MCTA003-17 - ANÁLISE DE ALGORITMOS
CENTRO DE MATEMÁTICA, COMPUTAÇÃO E COGNIÇÃO
UNIVERSIDADE FEDERAL DO ABC
PROF. MAYCON SAMBINELLI

UFABC

Notação Assintótica

1. Ordene a lista de funções a seguir por ordem crescente de taxa de crescimento:

$$f_1(n) = n^{2.5},\ f_2(n) = \sqrt{2n},\ f_3(n) = 10^n,\ f_4(n) = 100^n,\ f_5(n) = n^2 \log_2 n$$

$$f_6(n) = 34n^0,\ f_7(n) = 18902n^2,\ f_8(n) = 3\frac{n^{127}}{n^{34}},\ f_9(n) = 456^{8478},\ f_{10}(n) = \frac{n(n+1)}{5}$$

2. Em cada situação a seguir, prove se $f(n) = O(g(n))$ ou $f(n) \neq O(g(n))$, e se $f(n) = \Omega(g(n))$ ou $f(n) \neq \Omega(g(n))$. Comente quando $f(n) = \Theta(g(n))$. Considere que a e b são constantes positivas:

(a) $f(n) = n^2 + 10n + 20$ e $g(n) = n^2$
(b) $f(n) = n^{1/2}$ e $g(n) = n^{2/3}$
(c) $f(n) = \frac{n(n+1)(n+2)}{2}$ e $g(n) = n^2 \log n$
(d) $f(n) = \frac{n}{1000}$ e $g(n) = 50^{100}$
(e) $f(n) = \log_a n$ e $g(n) = \log_b n$ (o que esse resultado significa?)
(f) $f(n) = n^{1.05}$ e $g(n) = n \ln n$
(g) $f(n) = 100^{n+a}$ e $g(n) = 100^n$
(h) $f(n) = 100^{an}$ e $g(n) = 100^n$
(i) $f(n) = 99^{n+a}$ e $g(n) = 100^n$
(j) $f(n) = n^{1.01}$ e $g(n) = n \log^2 n$
(k) $f(n) = \log \sqrt{n}$ e $g(n) = \log(100n)$
(l) $f(n) = 2^{n+1}$ e $g(n) = 2^n$
(m) $f(n) = \log \sqrt{n}$ e $g(n) = \log 10.000n$
(n) $f(n) = 2^{2n}$ e $g(n) = 2^n$
(o) $f(n) = (n+a)^b$ e $g(n) = \Theta(n^b)$ com $b > 0$
(p) $f(n) = n \log n$ e $g(n) = 10n \log 10n$
(q) $f(n) = n!$ e $g(n) = n \log_2 n^2$
(r) $f(n) = 10 \log n$ e $g(n) = \log(n^2)$
(s) $f(n) = n!$ e $g(n) = n \log n$ (dica: n^n e $(n/2)^{n/2}$ podem ser úteis)

3. Usando **limites**, em cada situação a seguir, prove se $f(n) = O(g(n))$ ou $f(n) \neq O(g(n))$, e se $f(n) = \Omega(g(n))$ ou $f(n) \neq \Omega(g(n))$. Comente quando $f(n) = \Theta(g(n))$.

(a) $f(n) = 100n + \log n$ e $g(n) = n + \log^2 n$
(b) $f(n) = 50 \lg \sqrt{n}$ e $g(n) = \lg n^3$
(c) $f(n) = 1.01^n$ e $g(n) = n^{1.01}$
(d) $f(n) = 2^n$ e $g(n) = 2^{\sqrt{\lg n}}$

4. Prove que

(a) $\sum_{i=1}^{n} i^k$ é $\Theta(n^{k+1})$

(b) $\sum_{i=1}^{n} \frac{i}{2^i}$ é $O(1)$

5. Sejam $f(n)$ e $g(n)$ funções crescentes e maiores do que 1 tais que $f(n)$ é $O(g(n))$. Isto é, existem constantes d e n_0 tais que $f(n) \leq dg(n)$ sempre que $n \geq n_0$. Note que se tomarmos $c = \max\{1, d\}$, também vale que $f(n) \leq cg(n)$. Para cada item a seguir, decida se o mesmo é verdadeiro ou falso e dê uma prova ou contraexemplo:

(a) se $f(n)$ é $O(g(n))$, então $\log f(n)$ é $O(\log g(n))$

(b) $2^{f(n)}$ é $O(2^{g(n)})$

(c) $f(n)^2$ é $O(g(n)^2)$

(d) $\log \sqrt{n} = O(\log n)$

(e) se $f(n) = O(g(n))$ e $g(n) = O(h(n))$, então $f(n) = O(h(n))$

(f) se $f(n) = O(g(n))$ e $g(n) = \Theta(h(n))$, então $f(n) = \Theta(h(n))$

6. Considere um polinômio $P(n)$ de grau k, isto é, $P(n) = \sum_{i=0}^{k} a_i n^i$, onde cada a_i é uma constante e $a_k > 0$. Seja t uma constante. Prove que

(a) se $t \geq k$, então $P(n)$ é $O(n^t)$.

(b) se $t \leq k$, então $P(n)$ é $\Omega(n^t)$.

(c) se $t = k$, então $P(n)$ é $\Theta(n^t)$.

7. Sejam $f(n)$ e $g(n)$ funções assintoticamente não negativas. Usando a definição da notação Θ, prove que $\max\{f(n), g(n)\} = \Theta(f(n) + g(n))$. O que esse resultado significa?

8. Sejam f e g funções positivas. Prove as seguintes afirmações

(a) Se $\lim_{n\to\infty} \frac{f(n)}{g(n)} \neq \infty$, então $f = O(g(n))$

(b) Se $\lim_{n\to\infty} \frac{f(n)}{g(n)} \neq 0$, então $f = \Omega(g(n))$

1 Rapidinhas

9. Você provou que seu algoritmo tem tempo de execução $\Omega(n^2)$ no pior caso. O que isso diz sobre o tempo de execução do algoritmo sobre qualquer entrada? Você acredita que o tempo de execução no melhor caso é $\Omega(n^3)$. Isso é uma contradição com o resultado anterior?

10. É possível que um algoritmo tenha tempo de execução $\Omega(n^3)$ e $O(n^2 \log n)$ no pior caso? Justifique.

11. Se um algoritmo tem tempo de execução $O(n^2)$ sobre todas as entradas de tamanho n, ele pode ter tempo de execução $\Omega(n \log n)$ no pior caso?

12. Se um algoritmo tem tempo de execução $\Theta(n^2)$ sobre todas as entradas de tamanho n, ele pode ter tempo de execução $\Omega(n \log n)$ no pior caso?

13. Explique por que a declaração "O tempo de execução do algoritmo A é no mínimo $O(n^2)$" não faz sentido.