ICMC - Instituto de Ciências Matemáticas e da Computação
BCC - Bacharelado em Ciências de Computação
Aluno: Bernardo Marques Costa
Número USP: 11795551
Docente: Leonardo Pereira
Disciplina: Laboratório de Introdução a Ciência da Computação II

RELATÓRIO 4 - ORDENAÇÃO DE PALAVRAS

Introdução

Neste trabalho é apresentado a comparação entre os algoritmos de ordenação: **bubblesort**, **insertion sort** e **merge sort** no que se refere a ordenação de matrizes (vetores de palavras).
É fornecido ao programa 1 dos 4 arquivos: 14k.txt, 29k.txt, 58k.txt e 116k.txt. Cada um contém um número total de palavras próximo ao valor descrito no título.
Para a parte do relatório, é feito a análise da média de 10 execução de cada um dos algoritmos para cada um das quantidades possíveis de inputs, comparando o valor do tempo e a complexidade dos algoritmos estudados.

Bubblesort

Após executar o programa, obtemos um arquivo CSV que contém a seguinte tabela:

	Number of Inputs	Time
0	14	1.997850
1	29	9.563426
2	58	43.276890
3	116	165.308141

Como podemos ver, o número de inputs cresce, aproximadamente, de 2 em dois. Considerando a complexidade do algoritmo bubblesort de $O(n^2)$, podemos observar que o tempo também cresce proporcionalmente com 2^2

Insertion sort

Após executar o programa, obtemos a seguinte tabela de dados:

	Number of Inputs	Time
0	14	1.568970
1	29	7.416694
2	58	38.049748
3	116	148.138092

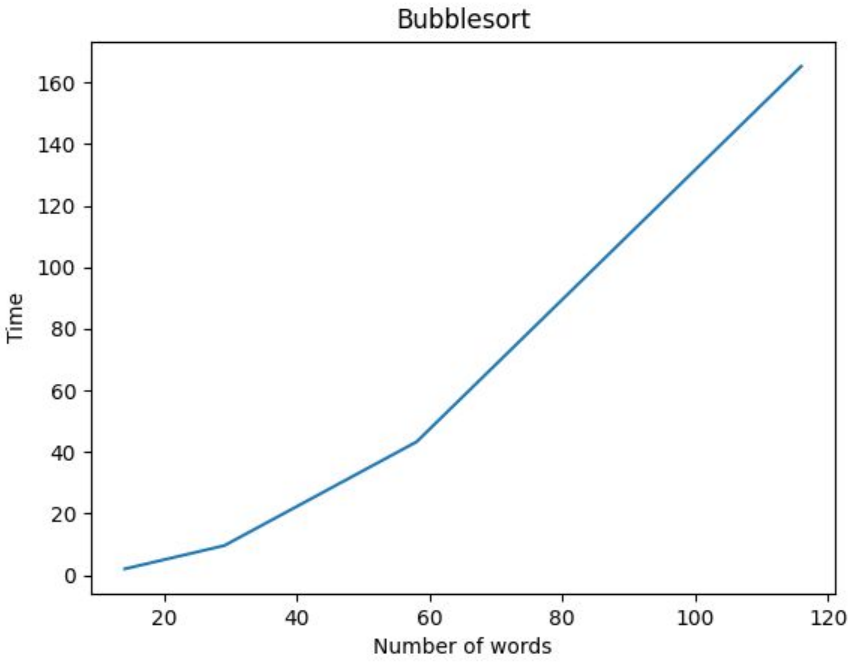
O algoritmo insertion sort possui uma complexidade e tempos de execução muito semelhantes ao bubblesort, tendo uma complexidade $O(n^2)$, sendo sutilmente mais otimizado

