Análise de Algoritmos – DCC606

Algoritmo – Faz Algo

Lucas Anderson Ladislau Aguiar
Lucas Prado Ribeiro

O Algoritmo

FazAlgo

```
void FazAlgo (int n) {
    int i, j, k;
    FOR (i= 1; i<n - 1; i++) {
        FOR (j= i + 1; j<= n; j++) {
            FOR (k = 1; k<=j;k++) {
                Algum comando de custo O(1)
    } } }
}
```

$$C(n) = \sum_{i=1}^{n-2} \sum_{j=i+1}^{n} \sum_{k=1}^{j} O(1)$$

Complexidade

$$C(n) = \frac{1}{3}n^3 - \frac{4}{3}n$$

$$O(n^3)$$

Algoritmo para o teste

```
void FazAlgo(int n) {
  int i, j, k;
  long contador = 0;
  for (i = 1; i < n - 1; i++) {
      for (j = i + 1; j <= n; j++) {
          for (k = 1; k <= j; k++) {
             contador++;
           }
       }
   }

 printf("%ld\n", contador);
}
```

```
void FazAlgoOtimizado(int n) {
  int i, j, k;
  long contador = 0;
  for (i = 1; i < n - 1; i++) {
      for (j = i + 1; j <= n; j++) {
             contador+=j;
       }
   }

 printf("%ld\n", contador);
}
```

Testes – Faz Algo $O(n^3)$

Entrada(n)	Tempo para executar(s)
500	~0.065484
1000	~0.519335
2000	~4.134691
2500	~8.062164
3000	~13.92312
4000	~33.01765
5000	~64.44222

Entrada(n)	Tempo para executar(s)
6000	~111.292495
7000	~177.310919
8000	~263.98722
9000	~375.951576
10000	~516.03524
11000	~685.58918
12000	~890.5404

Testes – Otimizado $O(n^2)$

Entrada(n)	Tempo para executar(s)
500	0.000256
1000	~0.000956
2000	0.003767
2500	~0.006930
3000	~0.009019
4000	~0.0150286
5000	~0.023361

Entrada(n)	Tempo para executar(s)
6000	0.033666
7000	~0.046191
8000	~0.061346
9000	~0.076954
10000	0.0948915
11000	0.1134005
12000	~0.1345881

