



Análise de Algoritmos – DCC606

Análise do Algoritmo “VerificaAlgo”

Apresentado por:
George Lucas
Zambonin e
Wandressa Reis

Data:
26 de agosto de 2024

Conteúdo

- Função de custo e complexidade;
- Código em C do algoritmo proposto;
- Experimentação com a execução do algoritmo com diferentes entradas e coleta de tempo de execução;
- Gráfico de linha com o tempo de execução em relação a cada entrada e análise da tendência de comportamento assintótico.

Pseudocódigo

```
VerificaAlgo (n: int);  
i, j, k, l: int;  
para l := 1 TO 10.000 faça  
    para i := 1 TO n-5 faça  
        para j := i+2 TO n/2 faça  
            para k := 1 TO n faça  
                {Inspeçione elemento}
```

Função de Custo (passo- a-passo slide)

$$\sum_{l=1}^{10000} \sum_{i=1}^{n-5} \sum_{j=i+1}^{n/2} \sum_{k=1}^n =$$

$$\sum_{l=1}^{10000} \sum_{i=1}^{n-5} \sum_{j=i+1}^{n/2} n =$$

$$\sum_{l=1}^{10000} \sum_{i=1}^{n-5} \left[\frac{n}{2} - (i+1) + 1 \right] n =$$

$$\sum_{l=1}^{10000} \sum_{i=1}^{n-5} \left[\frac{n^2}{2} - in \right] =$$

$$\sum_{l=1}^{10000} \left[\sum_{i=1}^{n-5} \frac{n^2}{2} - n \sum_{i=1}^{n-5} i \right] =$$

...

$$20000n^2 - 100000n$$

Função de Custo

$$\begin{aligned}
 & \sum_{l=1}^{10000} \sum_{i=1}^{n-5} \sum_{j=i+2}^{\frac{n}{2}} \sum_{k=1}^n = \sum_{l=1}^{10000} \sum_{i=1}^{n-5} \sum_{j=i+2}^{\frac{n}{2}} (n-1+1) \\
 & = \sum_{l=1}^{10000} \sum_{i=1}^{n-5} \sum_{j=i+2}^{\frac{n}{2}} \cdot n = \sum_{l=1}^{10000} \sum_{i=1}^{n-5} \left(\left(\frac{n}{2} - (i+2) + 1 \right) \cdot n \right) \\
 & = \sum_{l=1}^{10000} \sum_{i=1}^{n-5} \left(\left(\frac{n}{2} - i - 1 \right) n \right) = \sum_{l=1}^{10000} \sum_{i=1}^{n-5} \left(\frac{n^2}{2} - ni - n \right) \\
 & = \sum_{l=1}^{10000} \left[\frac{n^2}{2} \cdot \sum_{i=1}^{n-5} 1 - n \sum_{i=1}^{n-5} i - n \sum_{i=1}^{n-5} 1 \right] \\
 & = \sum_{l=1}^{10000} \left[\frac{n^2}{2} ((n-5) - 1 + 1) - n((n-5)(n-5+1) - n((n-5) - 1 + 1)) \right] \\
 & = \sum_{l=1}^{10000} \left[\frac{n^2}{2} (n-5) - \frac{n(n-5)(n-4) - n(n-5)}{2} \right] \\
 & = \sum_{l=1}^{10000} \left[\frac{n^3 - 5n^2}{2} - \frac{n^3 + 4n^2 + 5n^2 - 20n}{2} - n^2 + 5n \right] \\
 & = \sum_{l=1}^{10000} \left[\frac{4n^2 - 20n}{2} - n^2 + 5n \right] \\
 & = \sum_{l=1}^{10000} \left[\frac{4n^2 - 20n}{2} - \frac{2n^2 + 10n}{2} \right] \\
 & = \sum_{l=1}^{10000} \left[\frac{2n^2 - 10n}{2} \right] = \left[\frac{2n^2 - 10n}{2} \right] \sum_{l=1}^{10000} 1
 \end{aligned}$$

$$\begin{aligned}
 & = \left[\frac{2n^2 - 10n}{2} \right] 10000 - 1 + 1 \\
 & = \frac{20000n^2 - 100000n}{2} = 10000n^2 - 50000n
 \end{aligned}$$

Função de Complexidade

A partir da função de custo apresentada, podemos concluir:

- $O(n^2)$

$$= \left[\frac{2n^2 - 10n}{2} \right] 10000 - \cancel{1} + \cancel{1}$$
$$= \frac{20000n^2 - 100000n}{2} = 10000n^2 - 50000n //$$

