Relatório AEDS II - CEFET-MG

Alunos: Bernard Menezes.

Prática 1

25 de agosto de 2017

Sumário

1 Introdução **1**

2 Códigos **1**

3 Resultados **1**

3.1 Comparações de registros 2

3.2 Tempo 3

4 Conclusão **3**

1 INTRODUÇÃO

Este relatório refere-se à Prática 1 da matéria de Algorítmos e Estrutura de Dados II. Toda a codificação a qual este relatório faz referência foi feita na linguagem de programação JAVA e a IDE Eclipse, utilizando como base as implementações disponíveis nesse site.

Nesta prática foi pedido a implementação de um **árvore binária** com uma codificação específica para a classe Item, e também foram solicitados os métodos de inicialização, inserção e pesquisa. Todas as simulações presentes neste documento referem-se a experimentos feitos em um computador com a seguinte arquitetura: processador Intel Atom N470 (512K Cache, 1.83 GHz) 2GB de RAM e 320GB de memória secundária.

2 CÓDIGOS

Os códigos com os respectivos comentários encontram-se no mesmo repositório que este relatório, e como já dito anteriormente, utilizam-se de implementações disponibilizadas pelo Professor Emérito do Departamento de Ciência da UFMG, Nivio Ziviani, em site. A saída dos programas utilizados para gerar os gráficos presentes neste relatório podem ser encontrados no arquivo: `"Arvore_Binaria/saida"`.

3 RESULTADOS

Como pedido, os elementos foram inseridos de duas formas diferentes na árvore, de maneira ordenada e de maneira aleatória. Após as inserções na árvore o método de pesquisa foi chamado para um elemento que não estava presente nela, e durante essa procura foram realizado dois testes medindo: o número de comparação de registros e de tempo (milisegundos). Os teste foram realizados com o número de elementos a serem inseridos variando de 1000 até 9000 com passo de 1000 elementos. Foram gerados gráficos comparativos para mensurar e diferenciar o desempenho nos dois tipos de inserção.