Gráfico – Faz Algo

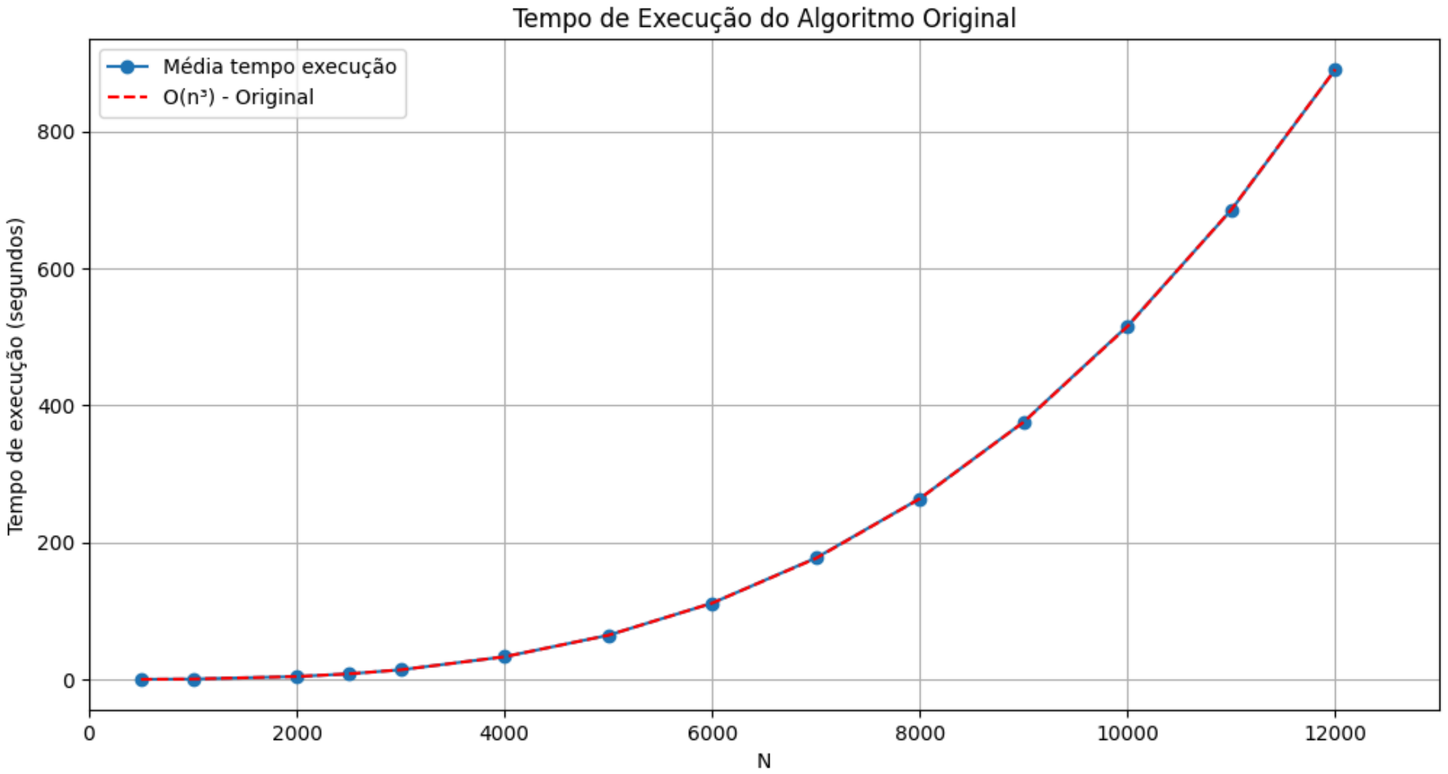

Gráfico – Otimizado

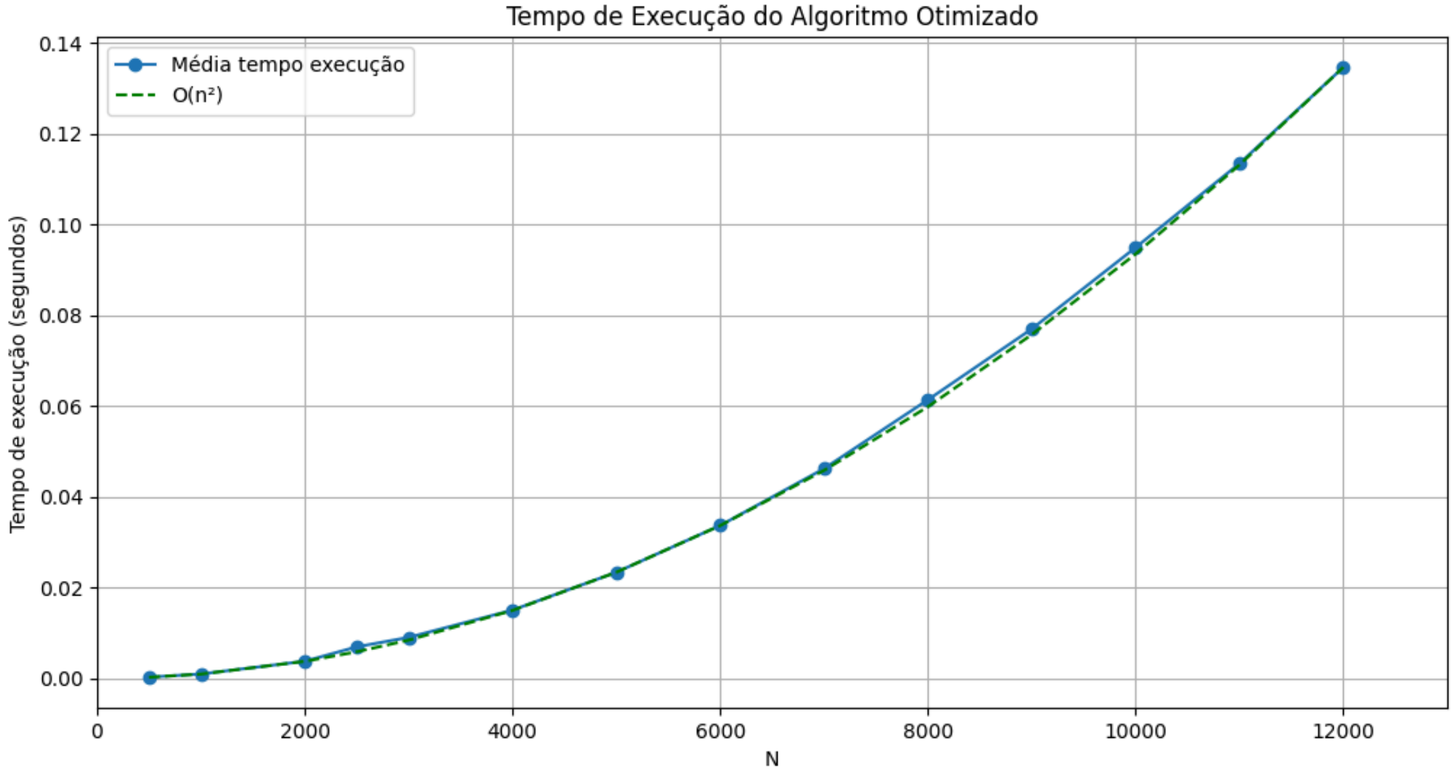

Gráfico – Comparação