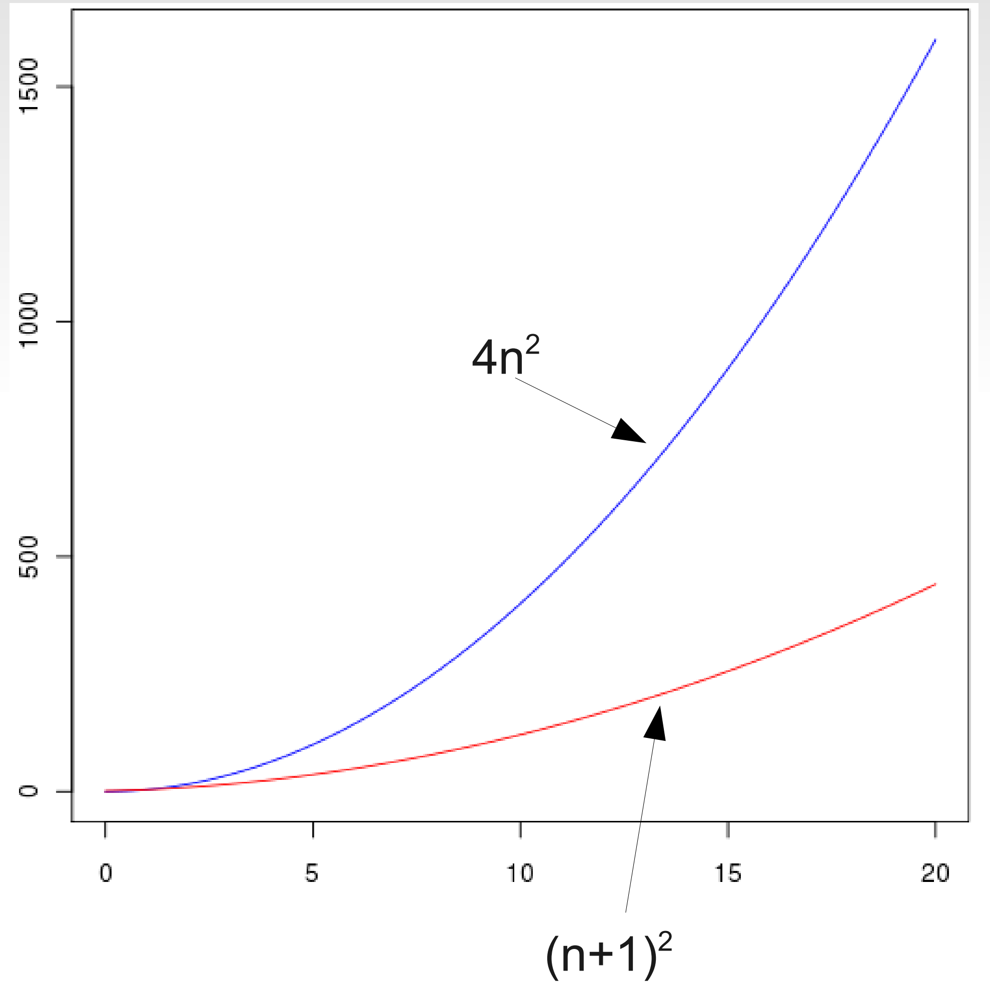


Relacionamento Assintótico

- $g(n) = (n+1)^2$ e $f(n) = n^2$
- Alguém domina alguém?
  - Não há como f(n) dominar g(n)?
    - Lembre que a definição envolve também uma constante.
    - Suponha que queremos $g(n) \leq cf(n)$
    - Então $|(n+1)^2| \leq |cn^2|$
    - Mas, para isso, basta que $|(n+1)^2| \leq |(\sqrt{c}\, n)^2|$,
      - ou $|n+1| \leq |\sqrt{c}\, n|$
    - Se $\sqrt{c} = 2$, ou seja, $c=4$, isso é verdade

Relacionamento Assintótico

- $g(n) = (n+1)^2$ e $f(n) = n^2$
- Alguém domina alguém?
  - E $|(n+1)^2| \le |4n^2|$, para $n \ge 0$
  - f(n) domina g(n), para $n \ge 1$
- Nesse caso, dizemos que f(n) e g(n) dominam assintoticamente uma a outra.

