CTC12 Lab Balanced Tree

Aluno: Douglas Bergamim Fernandes

Relatório **(gere um único PDF)**

1) Descreva a estrutura que escolheu:

A estrutura de dados escolhida foi de uma árvore Vermelho-Preto. Esse tipo de estrutura de dado é um árvore de busca binária que satisfaz as seguintes propriedades:

- Todo nó é vermelho ou preto;
- a raiz é preta;
- toda folha (NIL) é preta;
- se um nó é vermelho, então seus filhos são pretos;
- para cada nó, todos os caminhos simples do nó até as folhas descendentes têm a mesma quantidade de nós pretos.

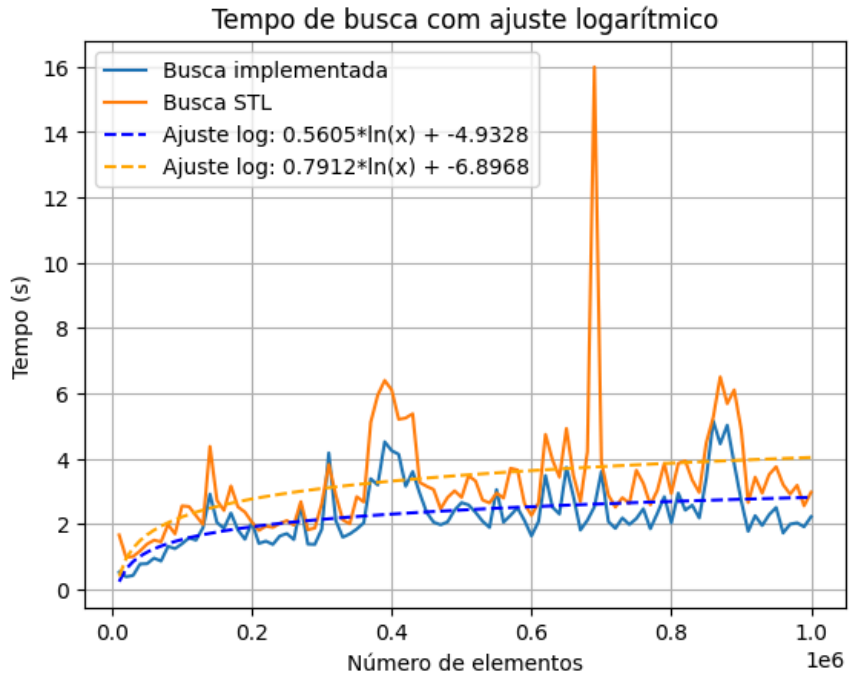
O livro base utilizado para implementar o algoritmo foi:
Cormen, T.H., Leiserson, C.E., Rivest, R. L., & Stein, C. *Algoritmos: Teoria e Prática* (3ª ed.).

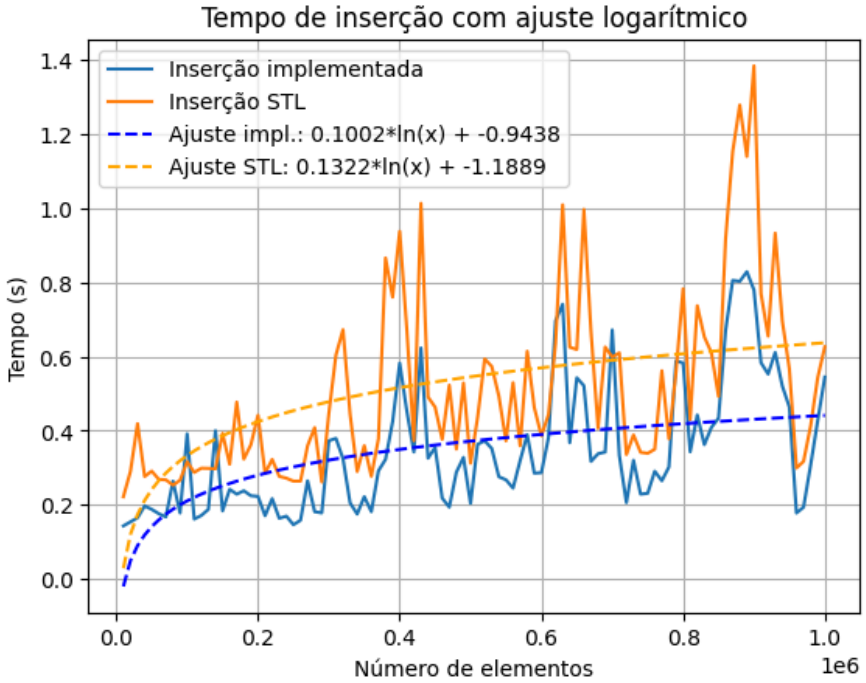
- Alguma consideração que ajude a entender o seu código (implementou algo diferente do usual? Há classes diferentes que mereçam um diagrama de classes ou uma explicação? Vale desenhos a mão fotografados ou diagramas feitos em software.).

Para auxiliar a busca na função find(), foi feita uma implementação parecida com o do multimap. Uma struct iterator foi criado na classe `IndexPointsAluno`. Dessa forma, na implementação de find(), o código primeiro acha, por uma busca binária, o nó que está o primeiro elemento maior que first e depois usa o iterador para achar os elementos com chave imediatamente maior e que ainda estão abaixo do limite superior second.

2) Mostre na forma de gráficos as curvas de tempo do teste OakByNorm, comparando a sua implementação com a minha implementação baseada em stl::multimap.

Mostre que as buscas **e** inserções são O(log n) em ambas as implementações - use um programa de fit em função logaritmica. “Buscas **e** inserções” significa que devem plotar as duas medidas.





Nos dois casos, o algoritmo implementado é mais rápido que o algoritmo STL.