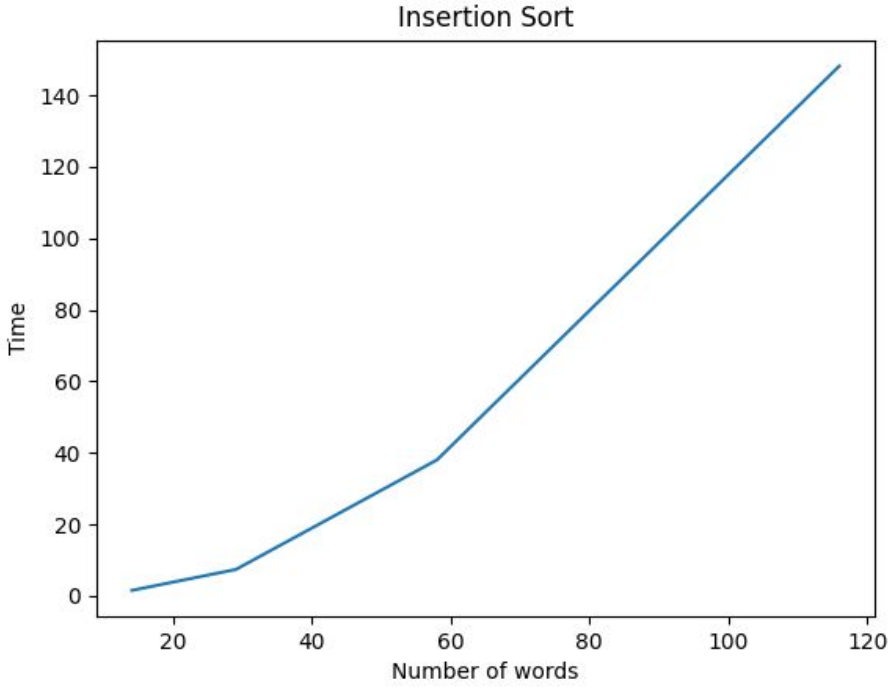
Mergesort

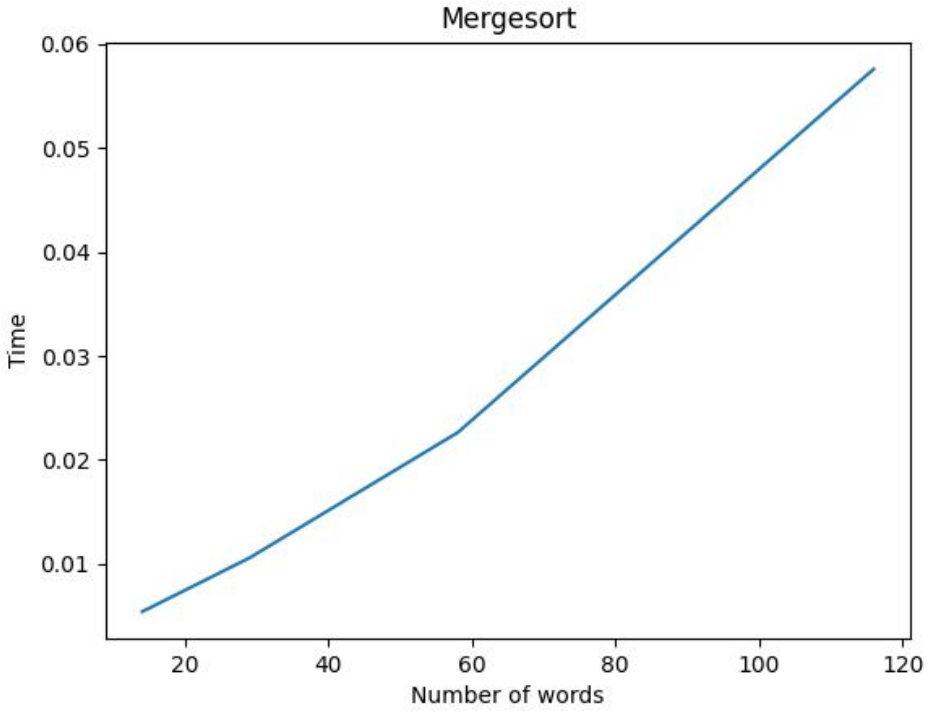
Ao contrário dos algoritmos anteriores, podemos observar que o tempo de execução é muito menor, e seu crescimento não é exponencial. De fato, como podemos observar pela complexidade do mergesort, temos $O(n \cdot \log n)$ como a função big O do algoritmo de merge sort

	Number of Inputs	Time
0	14	0.005418
1	29	0.010603
2	58	0.022643
3	116	0.057575

Gráficos







Conclusão

Como podemos observar a partir do tempo e da construção dos gráficos, temos uma leve otimização do bubble sort para o insertion sort e uma gigantesca diferença de tempo entre os dois primeiros algoritmos e o merge sort, em que o valor correspondente ao tempo de execução cai de vários segundos para menos de 1 segundo.
Assim, o método de divisão e conquista do mergesort aumenta consideravelmente a eficácia de ordenação, em comparação aos métodos sequenciais estudados agora.