Laboratório de Estrutura de Dados

Primeira versão do projeto da disciplina

Comparação entre os algoritmos de ordenação elementar

Álvaro Dias, Helânio Renê e Luiz José

1. Introdução

Esse relatório corresponde ao relato dos resultados obtidos no primeiro projeto da disciplina de Estrutura de Dados, cujos objetivos são estudar o desempenho dos algoritmos de ordenação utilizando dados de projetos voltados para Data Science.

Visão geral do projeto:

- **Introdução**: Introdução breve sobre o que foi trabalhado no projeto.
- **Descrição geral sobre o método utilizado**: Parte descritiva com base no objetivo geral, arquivos que foram utilizados, métodos do código, tabelas comparando os tempos de execução e informações acerca do ambiente de testes.
- **Resultado e análise**: Resultados e informações obtidas a partir das implementações que foram feitas no projeto.

2. Descrição geral sobre o método utilizado

Inicialmente, as planilhas com os dados foram baixadas e foram realizadas as classificações das senhas contidas no arquivo “passwords.csv” gerando um novo arquivo chamado “passwords_classifier.csv”, contendo uma nova coluna “class”, com as respectivas classificações de cada senha seguindo as regras especificadas no projeto.

Após a classificação, duas transformações foram realizadas: mudar a data para o formato “DD/MM/AAAA” e filtrar as senhas pelas categorias ”Boa” e “Muito Boa”. Após cada transformação, dois arquivos resultantes foram gerados e nomeados como “passwords_formated_data.csv” e “passwors_classifier.csv”.

Em seguida foram implementados e utilizados todos os algoritmos de ordenação estudados na disciplina (Selection Sort, Insertion Sort, Merge Sort, Quick Sort, Quick Sort, Counting, e Heap Sort), gerando um arquivo para cada tipo de ordenação dentro de três categorias:

Length - Ordenando as senhas pelo seu comprimento.

Month - Ordenando a tabela pelo Mês.

Date. - Ordenando pela Data inteira.

Com o objetivo de agilizar a execução dos algoritmos de ordenação e obtenção dos resultados, foi utilizado uma versão reduzida do arquivo “passwords_formated_data.csv”, com um total de 10.001 senhas.

Descrição da implementação da ferramenta (IDE) utilizada:

- A ferramenta utilizada no projeto foi o IntelliJ IDEA 2023.1.

Descrição geral do ambiente de testes:

- Processador - Intel(R) Core(™) i3-9100F CPU @ 3.60GHz
- Memória - RAM instalada 8,00GB
- Tipo de Sistema - Sistema operacional de 64 bits, processador baseado em x64
- Epecificações do Windows - Windows 10 Home versão 22H2

3. Resultados e Análise

Foi elaborada uma tabela comparando o tempo de execução dos algoritmos utilizados, para cada tipo de ordenação. Nas tabelas abaixo é possível ver os algoritmos e seus respectivos tempos de execução , tudo isso comparando com os tipos de caso(Médio caso, Melhor caso e Pior caso).

Length

-	Médio Caso	Melhor Caso	Pior Caso
SelectionSort	2,024s	1,191s	1,213s
InsertionSort	0,564s	0,135s	0,377s
MergeSort	0,286s	0,132s	0,076s
QuickSort	0,346s	0,650s	0,242s
QuickSort(Mediana de 3)	0,334s	0,160s	0,074s
CountingSort	0,265s	0,171s	0,074s
HeapSort	1,511s	0,954s	1,000s

Month

-	Médio Caso	Melhor Caso	Pior Caso
SelectionSort	87,051s	85,077s	84,114s
InsertionSort	36,485s	0,140s	66,024s
MergeSort	0,868s	0,307s	0,191s
QuickSort	7,053s	6,688s	6,581s
QuickSort(Mediana de 3)	7,365s	7,052s	6,250s
CountingSort	0,307s	0,127s	0,090s
HeapSort	0,403s	0,179s	0,117s

Date

-	Médio Caso	Melhor Caso	Pior Caso
SelectionSort	122,005s	113,068s	107,975s
InsertionSort	41,504s	0,125s	79,500s
MergeSort	0,752s	0,414s	0,187s
QuickSort	0,763s	72,920s	74,405s
QuickSort(Mediana de 3)	0,853s	0,463s	0,455s
CountingSort	0,345s	0,154s	0,097s
HeapSort	0,953s	0,897s	0,484s

- Com base nos tempos de execução analisados, foi possível calcular as médias obtidas por cada algoritmo nos três casos:

Length

Algoritmos	Médias obtidas
Selection	1,071s
Insertion	0,358s
Merge	0,164s
Quick	0,412s
Quick(3)	0,189s
Count	0,170s
Heap	1,155s

- Com base nas médias de execução do Length e dos respectivos algoritmos, os mais eficientes estão a seguir, onde o primeiro representa o mais eficiente em relação ao tempo de execução e o último o menos eficiente:
    1. Merge
    2. Count
    3. Quick(3)
    4. Insertion
    5. Quick
    6. Selection
    7. Heap

Month

Algoritmos	Médias obtidas
Selection	85,414s
Insertion	34,216s
Merge	0,455s
Quick	6,774s
Quick(3)	6,878s
Count	0,174s
Heap	0,233s

- Com base nas médias de execução do Month e dos respectivos algoritmos, os mais eficientes estão a seguir, onde o primeiro representa o mais eficiente em relação ao tempo de execução e o último o menos eficiente:
    1. Count
    2. Heap

3. Merge
4. Quick
5. Quick(3)
6. Insertion
7. Selection

Date

Algoritmos	Médias obtidas
Selection	114,349s
Insertion	40,376s
Merge	0,451s
Quick	49,362s
Quick(3)	0,590s
Count	0,198s
Heap	0,778s

- Com base nas médias de execução do Date e dos respectivos algoritmos, os mais eficientes estão a seguir, onde o primeiro representa o mais eficiente em relação ao tempo de execução e o último o menos eficiente:
    1. Count
    2. Merge
    3. Quick(3)
    4. Heap
    5. Insertion
    6. Quick
    7. Selection

Análise/Comentários acerca dos resultados obtidos:

Com base nos resultados obtidos pelo experimento, foi possível perceber que em cada caso um dos métodos se prevaleceu e teve um tempo de execução menor, significando um melhor desempenho, no caso do Length foi o Merge sort, no Month foi o Count sort e no Date foi o Count sort também. Com base nos resultados e tempos de execução é possível perceber que o Count sort apresenta os melhores resultados com base nos tempos de execução, logo após, temos o Merge sort, que também apresenta um ótimo resultado no geral dos 3 casos.