Notação Assintótica

Noções Preliminares

- O que é um algoritmo correto?
  - Aquele que pára para toda instância do problema, retornando uma solução correta.
- O que é analisar um algoritmo?
  - Predizer a quantidade de recursos utilizados (memória, tempo de execução, número de processadores, acessos a disco, etc).
  - Na maioria dos casos estaremos interessados em avaliar o tempo de execução gasto pelo algoritmo.
  - Contaremos o número de operações efetuadas.

Como representar um algoritmo?

Pseudo-código (abstrato, independe de implementação).

Entrada: Vetor A de n inteiros.
Saída: Vetor A ordenado.

```
Ordenação(A, n)
    para j ← 2 até n faça
        valor ← A[j]
        /* Insere A[j] na seqüência ordenada A[1..j − 1] */
        i ← j − 1
        enquanto i > 0 e A[i] > valor faça
            A[i + 1] ← A[i]
            i ← i − 1
        A[i + 1] ← valor
```

Modelo Computacional

- A análise de um algoritmo depende do modelo computacional subjacente.
- O modelo computacional define quais são os recursos disponíveis e quanto custam.
- Queremos analisar algoritmos independente de implementação.
- Utilizaremos o modelo abstrato RAM:
  - Simula máquinas convencionais.
  - Um único processador que executa instruções seqüencialmente.
  - Somas, subtrações, multiplicações, divisões, comparações e atribuições são feitas em tempo constante.

Funções

Qual é o maior: $n^2 - 9$ ou $4n + 8$?

Depende do valor de n.

Funções

Qual é o maior: $n^2 - 9$ ou $4n + 8$?

Qual cresce mais?

$\Rightarrow$ comparação assintótica $\Rightarrow$ n ENORME

Intuitivamente:
$n^2 \quad (3/2)n^2 \quad 9999n^2 \quad n^2/1000$
crescem todas com a mesma ordem

Quando estamos considerando entradas grandes o suficiente para tornar relevante apenas a ordem de crescimento do tempo de execução, estamos estudando a eficiência assintótica dos algoritmos.

Pior Caso e Notação Assintótica

- O tempo de execução de um algoritmo depende da instância. Faremos análises de pior caso.
- A complexidade dos algoritmos será expressa como uma função do tamanho da entrada, já que isso espelha, de alguma forma, quão "inteligente" é o algoritmo.
- Basta compararmos a ordem de grandeza das funções de complexidade. Para isso utilizaremos a notação assintótica.

Notação Assintótica

- Vamos expressar complexidade através de funções em variáveis que descrevam o tamanho de instâncias do problema. Exemplos:
  - Problemas de aritmética de precisão arbitrária: número de bits (ou bytes) dos inteiros.
  - Problemas de ordenação de vetores: tamanho do vetor.
  - Busca em textos: número de caracteres do texto ou padrão de busca.
- Vamos supor que funções que expressam complexidade são sempre positivas, já que estamos medindo número de operações.

A complexidade assintótica é definida pelo crescimento da complexidade para entradas suficientemente grandes. O algoritmo assintoticamente mais eficiente é melhor para todas as entradas, exceto para as entradas relativamente pequenas.

Comparação de Funções

	$n = 100$	$n = 1000$	$n = 10^4$	$n = 10^6$	$n = 10^9$
$\log n$	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	$\approx 1,26 \cdot 10^{30}$	$\approx 1,07 \cdot 10^{301}$	?	?	?

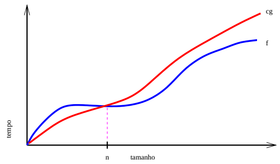
Na maioria dos casos práticos, quando queremos resolver um problema computacionalmente, queremos ser capazes de resolver instâncias muito grandes do problema. Por isso, estaremos interessados em comparar as funções de complexidade assintoticamente, quando as variáveis das funções comparadas tendem a $+$infinito.