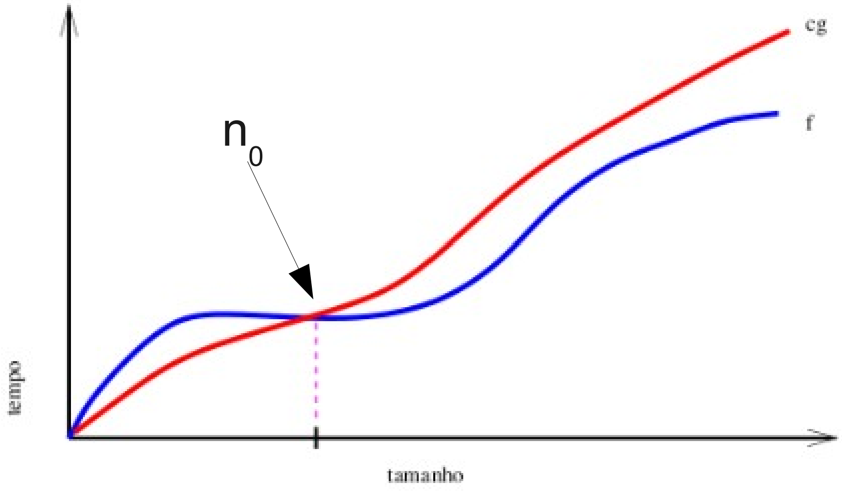


Notação O

- Knuth criou a notação O (O grande) para expressar que g(n) domina assintoticamente f(n)
  - Escreve-se f(n) = O(g(n)) e lê-se: "f(n) é da ordem no máximo g(n)".
- Para que serve isto para o bacharel em Sistemas de Informação?
  - Muitas vezes calcular a função de complexidade g(n) de um algoritmo A é complicado.
  - É mais fácil determinar que f (n) é O(g(n)), isto é, que assintoticamente f(n) cresce no máximo como g(n).

Notação O

- Definição:
  - $O(g(n)) = \{f(n)$: existem constantes positivas c e n_0 tais que $0 \leq f(n) \leq cg(n)$, para todo $n \geq n_0\}$.
- Informalmente, dizemos que, se $f(n) \in O(g(n))$, então $f(n)$ cresce no máximo tão rapidamente quanto $g(n)$.



Notação O

- Definição:
  - O(g(n)) = {f(n): existem constantes positivas c e n_0 tais que $0 \leq f(n) \leq cg(n)$, para todo $n \geq n_0$}.
  - $\frac{3}{2}n^2 - 2\mathrm{n} \in O(n^2)$?
    - ???

Notação O

- Definição:
  - $O(g(n)) = \{f(n)$: existem constantes positivas c e n_0 tais que $0 \leq f(n) \leq cg(n)$, para todo $n \geq n_0\}$.
  - $\frac{3}{2}n^2 - 2n \in O(n^2)$?
    - Fazendo c = 3/2, teremos $|\frac{3}{2}n^2 - 2n| \leqslant |\frac{3}{2}n^2|$, para $n_0 \geq 2$
    - Outras constantes podem existir, mas o que importa é que <u>existe alguma escolha para as constantes</u>

Operações com a notação O

$$
\begin{aligned}
f(n) &= O(f(n)) \\
c \times f(n) &= O(f(n)), c \text{ é uma constante} \\
O(f(n)) + O(f(n)) &= O(f(n)) \\
O(O(f(n))) &= O(f(n)) \\
O(f(n)) + O(g(n)) &= O(max(f(n), g(n))) \\
O(f(n))O(g(n)) &= O(f(n)g(n)) \\
f(n)O(g(n)) &= O(f(n)g(n))
\end{aligned}
$$

