

Estudo Comparativo do Desempenho Entre Os Algoritmos de Árvores AVL e Árvores Red Black

Lucas Gabriel Eloi Viera¹, Manoel Lucas Lopes Reis²

Departamento de Computação – Campus São Luís-Monte Castelo Av. Getúlio Vargas, nº 04,
Monte Castelo – CEP 65030-005 – São Luís – MA – Brazil

²Department of Computing – Federal Institute of Education, Science and Technology of
Maranhão

³Curso de Sistemas de Informação
Instituto Federal de Educação, Ciência e Tecnologia do Maranhão(IFMA) – São Luís, MA –
Brazil

eloi.lucas@acad.ifma.edu.br, manael.reis@acad.ifma.edu.br.

Abstract. *This paper presents a theoretical and experimental analysis comparing Red-Black Trees and AVL Trees. The objective is to empirically verify the $O(\log n)$ time complexity and the effectiveness of color-based balancing versus strict height balancing. Practical experiments using Java with inputs of 100,000 elements measured execution time, rotation volume, and final tree height. The results confirm that Red-Black Trees provide lower restructuring overhead and faster execution in write-intensive scenarios, validating their efficiency for dynamic data management.*

Resumo. *Este artigo apresenta uma análise teórica e experimental do desempenho das Árvores Red-Black em comparação com as Árvores AVL. O objetivo primordial é verificar empiricamente a complexidade de tempo $O(\log n)$ e a eficácia do mecanismo de balanceamento baseado em bits de cor e rotações. Foram realizados experimentos práticos utilizando a linguagem Java com entradas de 100.000 elementos, mensurando variáveis como o tempo médio de execução, o número médio de rotações e a altura final das estruturas em cenários de inserção, busca e remoção. Os resultados confirmam que a Árvore Red-Black, embora possua um balanceamento menos rígido que a AVL, proporciona um menor overhead de reestruturação e tempos de execução inferiores em operações de escrita intensiva, validando sua eficiência assintótica e estrutural para o gerenciamento de dados dinâmicos.*

1. Introdução:

Algoritmos de computador representam um conjunto de etapas para executar uma tarefa, descritas com a precisão necessária para que uma máquina possa processá-las. O propósito primordial de um algoritmo é não apenas produzir uma solução correta para um problema de computação, mas também utilizar os recursos computacionais de forma eficiente, com foco principal no tempo de execução. Para avaliar essa eficiência, a análise de algoritmos recai sobre a taxa de crescimento do tempo de execução em função do tamanho da entrada (n). Essa análise assintótica permite ignorar coeficientes constantes e termos de ordem inferior, destacando a ordem de crescimento do tempo de execução por meio de notações como O , Ω e Θ .

Dentro das operações fundamentais em ciência da computação, a busca de informações é uma tarefa primordial que visa determinar a localização de um valor específico

em uma estrutura de dados. Enquanto estruturas lineares elementares podem se tornar ineficientes à medida que n cresce, as Árvores Red-Black surgem como uma evolução das árvores binárias de busca comuns. Elas utilizam um bit extra de armazenamento em cada nó para indicar sua cor (vermelho ou preto), impondo restrições que garantem o balanceamento da estrutura. Esse mecanismo assegura que a altura da árvore permaneça $O(\log n)$, permitindo que operações de busca, inserção e remoção sejam executadas de forma eficiente no pior caso.

O presente trabalho visa analisar o funcionamento e a complexidade da Árvore Red-Black, realizando a verificação empírica da complexidade teórica e das propriedades de integridade da estrutura. O estudo se propõe a implementar e comparar o desempenho deste algoritmo com a Árvore AVL, buscando medir, comparar e interpretar seu comportamento em cenários de escrita e leitura intensiva. O objetivo central é confrontar os resultados empíricos de tempo de execução e volume de rotações com a análise teórica, que prediz que a Árvore Red-Black oferece um rebalanceamento mais eficiente e veloz para inserções em relação à rigidez do balanceamento da AVL.

