Instituto Tecnológico de Aeronáutica - ITA
CTC-12 - Projeto e Análise de Algoritmos
Aluno: Ulisses Lopes da Silva

Relatório do Laboratório 2 - Árvores Balanceadas

---

1 Estrutura de Dados Escolhida

A estrutura de dados escolhida para realizar o laboratório foi a Árvore Rubro-Negra. A implementação das funções principais, como *Insert*, *InsertFixUp* etc., foi realizada com base no pseudocódigo disponibilizado no livro **Algoritmos - Teoria e Prática**, de Thomas M. Cormen.

Sobre a estrutura em questão, foram implementados os métodos básicos da classe, tais como rotações para a esquerda e para a direita, inserções e correções de balanceamento. Contudo, o método mais importante, além do de inserção e rebalanceamento, foi o da função privada ***findInOrder***, que é instanciada num segundo nível pelo método ***find***, da classe ***IndexPointsAlunos***. Essa função consiste em uma busca recursiva, *In Order*, pelos nós da árvore; caso o nó visitado possua uma chave que pertença ao intervalo [*first*, *last*], o elemento correspondente é inserido no vetor recebido por referência.

2 Curvas de Tempo Obtidas

- Tempo medido para Buscas

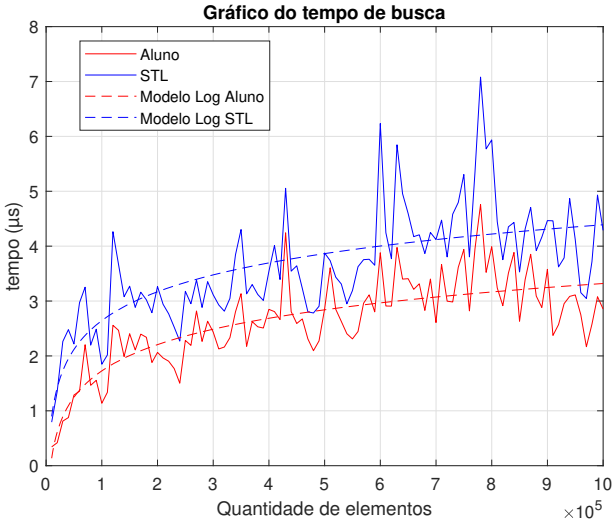

Fig. 1: Tempo medido para buscas

- Tempo medido para Inserções

Fig. 2: Tempo medido para inserções

Como se pode verificar analisando as figuras, o tempo médio para cada uma das categorias, tanto pela implementação do aluno, quanto pela implementação pela biblioteca **std::multimap** têm um crescimento logarítimico. Dessa forma, é notável perceber que, de fato, as árvores balanceadas têm uma ordem $O(logn)$, com a do aluno sendo superior em questão de otimização, obtendo um tempo um pouco menor para as operações, conforme mostra a figura abaixo:

```
ulissesuls@Ulisses:~/Projects/CTC-12/Lab 02/build$ ./labTreetests
Running main() from /home/ulissesuls/Projects/CTC-12/Lab 02/build/_deps/googl
[==========] Running 7 tests from 2 test suites.
[----------] Global test environment set-up.
[----------] 4 tests from SanityTests
[ RUN      ] SanityTests.readAscFile
[       OK ] SanityTests.readAscFile (0 ms)
[ RUN      ] SanityTests.indexFile
[       OK ] SanityTests.indexFile (0 ms)
[ RUN      ] SanityTests.indexFileALU
[       OK ] SanityTests.indexFileALU (0 ms)
[ RUN      ] SanityTests.AllSearches
[       OK ] SanityTests.AllSearches (0 ms)
[----------] 4 tests from SanityTests (0 ms total)

[----------] 3 tests from OakTests
[ RUN      ] OakTests.OakReadFile
[       OK ] OakTests.OakReadFile (215 ms)
[ RUN      ] OakTests.OakByNorm
[       OK ] OakTests.OakByNorm (835 ms)
[ RUN      ] OakTests.VoidSphereSelection
 VoidSphere time (ms) MMP: 2.17331 aluno: 1.14958
[       OK ] OakTests.VoidSphereSelection (834 ms)
[----------] 3 tests from OakTests (1885 ms total)

[----------] Global test environment tear-down
[==========] 7 tests from 2 test suites ran. (3106 ms total)
[  PASSED  ] 7 tests.
```

Fig. 3: Registro do tempo de execução em ambos os métodos

3 Referências Bibliográficas

[1]. **CORMEN**, Thomas M. Algoritmos - Teoria e Prática. 3a edição, Editora ELSEVIER, Rio de Janeiro, RJ, 2012.