Operações com a notação O

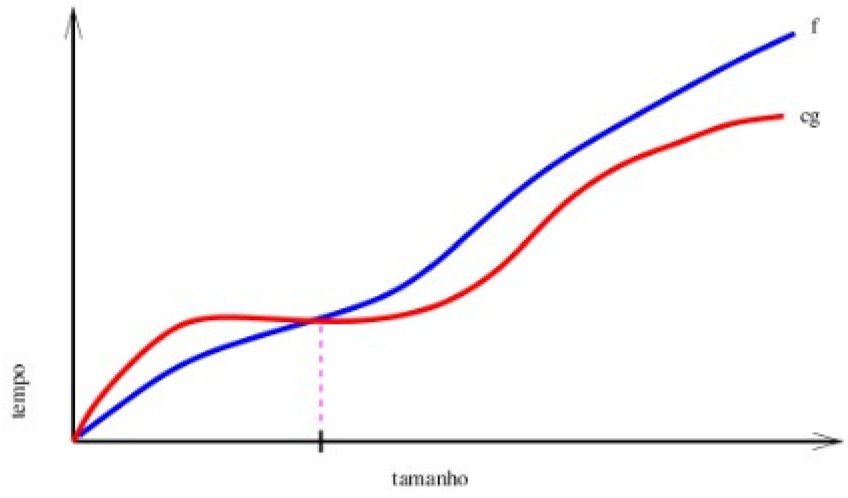
- A regra $O(f(n)) + O(g(n)) = O(\max(f(n),g(n)))$ pode ser usada para calcular o tempo de execução de uma seqüência de trechos de um programa
  - Suponha 3 trechos: $O(n)$, $O(n^2)$ e $O(n\log n)$
  - Qual o tempo de execução do algoritmo como um todo?
    - ???

Operações com a notação O

- A regra $O(f(n)) + O(g(n)) = O(\max(f(n),g(n)))$ pode ser usada para calcular o tempo de execução de uma seqüência de trechos de um programa
  - Suponha 3 trechos: $O(n)$, $O(n^2)$ e $O(n\log n)$
  - Qual o tempo de execução do algoritmo como um todo?
    - Lembre-se que o tempo de execução é a soma dos tempos de cada trecho
    - $O(n) + O(n^2) + O(n\log n) = \max(O(n), O(n^2), O(n\log n)) = O(n^2)$

Notação Ω

- Definição:
  - $\Omega(g(n)) = \{f(n)$: existem constantes positivas c e n_0 tais que $0 \leq cg(n) \leq f(n)$, para todo $n \geq n_0\}$.
- Informalmente, dizemos que, se $f(n) \in \Omega(g(n))$, então f(n) cresce no mínimo tão lentamente quanto g(n).
  - Note que se $f(n) \in O(g(n))$ define um limite superior para f(n), $\Omega(g(n))$ define um limite inferior