2. Análise Teórica do Funcionamento e Complexidade

A análise da eficiência de algoritmos é um aspecto central da Ciência da Computação, concentrando-se na taxa de crescimento do tempo de execução em função do tamanho da entrada (n). Diferentemente das Árvores Binárias de Busca (ABB) tradicionais, que podem se degenerar em estruturas lineares com custo temporal de $O(n)$ no pior caso, as Árvores Red-Black empregam um mecanismo de balanceamento baseado em propriedades de coloração e operações de rotação.

Cada nó da árvore armazena um bit adicional que representa sua cor (vermelha ou preta), e o cumprimento rigoroso de um conjunto de invariantes estruturais garante que a altura da árvore permaneça limitada por uma função logarítmica do número de elementos armazenados.

2.1 Propriedades Invariantes e Altura Preta

A estrutura de uma Árvore Red-Black é definida pelas seguintes cinco propriedades invariantes:

- Todo nó é colorido de vermelho ou preto;
- A raiz da árvore é sempre preta;
- Todas as folhas (nós NIL) são pretas;
- Se um nó é vermelho, então ambos os seus filhos são pretos;
- Para cada nó, todos os caminhos simples desse nó até suas folhas descendentes contêm o mesmo número de nós pretos.

A noção de altura preta, denotada por $bh(x)$, é definida como o número de nós pretos em qualquer caminho de um nó x até uma folha, excluindo o próprio nó x . A manutenção dessa altura uniforme em todos os caminhos garante que nenhum caminho da raiz até uma folha seja mais do que o dobro do comprimento de outro.

Como consequência direta dessas propriedades, a altura total h da árvore é limitada por $h \leq 2 \log_2(n + 1)$. Dessa forma, as operações fundamentais de busca, inserção e remoção possuem complexidade assintótica de $O(\log n)$ no pior caso.

2.4 Comparação de Complexidades

Em termos de limites assintóticos, a Árvore Red-Black oferece garantias superiores às estruturas lineares e ABB comuns não balanceadas:

Tabela 1. Comparativo entre Caso Médio e Pior Caso no Algoritmo de Árvores Red Black

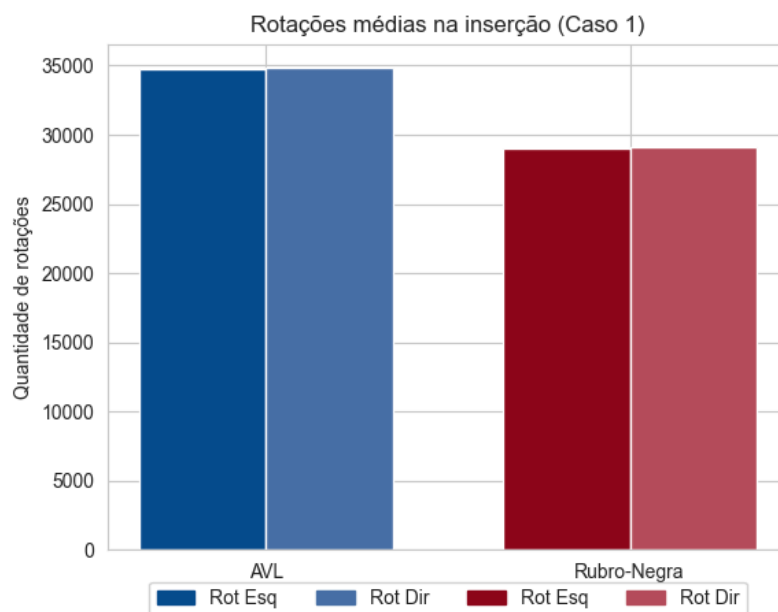
Operação	Caso Médio	Pior Caso
Busca	$\Theta(\log n)$	$\Theta(\log n)$
Inserção	$\Theta(\log n)$	$\Theta(\log n)$
Remoção	$\Theta(\log n)$	$\Theta(\log n)$

3. Resultados e Discussão

Os experimentos foram realizados em um computador com sistema operacional PopOS, equipado com um processador i5-10210u e 8GB de memória RAM. Para garantir consistência e confiabilidade nas medições, cada algoritmo foi executado 5 vezes, tendo cada execução uma entrada fixa de 100.000 elementos. Em cada repetição, foi utilizada uma *seed* diferente no gerador de números aleatórios, assegurando que as 5 entradas analisadas em cada fossem distintas entre si.

