CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CAMPUS TIMÓTEO

Raphael Gomes Wagner e José Geraldo Duarte Junior e Pedro Arthur Diniz

ARVORES BINÁRIA: EXCLUSÃO

Timóteo

2022

Raphael Gomes Wagner e José Geraldo Duarte Junior e Pedro Arthur Diniz

ARVORES BINÁRIA: EXCLUSÃO

Trabalho 3 de Algoritmos e Estrutura de Dados sobre exclusão em Arvores Binarias no curso de Engenharia de Computação no Centro Federal de Educação Tecnológica de Minas Gerais

Orientador: Gustavo Martins

Timóteo

2022

.

.

.

Dedico a
Leandro Santana, Arthur Gomes, José Geraldo Duarte Junior
Felipe Reggiane, Daniel Vidal, Guilherme Heringer, Nathan Teixeira e Pedro Arthur.

Agradecimentos

Agradeço a todos da dedicação por terem passado em AED I comigo, e aos dois primeiros por terem ajudado no estudo da matéria, me ensinando.

1 Introdução

Esse trabalho foi desenvolvido em conjunto, feito por Raphael Gomes, José Geraldo e Pedro Arthur, no intuito de executar exclusão em uma Arvore Binária, desenvolvida pela Professora Divani Barbosa Gavinier(GAVINIER, 2020), porém editada, adicionando métodos que permitissem a execução do trabalho. Essa implementação foi utilizada para medir tempos de execução sobre a exclusão de 5% dos elementos de cada arvore.

Com o intuito de executar a atividade em diferentes ambientes, foram utilizadas 3 tipos de arvores diferentes, e elas são: Arvore com elementos sequenciais, Arvore com elementos randômicos, e Arvore balanceada com elementos sequenciais. Em teoria, deveriam ser utilizados arvores com os seguintes tamanhos:

1. 10^1 elementos
2. 10^3 elementos
3. 10^5 elementos
4. 10^7 elementos
4. 10^9 elementos