Notação Ω

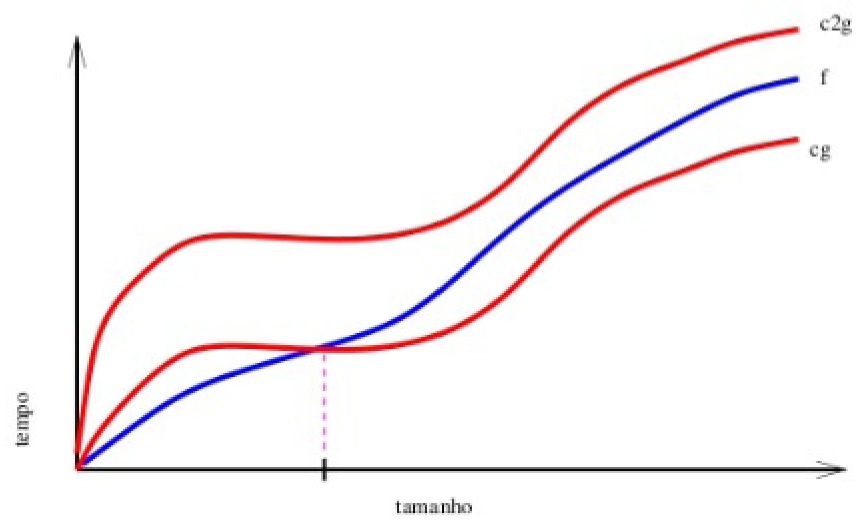
- Definição:
  - $\Omega(g(n)) = \{f(n)$: existem constantes positivas c e n_0 tais que $0 \leq cg(n) \leq f(n)$, para todo $n \geq n_0\}$.
  - $\frac{3}{2}n^2 - 2n \in \Omega(n^2)$
    - ???

Notação Ω

- Definição:
  - $\Omega(g(n)) = \{f(n)$: existem constantes positivas c e n_0 tais que $0 \leq cg(n) \leq f(n)$, para todo $n \geq n_0\}$.
  - $\frac{3}{2}n^2 - 2n \in \Omega(n^2)$
    - Fazendo c = 1/2, teremos $|\frac{3}{2}n^2 - 2n| \geqslant |\frac{1}{2}n^2|$, para $n_0 \geq 2$

Notação Θ

- Definição:
  - $\Theta(g(n)) = \{f(n)$: existem constantes positivas c_1, c_2 e n_0 tais que $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$, para todo $n \geq n_0\}$
- Informalmente, dizemos que, se $f(n) \in \Theta(g(n))$, então f(n) cresce tão rapidamente quanto g(n).



Notação Θ

- Definição:
  - $\Theta(g(n)) = \{f(n)$: existem constantes positivas c_1 , c_2 e n_0 tais que $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$, para todo $n \geq n_0\}$
  - $\frac{3}{2}n^2 - 2n \in \Theta(n^2)$
    - ???

Notação Θ

- Definição:
  - $\Theta(g(n)) = \{f(n)$: existem constantes positivas c_1 , c_2 e n_0 tais que $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$, para todo $n \geq n_0\}$
  - $\frac{3}{2}n^2 - 2n \in \Theta(n^2)$
    - Fazendo $c_1 = 1/2$ e $c_2 = 3/2$ teremos