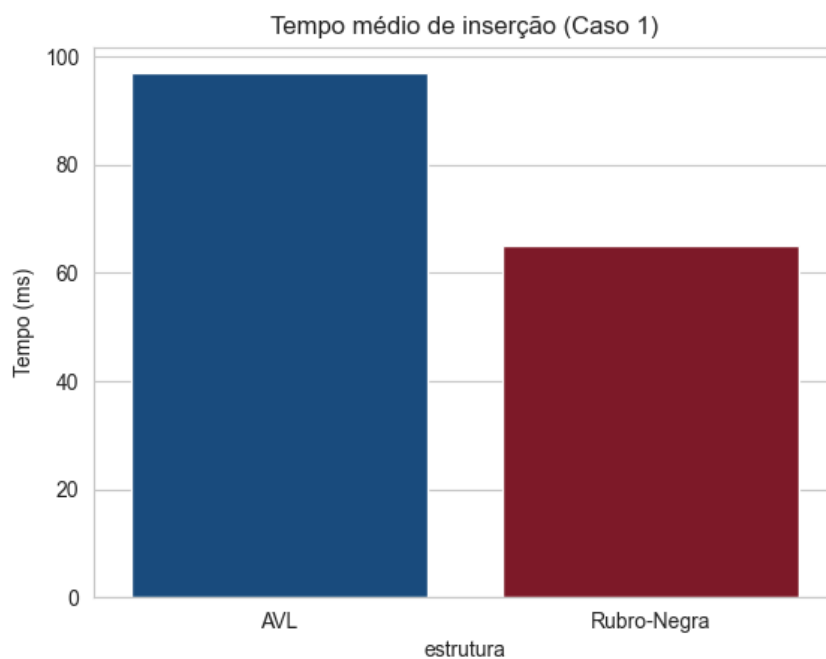
As implementações foram executadas utilizando a seguinte versão da ferramenta: JDK 25 para o programa em Java.

Figura 1. Número de Rotações Médias Na Fase de Inserção



Fonte: O autor

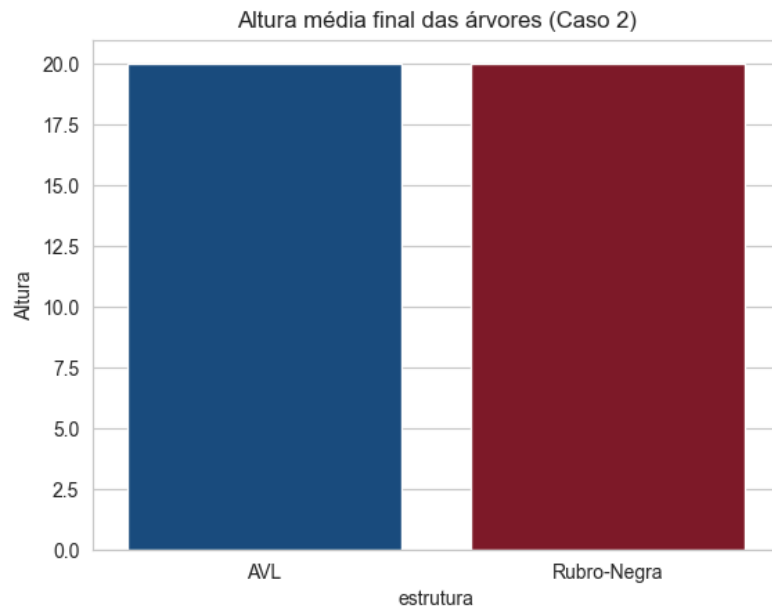
Figura 2. Número de Rotações Médias Na Fase de Inserção



