Introdução à Análise de Complexidade
Resolução de exercícios de testes

Eduardo Freitas Fernandes

2025

Exercício 1

Nesta análise, contamos o número de acessos ao array `v`:

- **melhor caso**: array estritamente crescente.

$$T_{melhor}(N) = \sum_{i=0}^{N-1} 1 = \Omega(N)$$

- **pior caso**: array descrescente.

$$T_{pior}(N) = \sum_{i=0}^{N-1} 1 + \sum_{i=0}^{N-1} \sum_{j=0}^{i-1} 1 = N + \sum_{i=0}^{N-1} i = N + \frac{N \times (N-1)}{2} = O(N^2)$$

Nesta análise, contamos o número de acessos ao array `a`:

$$T(N) = \sum_{i=0}^{N-1} \sum_{j=0}^{i-1} 2 = \sum_{i=0}^{N-1} 2 \times i = 2 \times \frac{N \times (N-1)}{2} = \Theta(N^2)$$

Exercício 2

$$T(N) = \begin{cases} 0 & N \leq 0 \\ T_{processa}(N) + 2 \times T(N/2) & N > 0 \end{cases} = \begin{cases} 0 & N \leq 0 \\ N + 2 \times T(N/2) & N > 0 \end{cases}$$

$$= \sum_{i=1}^{1+\log_2(N)} N = N \times (1 + \log_2(N)) = \Theta(N \times \log_2(N))$$

Exercício 3

Nesta análise, contamos o número de execuções da função `swap()`:

- **melhor caso**: array crescente.

$$T_{bubble}(N) = \Omega(1), \quad T_{bsort}(N) = \Omega(1)$$

- **pior caso**: array estritamente decrescente.

$$T_{bubble}(N) = O(N), \quad T_{bsort}(N) = \sum_{i=0}^{N} N = O(N^2)$$

Nesta análise, contamos o número de comparações feitas entre elementos do array `a` (i.e: `(a[i] < a[i-1])`)

- **melhor caso**: array crescente, `bubble()` executa uma vez, e `bsort()` termina o ciclo.

$$T_{bsort}(N) = \sum_{i=0}^{N-1} 1 = \Omega(N)$$

- **pior caso**: array estritamente decrescente, o ciclo em `bsort()` executa `N` vezes.

$$T_{bsort}(N) = \sum_{i=0}^{N-1} N = O(N^2)$$

Exercício 4

Nesta análise, contamos o número de vezes que a operação `k*=i` é executada:

$$T_{factorial}(N) = \Theta(N), \quad T_{factoriais}(N) = \sum_{i=0}^{N-1} N = \Theta(N^2)$$

Exercício 5

Exercício 6

Nesta análise, contamos o número de vezes que a condição `B[j] <= A[i]` é avaliada:

- **melhor caso**:

$$T(N) = \sum_{i=0}^{N-1} 1 = \Omega(N)$$

- **pior caso**:

$$T(N) = \sum_{i=0}^{N-1} \sum_{j=0}^{i-1} 1 = \sum_{i=0}^{N-1} i = \frac{N \times (N-1)}{2} = O(N^2)$$

Exercício 7

$$T(N) = \begin{cases} 0 & N \leq 0 \\ 1 + 2 \times T(N/2) & N > 0 \end{cases} = \sum_{i=0}^{\log_2(N)} 2^i = \frac{2^{1+\log_2 N} - 1}{2 - 1} = \Theta(\log_2 N)$$

Exercício 8

Nesta análise, iremos avaliar o pior caso, em que `k = N`:

$$T(N) = \begin{cases} 0 & N \leq 0 \\ N + T(N-1) & N > 0 \end{cases} = \sum_{i=1}^{N} i = \frac{N \times (N+1)}{2} = O(N^2)$$

Exercício 9

```
void shift(int u[], int n, int k) {
    int storage[N];
    int i;
    // calcular novas posicoes
    for (i = 0; i < N; i++)
        storage[(i + k) % n] = u[i];
    // copiar para o array original
    for (i = 0; i < N; i++)
        u[i] = storage[i];
}
```

Exercício 10

Nesta análise, iremos contar o número de vezes que o array `bits` é alterado (i.e: `bits[n] = ...`):

- **melhor caso**: bit menos significativo (à direita) a zero.

$$T(N) = \Omega(1)$$

- **pior caso**: todos os bits a 1, ou o mais significativo a 0, e os restantes a 1.

$$T(N) = \sum_{i=0}^{N-1} 1 = O(N)$$

i	input	output	c_i
1	$\boxed{0}\ \boxed{0}\ \boxed{0}\ \boxed{0}$	$\boxed{0}\ \boxed{0}\ \boxed{0}\ \boxed{1}$	$1 = 1$
2	$\boxed{0}\ \boxed{0}\ \boxed{0}\ \boxed{1}$	$\boxed{0}\ \boxed{0}\ \boxed{1}\ \boxed{0}$	$2 = 1 + 1$
3	$\boxed{0}\ \boxed{0}\ \boxed{1}\ \boxed{0}$	$\boxed{0}\ \boxed{0}\ \boxed{1}\ \boxed{1}$	$1 = 1$
4	$\boxed{0}\ \boxed{0}\ \boxed{1}\ \boxed{1}$	$\boxed{0}\ \boxed{1}\ \boxed{0}\ \boxed{0}$	$3 = 1 + 1 + 1$
5	$\boxed{0}\ \boxed{1}\ \boxed{0}\ \boxed{0}$	$\boxed{0}\ \boxed{1}\ \boxed{0}\ \boxed{1}$	$1 = 1$
6	$\boxed{0}\ \boxed{1}\ \boxed{0}\ \boxed{1}$	$\boxed{0}\ \boxed{1}\ \boxed{1}\ \boxed{0}$	$2 = 1 + 1$
7	$\boxed{0}\ \boxed{1}\ \boxed{1}\ \boxed{0}$	$\boxed{0}\ \boxed{1}\ \boxed{1}\ \boxed{1}$	$1 = 1$
8	$\boxed{0}\ \boxed{1}\ \boxed{1}\ \boxed{1}$	$\boxed{1}\ \boxed{0}\ \boxed{0}\ \boxed{0}$	$4 = 1 + 1 + 1 + 1$
9	$\boxed{1}\ \boxed{0}\ \boxed{0}\ \boxed{0}$	$\boxed{1}\ \boxed{0}\ \boxed{0}\ \boxed{1}$	$1 = 1$
10	$\boxed{1}\ \boxed{0}\ \boxed{0}\ \boxed{1}$	$\boxed{1}\ \boxed{0}\ \boxed{1}\ \boxed{0}$	$2 = 1 + 1$

O que pretendemos calcular é a soma dos elementos da última coluna $\sum c_i$ para depois calcular a sua média. Neste caso, podemos ver que:

- todas as linhas têm a parcela 1
- metade das linhas têm também a parcela +1
- um quarto das linhas têm também mais uma parcela +1
- ...

Generalizando, a soma dos valores da última coluna pode ser calculada por:

$$C_N = N + \frac{N}{2} + \frac{N}{4} + ... = \sum_{i=0}^{\log_2 N} \frac{N}{2^i} = N \times (2 - \frac{1}{2^{\log_2 N}}) = 2 \times N - 1$$

Para sabermos o custo amortizado de cada operação, temos que dividir este custo pelo número de operações em causa:

$$\hat{c}_i = \frac{C_N}{N} = \frac{2 \times N - 1}{N} = 2 - \frac{1}{N} = \Theta(1)$$

Exercício 11