      $$|\frac{1}{2}n^2| \leqslant |\frac{3}{2}n^2 - 2n| \leqslant |\frac{3}{2}n^2|$$

      para $n_0 \geq 2$

Notação Θ

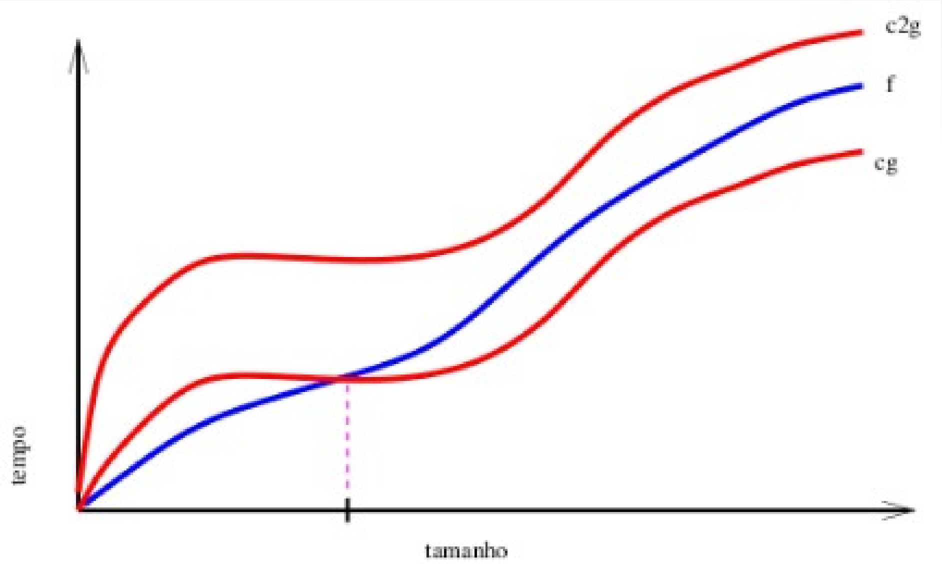
- Mas:
  - $\frac{3}{2}n^2 - 2\mathrm{n} \in O(n^2) \rightarrow |\frac{3}{2}n^2 - 2\mathrm{n}| \leqslant |\frac{3}{2}n^2|$
  - $\frac{3}{2}n^2 - 2\mathrm{n} \in \Omega(n^2) \rightarrow |\frac{1}{2}n^2| \leqslant |\frac{3}{2}n^2 - 2\mathrm{n}|$
  - e $\frac{3}{2}n^2 - 2\mathrm{n} \in \Theta(n^2) \rightarrow |\frac{1}{2}n^2| \leqslant |\frac{3}{2}n^2 - 2\mathrm{n}| \leqslant |\frac{3}{2}n^2|$
  - Será coincidência?
    - ???

Notação Θ

- Mas:
  - $\frac{3}{2}n^2-2\mathrm{n}\in O(n^2)\rightarrow|\frac{3}{2}n^2-2\mathrm{n}|\leqslant|\frac{3}{2}n^2|$
  - $\frac{3}{2}n^2-2\mathrm{n}\in \Omega(n^2)\rightarrow|\frac{1}{2}n^2|\leqslant|\frac{3}{2}n^2-2\mathrm{n}|$
  - e $\frac{3}{2}n^2-2\mathrm{n}\in \Theta(n^2)\rightarrow|\frac{1}{2}n^2|\leqslant|\frac{3}{2}n^2-2\mathrm{n}|\leqslant|\frac{3}{2}n^2|$
  - Será coincidência?
    - Não!
    - Se f(n) ∈ O(g(n)) e f(n) ∈ Ω(g(n)), então f(n) ∈ Θ(g(n))

Notação Θ

- Mas:
  - Será coincidência?
    - Não!
    - Se $f(n) \in O(g(n))$ e $f(n) \in \Omega(g(n))$, então $f(n) \in \Theta(g(n))$



Notação o

- Definição:
  - $o(g(n)) = \{f(n)$: para toda constante positiva c, existe uma constante $n_0 > 0$ tal que $0 \leq f(n) < cg(n)$, para todo $n \geq n_0\}$.
- Informalmente, dizemos que, se $f(n) \in o(g(n))$, então f(n) cresce mais lentamente que g(n).
  - Intuitivamente, na notação o, a função f(n) tem crescimento muito menor que g(n) quando n tende para o infinito

Notação o

- $1000n^2 \in o(n^3)$
  - ???

Notação o

- $1000\,n^{٢} \in o(n^3)$
  - Para todo valor de c, um n_0 que satisfaz a definição é:

$$n_0 = \lceil \frac{1000}{c} \rceil + 1$$

Notação o

- Qual a diferença entre O e o?
  - O: **<u>existem</u>** constantes positivas c e n_0 tais que $0 \leq f(n) \leq cg(n)$, para todo $n \geq n_0$
    - A expressão $0 \leq f(n) \leq cg(n)$ é válida para <u>alguma</u> constante $c>0$
  - o: **<u>para toda</u>** constante positiva c, existe uma constante $n_0 > 0$ tal que $0 \leq f(n) < cg(n)$, para todo $n \geq n_0$
    - A expressão $0 \leq f(n) < cg(n)$ é válida para <u>toda</u> constante $c>0$