Quando comparamos assintoticamente duas funções, podemos desprezar as constantes multiplicativas das variáveis e termos de menor grau das funções, pois estes não alteram a ordem de grandeza do crescimento das funções. É exatamente para comparar ordens de grandeza das funções que utitizamos as classes $\mathcal{O}$, Ω, Θ, o e ω.

Classe $\mathcal{O}$

$\mathcal{O}(g(n)) = \{f(n) :$ existem constantes positivas c e n_0 tais que $0 \le f(n) \le cg(n)$, para todo $n \ge n_0\}$.

Informalmente, dizemos que, se $f(n) \in \mathcal{O}(g(n))$, então $f(n)$ cresce no máximo tão rapidamente quanto $g(n)$.



Quando $f(n) \in \mathcal{O}(g(n))$, dizemos que $g(n)$ dá um limite superior para o crescimento de $f(n)$, ou seja, assintoticamente, $f(n)$ não cresce mais que $g(n)$.

Classe $\mathcal{O}$

$\mathcal{O}(g(n)) = \{f(n) :$ existem constantes positivas c e n_0 tais que $0 \le f(n) \le cg(n)$, para todo $n \ge n_0\}$.

Informalmente, dizemos que, se $f(n) \in \mathcal{O}(g(n))$, então $f(n)$ cresce no máximo tão rapidamente quanto $g(n)$.

Exemplo: $\frac{1}{2}n^2 - 3n \in \mathcal{O}(n^2)$

Valores de c e n_0 que satisfazem a definição são

Classe $\mathcal{O}$

$\mathcal{O}(g(n)) = \{f(n) :$ existem constantes positivas c e n_0 tais que $0 \le f(n) \le cg(n)$, para todo $n \ge n_0\}$.

Informalmente, dizemos que, se $f(n) \in \mathcal{O}(g(n))$, então $f(n)$ cresce no máximo tão rapidamente quanto $g(n)$.

Exemplo: $\frac{1}{2}n^2 - 3n \in \mathcal{O}(n^2)$

Valores de c e n_0 que satisfazem a definição são

$$c = \frac{1}{2} \text{ e } n_0 = 7.$$

Exemplo: $\frac{1}{2}n^2 - 3n \in \mathcal{O}(n^2)$

Queremos provar que, para algum par de constantes positivas c e n_0, temos

$$0 \leq \frac{1}{2}n^2 - 3n \leq cn^2, \text{ para todo } n \geq n_0.$$

Dividindo ambos os lados da equação por n^2, percebemos que queremos encontrar c e n_0 tais que

$$0 \leq \frac{1}{2} - \frac{3}{n} \leq c, \text{ para todo } n \geq n_0.$$

Como a função $\frac{1}{2} - \frac{3}{n}$ nunca é maior que $\frac{1}{2}$ e é positiva para $n \geq 7$, então $c = \frac{1}{2}$ e $n_0 = 7$ certificam que $\frac{1}{2}n^2 - 3n \in \mathcal{O}(n^2)$. Note que esse não é o único par de constantes que garante a pertinência de $\frac{1}{2}n^2 - 3n$ a $\mathcal{O}(n^2)$; existem infinitos pares.

Nomes das classes

classe	nome
$\mathcal{O}(1)$	constante
$\mathcal{O}(\lg n)$	logarítmica
$\mathcal{O}(n)$	linear
$\mathcal{O}(n \lg n)$	$n \log n$
$\mathcal{O}(n^2)$	quadrática
$\mathcal{O}(n^3)$	cúbica
$\mathcal{O}(n^k)$ com $k \geq 1$	polinomial
$\mathcal{O}(2^n)$	exponencial
$\mathcal{O}(a^n)$ com $a > 1$	exponencial

