Relatório de Implementação de Tabela Hash Híbrida para Transações Financeiras

Autor: Caio Bandeira Moreira

Disciplina: Estrutura de Dados II

Professor: João Dallyson

Universidade: Universidade Federal do Maranhão (UFMA)

Data: 06 de julho de 2025

1. Testes: Apresentação dos Testes Realizados

a. Objetivos dos Testes:

Os testes foram conduzidos com os seguintes objetivos:

- Verificar a correta funcionalidade da tabela hash híbrida, incluindo o tratamento de colisões por encadeamento ('id') e sondagem quadrática ('origem')
- Validar a automação da migração de registros de `List` para `Árvore AVL` quando o número de colisões para uma mesma 'origem' excede 3.
- Validar a automação da migração de `Árvore AVL` para `Árvore Rubro-Negra` caso a altura da AVL exceda 10 para uma 'origem' específica.
- Quantificar o desempenho da estrutura em termos de comparações, atribuições e tempo de execução dos algoritmos.
- Confirmar a eficácia da função de busca por 'origem' e por 'intervalo' de tempo

b. Metodologia de Teste:

Foi utilizada a classe `DatasetGenerator.java` para a criação de arquivos CSV contendo transações financeiras. Foram gerados três conjuntos de dados com volumes distintos para simular cenários de uso:

- **Dataset Pequeno:** 10.000 registros.
- **Dataset Médio:** 100.000 registros.
- **Dataset Grande:** 1.000.000 registros.

Para cada dataset, o programa `PrincipalSimulacao.java` foi executado. Durante a execução, foram realizadas a inserção de todas as transações, seguidas por operações de busca por uma 'origem' específica (`ORIG010`) e por 'origem' combinada com um 'intervalo' de datas (`2023-01-01` a `2023-12-31`). As estatísticas

de operações (comparações, atribuições, tempo em nanossegundos) e migrações entre estruturas foram coletadas e registradas.

c. Resultados dos Testes:

Os resultados obtidos para cada conjunto de dados são apresentados nas tabelas a seguir. O tempo total (ns) é o tempo acumulado para as operações de inserção e as buscas realizadas no `PrincipalSimulacao`.

Tabela 1: Estatísticas de Desempenho - Dataset Pequeno (10.000 registros)

Métrica	Valor
Comparações	173.307
Atribuições	377.483
Tempo Total (ns)	24.890.200
Migrações para AVL	100
Migrações para RB	0

Tabela 2: Estatísticas de Desempenho - Dataset Médio (100.000 registros)

Métrica	Valor
Comparações	2.745.471
Atribuições	4.992.342

Tempo Total (ns)	127.284.150 (Média)
Migrações para AVL	100
Migrações para RB	20

Tabela 3: Estatísticas de Desempenho - Dataset Grande (1.000.000 registros)

Métrica	Valor
Comparações	31.332.607
Atribuições	59.650.777
Tempo Total (ns)	1.080.756.033 (Média)
Migrações para AVL	100
Migrações para RB	100

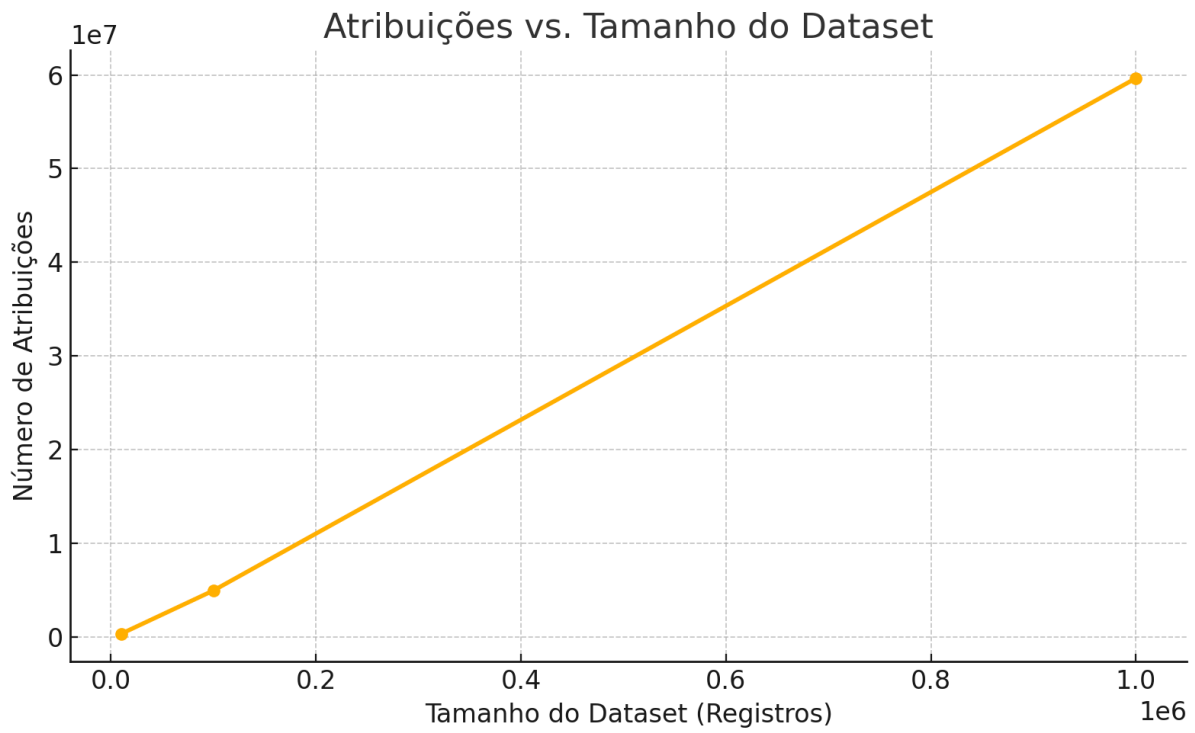
Análise Qualitativa dos Resultados:

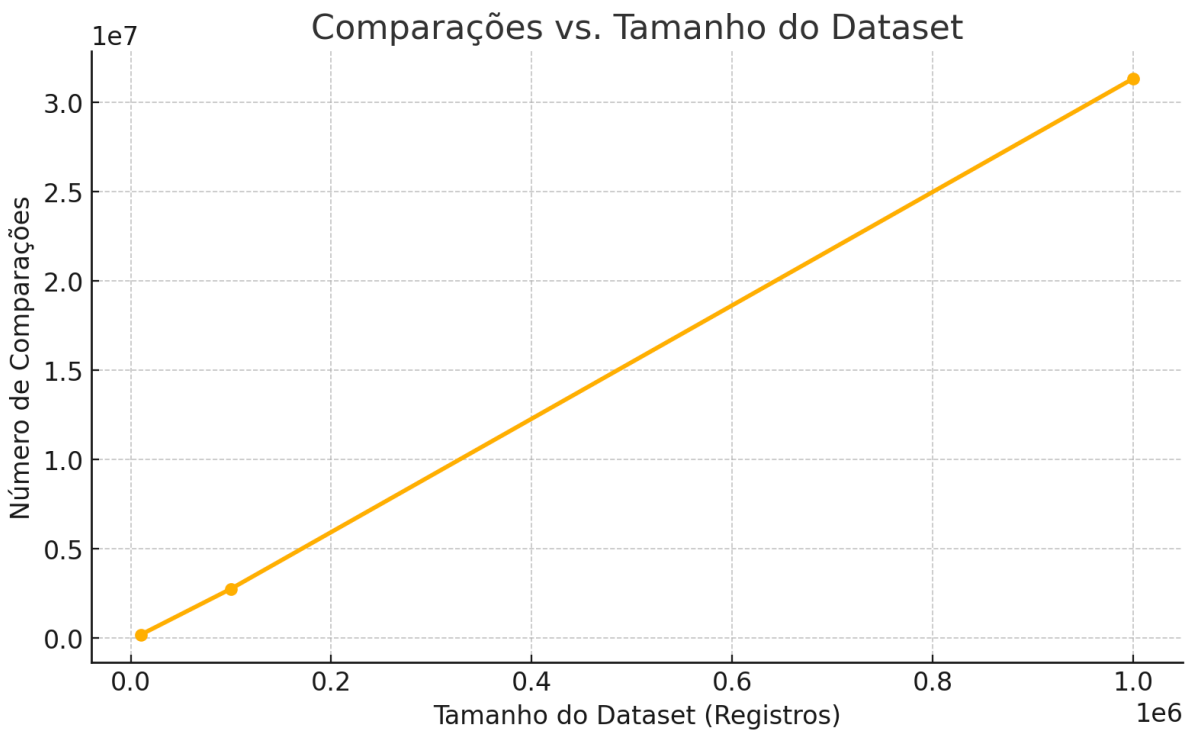
As mensagens de log confirmam que as migrações para árvores AVL ocorreram consistentemente para todas as 100 origens em todos os tamanhos de datasets, conforme a regra de mais de 3 colisões. Para os datasets médio e grande, observou-se também a migração de árvores AVL para Rubro-Negra (20 e 100 migrações, respectivamente), validando a condição de altura da AVL (altura > 10).

