**Análise Comparativa de Árvores Binárias de Busca: AVL vs. Rubro Negra**

**Nathalia Gomes Magro1;Rodrigo da Rosa²**

1 Programa de Pós-Graduação em Ciência da Computação (PPGComp)   
Universidade Estadual do Oeste do Paraná (Unioeste) – Cascavel, PR – Brasil

nathalia.magro@unioeste.br;rodrigo.rosa7@unioeste.br;

***Abstract.*** *This article conducts a comparative analysis of two approaches to implementing binary search trees: AVL Trees and Red-Black (RB) Trees. The objective is to examine the practical performance of these data structures during construction and query processes. The analysis focuses on the number of key comparisons and the time spent at each stage. Through experiments, the study highlights the efficiency and suitability of each method in different scenarios. Implementations were done in Python, and the input data was standardized to ensure a fair comparison. The results reveal the circumstances in which each type of tree performs better, providing important information for choosing the most appropriate data structure according to specific needs..*

*Keywords: Binary Search Trees, AVL Trees, Red-Black Trees, Data Structures, Construction Algorithms, Query Algorithms, Performance Comparison, Python, Algorithm Efficiency.*

***Resumo.*** *Este artigo realiza uma análise comparativa de duas abordagens para a implementação de árvores binárias de busca: Árvores AVL e Árvores Rubro Negra (RB). O objetivo é examinar o desempenho prático dessas estruturas de dados durante os processos de construção e consulta. A análise enfoca o número de comparações de chaves e o tempo gasto em cada etapa. Por meio de experimentos, o estudo destaca a eficiência e a adequação de cada método em diferentes cenários. As implementações foram feitas em Python e os dados de entrada foram uniformizados para garantir uma comparação justa. Os resultados revelam as circunstâncias nas quais cada tipo de árvore apresenta melhor desempenho, fornecendo informações importantes para a escolha da estrutura de dados mais adequada conforme as necessidades específicas.*

*Palavras-chave: Árvores Binárias de Busca, Árvores AVL, Árvores Rubro Negra, Estruturas de Dados, Algoritmos de Construção, Algoritmos de Consulta, Comparação de Desempenho, Python, Eficiência de Algoritmos.*

1. **Introdução**

Árvores binárias de busca são estruturas de dados hierárquicos que facilitam operações eficientes de busca, inserção e exclusão. Duas variantes populares dessas árvores são as Árvores AVL e Red-Black. As Árvores AVL, introduzidas por Adelson-Velsky e Landis em 1962, são auto-balanceadas, garantindo que a diferença de altura entre as subárvores de qualquer nó seja no máximo 1 . As Árvores Red-Black, introduzidas por Rudolf Bayer em 1972, são uma variante das árvores B, garantindo que a altura da árvore seja no máximo 2 log(n+1).

1. **Objetivo**

Este relatório técnico tem como objetivo aplicar e corroborar conceitos adquiridos sobre algoritmos de manipulação de dados de forma hierárquica. Para tanto, foram implementadas duas soluções para árvores binárias de busca. A primeira solução implementa uma Árvore AVL e, em seguida, para fins de comparação, a segunda solução implementa a estratégia de Árvore Rubro Negra (RB).

1. **Materiais e Métodos**

O ambiente computacional utilizado para a execução dos testes e experimentos consiste em um *desktop* com a seguinte configuração: um processador Intel(R) Core(TM) i5-8400 CPU operando a 2.80GHz, 8,00 GB de RAM instalada, e sistema operacional Windows 10 Pro de 64 bits. Esta configuração do dispositivo garante a precisão dos testes realizados e a replicabilidade dos resultados. Além do *hardware* local, os testes foram executados utilizando o *Google Colab*, disponível em *Google Colab,* o que permitiu a utilização de recursos computacionais adicionais baseados em nuvem para a execução do código.

Para a implementação dos algoritmos de busca e ordenação, foi utilizada a linguagem de programação Python na versão 3.8, escolhida pela sua versatilidade e amplo suporte para manipulação de dados e análise estatística. Bibliotecas padrão do Python, como random para a geração de dados aleatórios e time para monitoramento do desempenho temporal dos algoritmos, foram empregadas. Além disso, a biblioteca *statistics* foi usada para calcular médias e desvios padrões dos resultados dos testes.

* 1. **Conjunto de Dados**

Os dados de entrada foram disponibilizados pelo Professor André em seu *Drive Online*, pasta 04. Foram fornecidos em arquivos de texto, contendo listas de chaves a serem inseridas e buscadas nas árvores. Estes arquivos foram divididos em conjuntos para construção e consulta, sendo os vetores de tamanho 50, 100, 200, 300, 500, 750, 1000, 1500, 2000, 3000, 5000, 7500, 10000, 12500, 15000, 20000, 25000, 30000, 40000, 50000, 75000, 100000, 125000, 150000, 175000, 200000, 225000, 250000 elementos.

