MCTA003-17 - Análise de Algoritmos

Aula 03 - Tempo de Execução

Maycon Sambinelli
m.sambinelli@ufabc.edu.br
https://professor.ufabc.edu.br/~m.sambinelli/

2022.Q2

Centro de Matemática, Computação e Cognição
Universidade Federal do ABC

UFABC

Outline

Introdução

Introdução

UFABC

Vários fatores afetam o *tempo de execução* de um programa:

- Hardware
- Linguagem
- Tamanho da entrada
- Algoritmo

Queremos um conceito de tempo que seja independente de:

- Hardware
- Linguagem

Assim, podemos realmente comparar o *tempo de execução de um algoritmo* em relação ao *tamanho da entrada*.

Nosso modelo de computação

UFABC

- Nosso modelo de computador teórico
  - Captura a essência de um computador real
- Operações primitivas: operações que podem ser realizadas rapidamente sobre um número pequeno (ex: 32, 64, 128 bits).

Operações primitivas

UFABC

- Operações aritméticas: soma, subtração, multiplicação, divisão, resto, piso, teto
- Operações relacionais: $<, \leq, >, \geq, =$
- Operações lógicas: and, or, not
- Atribuição de valores, carregamento de valores (acesso a memória)
- Operações de controle de fluxo de execução

Tempo de Execução

UFABC

O tempo de execução de um algoritmo é dado pela quantidade de operações primitivas (passos simples) executadas por ele sobre uma certa instância de entrada.

- O tempo de execução cresce junto com a entrada.
- O tempo de execução é uma função T sobre o tamanho da entrada.
  - $T(n)$ denota o número de operações primitivas realizadas pelo algoritmo quando executada sobre uma entrada de tamanho n.

Receita de bolo: tamanho da entrada

- vetor, lista, conjunto: n é a quantidade de elementos
- números: n é a quantidade de bits na representação binária de n.

busca linear

UFABC

busca binária

UFABC

Análise por caso

Análise por caso

O **tempo de melhor caso** de um algoritmo é o menor tempo de execução do algoritmo dentre todos os tempos de execução de todas as instâncias de um dado tamanho n.

O **tempo de pior caso** de um algoritmo é o maior tempo de execução do algoritmo dentre os tempos de execução de todas as instâncias de um dado tamanho n.

tempo no melhor caso $\leq T(n) \leq$ tempo no pior caso

Caso Médio

O tempo de caso médio de um algoritmo é a média do tempo de execução de todas as instâncias de tamanho n.

- consideramos algor sobre a distribuição das entradas e fazemos uma análise probabilística
- pode ser tão ruim quanto o pior caso
- análise de caso médio costuma ser mais complicada

Algoritmo eficiente

UFABC

Um algoritmo é eficiente se seu tempo de execução no pior caso puder ser descrito por uma função que é limitadas superiormente por uma função polinomial no tamanho da entrada.

- ambas as buscas são eficientes?
  - qual é melhor?

Notação Assintótica

Notação Assintótica

UFABC

- É uma abstração que nos permite focar no que ocorre com $T(n)$ quando n cresce indefinidamente.
  - termos de menor ordem não importam
  - constantes não importam

exemplo 1:

$$f(n) = n^4 + 786n^3 + 2n^2 - 999n + 12$$

exemplo 2:

$$f(n) = 475n + 34 \qquad \text{e} \qquad g(n) = n^2 + 3$$

Notação O

Seja n uma inteiro positivo e sejam $f(n)$ e $g(n)$ funções positivas. Dizemos que

$$f = O(g) \text{ ou } f \text{ é } O(g)$$

se existem constantes positivas C e n_0 tais que

$$f(n) \leq Cg(n) \qquad \forall n \geq n_0.$$

Exemplo

$$f(n) = 5n + 3$$

$$g_1(n) = n \qquad g_2(n) = n^2 \qquad g_3(n) = \sqrt{n}$$

Notação Ω

UFABC

Seja n uma inteiro positivo e sejam $f(n)$ e $g(n)$ funções positivas. Dizemos que