Fonte: O autor

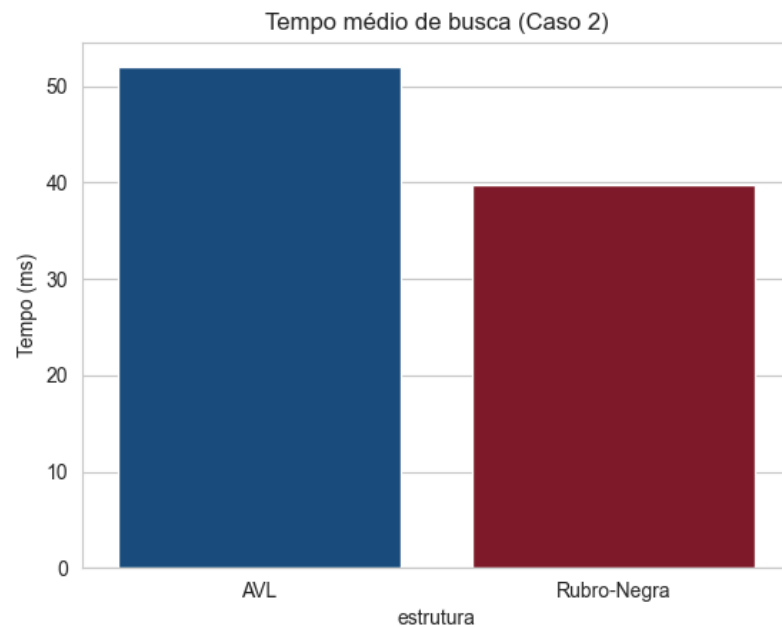
De acordo com a execução prática, a árvore rubro-negra demandou de **um tempo e um número de rotações menor** quando comparada com o desempenho de uma árvore AVL. Quanto à altura de ambas as árvores, percebe-se que as duas obtiveram o mesmo resultado.