* 1. **Procedimentos de Implementação**
* Árvore AVL: Implementamos funções de inserção, busca, rotações (simples e duplas) e balanceamento, contabilizando o número de comparações e o tempo gasto em cada operação.
* Árvore Rubro-Negra: Implementamos funções de inserção, busca, rotações, recoloração e balanceamento, também contabilizando o número de comparações e o tempo gasto.
  1. **Critérios de Avaliação**

A comparação entre as duas estruturas foi realizada em duas etapas:

* Construção: Medimos o número de comparações e o tempo gasto para inserir todos os elementos na árvore.
* Consulta: Medimos o número de comparações e o tempo gasto para buscar todos os elementos na árvore já construída.

1. **Fundamentação Teórica**

As estruturas de dados hierárquicas desempenham um papel importante em diversos algoritmos e aplicações de ciência da computação, proporcionando uma maneira eficiente de organizar e manipular grandes volumes de dados (Knuth, 1998). Entre essas estruturas, as árvores binárias de busca destacam-se pela sua capacidade de manter os dados ordenados e permitir operações rápidas de inserção, busca e remoção (Cormen et al., 2009). No entanto, à medida que novos elementos são inseridos, uma árvore binária de busca simples pode se tornar desbalanceada, resultando em uma degradação do desempenho (Tarjan, 1983).

Para mitigar esse problema, foram desenvolvidas diversas variações de árvores binárias de busca balanceadas. Duas das mais importantes e amplamente utilizadas são a Árvore AVL e a Árvore Rubro-Negra (Adelson-Velsky & Landis, 1962; Guibas & Sedgewick, 1978). Ambas as estruturas têm como objetivo principal manter a árvore balanceada durante as operações de inserção e remoção, garantindo assim tempos de execução eficientes para as operações fundamentais.

A Árvore AVL, introduzida por Adelson-Velsky e Landis em 1962, é uma das primeiras árvores binárias de busca auto-balanceadas (Adelson-Velsky & Landis, 1962). Ela mantém um balanceamento estrito ao garantir que a diferença de altura entre as subárvores esquerda e direita de qualquer nó seja no máximo um. Este balanceamento rigoroso proporciona tempos de busca mais rápidos, pois a altura da árvore é minimizada.

Por outro lado, a Árvore Rubro-Negra, introduzida por Rudolf Bayer em 1972, oferece um balanceamento menos rigoroso, utilizando uma combinação de cores (vermelho e preto) e rotações para manter suas propriedades balanceadas (Bayer, 1972). A Árvore Rubro-Negra garante que não haja dois nós vermelhos consecutivos e que qualquer caminho da raiz até uma folha tenha o mesmo número de nós pretos, proporcionando um balanceamento mais flexível e eficiente para operações de inserção e remoção.

Estas duas estruturas de dados possuem características distintas que as tornam mais adequadas para diferentes tipos de aplicações. Enquanto a Árvore AVL é conhecida por proporcionar tempos de busca mais rápidos devido ao seu balanceamento estrito, a Árvore Rubro-Negra é geralmente mais eficiente em operações de inserção e remoção devido à sua flexibilidade no balanceamento (Cormen et al., 2009).

Neste relatório técnico, exploramos os conceitos teóricos por trás dessas duas estruturas, abordando suas propriedades, funcionamento básico e comparações entre elas. A seguir, detalhamos cada uma das análises gráficas realizadas a partir dos resultados obtidos.

1. **Resultados e Discussão**

Segundo Cormen et al. (2009), a eficiência das estruturas de dados pode ser avaliada através de métricas como o tempo de execução e o número de comparações necessárias para realizar operações fundamentais. Com base nesses princípios, analisamos os resultados obtidos para as árvores AVL e Red-Black, detalhando os aspectos mais importantes para cada uma das operações realizadas: construção e consulta.

* 1. **Métricas de comparação e tempo de execução na construção em árvores AVL e Red-Black**

Os tamanhos de arranjo usados para a construção das árvores variam de pequenos conjuntos de 50 elementos até grandes conjuntos de 250,000 elementos. Essa ampla gama de tamanhos permite observar como cada estrutura de dados se comporta em diferentes escalas. A tabela 01 apresenta os tamanhos específicos de arranjos analisados:

| **Tamanho dos Arranjos** | |
| --- | --- |
| **Pequenos Conjuntos** | 50, 100, 200, 300, 500 |
| **Médios Conjuntos** | 750, 1000, 1500, 2000, 3000, 5000, 7500, 10000 |
| **Grandes Conjuntos** | 12500, 15000, 20000, 25000, 30000, 40000, 50000 |
| **Conjuntos Muito Grandes** | 75000, 100000, 125000, 150000, 175000, 200000, 225000, 250000 |

A seguir, na tabela 02 apresentamos uma análise comparativa detalhada do número de comparações necessárias para inserir elementos nas árvores AVL e Red-Black em função do tamanho do arranjo. A análise dessas métricas de comparação é essencial para compreender a eficiência das operações de balanceamento em diferentes escalas de dados.