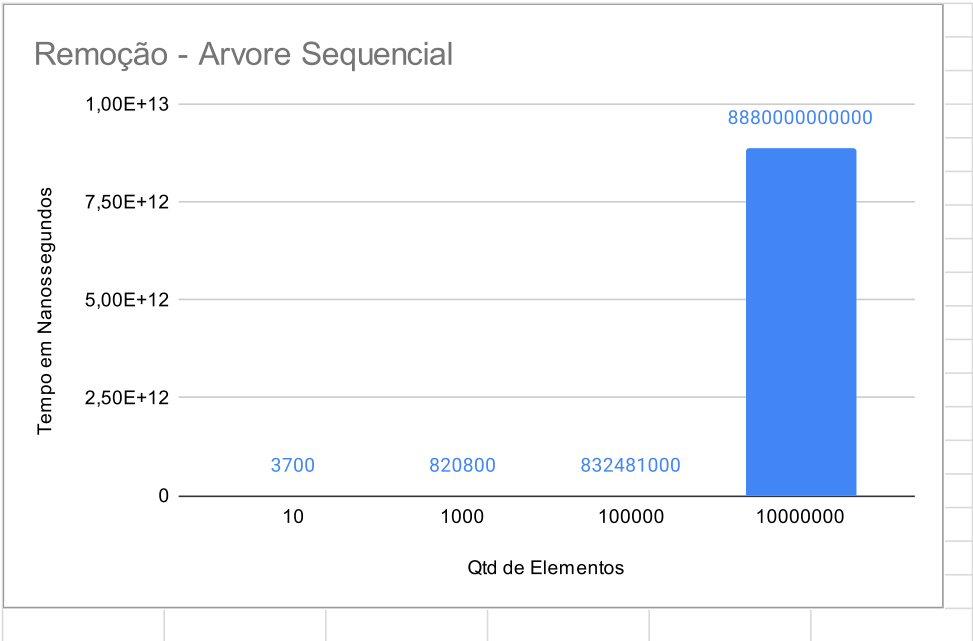
2 Desenvolvimento

2.1 Remoção na Arvore Sequencial

Iniciando o desenvolvimento da prática, as arvores foram criadas na IDE assim como os vetores de tamanhos diversos para armazenar os valores que serão inseridos na arvore. Porém, o Java não permite a criação de vetores em 10^9 elementos, então todas as arvores criadas, foram geradas até 10^7 elementos. A partir disso, foram utilizados métodos de inserção de dados da Pratica 3, desenvolvida anterior a esta, já que este trabalho trata-se de uma continuação direto com estudos sobre remoção.

A os dados selecionados para a remoção foram gerados utilizando a biblioteca interna do Java "Random", que permitiu gerar valores aleatórios entre 0 e n (n = quantidade de elementos), aumentando a taxa de sucesso para gerar valores validos para a remoção. Os dados apresentados a seguir inicialmente trazem os resultados do tempo de execução da da remoção de 5% de elementos aleatórios de arvores de diferentes tamanhos que receberam seus valores de forma ordenada:

Quantidade de elementos	Tempo em Nanossegundos
10^1	3700
10^3	820800
10^5	832481000
10^7	8880000000000



2.2 Remoção na Arvore Randômica

Para a remoção de elementos aleatórios desta arvore foi gerado um vetor com valores aleatórios utilizando um gerador congruencial pseudo aleatório com os seguintes parâmetros:

Semente = 0;

Modulo = 1073741824;

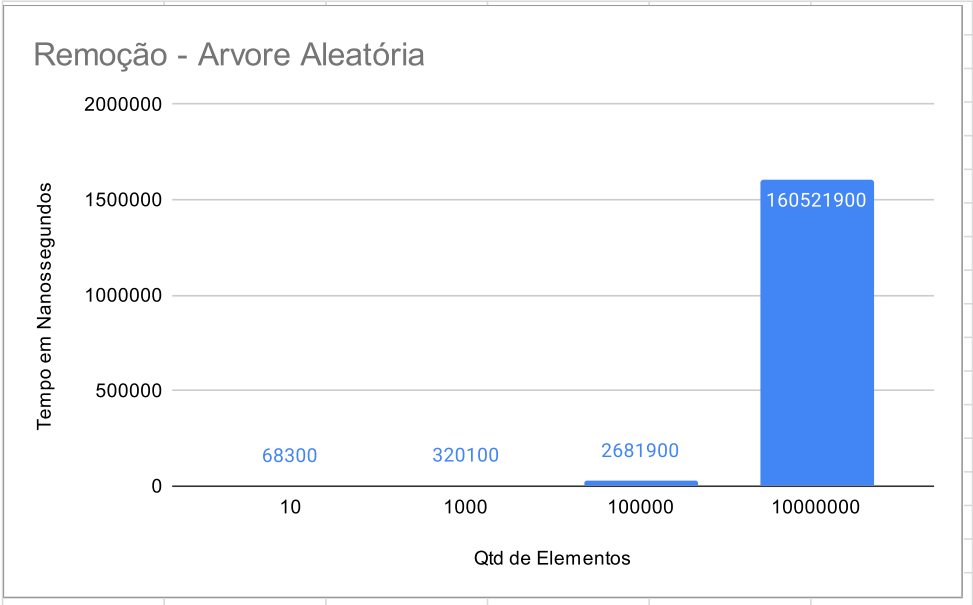
Multiplicador = 843314861;

Incremento = 453816693;

estes parâmetros também foram utilizados na inserção, então é certo que os mesmos estarão na arvore, mas não foram removidos na mesma ordem que foram inseridos, e sim de forma aleatória também.