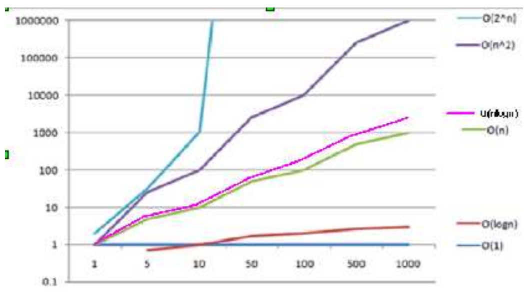
Observações

- A notação assintótica não está interessada na função específica $f(n)$, mas na sua taxa de crescimento com relação a n. As constantes da função $f(n)$ são ignoradas.
- Os termos de mais baixa ordem são ignorados. Se $f(n) < h(n)$:

$$\lim_{n \to \infty} \frac{f(n)}{h(n)} = 0$$

Principais limites assintóticos

$\mathcal{O}(1) < \mathcal{O}(logn) < \mathcal{O}(n) < \mathcal{O}(nlogn) < \mathcal{O}(n2) < \mathcal{O}(2^n)$



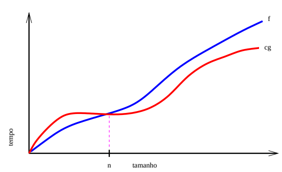
Uso da notação $\mathcal{O}$

- "$f(n)$ é $\mathcal{O}(g(n))$" deve ser entendido como "$f(n) \in \mathcal{O}(g(n))$"
- "$f(n) = \mathcal{O}(g(n))$" deve ser entendido como "$f(n) \in \mathcal{O}(g(n))$"
- "$f(n) \leq \mathcal{O}(g(n))$" é feio
- "$f(n) \geq \mathcal{O}(g(n))$" não faz sentido
- "$f(n)$ é $h(n) + \mathcal{O}(g(n))$" significa que existe constantes positivas c e n_0 tais que: $f(n) \leq h(n) + cg(n)$ para todo $n \geq n_0$

Classe Ω

$$\Omega(g(n)) = \{f(n) : \text{ existem constantes positivas } c \text{ e } n_0 \text{ tais que } 0 \leq cg(n) \leq f(n), \text{ para todo } n \geq n_0\}.$$

Informalmente, dizemos que, se $f(n) \in \Omega(g(n))$, então $f(n)$ cresce no mínimo tão lentamente quanto $g(n)$.



Quando $f(n) \in \Omega(g(n))$, dizemos que $g(n)$ dá um limite inferior para o crescimento de $f(n)$, ou seja, assintoticamente, $f(n)$ não cresce menos que $g(n)$.

Classe Ω

$$\Omega(g(n)) = \{f(n) : \text{ existem constantes positivas } c \text{ e } n_0 \text{ tais que } 0 \leq cg(n) \leq f(n), \text{ para todo } n \geq n_0\}.$$

Informalmente, dizemos que, se $f(n) \in \Omega(g(n))$, então $f(n)$ cresce no mínimo tão lentamente quanto $g(n)$.

Exemplo: $\frac{1}{2}n^2 - 3n \in \Omega(n^2)$

Valores de c e n_0 que satisfazem a definição são

Classe Ω

$$\Omega(g(n)) = \{f(n) : \text{ existem constantes positivas } c \text{ e } n_0 \text{ tais que } 0 \leq cg(n) \leq f(n), \text{ para todo } n \geq n_0\}.$$

Informalmente, dizemos que, se $f(n) \in \Omega(g(n))$, então $f(n)$ cresce no mínimo tão lentamente quanto $g(n)$.