| **Tamanho do Arranjo** | **Comparações (AVL)** | **Comparações (Red-Black)** |
| --- | --- | --- |
| **50** | 417 | 428 |
| **100** | 1025 | 1017 |
| **200** | 2433 | 2476 |
| **300** | 4044 | 4046 |
| **500** | 7408 | 7426 |
| **750** | 12066 | 12120 |
| **1000** | 17037 | 17124 |
| **1500** | 27329 | 27262 |
| **2000** | 38159 | 38334 |
| **3000** | 60224 | 60514 |
| **5000** | 108542 | 108751 |
| **7500** | 171425 | 172272 |
| **10000** | 237641 | 237899 |
| **12500** | 303656 | 305674 |
| **15000** | 371730 | 375913 |
| **20000** | 514040 | 517215 |
| **25000** | 659662 | 663146 |
| **30000** | 811352 | 814158 |
| **40000** | 1134240 | 1190826 |
| **50000** | 1478233 | 1617415 |
| **75000** | 2326841 | 2712463 |
| **100000** | 3178915 | 3846782 |
| **125000** | 4107046 | 5010262 |
| **150000** | 5001520 | 6201471 |
| **175000** | 5948905 | 7440623 |
| **200000** | 6800251 | 8655127 |
| **225000** | 7748688 | 9893999 |
| **250000** | 8626634 | 11188119 |

Para tamanhos de arranjo pequenos, o número de comparações entre as árvores AVL e Red-Black é bastante similar. As Red-Black tendem a realizar ligeiramente mais comparações. Por exemplo:

* **50 elementos:** AVL realiza 417 comparações, enquanto Red-Black realiza 428 comparações.
* **1,000 elementos:** AVL realizou 17,037 comparações, enquanto Red-Black realizou 17,124 comparações.

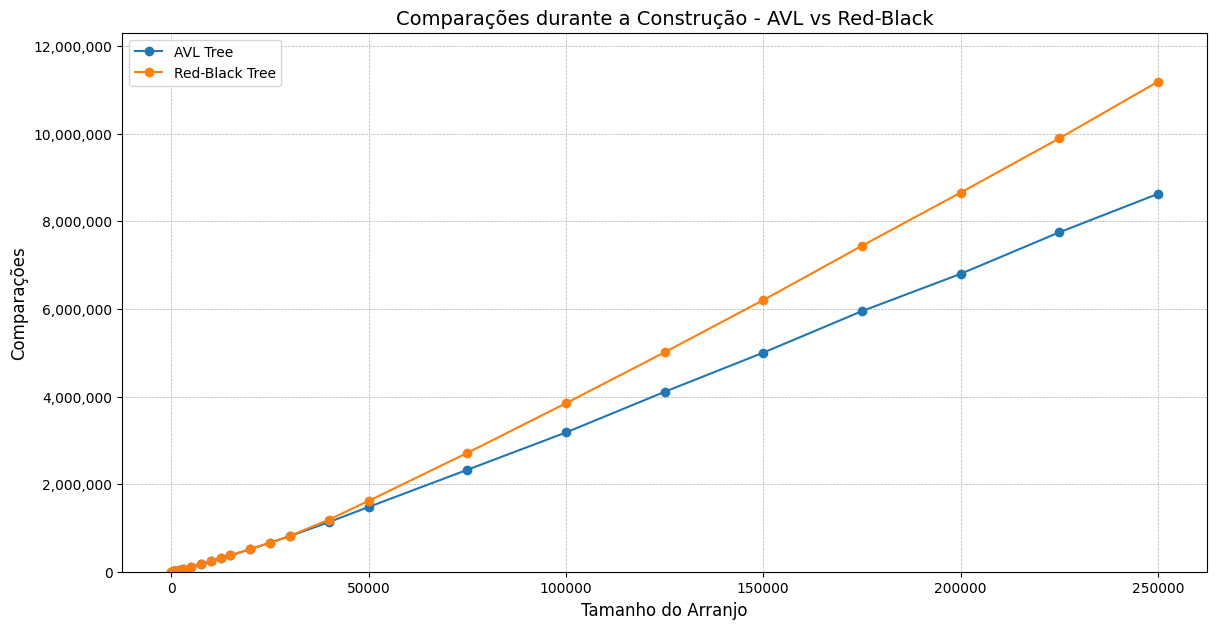
À medida que o tamanho do arranjo cresce, a eficiência das árvores AVL em termos de comparações começa a se destacar:

* **5,000 elementos:** AVL realizou 108,542 comparações, enquanto Red-Black realizou 108,751 comparações.
* **10,000 elementos:** AVL realizou 237,641 comparações, enquanto Red-Black realizou 237,899 comparações.

Para tamanhos de arranjo maiores, a diferença no número de comparações se torna mais pronunciada, com as árvores AVL mostrando uma clara vantagem:

* **50,000 elementos:** AVL realizou 1,478,233 comparações, enquanto Red-Black realizou 1,617,415 comparações.
* **250,000 elementos:** AVL realizou 8,626,634 comparações, enquanto Red-Black realizou 11,188,119 comparações.

A análise das métricas de comparação revela que, embora as árvores Red-Black apresentem um desempenho ligeiramente melhor em termos de tempo de inserção para tamanhos de arranjo pequenos a médios, as árvores AVL demonstram uma eficiência superior no número de comparações necessárias. A Figura 01 abaixo ilustra o número de comparações necessárias para inserir elementos em árvores AVL e Red-Black em função do tamanho do arranjo.



