Relatório EP3

Aluno: Luis Vitor Zerkowski
NUSP: 9837201
Professora: Cristina G. Fernandes
Disciplina: Algoritmos e Estruturas de Dados II (MAC-0323)

Este relatório foi escrito no intuito de documentar todos os experimentos feitos para o exercício programa três da disciplina. De maneira geral, este arquivo pode ser dividido em duas partes. A primeira trata-se de uma tabela com todos os tempos devidamente medidos para cada uma das estruturas de dados construírem seu mapa de palavras em frequências. Já a segunda, é uma análise crítica dos resultados obtidos, avaliando a qualidade de cada uma das estruturas em cada um dos diferentes contextos. Segue, portanto, uma breve descrição de cada um dos arquivos utilizados no experimento.

1. teste_pequeno.txt -> Testa se as estruturas estão funcionando. Contém apenas 30 palavras e 23 diferentes chaves.

2. dicionario.txt -> Arquivo com não muitas palavras - 261797 -, mas alto número de chaves - 245118 - e, além disso, organizadas em ordem quase alfabética.

3. dictionary.txt -> Arquivo organizado completamente em ordem alfabética. Conta com 174539 palavras e 174539 chaves.

4. les_miserables.txt -> Primeiro arquivo de texto não pré-fabricado. Conta com 578851 palavras e 22813 chaves organizadas de maneira aleatória.

5. bible.txt -> Arquivo de texto não pré-fabricado sutilmente maior que o anterior, porém com menos chaves. Conta com 794948 palavras e 12905 chaves.

6. sherlock.txt -> Último arquivo de texto não pré-fabricado. Maior que os anteriores em tamanho e número de chaves, conta com 1105285 palavras e 29157 chaves.

Nas tabelas a seguir, os tempos foram todos medidos em segundos e a busca realizada foi feita para a palavra “a” em todos os textos e para todas as estruturas. Vale notar, ainda, que todos os testes foram realizados pelo menos vinte vezes para garantir valores médios e não ápices de qualidade ou falha, a menos do arquivo “dictionary.txt” para a ABB. Como essa estrutura demorava cerca de 7 minutos para ser resolvida - podendo chegar até 15 minutos - os experimentos na mesma foram realizados apenas duas vezes. Vale notar, ainda, que as cores da tabela indicam qual estrutura teve o melhor desempenho para cada arquivo, sendo verde um indicativo de menor tempo de construção e vermelho um indicativo do maior tempo de construção.

Textos e Estruturas	teste_pequeno.txt	dicionario.txt	dictionary.txt
ABB	Construção: 30e-6 Busca: 10e-6	Construção: 4,831 Busca: 20e-6	Construção: 388,14 Busca: 25e-6
ARN	Construção: 39e-6 Busca: 18e-6	Construção: 354e-3 Busca: 25e-6	Construção: 176e-3 Busca: 20e-6
Hashing Encadeamento	Construção: 24e-6 Busca: 10e-6	Construção: 386e-3 Busca: 17e-6	Construção: 333e-3 Busca: 18e-6
Hashing Encadeamento MTF	Construção: 25e-6 Busca: 23e-6	Construção: 385e-3 Busca: 21e-6	Construção: 257e-3 Busca: 12e-6
Hashing Linear	Construção: 31e-6 Busca: 9e-6	Construção: 16,928 Busca: 14e-6	Construção: 5,699 Busca: 19e-6
Tries	Construção: 45e-6 Busca: 14e-6	Construção: 346e-3 Busca: 6e-6	Construção: 236e-3 Busca: 18e-6