Função de Custo e Complexidade

Função de custo correta:

$$S = \frac{n \times (\min(n - 5, \lfloor n/2 \rfloor - 1)) \times (\min(n - 5, \lfloor n/2 \rfloor - 2))}{2}$$

Complexidade:

- $O(n^3)$

Algoritmo em C

```
8  #define BILLION 1000000000L
9
10 // return comparisons_count;
11 int verifica_algo(int n){
12     int i, j, k, l;
13
14     int comparisons_count = 0;
15
16     for (l=0; l<10000; l++) {
17         for(i=0; i < n-5; i++) {
18             for (j=i+2; j<n/2; j++) {
19                 for (k=0; k < n; k++) {
20                     comparisons_count++;
21                     // inspeciona elemento
22                 }
23             }
24         }
25     }
26     return comparisons_count;
27 }
```

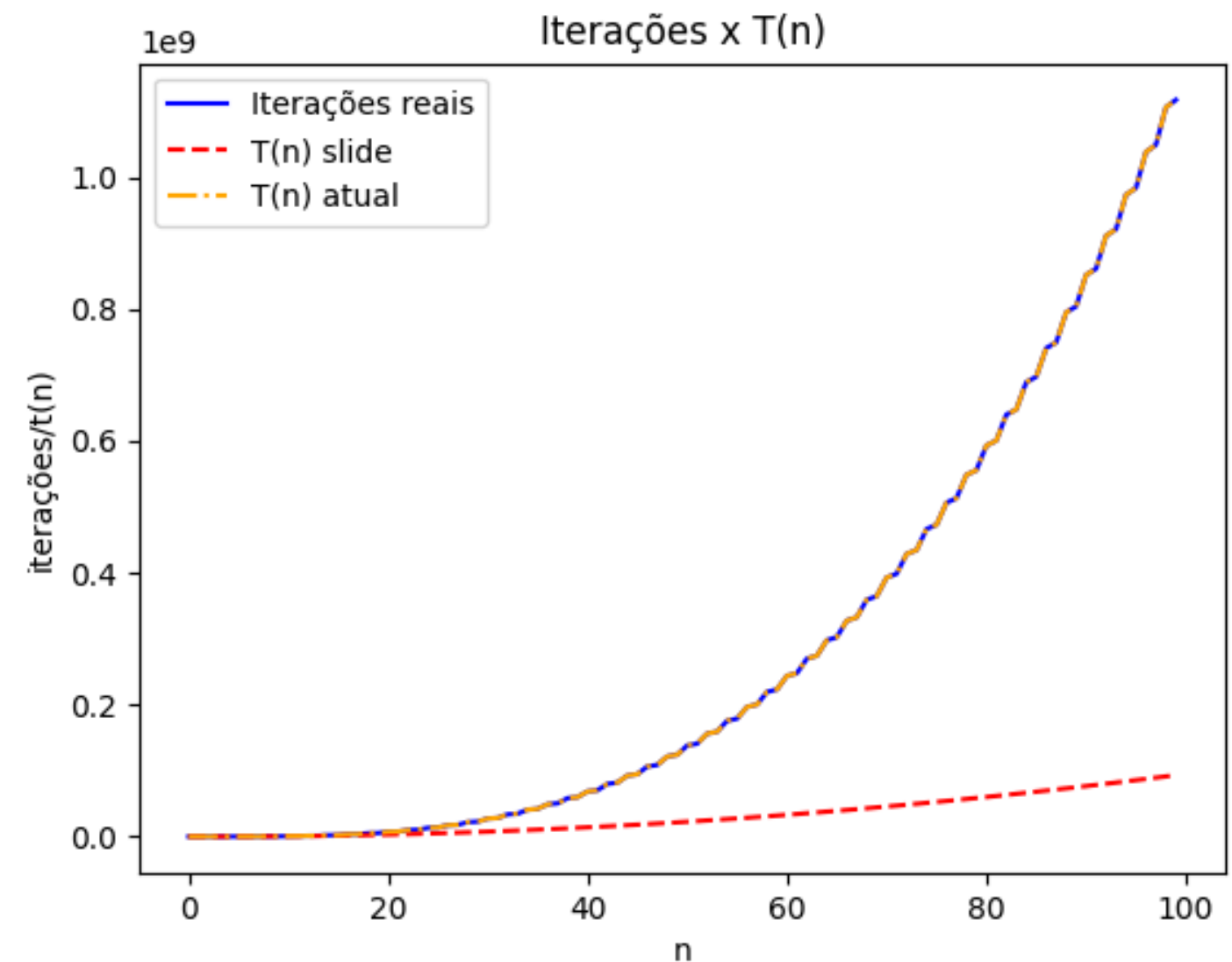

Comparação T(n)