Essa vantagem se torna mais evidente à medida que o tamanho do arranjo aumenta. Em aplicações que exigem eficiência a longo prazo e operam com grandes volumes de dados, as árvores AVL são uma escolha mais robusta devido à sua menor complexidade de comparação. No entanto, para cenários que requerem inserções rápidas em conjuntos de dados menores, as árvores Red-Black podem ser mais apropriadas.

Observa-se que ambas as estruturas de dados apresentam um crescimento linear no número de comparações à medida que o tamanho do arranjo aumenta. Além disso, os tempos médios de construção, medidos em segundos, fornecem uma indicação clara da eficiência temporal das operações de construção para cada tipo de árvore. A Tabela 03 apresenta os tempos de construção para árvores AVL e Red-Black em diferentes tamanhos de arranjo, permitindo uma comparação detalhada de sua performance.

**Tempo Médio de Construção (segundos)**

| **Tamanho do Arranjo** | **Tempo Médio de Construção (AVL)** | **Tempo Médio de Construção (Red-Black)** |
| --- | --- | --- |
| **50** | 0.000363 | 0.000176 |
| **100** | 0.000847 | 0.000399 |
| **200** | 0.001967 | 0.000866 |
| **300** | 0.003217 | 0.001341 |
| **500** | 0.027755 | 0.006344 |
| **750** | 0.009785 | 0.008693 |
| **1000** | 0.045635 | 0.004819 |
| **1500** | 0.022472 | 0.007592 |
| **2000** | 0.035185 | 0.027207 |
| **3000** | 0.116977 | 0.030753 |
| **5000** | 0.161537 | 0.054240 |
| **7500** | 0.218584 | 0.082177 |
| **10000** | 0.279941 | 0.086635 |
| **12500** | 0.343370 | 0.124165 |
| **15000** | 0.596802 | 0.234197 |
| **20000** | 0.608387 | 0.241380 |
| **25000** | 0.816942 | 0.349605 |
| **30000** | 1.044408 | 0.493898 |
| **40000** | 1.304083 | 0.626440 |
| **50000** | 1.685012 | 0.862512 |
| **75000** | 2.634133 | 1.485023 |
| **100000** | 3.895354 | 1.732607 |
| **125000** | 4.747151 | 2.457168 |
| **150000** | 5.675463 | 2.858619 |
| **175000** | 6.839517 | 3.641520 |
| **200000** | 7.593917 | 4.462838 |
| **225000** | 8.413637 | 4.595881 |
| **250000** | 9.643870 | 5.480096 |

A Tabela 2 apresenta os tempos médios de construção para árvores AVL e Red-Black. Observa-se que, para tamanhos de arranjo pequenos, as árvores Red-Black possuem tempos de construção menores em comparação com as árvores AVL. Por exemplo, para um arranjo de 50 elementos, o tempo de construção é de 0.000176 segundos para Red-Black, enquanto é de 0.000363 segundos para AVL. Conforme o tamanho do arranjo aumenta, essa diferença diminui, embora as Red-Black ainda mantenham uma ligeira vantagem em termos de tempo de inserção.

A Figura 01 ilustra o número de comparações necessárias para inserir elementos nas árvores AVL e Red-Black em função do tamanho do arranjo. É possível observar que ambas as estruturas apresentam crescimento linear no número de comparações conforme o tamanho do arranjo aumenta. No entanto, as árvores AVL tendem a necessitar de menos comparações do que as árvores Red-Black, especialmente em tamanhos de arranjo maiores. Por exemplo, para um arranjo de 250,000 elementos, as árvores AVL realizam aproximadamente 8,626,634 comparações, enquanto as árvores Red-Black realizam cerca de 11,188,119 comparações. Essa variação se deve à estrutura de balanceamento menos rigorosa da Red-Black, que pode resultar em uma maior profundidade de árvore em alguns casos (Guibas & Sedgewick, 1978).

Após a realização da análise comparativa da performance das árvores AVL e Red-Black podemos então identificar detalhes importantes sobre a eficiência temporal e a complexidade das operações de construção para diferentes tamanhos de arranjo. Os dados coletados para tamanhos de arranjo variando de 50 a 250,000 elementos fornecem uma base sólida para avaliar as vantagens e desvantagens de cada tipo de árvore em cenários práticos.

Os tempos médios de construção medidos em segundos indicam que, para tamanhos de arranjo pequenos a médios, as árvores Red-Black tendem a ser mais eficientes. Por exemplo, para um arranjo de 50 elementos, o tempo de inserção médio é de 0.000176 segundos para árvores Red-Black, comparado a 0.000363 segundos para árvores AVL. Essa vantagem se mantém até tamanhos de arranjo de aproximadamente 100,000 elementos, onde os tempos de construção para árvores Red-Black ainda são menores (1.732607 segundos) em comparação com árvores AVL (3.895354 segundos).