Textos e Estruturas	les_miserables.txt	bible.txt	sherlock.txt
ABB	Construção: 306e-3 Busca: 29e-6	Construção: 359e-3 Busca: 25e-6	Construção: 598e-3 Busca: 26e-6
ARN	Construção: 525e-3 Busca: 27e-6	Construção: 654e-3 Busca: 23e-6	Construção: 1,078 Busca: 25e-6
Hashing Encadeamento	Construção: 315e-3 Busca: 11e-6	Construção: 404e-3 Busca: 10e-6	Construção: 575e-3 Busca: 18e-6
Hashing Encadeamento MTF	Construção: 233e-3 Busca: 11e-6	Construção: 245e-3 Busca: 17e-6	Construção: 442e-3 Busca: 17e-6
Hashing Linear	Construção: 377e-3 Busca: 13e-6	Construção: 491e-3 Busca: 20e-6	Construção: 863e-3 Busca: 17e-6
Tries	Construção: 315e-3 Busca: 21e-6	Construção: 331e-3 Busca: 20e-6	Construção: 533e-3 Busca: 12e-6

Analisando os resultados obtidos para todas as estruturas de dados alguns vários fenômenos puderam ser notados:

1. Em primeiro lugar, é possível observar a constante alta qualidade das estruturas *Hashing com Encadeamento e Move To Front* e *Tries*. Ainda que não tivessem o melhor dos tempos naquele experimento, estavam sempre muito perto do melhor tempo.

2. Em seguida, faz-se perceptível a qualidade das árvores balanceadas com a *ARN*, mas também o preço a pagar-se por elas. Para textos cuja quantidade de chaves não passava da ordem de 10e4, seu desempenho deixa um tanto a desejar, aproximando-se de uma *ABB*, mas ainda ficando para trás por conta de todos os remanejos que deveriam ser feitos para que a árvore permanecesse balanceada. Em se tratando de textos com número de chaves na ordem de 10e5 ou mais, no entanto, essas estruturas mostraram-se altamente qualificadas, ganhando muito em tempo das *ABBs* e, por vezes, ganhando inclusive do *Hashing com Encadeamento e Move To Front* e *Tries* também.

3. É possível perceber também a qualidade das estruturas de *Hashing* em se tratando de buscas. Apesar dos tempos serem muito pequenos e razoavelmente parecidos em valores absolutos para todas as estruturas, existe uma sensível diferença relativa nesse tempo comparando-se as estruturas de *Hashing* com as estruturas de *Árvore*, com arquivos nos quais os *Hashings* chegam a ter metade do tempo de busca das estruturas em *Árvore*.

4. Por fim, pode-se fazer um apanhado geral de todas as estruturas:
    a. ABB -> Altamente funcional para textos com poucas chaves, independente do número de palavras, e vai muita perdendo qualidade para textos com muitas chaves, em particular aqueles organizados alfabeticamente.

    b. ARN -> Bom desempenho para qualquer texto com qualquer quantidade de chaves e qualquer quantidade de palavras. Sua eficiência torna-se notável, no entanto, quando o número de chaves cresce muito - na ordem de 10e5 ou mais - e quando as chaves estão organizadas de maneira alfabética.

    c. Hashing com encadeamento -> Excelente desempenho para todo tipo de texto, mantendo-se sempre entre as melhores estruturas independentemente do número de chaves e palavras. Apesar do bom desempenho geral, tipicamente não se destaca como a melhor das estruturas para nenhum tipo de texto, a não ser aqueles absolutamente minúsculos - da ordem de 10 chaves e palavras.

    d. Hashing com encadeamento e MTF -> Desempenho geral estupendo independente do tipo de texto e da quantidade de chaves ou palavras. Destacou-se diversas vezes como a melhor das estruturas e mesmo quando não era a melhor, estava muito próximo dos melhores tempos.

    e. Hashing linear -> Estrutura boa para textos convencionais, poucas chaves - da ordem de até 10e4 -, muitas palavras - de ordem a partir de 10e6 - e ordem aleatória dessas palavras. Para textos organizados de forma alfabética, seu desempenho cai bastante, podendo, inclusive, ser pior do que a *ABB*.

    f. Tries -> Desempenho geral magnífico independente do número de chaves e palavras. Se não era a melhor das estruturas, estava sempre entre as melhores. Sua única ressalva faz-se para textos minúsculos - da ordem de

10 chaves e palavras. Por conta da complexidade da estrutura, acaba por não ter desempenho muito bom nesses textos, porém o tempo absoluto para os mesmos é tão pequeno que acarreta grandes prejuízos para a alta qualidade das *Tries*.