Logo os dados apresentados a seguir trazem os resultados do tempo de execução da da remoção de 5% de elementos aleatórios de arvores de diferentes tamanhos que receberam seus valores de forma aleatória:

Quantidade de elementos	Tempo em Nanossegundos
10^1	68300
10^3	320100
10^5	2681900
10^7	160521900

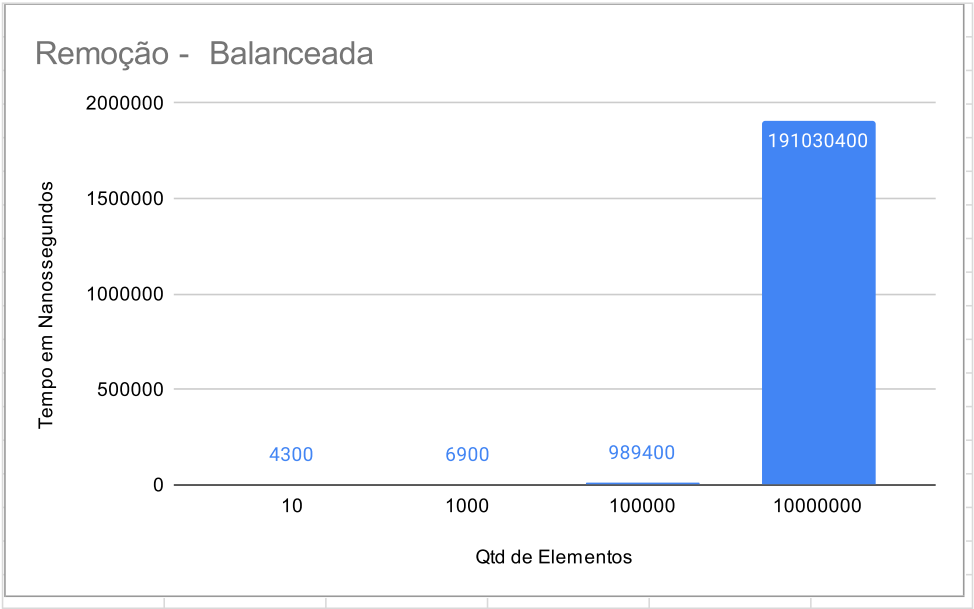


2.3 Remoção na Arvore Balanceada

Para a remoção de elementos de dentro da arvore balanceada a logica utilizada segue a mesma da arvore sequencial, já que os valores inseridos na mesma estão entre 0 e n (n = quantidade de elementos), porem foram inseridos em uma condição especial que força a arvore a ser formada de já balanceada, como foi visto na atividade pratica 3.

Logo os dados apresentados a seguir trazem os resultados do tempo de execução da remoção de 5% de elementos aleatórios de arvores balanceadas de diferentes tamanhos:

Quantidade de elementos	Tempo em Nanossegundos
10^1	4300
10^3	6900
10^5	989400
10^7	191030400


Remoção - Balanceada
2000000
1500000
1000000
500000
0
Tempo em Nanossegundos
4300
6900
989400
191030400
10
1000
100000
10000000
Qtd de Elementos

3 Conclusão

Ao fim deste trabalho, foi abortado como foi realizada a remoção de elementos nas arvore com elementos sequenciais, arvore com elementos randômicos, e arvore balanceada com elementos sequenciais e seus diversos tamanhos. Foi observado o tempo de execução da remoção de 5% de elementos aleatórios em cada arvore cujo o tamanho varia de 10^1 elementos a 10^7 elementos, pois como relatado anteriormente o Java não permite a criação de uma arvore de tamanho 10^9. É notado também como a exclusão na arvore sequencial demora mais do que as outras pra realizar o processo, assim como foi na inserção sem que fosse utilizada a inserção otimizada. Em seguida, a arvore balanceada tem desempenho melhor do que a arvore randômica, exceto na exclusão com 10^7 elementos na arvore.

Referências

GAVINIER, D. B. *Arvore Binaria em Java*. 2020. Disponível em: <https://gist.github.com/divanibarbosa/819e7cfcf1b9bae48c4e0f5bd74fb658>. Citado na página 4.