$$T(n) = 10000n^2 - 50000n$$

↻ Baseada no cálculo disponibilizado nos slides

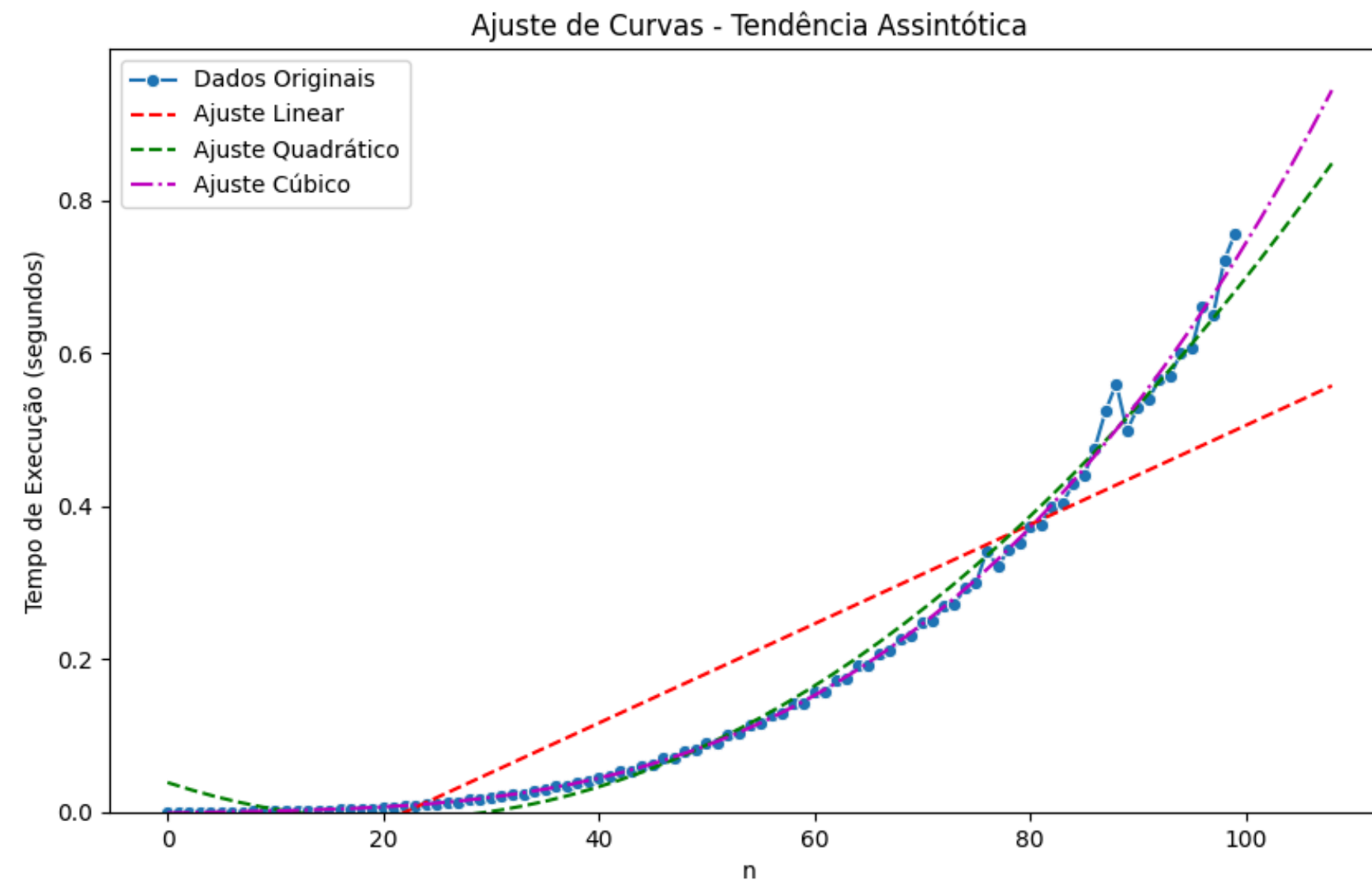
$$T(n) = \frac{n \cdot \left(\min \left(n - 5, \left\lfloor \frac{n}{2} \right\rfloor - 1 \right) \right) \cdot \left(\min \left(n - 5, \left\lfloor \frac{n}{2} \right\rfloor - 2 \right) \right)}{2}$$

↻ Baseada em nossas análises



Complexidade

↳ Obedece a curva “Ajuste Cúbico”



Projeção assintótica $O(n^3)$

