Relatório de Análise de Algoritmos

Daniel Marques, Miguel Brito, Jefferson Oliveira, Vinicius Gonzaga June 29, 2017

1 Introdução

1.1 Quick Sort

Analise

O Quicksort é um algoritmo de ordenação por divisão e conquista. Seu funcionamento está relacionado diretamente ao particionamento do vetor, que consiste em selecionar um elemento pivô e separar todos os elementos que sejam menores que esse pivo de todos os elementos que sejam maiores, em duas partições separadas. Recursivamente realiza o particionamento, até que se atinja o caso base.

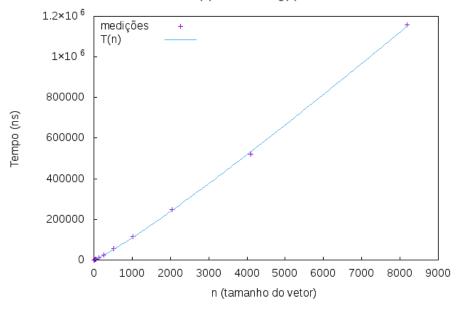
- $\bullet\,$ Tempo no melhor caso: $\mathcal{O}(nlgn)$
- Tempo no pior caso: $O(n^2)$
- Tempo no caso médio: O(nlgn)

2 Tempos

2.1 Vetores aleatórios

Figure 1: Quicksort vetor aleatório

Método QuickSort com vetor aleatorio T(n) = n*15.59Log(n) + 1692



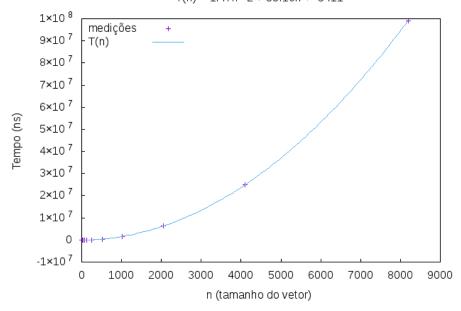
Para $n = 2^{32}$:

$$T(n) = 2^{32} * 15.59 * lg 2^{32} + 1692 = 2.1426733e + 12 ns = 2142 s = 35,7 min.$$

2.2 Vetores totalmente crescentes

Figure 2: Quicksort vetor crescente

Método QuickSort com vetor em ordem crescente T(n) = 1.47n**2 + 55.16n + -5411



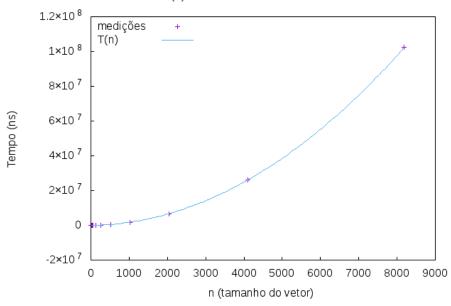
Para $n = 2^{32}$:

 $T(n) = 2^{64} * 1.47 + 55.16 * 2^{32} - 5411 = 2.7116714e + 19 \text{ ns} = 2.7116714e + 10 \text{ s} = 8,59 \text{ sécs.}$

2.3 Vetores totalmente decrescentes

Figure 3: Quicksort vetor decrescente

Método QuickSort com vetor em ordem decrescente T(n) = 1.496n**2 + 237n + -3.484e+04



Para $n = 2^{32}$:

 $T(n) = 2^{64} * 1.496 + 237 * 2^{32} - 3480 = 2.7596329e + 19 \text{ ns} = 2.7596329e + 10 = 8,75 \text{ sécs.}$

2.4 Vetores parcialmente crescentes

Figure 4: Quicksort vetor parcialmente crescente 60 %

Método QuickSort com vetor parcialmente em ordem crescente(60) T(n) = 0.002853n**2 + 139.6n + -6346

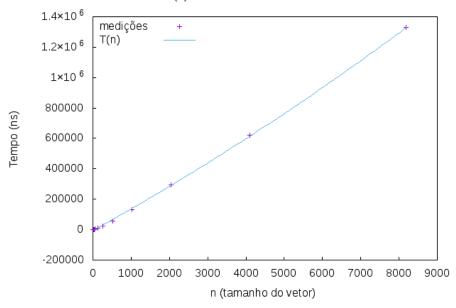


Figure 5: Quicksort vetor parcialmente crescente 70 %

Método QuickSort com vetor parcialmente em ordem crescente(70) T(n) = 0.0001523n**2 + 170.4n + -1.181e+04