Notação ω

- Definição:
  - $\omega(g(n)) = \{f(n)$: para toda constante positiva c, existe uma constante $n_0 > 0$ tal que $0 \leq cg(n) < f(n)$, para todo $n \geq n_0\}$.
- Informalmente, dizemos que, se $f(n) \in \omega(g(n))$, então f(n) cresce mais rapidamente que g(n).
  - Intuitivamente, na notação ω, a função f(n) tem crescimento muito maior que g(n) quando n tende para o infinito

Notação ω

- ω está para Ω, da mesma forma que o está para O
  - O e Ω são chamados de assintoticamente firmes
- $$\frac{1}{1000}n^2 \in \omega(n)$$
  - ???

Notação ω

- ω está para Ω, da mesma forma que o está para O
  - O e Ω são chamados de assintoticamente firmes
- $\frac{1}{1000} n^2 \in \omega(n)$
  - Para todo valor de c, um n_0 que satisfaz a definição é:

$$n_0 = \lceil 1000 c \rceil + 1$$

Definições equivalentes

$$f(n) \in o(g(n)) \quad se \lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

$$f(n) \in O(g(n)) \quad se \lim_{n \to \infty} \frac{f(n)}{g(n)} < \infty$$

$$f(n) \in \Theta(g(n)) \quad se \quad 0 < \lim_{n \to \infty} \frac{f(n)}{g(n)} < \infty$$

$$f(n) \in \Omega(g(n)) \quad se \lim_{n \to \infty} \frac{f(n)}{g(n)} > 0$$

$$f(n) \in \omega(g(n)) \quad se \lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty$$

Propriedades das Classes

Reflexividade:

$f(n) \in O(f(n))$.
$f(n) \in \Omega(f(n))$.
$f(n) \in \Theta(f(n))$.

Simetria:

$f(n) \in \Theta(g(n))$ se, e somente se, $g(n) \in \Theta(f(n))$.

Simetria Transposta:

$f(n) \in O(g(n))$ se, e somente se, $g(n) \in \Omega(f(n))$.
$f(n) \in o(g(n))$ se, e somente se, $g(n) \in \omega(f(n))$.

Propriedades das Classes

Transitividade:

Se $f(n) \in O(g(n))$ e $g(n) \in O(h(n))$, então $f(n) \in O(h(n))$.
Se $f(n) \in \Omega(g(n))$ e $g(n) \in \Omega(h(n))$, então $f(n) \in \Omega(h(n))$.
Se $f(n) \in \Theta(g(n))$ e $g(n) \in \Theta(h(n))$, então $f(n) \in \Theta(h(n))$.
Se $f(n) \in o(g(n))$ e $g(n) \in o(h(n))$, então $f(n) \in o(h(n))$.
Se $f(n) \in \omega(g(n))$ e $g(n) \in \omega(h(n))$, então $f(n) \in \omega(h(n))$.

Exercícios

Quais as relações de comparação assintótica (O, Ω, Θ) das funções:

$$f_1(n) = 2^\pi$$
$$f_2(n) = 2^n$$
$$f_3(n) = n \log n$$
$$f_4(n) = \log n$$
$$f_5(n) = 100n^2 + 150000n$$
$$f_6(n) = n + \log n$$
$$f_7(n) = n^2$$
$$f_8(n) = n$$

	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
f_1	Θ							
f_2		Θ						
f_3			Θ					
f_4				Θ				
f_5					Θ			
f_6						Θ		
f_7							Θ	
f_8								Θ

Referências

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest & Clifford Stein. Algoritmos - Tradução da 2a. Edição Americana. Editora Campus, 2002 (Capítulo 3).
- Michael T. Goodrich & Roberto Tamassia. Estruturas de Dados e Algoritmos em Java. Editora Bookman, 4a. Ed. 2007 (Capítulo 4).
- Nívio Ziviani. Projeto de Algoritmos com implementações em C e Pascal. Editora Thomson, 2a. Edição, 2004 (Seção 1.3).