Exemplo: $\frac{1}{2}n^2 - 3n \in \Omega(n^2)$

Valores de c e n_0 que satisfazem a definição são

$$c = \frac{1}{14} \text{ e } n_0 = 7.$$

Exemplo: $\frac{1}{2}n^2 - 3n \in \Omega(n^2)$

Queremos provar que, para algum par de constantes positivas c e n_0, temos

$$0 \leq cn^2 \leq \frac{1}{2}n^2 - 3n, \text{ para todo } n \geq n_0.$$

Dividindo ambos os lados da equação por n^2, percebemos que queremos encontrar c e n_0 tais que

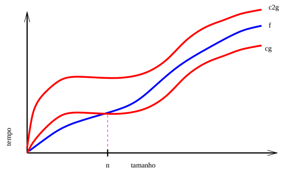
$$0 \leq c \leq \frac{1}{2} - \frac{3}{n}, \text{ para todo } n \geq n_0.$$

Note que, para $n \leq 6$, temos $\frac{1}{2} - \frac{3}{n} \leq 0$. Como c deve ser uma constante positiva, teremos $n_0 \geq 7$. Mas $\frac{1}{2} - \frac{3}{n} \leq 0$ cresce monotonicamente para $n \geq 7$, então $c = \frac{1}{2} - \frac{3}{7} = \frac{1}{14}$ e $n_0 = 7$ certificam que $\frac{1}{2}n^2 - 3n \in \Omega(n^2)$.

Classe Θ

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

Informalmente, dizemos que, se $f(n) \in \Theta(g(n))$, então $f(n)$ cresce tão rapidamente quanto $g(n)$.



Comparando as definições apresentadas anteriormente das funções $f(n)$ que compõem as classes $O(g(n))$ e $\Omega(g(n))$ com a definição das funções $f(n)$ que compõem as classes $\Theta(g(n))$, verificamos que $f(n) \in \Theta(g(n))$ se, e somente se, $f(n) \in O(g(n)) \cap \Omega(g(n))$.

Classe Θ

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

Informalmente, dizemos que, se $f(n) \in \Theta(g(n))$, então $f(n)$ cresce tão rapidamente quanto $g(n)$.

Exemplo: $\frac{1}{2}n^2 - 3n \in \Theta(n^2)$

Classe Θ

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

Informalmente, dizemos que, se $f(n) \in \Theta(g(n))$, então $f(n)$ cresce tão rapidamente quanto $g(n)$.

Exemplo: $\frac{1}{2}n^2 - 3n \in \Theta(n^2)$

$$c_1 = \frac{1}{14}, \; c_2 = \frac{1}{2} \text{ e } n_0 = 7.$$

Exemplo: $\frac{1}{2}n^2 - 3n \in \Theta(n^2)$

Já provamos anteriormente que $\frac{1}{2}n^2 - 3n \in \mathcal{O}(n^2)$ e $\frac{1}{2}n^2 - 3n \in \Omega(n^2)$, logo $\frac{1}{2}n^2 - 3n \in \Theta(n^2)$.

No entanto, quais seriam as constantes positivas c_1, c_2 e n_0 que garantem que $\frac{1}{2}n^2 - 3n \in \Theta(n^2)$?

A resposta é simples: Os valores de c_1 e c_2 são, respectivamente, os mesmos que garantiram que a função estava em $\Omega(n^2)$ e $\mathcal{O}(n^2)$, ou seja $c_1 = \frac{1}{14}$ e $c_2 = \frac{1}{2}$; o valor de n_0 deve ser maior entre os dois valores de n_0 utilizados na demonstração de pertinência a $\mathcal{O}(n^2)$ e $\Omega(n^2)$, ou seja $n_0 = \max\{7, 7\} = 7$.

Verdadeiro ou Falso?

- $3n^5 - 16n + 2 = \mathcal{O}(n^5)$
- $3n^5 - 16n + 2 = \mathcal{O}(n)$
- $3n^5 - 16n + 2 = \mathcal{O}(n^{17})$
- $3n^5 - 16n + 2 = \Omega(n^5)$
- $3n^5 - 16n + 2 = \Omega(n)$
- $3n^5 - 16n + 2 = \Omega(n^{17})$
- $3n^5 - 16n + 2 = \Theta(n^5)$
- $3n^5 - 16n + 2 = \Theta(n)$
- $3n^5 - 16n + 2 = \Theta(n^{17})$

Classe o

$o(g(n)) = \{f(n) :$ para toda constante positiva c, existe uma constante $n_0 > 0$ tal que $0 \le f(n) < cg(n)$, $\forall\ n \ge n_0\}$.