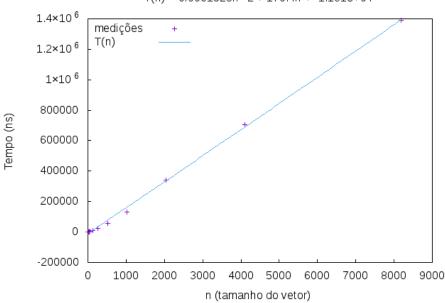


Figure 6: Quicksort vetor parcialmente crescente 80 %

Método QuickSort com vetor parcialmente em ordem crescente(80) T(n) = -0.002539n**2 + 229.7n + -1.844e+04

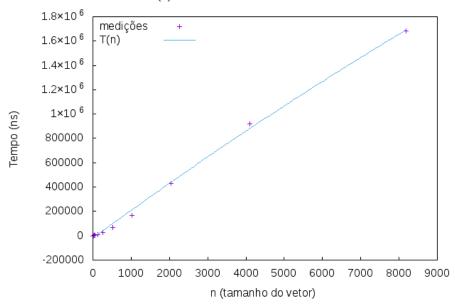
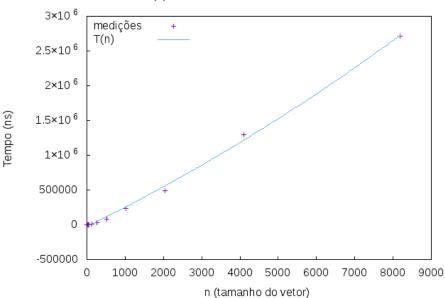


Figure 7: Quicksort vetor parcialmente crescente 90 %

Método QuickSort com vetor parcialmente em ordem crescente(90) T(n) = 0.008174n**2 + 268.9n + -2.346e+04



2.5 Vetores parcialmente decrescentes

Figure 8: Quicksort vetor parcialmente decrescente 60 %

Método QuickSort com vetor parcialmente em ordem decrescente(60) T(n) = 0.004195n**2 + 101.8n + -2567

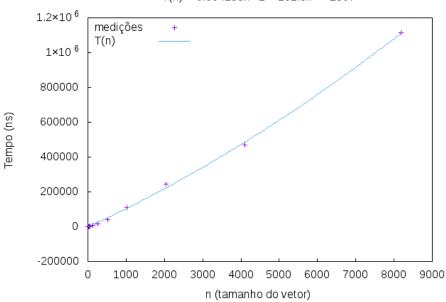


Figure 9: Quicksort vetor parcialmente decrescente 70 %

Método QuickSort com vetor parcialmente em ordem decrescente(70) T(n) = 0.002453n**2 + 128.1n + -1.089e+04

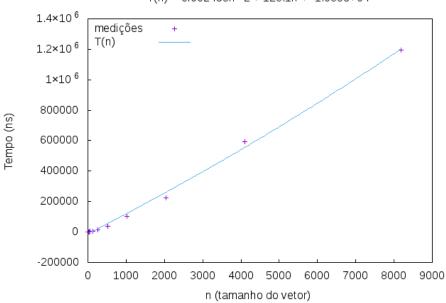


Figure 10: Quicksort vetor parcialmente decrescente 80 %

Método QuickSort com vetor parcialmente em ordem decrescente(80) T(n) = 0.008449n**2 + 101.8n + -3582

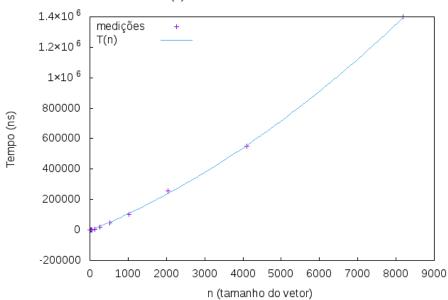


Figure 11: Heapsort vetor parcialmente decrescente 90 %

Método QuickSort com vetor parcialmente em ordem decrescente(90) T(n) = 0.002062n**2 + 173.2n + -1.336e+04

