Relatório de Estruturas de dados II

Creative Problem 3.3.30 (Software Caching) – Rodrigo Alves Souza (6800149)

Resultados obtidos

Foram feitos experimentos utilizando:

1. ST sem caching
2. RedBlackBST sem caching
3. RedBlackBST com caching

Também foi testado para os seguintes tamanhos mínimos de palavras (tresholds): 1, 5 e 10.

Foi utilizado o arquivo **leipzig1M.txt** retirado do Algs4 com 1 milhão de linhas e aproximadamente 130MB.

NOTA: TODOS AS IMPLEMENTAÇÕES FORAM TIRADAS DO ALGS4 DO SEDGEWICK E MODIFICADAS PARA O PROPÓSITO DO EXERCÍCIO

FrequencyCounter usando ST e sem caching

Treshold = 1:

```
the 48058
distinct = 85175
words    = 874238
Tempo: 1.609994485s
```

Treshold = 5

```
would 1700
distinct = 60422
words    = 300803
Tempo: 1.469427557s
```

Treshold = 10

```
government 2791
distinct = 40933
words    = 176292
Tempo: 2.60368188s
```

FrequencyCounter usando RedBlackBST e sem caching

Treshold = 1:

```
the 52327
distinct = 89620
words    = 950708
Tempo: 1.929531321s
```

Treshold = 5

```
would 627
distinct = 32246
words    = 111680
Tempo: 2.023279443s
```

Treshold = 10

```
government 847
distinct = 18269
words    = 52458
Tempo: 1.870015418s
```

FrequencyCounter usando RedBlackBST e caching

Treshold = 1:

```
the 7722
distinct = 27188
words    = 140044
Tempo: 1.425244683s
```

Treshold = 5

```
would 2628
distinct = 79204
words    = 465967
Tempo: 1.557238869s
```

Treshold = 10

```
government 648
distinct = 15566
words    = 41576
Tempo: 0.84517271s
```

Conclusões

A implementação de ST original do livro utiliza o TreeMap como estrutura de dados para guardar os pares chave-valor. O TreeMap do java também utiliza uma BST como implementação. A diferença do ST para o RedBlackBST é que o Sedgewick utiliza LeftLeaning RedBlack BST como implementação, onde suas ligações vermelhas são sempre com o nó da esquerda. Portanto, em questões de eficiência, as versões implementadas são bem próximas uma das outras.

O Software Caching da BST guarda o último Node acessado para que o acesso e modificação seja mais rápido. Porém como o caching é feito apenas para a última palavra acessada, o algoritmo tem efeito apenas quando temos palavras repetidas uma seguida da outra no arquivo de entrada. Para textos comuns isto não ocorre.

NOTA: AS FLUTUAÇÕES DE TEMPO E DIFERENTES RESULTADOS PARA O MESMO ARQUIVO DE ENTRADA E ARGUMENTOS JÁ OCORREM NA IMPLEMENTAÇÃO ORIGINAL DO FREQUENCYCOUNTER, DIFICULTANDO A ANALISE