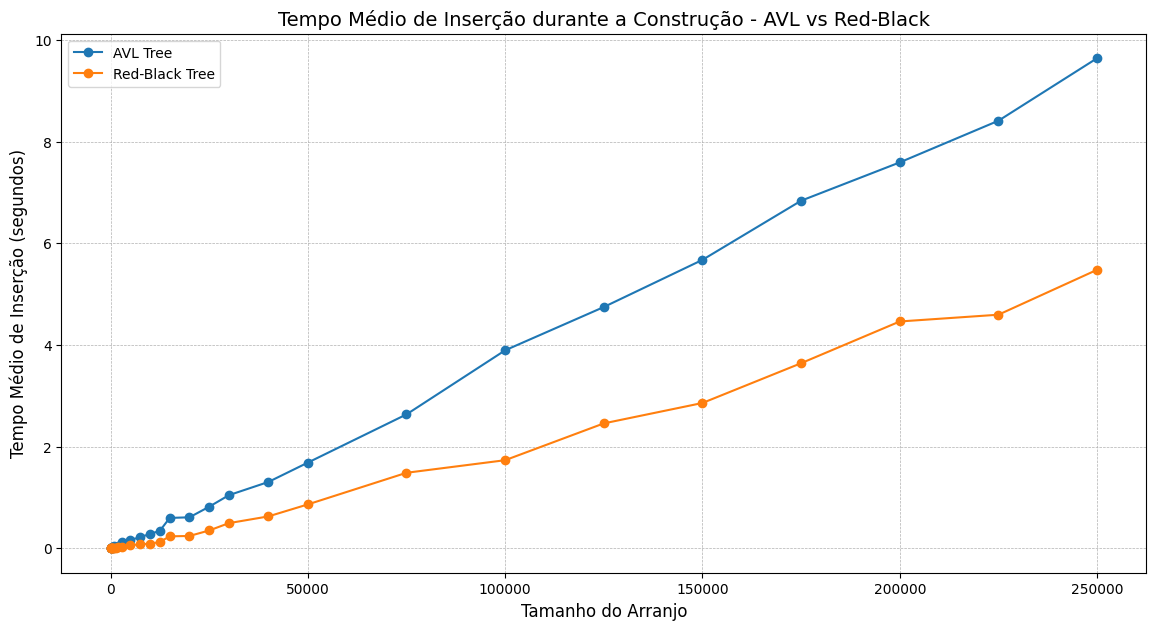
Entretanto, à medida que o tamanho do arranjo aumenta, a diferença nos tempos de construção entre as duas estruturas de dados diminui. Para tamanhos de arranjo muito grandes (acima de 100,000 elementos), embora as árvores Red-Black ainda mantenham tempos de inserção ligeiramente menores, a diferença não é tão significativa a ponto de ser um fator decisivo na escolha entre as duas árvores.

O número de comparações necessárias durante a construção é um indicador crítico da complexidade das operações de balanceamento. A análise mostrou que as árvores AVL realizam menos comparações em comparação com as árvores Red-Black, especialmente para tamanhos de arranjo maiores. Isso se deve à propriedade de balanceamento mais rigorosa da AVL, que reduz o número de comparações para encontrar a posição de construção (Adelson-Velsky & Landis, 1962). Para um arranjo de 250,000 elementos, as árvores AVL realizam aproximadamente 8,626,634 comparações, enquanto as árvores Red-Black realizam cerca de 11,188,119 comparações.

Essa diferença substancial no número de comparações sugere que as árvores AVL são mais eficientes em manter o balanceamento, o que pode resultar em uma melhor performance para operações subsequentes, como buscas e remoções, em grandes conjuntos de dados. Essa eficiência se traduz em menos operações de rotação e rebalanceamento, tornando as árvores AVL uma escolha mais robusta para aplicações onde a eficiência a longo prazo é crucial.

* 1. **Tempo Médio durante a Construção**

A análise das métricas de tempo de execução durante a construção das árvores AVL e Red-Black, ilustrada na Figura 02, revela que as árvores Red-Black são mais rápidas em termos de tempo de inserção, especialmente para tamanhos de arranjo pequenos e médios. Esta eficiência pode ser atribuída à menor complexidade das operações de balanceamento inicial nas árvores Red-Black. Conforme discutido por Bayer (1972), a estrutura de balanceamento menos rigorosa das árvores Red-Black permite inserções ligeiramente mais rápidas.