3.1 COMPARAÇÕES DE REGISTROS

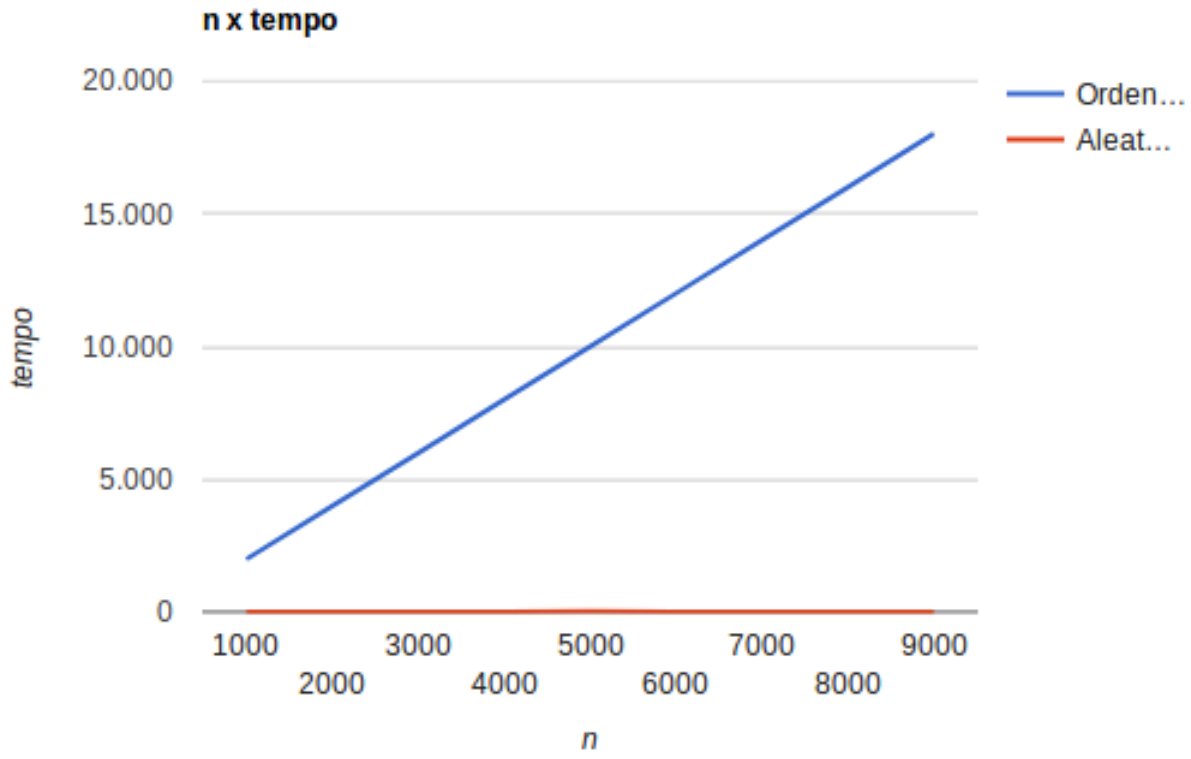

IMAGEM 1: GRÁFICO DE COMPARAÇÕES X TAMANHO

Como a imagem 1 mostra, quando a inserção em uma árvore binária ocorre de maneira ordenada, o número de comparações entre registros aumenta de maneira linear. Isso era esperado já que a inserção de número em ordem crescente ou decrescente em uma árvore binária não explora um dos ponteiros dos nós (esquerda se for crescente e direita se for decrescente) e por isso cresce apenas verticalmente. Dessa forma a árvore binária vira uma espécie de lista encadeada, o que torna a procura muito custosa em relação a comparações de registros. Já com a inserção aleatória a árvore cresce lateralmente também, o que permite que a principal característica da árvore seja mantida e fique totalmente diferente de uma lista encadeada, por isso a comparação de registros em uma árvore binária com inserção de elementos aleatórios é menos custosa.

3.2 TEMPO

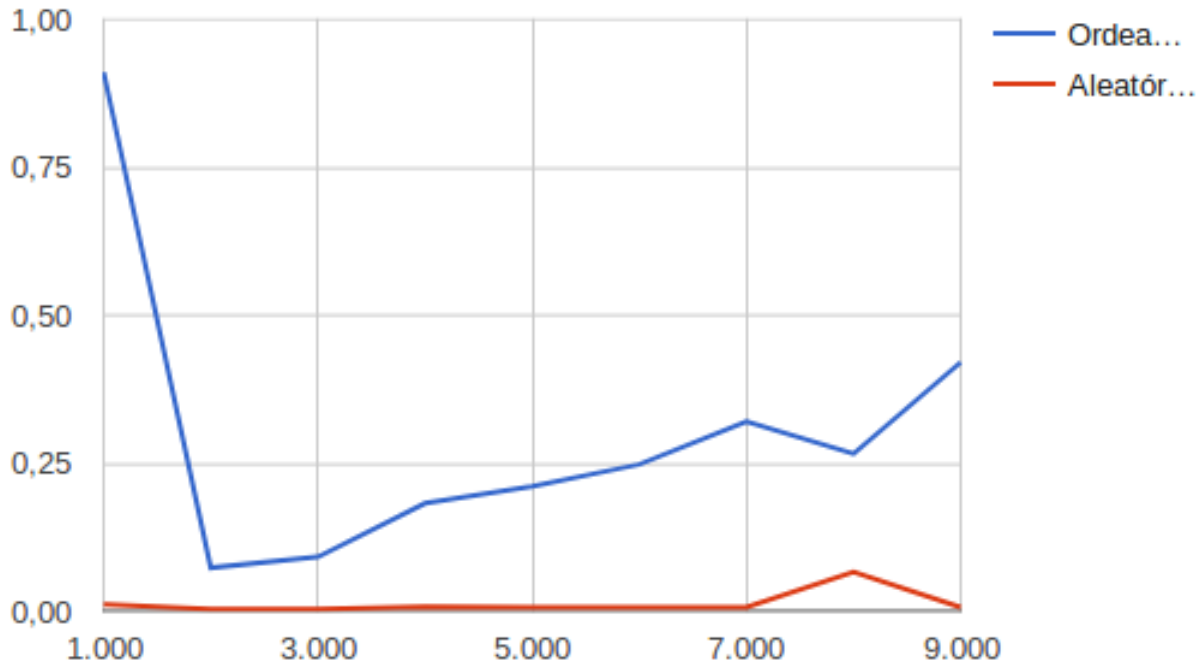

IMAGEM 2: GRÁFICO DE COMPARAÇÕES X TAMANHO

O gráfico 2 é consequência do que foi dito na seção 3.1, como o método de inserção ordenada de elementos gera na pesquisa um número maior de comparações, o custo computacional do programa aumenta, consequentemente o seu tempo de execução é alterado também.

4 CONCLUSÃO

Através dessa prática é possível concluir que árvores binárias são eficientes quanto à pesquisa desde que a inserção de elementos nela ocorra de maneira minimamente aleatória, para que ela cresça lateralmente também. Caso a inserção tenha caráter ordenado (crescente ou decrescente) a pesquisa pela árvore demora muito já que não será mais uma pesquisa $O(n \log n)$ (melhor caso) e sim $O(n)$ (pior caso).