Informalmente, dizemos que, se $f(n) \in o(g(n))$, então $f(n)$ cresce mais lentamente que $g(n)$.

Exemplo: $1000n^2 \in o(n^3)$

Para todo valor de c, um n_0 que satisfaz a definição é

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

A relação $f(n) \in o(g(n))$ denota que a função $g(n)$ dá um limite superior assintótico para o crescimento de $f(n)$, mas esse limite não é justo. Em outras palavras, se $f(n) \in o(g(n))$, então certamente $f(n) \in \mathcal{O}(g(n))$. Mas, se $f(n) \in \mathcal{O}(g(n))$, então ou $f(n) \in o(g(n))$ e $g(n)$ não é um limitante superior justo para $f(n)$, ou $f(n) \in \Theta(g(n))$ e $g(n)$ é um limitante superior justo para $f(n)$.

Exemplo: $1000n^2 \in o(n^3)$

Queremos provar que, para toda constante positiva c, existe uma constante $n_0 > 0$, tal que

$$0 \le 1000n^2 < cn^3, \text{ para todo } n \ge n_0.$$

Dividindo ambos os lados da equação por n^3, percebemos que queremos encontrar valor de n (n_0) a partir do qual a inequação

$$0 \le \frac{1000}{n} < c. \tag{1}$$

Como $\frac{1000}{n}$ é uma função positiva, decrescente e se iguala a c quando $n = \frac{1000}{c}$, basta tomarmos

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

para que a inequação (1) se verifique sempre que $n \ge n_0$.

Classe ω

$\omega(g(n)) = \{f(n) :$ para toda constante positiva c, existe uma constante $n_0 > 0$ tal que $0 \le cg(n) < f(n)$, $\forall n \ge n_0.\}$

Informalmente, dizemos que, se $f(n) \in \omega(g(n))$, então $f(n)$ cresce mais rapidamente que $g(n)$.

Exemplo: $\frac{1}{1000}n^2 \in \omega(n)$

Para todo valor de c, um n_0 que satisfaz a definição é

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

A relação $f(n) \in \omega(g(n))$ denota que a função $g(n)$ dá um limite inferior assintótico para o crescimento de $f(n)$, mas esse limite não é justo. Em outras palavras, se $f(n) \in \omega(g(n))$, então certamente $f(n) \in \Omega(g(n))$. Mas, se $f(n) \in \Omega(g(n))$, então ou $f(n) \in \omega(g(n))$ e $g(n)$ não é um limitante inferior justo para $f(n)$, ou $f(n) \in \Theta(g(n))$ e $g(n)$ é um limitante inferior justo para $f(n)$.

Exemplo: $\frac{1}{1000}n^2 \in \omega(n)$

Queremos provar que, para toda constante positiva c, existe uma constante $n_0 > 0$, tal que

$$0 \le cn < \frac{1}{1000}n^2, \text{ para todo } n \ge n_0.$$

Dividindo ambos os lados da equação por n^2, percebemos que queremos encontrar o valor de n (n_0) a partir do qual a inequação

$$0 \le c < \frac{n}{1000}. \tag{2}$$

Como $\frac{n}{1000}$ é uma função positiva, crescente e se iguala a c quando $n = \frac{c}{1000}$, basta tomarmos

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

para que a inequação (2) se verifique sempre que $n \ge n_0$.

Definições equivalentes

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

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

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

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

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

Propriedades das Classes

Transitividade:

Se $f(n) \in \mathcal{O}(g(n))$ e $g(n) \in \mathcal{O}(h(n))$, então $f(n) \in \mathcal{O}(h(n))$.

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 $f(n) \in \omega(h(n))$.