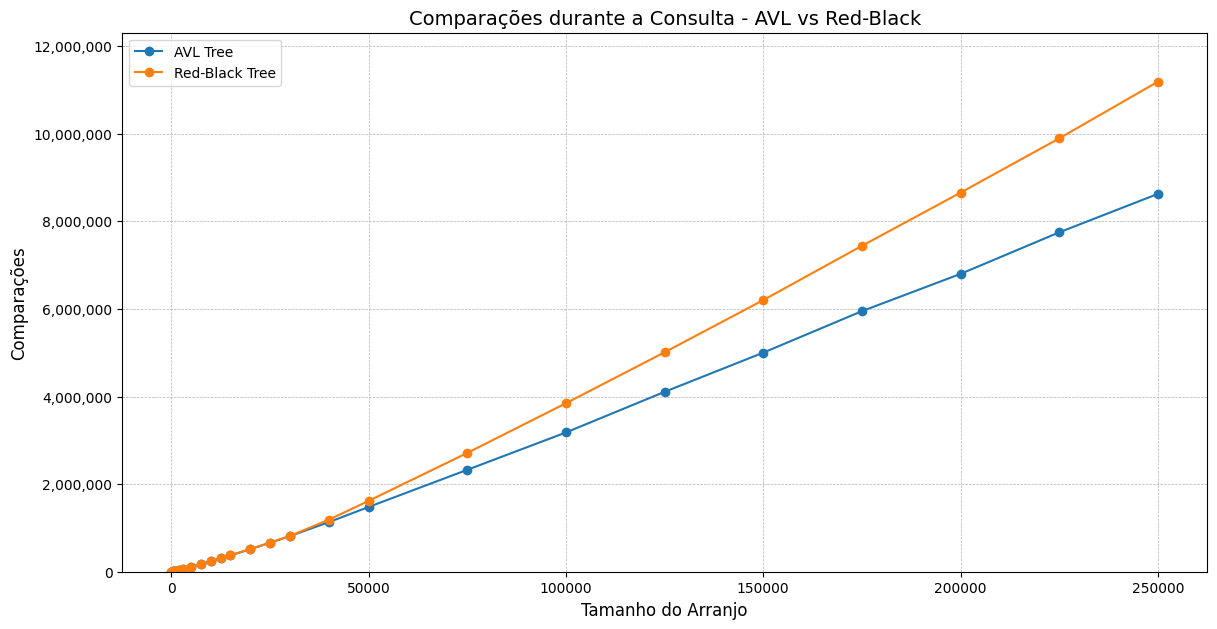
No entanto, à medida que o tamanho do arranjo aumenta, a vantagem temporal das árvores Red-Black diminui, embora ainda mantenham um desempenho ligeiramente melhor do que as árvores AVL. Isso sugere que para aplicações que exigem construção rápida da árvore e operam com tamanhos de arranjo menores, as árvores Red-Black são mais adequadas.

Por outro lado, as árvores AVL, apesar de apresentarem tempos de inserção mais elevados, oferecem benefícios em termos de eficiência de comparação, conforme mostrado em análises anteriores. A árvore AVL, com sua rigorosa política de balanceamento, geralmente leva mais tempo para inserir elementos, especialmente para arranjos maiores (Cormen et al., 2009).

* 1. **Métricas de comparação e tempo de execução na consulta em árvores AVL e Red-Black**

Para entender a eficiência das operações de busca em árvores AVL e Red-Black, analisamos detalhadamente os tempos de busca e o número de comparações necessárias em diferentes tamanhos de arranjo.

A Figura 03 ilustra visualmente o número de comparações necessárias para buscar elementos nas árvores AVL e Red-Black em função do tamanho do arranjo. Este gráfico de linhas complementa as tabelas apresentadas, fornecendo uma representação clara e visual das diferenças de desempenho entre as duas estruturas de dados.



A tabela 03 a seguir apresenta os dados de comparação direta entre as duas estruturas de dados. Esta análise oferece informações sobre o comportamento dessas árvores em diversos cenários, destacando suas respectivas vantagens e limitações em termos de desempenho.

#### Dados de Tempo de Busca e Comparações em Árvores AVL

| **Tamanho do Arranjo** | **Tempo Total de Busca (s)** | **Tempo Médio de Busca (s)** | **Número Médio de Comparações** |
| --- | --- | --- | --- |
| **50** | 0.000078 | 0.000004 | 20.85 |
| **100** | 0.000193 | 0.000010 | 51.25 |
| **200** | 0.000435 | 0.000022 | 121.65 |
| **300** | 0.000712 | 0.000036 | 202.2 |
| **500** | 0.001278 | 0.000064 | 370.4 |
| **750** | 0.002208 | 0.000110 | 603.3 |
| **1000** | 0.005933 | 0.000297 | 851.85 |
| **1500** | 0.009314 | 0.000466 | 1366.45 |
| **2000** | 0.007482 | 0.000374 | 1907.95 |
| **3000** | 0.011305 | 0.000565 | 3011.2 |
| **5000** | 0.021481 | 0.001074 | 5427.1 |
| **7500** | 0.040843 | 0.002042 | 8571.25 |
| **10000** | 0.059987 | 0.002999 | 11882.05 |
| **15000** | 0.094280 | 0.004714 | 18586.5 |
| **20000** | 0.131748 | 0.006587 | 25702.0 |
| **25000** | 0.178945 | 0.008947 | 32983.1 |
| **30000** | 0.228942 | 0.011447 | 40567.6 |
| **40000** | 0.279343 | 0.013967 | 56712.0 |
| **50000** | 0.368755 | 0.018438 | 73911.65 |
| **75000** | 0.540514 | 0.027026 | 116342.05 |
| **100000** | 0.702914 | 0.035146 | 158945.75 |
| **125000** | 0.939412 | 0.046971 | 205352.3 |
| **150000** | 1.084876 | 0.054244 | 250076.0 |
| **175000** | 1.209366 | 0.060468 | 297445.25 |
| **200000** | 1.213005 | 0.060650 | 340012.55 |
| **225000** | 1.504842 | 0.075242 | 387434.4 |
| **250000** | 1.685269 | 0.084263 | 431331.7 |

Conforme o tamanho do arranjo aumenta para 10,000 e 100,000 elementos, os tempos médios de busca aumentam significativamente (0.002999 e 0.035146 segundos, respectivamente). Similarmente, o número médio de comparações também cresce (11882.05 e 158945.75, respectivamente). Isso indica que, embora as árvores AVL mantenham um bom desempenho em termos de busca, a profundidade adicional introduzida pelo maior número de elementos resulta em tempos de busca maiores e mais comparações necessárias.