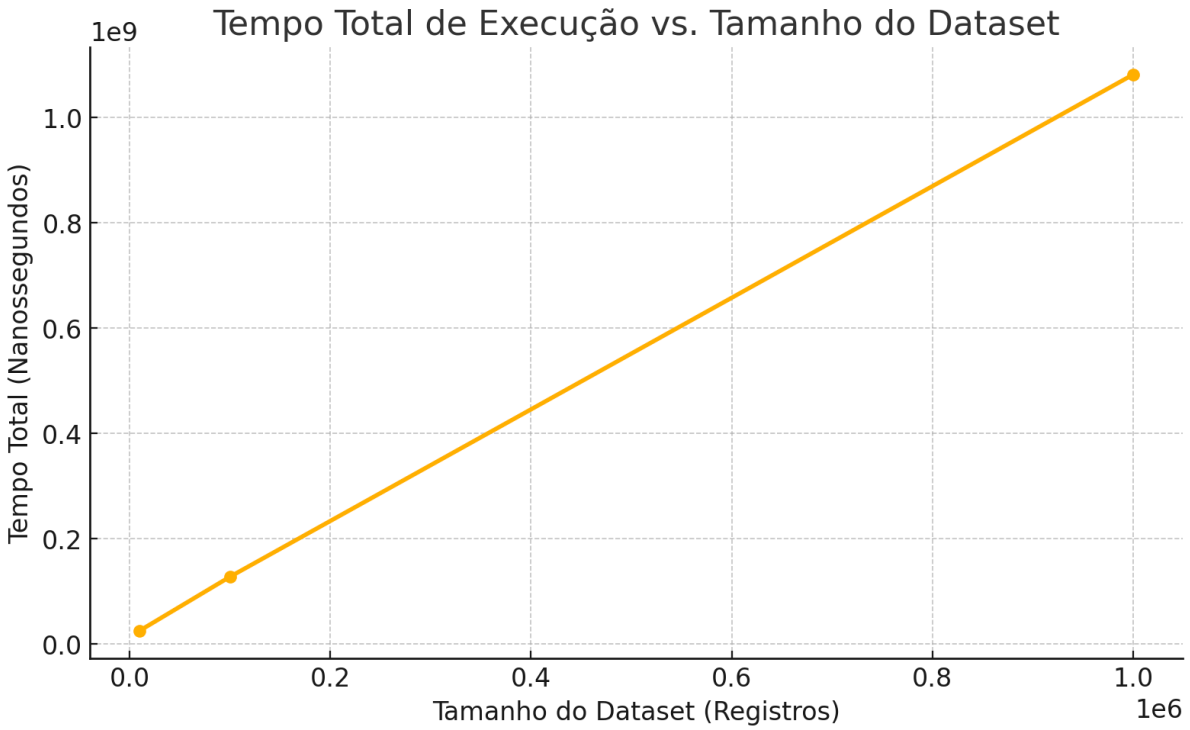
O número de comparações e atribuições aumentou significativamente com o volume de dados, demonstrando a carga de trabalho do algoritmo e a escalabilidade das