Vamos provar que a transitividade vale na classe $\mathcal{O}$. Se $f(n) \in \mathcal{O}(g(n))$ e $g(n) \in \mathcal{O}(h(n))$, então existem constantes positivas c_1, c_2, n_1, n_2 tais que

$$0 \leq f(n) \leq c_1 g(n), \text{ para todo } n \geq n_1,$$

e

$$0 \leq g(n) \leq c_2 h(n), \text{ para todo } n \geq n_2.$$

Concluímos então que

$$0 \leq f(n) \leq c_1 c_2 h(n), \text{ para todo } n \geq \max\{n_1, n_2\}.$$

Portanto, as constantes positivas $c = c_1 c_2$ e $n_0 = \max\{n_1, n_2\}$ certificam que $f(n) \in \mathcal{O}(h(n))$.

A demonstração da validade da transitividade para as classes Ω e Θ é análoga. Mas, e para as classes o e ω, como seria a demonstração?

Se $f(n) \in o(g(n))$, então para toda constante positiva c_1, existe uma constante $n_1 > 0$ tal que

$$0 \leq f(n) < c_1 g(n), \text{ para todo } n \geq n_1.$$

Como $g(n) \in o(h(n))$, então para toda constante positiva c_2, existe uma constante $n_2 > 0$ tal que

$$0 \leq g(n) < c_2 h(n), \text{ para todo } n \geq n_2.$$

Tomando então $c_1 = 1$ e o respectivo n_1, concluímos que, para toda constante positiva c_2, existe uma constante $n_0 = \max\{n_1, n_2\} > 0$ tal que

$$0 \leq f(n) < g(n) < c_2 h(n), \text{ para todo } n \geq n_0.$$

Ou seja, $f(n) \in o(h(n))$.

Propriedades das Classes

Reflexividade:

$f(n) \in \mathcal{O}(f(n))$.

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

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

Simetria:

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

Simetria Transposta:

$f(n) \in \mathcal{O}(g(n))$ se, e somente se, $g(n) \in \Omega(f(n))$.

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

A propriedade de reflexividade vale trivialmente para as classes $\mathcal{O}$, Ω e Θ e não vale trivialmente para as classes o e ω. Por quê?

$c = 1$ e $n_0 = 1$ prova a validade para $\mathcal{O}$, Ω e Θ e $c = 1$ prova a não validade para o e $omega$.

Vamos provar que a simetria vale na classe Θ. Se $f(n) \in \Theta(g(n))$ então existem constantes positivas c_1, c_2 e n_0 tais que

$$0 \leq c_1 g(n) \leq f(n), \text{ para todo } n \geq n_0, \tag{3}$$

e

$$0 \leq f(n) \leq c_2 g(n), \text{ para todo } n \geq n_0, \tag{4}$$

Dividindo as inequações (3) e (4) por c_1 e c_2, respectivamente, temos que:

$$0 \leq g(n) \leq \frac{1}{c_1} f(n), \text{ para todo } n \geq n_0,$$

e

$$0 \leq \frac{1}{c_2} f(n) \leq g(n), \text{ para todo } n \geq n_0.$$

Portanto, as constantes positivas $c_2' = \frac{1}{c_2}$, $c_1' = \frac{1}{c_1}$ e n_0 certificam que $g(n) \in \Theta(f(n))$.

Esse mesmo argumento prova a recíproca.

A demonstração da validade da simetria transposta para os pares de classes ($\mathcal{O}$, Ω) e (o, ω) é análoga. Pense nos detalhes!

Exercícios

Quais as relações de comparação assintótica das funções:

- 2^π
- $\log n$
- n
- $n \log n$
- n^2
- $100n^2 + 15n$
- 2^n

Referências

- Cid Carvalho de Souza e Cândida Nunes da Silva, notas de aula da disciplina de MC448 — Análise de Algoritmos I, IC-Unicamp.
- Ian Parberry, Lecture Notes on Algorithm Analysis and Computational Complexity (`http://www.eng.unt.edu/ian/books/free/license.html`)