$$f = \Omega(g) \text{ ou } f \text{ é } \Omega(g)$$

se existem constantes positivas C e n_0 tais que

$$f(n) \geq Cg(n) \qquad \forall n \geq n_0.$$

Exemplo

$$f(n) = 5n + 3$$

$$g_1(n) = n \qquad g_2(n) = n^2 \qquad g_3(n) = \sqrt{n}$$

Exemplo $f(n) = 2n^2$ e $g(n) = n^2 + 10n + 20$

Notação Θ

Seja n uma inteiro positivo e sejam $f(n)$ e $g(n)$ funções positivas. Dizemos que

$$f = \Theta(g) \text{ ou } f \text{ é } \Theta(g)$$

se existem constantes positivas C_1, C_2 e n_0 tais que

$$C_1 g(n) \leq f(n) \geq C_2 g(n) \qquad \forall n \geq n_0.$$

Obs f é $\Theta(g)$ se e somente se f é $O(g)$ e é $\Omega(g)$.

Obs $f(3n + 5) = O(n), O(n^2), O(n^3), ...$

Exemplo

$$f(n) = 5n + 3$$

$$g_1(n) = n \qquad g_2(n) = n^2 \qquad g_3(n) = \sqrt{n}$$

Notação o

UFABC

Seja n uma inteiro positivo e sejam $f(n)$ e $g(n)$ funções positivas. Dizemos que

$$f = o(g) \text{ ou } f \text{ é } o(g) \text{ ou } f << g$$

se para toda constante $C > 0$ existe um n_0 tal que

$$f(n) < Cg(n) \qquad \forall n \geq n_0.$$

Sumário

UFABC

$f = \Theta(g)$	f cresce com a mesma intensidade de g
$f = O(g)$	f não cresce mais rápido que g
$f = \Omega(g)$	f cresce ao menos tão rápido quanto g
$f = o(g)$	f cresce mais devagar que g

Propriedades

UFABC

Teorema

Sejam $f(n)$, $g(n)$ e $h(n)$ funções positivas. Temos que as seguintes afirmações são verdadeiras

1. $f(n) = \Theta(f(n))$
2. $f(n) = \Theta(g(n))$ se e somente se $g(n) = \Theta(f(n))$
3. $f(n) = O(g(n))$ se e somente se $g(n) = \Omega(f(n))$
4. $f(n) = O(g(n))$ e $g(n) = O(h(n))$, então $f(n) = O(h(n))$ (o mesmo vale se trocarmos O por Ω ou Θ)
5. $f(n) = O(g(n) + h(n))$ se e somente se $f(n) = O(g(n)) + O(h(n))$ (o mesmo vale se trocarmos O por Ω ou Θ)

UFABC

As definições de notação assintótica estão relacionadas ao conceito de limite. Assim, $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ revela a relação assintótica entre f e g (desde que esse limite exista). As seguintes relações são conhecidas

UFABC

Teorema

1. $$\lim_{n\to\infty} \frac{f(n)}{g(n)} \neq 0, \infty \Rightarrow f = \Theta(g)$$
2. $$\lim_{n\to\infty} \frac{f(n)}{g(n)} \neq \infty \Rightarrow f = O(g)$$
3. $$\lim_{n\to\infty} \frac{f(n)}{g(n)} \neq 0 \Rightarrow f = \Omega(g)$$
4. $$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \Rightarrow f = o(g)$$

UFABC

Exemplo: $f(n) = 3n^4 - 5n^3 + 4n$ e $g(n) = n^4$

demonstração:

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} \neq 0, \infty \Rightarrow f = \Theta(g)$$

Teorema Sejam f e g duas funções positivas. Se $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$, então $f = o(g)$.

Demonstração

Teorema Para todas $\alpha, k > 0$ pertencentes aos reais, vale que

$$(\ln n)^k = o(n^\alpha)$$

$$n^k = o((1+\alpha)^n)$$

Demonstração

UFABC

Corolário Para todo $\alpha, k > 0$ pertencentes aos reais e para todo $b > 0$ tal que $b \neq 1$, temos que

$$(\log_b n)^k << o(n^\alpha).$$

Receita de Bolo de Chocolate

UFABC

$$c << \lg n << \sqrt{n} << n^c << n^c \lg n << 2^n << n!,$$

onde $c \geq 1$ é uma constante.

Escrevemos $f = O(1)$ e $f = \Theta(1)$ para denotar que f é limitada superiormente e inferiormente, respectivamente, por uma constante C.

Quando $f = O(1)$ e $f = \Omega(1)$, escrevemos $f = \Theta(1)$.