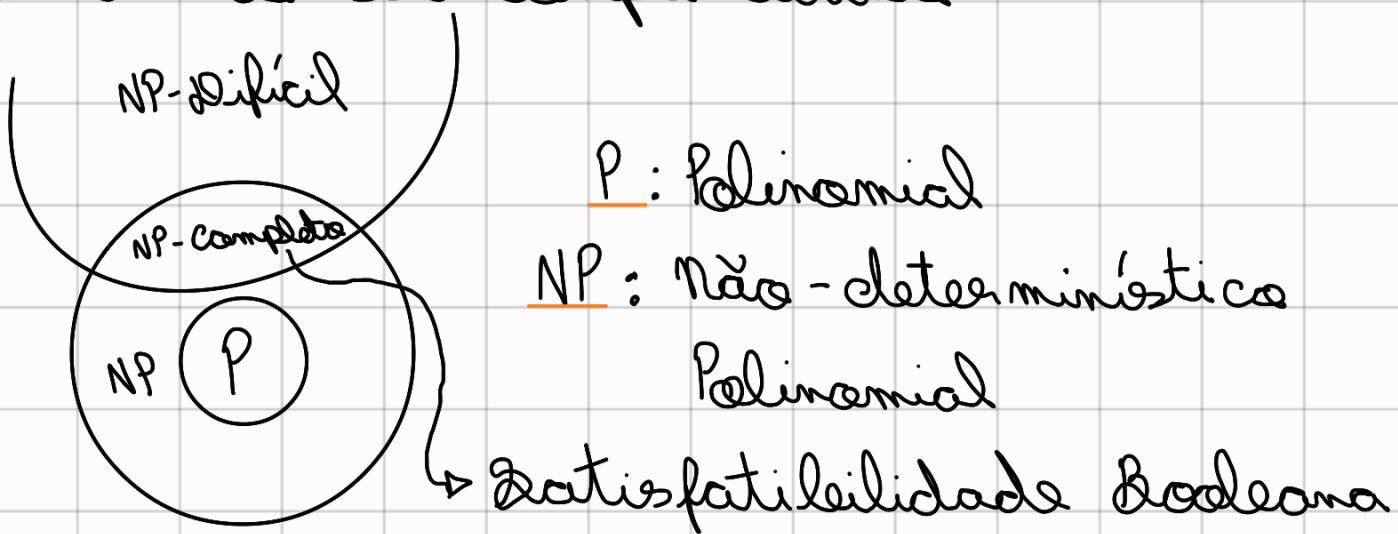


## Classes de complexidade

A complexidade assintótica é uma medida aproximada da quantidade de operações de um algoritmo. Entretanto em algoritmo existe para resolver um problema. Logo é comum referir-se à **complexidade do problema** como sendo a complexidade do melhor algoritmo para resolver aquele problema.

Isso "divide" os problemas em conjuntos a depender de sua complexidade.



Problema da classe:

- **P**: Há algoritmo polinomial para resolvê-los.
- **NP**: São problemas:
  - ↳ de decisão

- ↳ Se um círculo der uma solução, ela pode ser validada em tempo polinomial
- ↳ Não se conhece o algoritmo polinomial para resolvê-los.

- NP-difícil: problemas pelo menos tão difíceis quanto qualquer problema em NP, entretanto não necessariamente de decisão.

- NP-completo: problema da classe NP para o qual qualquer outro problema NP pode ser transformado em tempo polinomial.

**Questão aberta:  $P = NP?$**

**Casos comuns p/ análise de complexidade**

Regras de ouro:

- ↳ Se o seu algoritmo possuir trechos de diferentes complexidades, prevalece a mais alta.
- ↳ A complexidade é sempre medida em termos do:
  - tamanho da entrada.
  - Parâmetros variáveis relacionados ao problema.