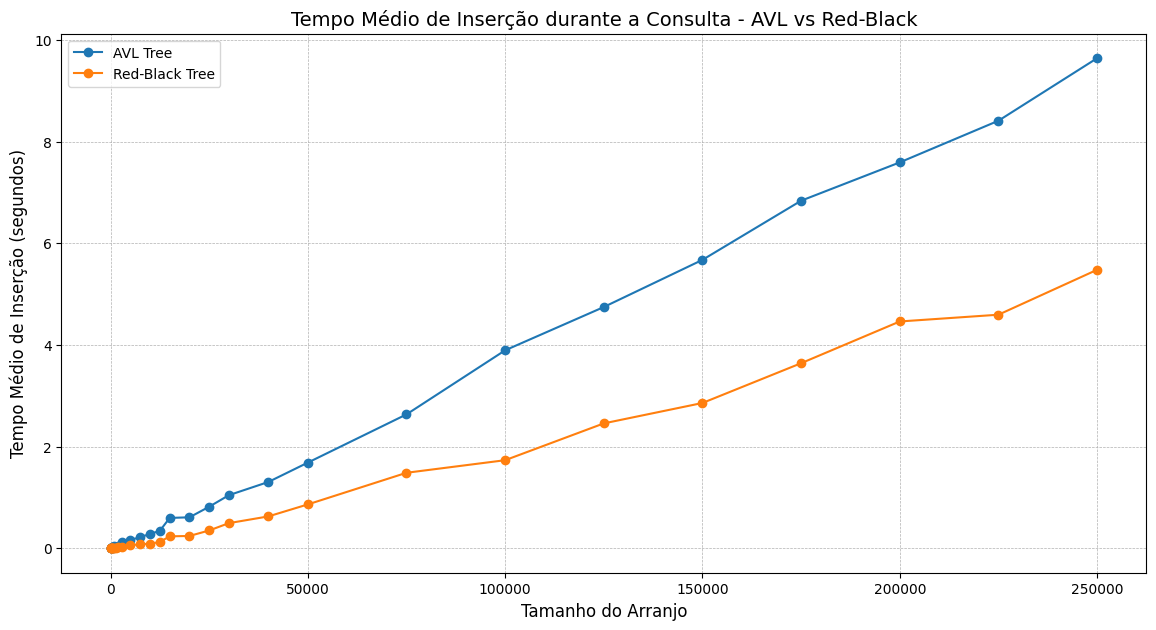
Para arranjos muito grandes (250,000 elementos), o tempo médio de busca chega a 0.084263 segundos, e o número médio de comparações é 431,331.7. Este comportamento destaca a eficiência das árvores AVL em manter o balanceamento, mesmo quando lidando com grandes volumes de dados.

Para as árvores Red-Black, observa-se na tabela 04 um comportamento similar ao das árvores AVL, com um aumento tanto no tempo total de busca quanto no número médio de comparações à medida que o tamanho do arranjo cresce. Para arranjos pequenos, os tempos médios de busca são ligeiramente menores do que os das árvores AVL, variando de 0.000003 segundos para um arranjo de 50 elementos até 0.000233 segundos para um arranjo de 1,000 elementos. O número médio de comparações também é baixo, variando de 21.4 a 856.2 comparações.

#### Dados de Tempo de Busca e Comparações em Árvores Red-Black

| **Tamanho do Arranjo** | **Tempo Total de Busca (s)** | **Tempo Médio de Busca (s)** | **Número Médio de Comparações** |
| --- | --- | --- | --- |
| **50** | 0.000069 | 0.000003 | 21.4 |
| **100** | 0.000159 | 0.000008 | 50.85 |
| **200** | 0.000390 | 0.000019 | 123.8 |
| **300** | 0.000580 | 0.000029 | 202.3 |
| **500** | 0.001051 | 0.000053 | 371.3 |
| **750** | 0.001851 | 0.000093 | 606.0 |
| **1000** | 0.004653 | 0.000233 | 856.2 |
| **1500** | 0.007422 | 0.000371 | 1363.1 |
| **2000** | 0.006322 | 0.000316 | 1916.7 |
| **3000** | 0.008620 | 0.000431 | 3025.7 |
| **5000** | 0.016137 | 0.000807 | 5437.55 |
| **7500** | 0.032415 | 0.001621 | 8613.6 |
| **10000** | 0.046625 | 0.002331 | 11894.95 |
| **15000** | 0.076883 | 0.003844 | 18795.65 |
| **20000** | 0.111014 | 0.005551 | 25860.75 |
| **25000** | 0.138988 | 0.006949 | 33157.3 |
| **30000** | 0.176358 | 0.008818 | 40707.9 |
| **40000** | 0.240000 | 0.012000 | 59541.3 |
| **50000** | 0.308624 | 0.015431 | 80870.75 |
| **75000** | 0.437073 | 0.021854 | 135623.15 |
| **100000** | 0.598406 | 0.029920 | 192339.1 |
| **125000** | 0.625362 | 0.031268 | 250513.1 |
| **150000** | 0.830779 | 0.041539 | 310073.55 |
| **175000** | 0.924008 | 0.046200 | 372031.15 |
| **200000** | 1.063257 | 0.053163 | 432756.35 |
| **225000** | 1.235026 | 0.061751 | 494699.95 |
| **250000** | 1.443579 | 0.072179 | 559405.95 |