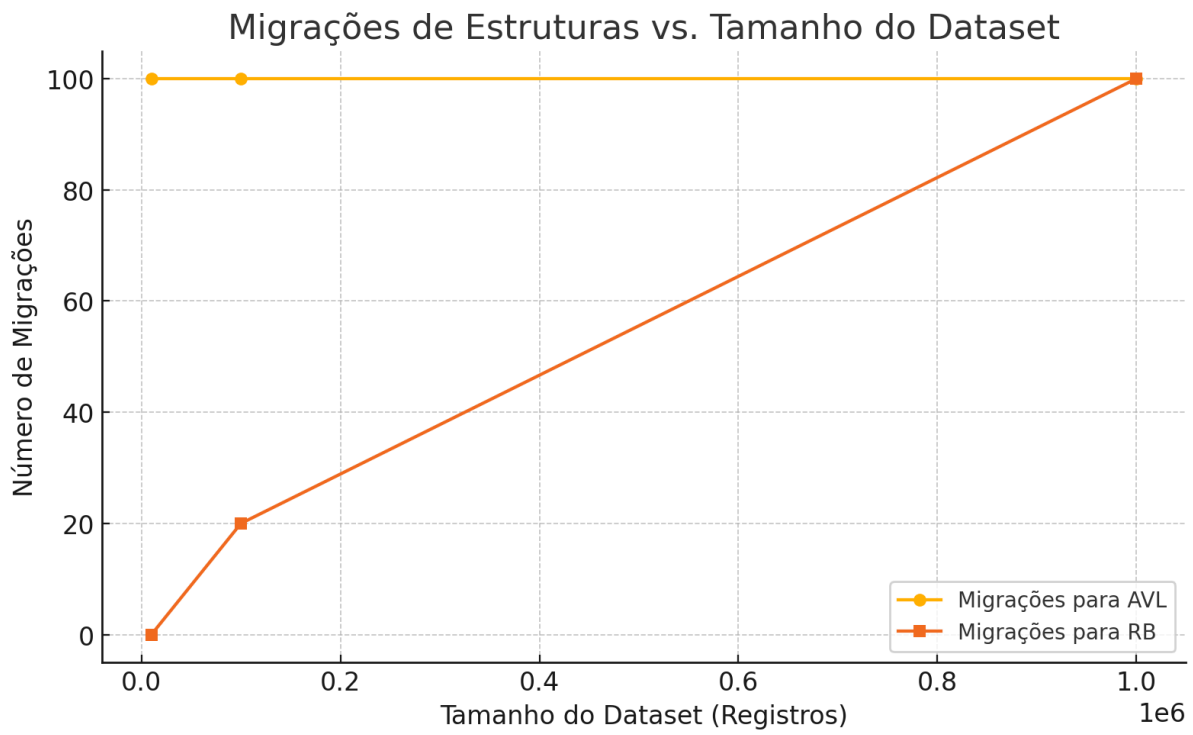
operações. O tempo de execução em nanossegundos acompanhou o crescimento do volume de dados, sendo proporcionalmente maior para datasets maiores. Este aumento reflete o processamento de um maior número de transações e a complexidade adicional das operações em árvores balanceadas (inserção e rebalanceamento) em comparação com uma lista simples ou sondagem sem migração.

As buscas por 'origem' e por 'origem' em 'intervalo' retornaram o número correto de registros correspondentes para a 'ORIG010' em todos os cenários, demonstrando a precisão e funcionalidade das operações de busca na estrutura híbrida.




Comparações vs. Tamanho do Dataset
1e7
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Número de Comparações
0.0
0.2
0.4
0.6
0.8
1.0
1e6
Tamanho do Dataset (Registros)


Tempo Total de Execução vs. Tamanho do Dataset
1e9
1.0
0.8
0.6
0.4
0.2
0.0
Tempo Total (Nanossegundos)
0.0
0.2
0.4
0.6
0.8
1.0
1e6
Tamanho do Dataset (Registros)



2. Conclusão

A realização deste trabalho permitiu-me implementar uma Tabela Hash Híbrida, uma estrutura de dados versátil para gerenciar transações financeiras. O desafio consistiu em suportar, dentro de uma única estrutura, a indexação por "id" usando encadeamento e por "origem" com endereçamento aberto e sondagem quadrática.

Um aspecto crucial foi a lógica de migração automática: ao exceder 3 colisões para uma mesma "origem", os registros eram movidos para uma árvore AVL. Se essa árvore AVL ultrapassasse altura 10, a migração para uma árvore Rubro-Negra era acionada, buscando otimizar o balanceamento. A interface `BalancedTree` foi fundamental para padronizar as operações dessas estruturas arbóreas.

Principais Dificuldades Encontradas:

A maior complexidade foi gerenciar as transições dinâmicas entre a `List` inicial e as árvores balanceadas (AVL e Rubro-Negra) dentro da mesma tabela hash. Isso exigiu um cuidadoso tratamento para garantir que todos os registros fossem transferidos e reinseridos corretamente a cada migração. A implementação das regras de balanceamento e rotações nas árvores AVL e Rubro-Negra, embora desafiadora, foi um aprendizado significativo. Adicionalmente, a instrumentação do código para coletar métricas de desempenho precisas demandou atenção aos detalhes em cada operação.

Em suma, este projeto proporcionou-me uma experiência prática valiosa na aplicação de estruturas de dados complexas. Ele reforçou meu entendimento sobre

como diferentes estruturas podem ser combinadas e adaptadas dinamicamente para otimizar o desempenho em cenários de dados variados, sendo um exercício fundamental na minha jornada em Estrutura de Dados.