PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Engenharia de Software
Fundamentos de Projeto e Análise de Algoritmos
Prof(a): João Caram Santos de Oliveira

Jully Ketely Alves da Silva

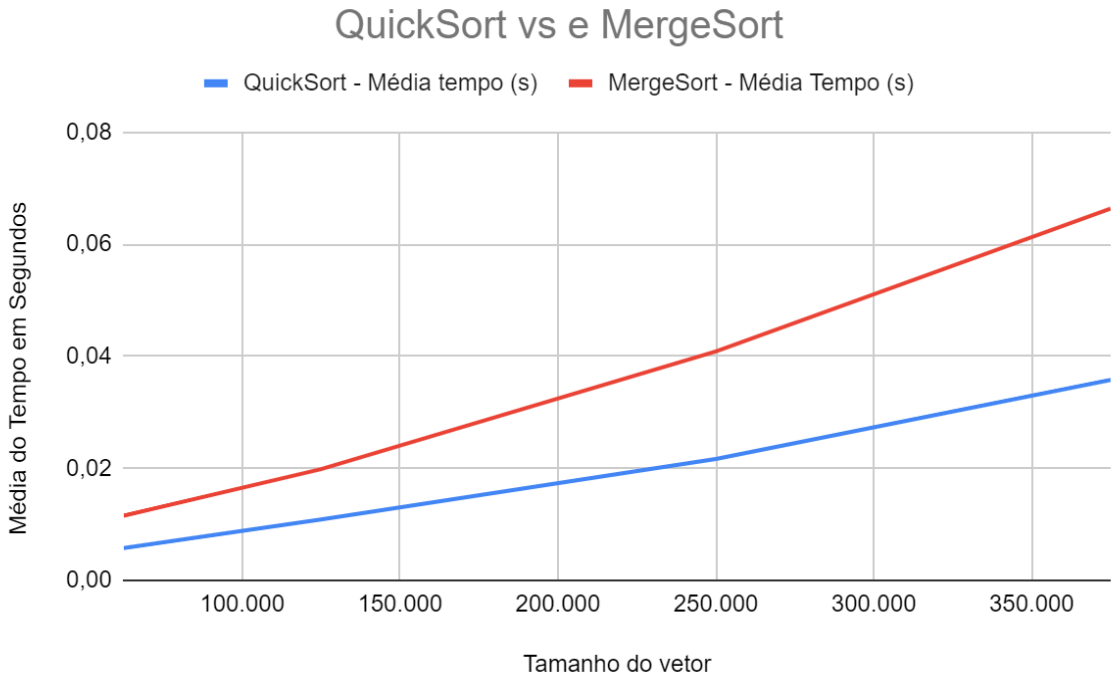
Especificação do equipamento

Processador: Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz
RAM instalada: 12,0 GB (utilizável: 11,8 GB)
Tipo de sistema: Sistema operacional de 64 bits, processador baseado em x64

Primeira parte

Resultados

Tamanho do vetor	***QuickSort* - Média tempo (s)**	***MergeSort* - Média Tempo (s)**
62.500	0,005759761	0,011540748
125.000	0,01087248	0,019860048
250.000	0,021663592	0,040859043
375.000	0,035770663	0,066375566



Análise

A escolha dos dois algoritmos (*MergeSort* e *QuickSort*) foi devido a que, mesmo os dois tendo a mesma ordem de complexidade, O(n log n), existem outros fatores que impactam no desempenho do algoritmo, uma delas é a forma de implementação.

Conforme a entrada cresce, o tempo de execução do *MergeSort* varia pouco, apresentando valores entre 0,005759761 e 0,035770663 segundos. Já o *QuickSort*, apresenta tempos de execução entre 0,011540748 e 0,066375566 com uma explosão de tempo de execução na entrada do tamanho de vetor 250.000.

Pode-se perceber que, mesmo analisando somente o primeiro vetor, a diferença na média do tempo de execução entre os algoritmos de complexidade **O(n log n)** já é evidente. Conforme o tamanho do vetor vai aumentando a média do tempo de execução dos algoritmos também. O MergeSort sofre um crescimento de **± 6x** o tempo inicial e o QuickSort por volta de **± 3,5x** o tempo inicial.

Segunda parte

Resultados

Tempo médio para vetor **aleatório**: 0,00097053 segundos
Tempo médio para vetor **ordenado**: 0,05829902 segundos

Análise

Para o vetor aleatório, o tempo médio de execução de aproximadamente 0,00097053 segundos é consistente com o desempenho esperado.

O pior caso do *QuickSort* é quando o particionamento criar uma partição com 1 elemento e outra com n-1, ou seja, quando o vetor já estiver ordenado, isso ocorre porque o pivô escolhido frequentemente se torna o elemento mínimo ou máximo, resultando em divisões desequilibradas. Tendo a complexidade $T(n) = \Theta(n^2)$. O tempo médio de execução de aproximadamente 0,05829902 segundos é consideravelmente maior do que para o vetor aleatório, o que é esperado