$f \in O(g) \Leftrightarrow f(n) \leq c \cdot g(n), \forall n \geq n_0$

$$n^3 + 2n^2 + n + 1 \in O(\underline{n^3})$$

$$\leq n^3 + 2n^3 + n^3 + 1, \forall n \geq 0$$

$$= 4n^3 + 1$$

$$\leq 5n^3, \forall n \geq 1$$

C  $n_0$

Exemplos: Qual a complexidade para:

① Acessar o  $i$ -ésimo elemento de um vetor.

$$O(1) \rightarrow v[i]$$

→ Posição

$$0 \quad 1 \quad 2 \quad 3 \quad 4 \quad \dots \quad i \quad \dots \quad n-1$$



(INT)

$\underbrace{100}_{\text{endereço inicial de } v} + \underbrace{4 \times i}_{\text{tamanho do tipo de dado}} \rightarrow \text{Posição}$  }  $O(1)$

② Encontrar um elemento  $x$  num vetor de tam.  $n$ .

FOR(INT  $i=0$ ;  $i < n$ ;  $i++$ )

IF ( $v[i] == x$ ) RETURN  $i$ ;

RETURN -1;

}  $O(n)$  → Pior caso  
                  → Caso médio

$\Omega(1)$

|     | I                      | #J |
|-----|------------------------|----|
| 0   | n                      |    |
| 1   | n                      |    |
| 2   | n                      |    |
| ⋮   |                        |    |
| n-1 | $\frac{n}{n \times n}$ |    |
|     | = n^2                  |    |

④ FOR( $i=0; i < 1000; i++$ )  
 /\* CÓDIGO  $O(1)$  \*/

Comp.:  $O(1)$

⑤ FOR( $i=1; i < n; i = i * 2$ )  
 /\* CÓDIGO  $O(1)$  \*/

$$i = \underbrace{1 \times 2 \times 2 \times 2 \dots \times 2}_k \geq n \rightarrow 2^k \geq n$$

$$\lg 2^k \geq \lg n$$

$$k \geq \lg n \Rightarrow O(\lg n)$$

⑥ FOR( $i=0$ ;  $i < n$ ;  $i++$ )

FOR( $j=i$ ;  $j < n$ ;  $j++$ )

/\* CÓDIGO  $\Theta(1)$  \*/

$$n + (n-1) + (n-2) + \dots + 2 + 1 \uparrow$$

| $i$      | # $J$       |
|----------|-------------|
| 0        | $n$         |
| 1        | $n-1$       |
| 2        | $n-2$       |
| 3        | $n-3$       |
| $\vdots$ | $\vdots$    |
| $n-1$    | $n-n+1 = 1$ |

$$S = (a_1 + a_m) \frac{n}{2} = (n+1) \frac{n}{2}$$

$$= \frac{n^2}{2} + \frac{n}{2} \rightarrow \Theta(n^2)$$

⑦ FOR( $i=n$ ;  $i>0$ ;  $i = i/2$ )

FOR( $j=0$ ;  $j < i$ ;  $j++$ )

/\* CÓDIGO  $\Theta(1)$  \*/

$$n + \frac{n}{2} + \frac{n}{4} + \dots + 0 \leq n + \frac{n}{2} + \frac{n}{4} + \dots + = \frac{Q_0}{1-q} = \frac{n}{1-\frac{1}{2}} = n$$

$$= \frac{n}{\frac{1}{2}} = 2n$$

Comp.:  $\Theta(n)$

⑧ Multiplicação de matrizes:  $A_{m \times p} \times B_{p \times n} = C_{m \times n}$

$$C_{ij} = \sum_{k=1}^p a_{ik} \times b_{kj}$$

FOR( $i=0$ ;  $i < m$ ;  $i++$ )

  FOR( $j=0$ ;  $j < n$ ;  $j++$ ) {

$C[i][j] = 0$ ;

    FOR( $k=0$ ;  $k < p$ ;  $k++$ )

$C[i][j] += a[i][k] * b[k][j]$ ;

}

Comp:  $O(m \cdot n \cdot p)$

Cúbica