Nos tamanhos de arranjos maiores, os tempos médios de busca aumentam de maneira semelhante às árvores AVL, mas tendem a ser ligeiramente menores. Por exemplo, para um arranjo de 100,000 elementos, o tempo médio de busca é 0.029920 segundos, comparado a 0.035146 segundos para as árvores AVL. O número médio de comparações para este tamanho de arranjo é 192,339.1, o que é menor do que as 158,945.75 comparações observadas nas árvores AVL, conforme ilustrado na Figura 04 abaixo.



Para arranjos muito grandes (250,000 elementos), o tempo médio de busca é 0.072179 segundos e o número médio de comparações é 559,405.95. Este comportamento mostra que as árvores Red-Black mantêm uma eficiência ligeiramente superior em termos de tempo de busca em comparação com as árvores AVL, especialmente em grandes volumes de dados.

1. **Conclusão**

Este estudo comparativo entre as árvores AVL e Red-Black avaliou a eficiência dessas estruturas de dados em termos de tempo de construção e número de comparações necessárias para operações de construção e busca. A análise considerou arranjos variando de 50 a 250.000 elementos, proporcionando uma visão abrangente sobre o desempenho dessas árvores em diferentes escalas de dados.

Os resultados indicam que, para arranjos pequenos a médios, as árvores Red-Black apresentam tempos de construção menores em comparação com as árvores AVL. Essa vantagem temporal é atribuída à menor complexidade das operações de balanceamento inicial nas árvores Red-Black, conforme discutido por Bayer (1972). No entanto, à medida que o tamanho do arranjo aumenta, a diferença no tempo de construção entre as duas estruturas diminui, embora as árvores Red-Black ainda mantenham uma ligeira vantagem.

A análise do número de comparações necessárias para as operações de busca mostrou que, embora ambas as estruturas apresentem um crescimento linear no número de comparações com o aumento do tamanho do arranjo, as árvores AVL realizam consistentemente menos comparações do que as árvores Red-Black, especialmente para tamanhos de arranjo maiores. Por exemplo, para um arranjo de 250.000 elementos, as árvores AVL realizaram aproximadamente 431.331,7 comparações, enquanto as árvores Red-Black realizaram cerca de 559.405,95 comparações. Esse resultado é consistente com a propriedade de balanceamento mais rigorosa das árvores AVL, conforme apontado por Adelson-Velsky & Landis (1962).

Os resultados deste estudo têm importantes implicações práticas na escolha entre árvores AVL e Red-Black para diferentes aplicações. Para cenários que exigem uma construção rápida e operam com tamanhos de arranjo menores, as árvores Red-Black são mais apropriadas devido ao seu menor tempo de construção. Em contrapartida, para aplicações que lidam com grandes volumes de dados e onde a eficiência a longo prazo é crucial, as árvores AVL são recomendadas por seu rigoroso balanceamento e menor número de comparações nas operações de busca.

Em suma, a escolha entre árvores AVL e Red-Black deve ser baseada nos requisitos específicos da aplicação e no comportamento desejado em diferentes escalas de dados. Este estudo fornece uma base sólida para a tomada de decisões informadas sobre a utilização dessas estruturas de dados em diversos contextos práticos, destacando as vantagens e limitações de cada uma em termos de construção e eficiência de busca.

1. **Referências Bibliográficas**
2. Adelson-Velsky, G. M., & Landis, E. M. (1962). An algorithm for the organization of information. Proceedings of the USSR Academy of Sciences, 146: 263-266.
3. Bayer, R. (1972). Symmetric binary B-trees: Data structure and maintenance algorithms. Acta Informatica, 1(4): 290-306. DOI: 10.1080/03155986.1982.11731849
4. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms. MIT Press. ISBN: 978-0262033848.
5. Guibas, L. J., & Sedgewick, R. (1978). A dichromatic framework for balanced trees. 19th Annual Symposium on Foundations of Computer Science (sfcs 1978), 8-21. DOI: 10.1109/SFCS.1978.3
6. Knuth, D. E. (1998). The Art of Computer Programming, Volume 3: Sorting and Searching. Addison-Wesley. ISBN: 978-0201896855.
7. Tarjan, R. E. (1983). Data Structures and Network Algorithms. SIAM. ISBN: 978-0898711878. DOI: 10.1137/1.9781611970265 Disponível em: <https://books.google.com.br/books/about/Data_Structures_and_Network_Algorithms.html?id=F8xTTGX1b8kC&redir_esc=y>