Figura 3. Altura Média Final das Árvores na Fase de Busca



Fonte: O autor

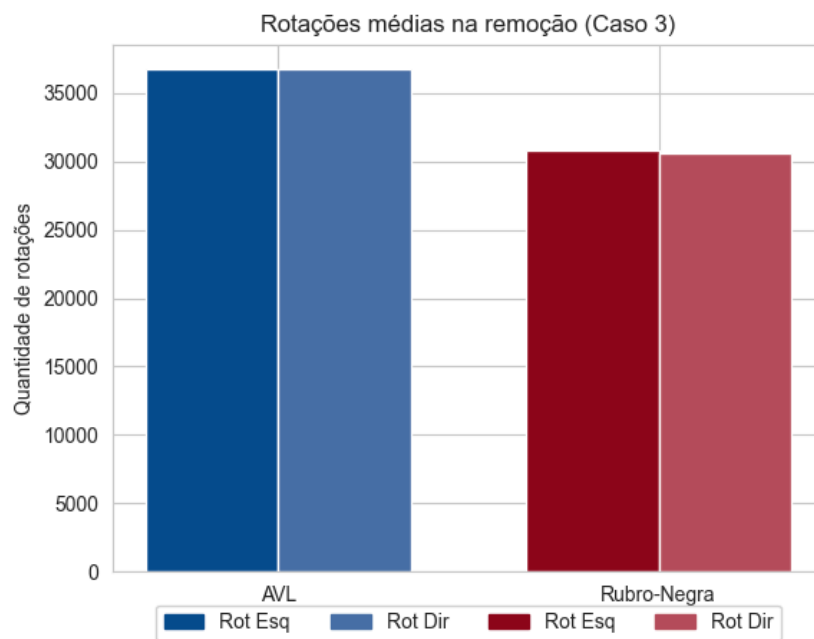
Figura 4. Tempo Médio na Fase de Busca



Fonte: O autor

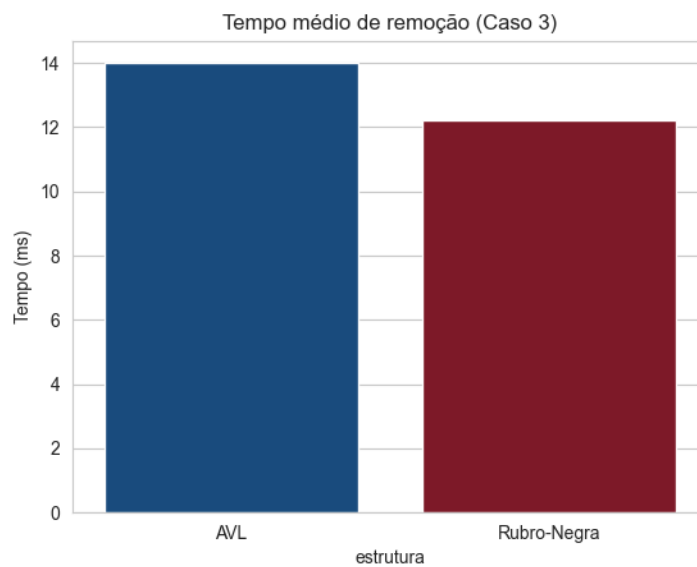
Na fase de busca, um comportamento semelhante foi encontrado. O tempo médio encontrado na Árvore Rubro-negra foi menor. Na fase de remoção, o comportamento continuou constante, como constatado a partir das figuras 5 e 6.

Figura 5. Número de Rotações Médias na Fase de Remoção



Fonte: O autor

Figura 6. Tempo Médio de Remoção



Fonte: O autor

4. Considerações finais

Os resultados práticos obtidos confirmam a análise teórica de que a Árvore Red-Black executa suas operações com um overhead de rotações significativamente menor quando comparada à Árvore AVL. Essa diferença decorre principalmente do mecanismo de balanceamento menos rígido da Red-Black, que prioriza a redução do número de reestruturações estruturais ao longo das operações de atualização.

Conforme evidenciado nos gráficos apresentados no relatório, especialmente na Figura 1, observa-se que, para uma entrada de 100.000 elementos, a Árvore Red-Black realizou um volume médio de rotações visivelmente inferior, situando-se aproximadamente entre 25.000 e 30.000 rotações, enquanto a Árvore AVL ultrapassou a marca de 30.000 rotações no mesmo cenário. Esse comportamento impacta diretamente o custo computacional da inserção.

Como consequência direta do menor número de rotações, o tempo médio de inserção da Árvore Red-Black apresentou desempenho superior, alcançando aproximadamente 65 ms, ao passo que a Árvore AVL demandou cerca de 100 ms para executar a mesma operação. Tal resultado reforça a eficiência do balanceamento da Red-Black em cenários com elevada taxa de escrita.

Na fase de remoção, o comportamento observado manteve-se consistente. De acordo com os dados apresentados nas Figuras 5 e 6, a Árvore Red-Black continuou a exigir um número médio de rotações inferior ao da AVL, refletindo-se também em um menor tempo de execução. Isso indica que, mesmo em operações estruturalmente mais complexas, a Red-Black preserva sua vantagem em termos de custo de reestruturação.

Em relação às operações de busca, embora a análise teórica sugira que a Árvore AVL possa apresentar desempenho ligeiramente superior devido ao seu balanceamento mais estrito, os testes práticos realizados demonstraram o contrário para o cenário avaliado. Conforme ilustrado na Figura 4, a Árvore Red-Black obteve um tempo médio de busca de aproximadamente 40 ms, enquanto a AVL registrou cerca de 52 ms, indicando que, na prática, o menor overhead de manutenção estrutural também pode beneficiar operações de leitura.

Referências Bibliográficas

CORMEN, Thomas H. (2014), Desmistificando algoritmos. Tradução: Arlete Simille Marques. 1. ed. Rio de Janeiro: Elsevier

CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford. (2022), Introduction to Algorithms. Fourth Edition. Cambridge, Massachusetts; London, England: The MIT Press