MAC323 EXERCÍCIO-PROGRAMA 4

PALAVRAS E SUAS FREQÜÊNCIAS II

Y. KOHAYAKAWA

Data de entrega: 18/6/2012 (23:55)

Introdução. Este EP é uma continuação do EP1. Neste EP, você deve usar as implementações de tabelas de símbolos vistas em sala, com adaptações adequadas, para resolver o problema proposto. Você deverá implementar tais tabelas usando: (*i*) árvores binárias de busca (simples), (*ii*) árvores binárias de busca aleatorizadas, (*iii*) skip-lists, (*iii*) árvores rubro-negras esquerdistas.

Descrição do problema. Dado um texto, queremos montar a lista de palavras que ocorrem no texto, com suas respectivas freqüências, em ordem decrescente de freqüência. Por exemplo, suponha que a entrada é o texto abaixo:

```
We must not underrate the gravity of the task which lies before us or the
temerity of the ordeal, to which we shall not be found unequal.  We must
expect many disappointments, and many unpleasant surprises, but we may be sure
that the task which we have freely accepted is one not beyond the compass and
the strength of the British Empire and the French Republic.
```

A saída deve então ser

```
the 9
and 3
not 3
of 3
we 3
which 3
We 2
be 2
many 2
must 2
task 2
...
```

Note que, por exemplo, *the* aparece 9 vezes no texto e é a palavra mais freqüente. No caso de empate, as palavras devem ser listadas em ordem alfabética (note que *and*, *not*, *of* etc ocorrem todas 3 vezes e que esta lista está ordenada).

Até este ponto, não há nada novo em relação ao EP1. Incluiremos um novo componente da seguinte forma. O usuário poderá também solicitar *remoções de palavras*, dando também como entrada um texto. Se o usuário pedir para remover

```
All of us share this world for but a brief moment in time.
```

o resultado passaria a ser

Versão preliminar de 27 de maio de 2012, 19:12.

```
the 9
and 3
not 3
we 3
which 3
We 2
be 2
many 2
must 2
of 2
task 2
...
```

O usuário fornecerá uma seqüência de textos, cada um com a especificação de ser adicionado ou removido de sua coleção de palavras, e em cada momento o usuário poderá pedir para ver a lista atual de palavras e suas freqüencias.

Seu programa. Seu programa deve receber o nome de um arquivo como um argumento na linha de comando, com a opção `-f`. Este arquivo deve ter o seguinte formato: cada linha deve começar com o caracter `+`, `-`, ou `p`.

Os caracteres `+` e `-` devem ser seguidos do nome de um arquivo texto a ser considerado: se o caracter for `+`, as palavras no arquivo texto devem ser adicionadas à coleção de palavras atual; se o caracter for `-`, as palavras no arquivo texto devem ser removidas da coleção de palavras atual. Aqui, naturalmente, *adicionar* e *remover* podem simplesmente significar incrementar e decrementar a freqüência da palavra em questão (inserções e remoções só ocorrem nos casos em que a palavra não pertence à coleção e nos casos em que a palavra está com freqüência 1 e pede-se sua remoção).

Se uma linha do arquivo de entrada começar com `p`, a coleção atual de palavras deve ser impressa, com suas respectivas freqüências, em ordem decrescente de freqüência (como descrito em mais detalhe acima). Toda saída do programa deve ser enviada para o `stdout`.

Assim, uma execução de seu programa poderia ser

```
prompt$ ep4 -farqs > out.txt
```

onde `arqs` contém as linhas

```
+ Churchill.txt
- Obama.txt
p
+ Churchill2.txt
p
```

A arquitetura de seu sistema. Seu sistema deve ser composto, pelo menos, das seguintes partes: `ep4main.c`, `ST.h`, `Item.h`, `Item.c`, `BST.c`, `rBST.c`, `SL.c` e `LLRB.c`. Ademais, você deve entregar um `Makefile`, para que seja fácil compilar seu sistema. Este `Makefile` deve especificar como gerar os executáveis `ep4BST`, `ep4rBST`, `ep4SL` e `ep4LLRB`.

O programa `ep4main.c` deve implementar o núcleo de seu sistema, que faz uso de uma tabela de símbolos, de forma abstrata (usando as funções em `ST.h`; aqui, pequenas adaptações

interessantes para este EP são permitidas). A tabela de símbolos deve manipular objetos do tipo `Item`, especificado por `Item.h` e `Item.c`.

Para gerar `ep4BST`, o usuário dirá

```
prompt$ make ep4BST
```

Esta chamada do `make` deve compilar `BST.c`, o programa que implementa uma tabela de símbolos com árvores binárias de busca simples, gerando `BST.o`. Ademais, esta chamada de `make` deve também produzir o executável `ep4BST` a partir de `ep4main.o`, `BST.o` e `Item.o`.

Os programas `rBST.c`, `SL.c` e `LLRB.c` devem implementar a tabela de símbolos com árvores binárias de busca aleatorizadas, skip-lists e árvores rubro-negras esquerdistas. Usando estas implementações, seu sistema deve ser capaz de gerar `ep4rBST`, `ep4SL` e `ep4LLRB`, de forma análoga a como `ep4BST` é gerado.

Opções. Implemente, pelo menos, a opção de linha de comando `-n`, que diz ao seu programa quantas palavras devem ser impressas. Por exemplo,

```
prompt$ ep4 -n7 -farqs > out.txt
```

deve imprimir as 7 palavras em resposta a todos os comandos `p` em `arqs`.

O usuário deve poder escolher entre os modos *case insensitive* e *case sensitive*: isto é, ignorando a diferença entre maiúsculas e minúsculas ou não. O padrão deve ser *case sensitive* e o modo *case insensitive* deve ser especificado com o argumento de linha de comando `-I`.

Relatórios de eficiência. Você deve comparar a eficiência de `ep4BST`, `ep4rBST`, `ep4SL` e `ep4LLRB`, executando testes com grandes volumes de dados (use livros do projeto Gutenberg). Elabore um relatório, descrevendo detalhadamente seus experimentos e resultados.

Observações

1. *Este EP é estritamente individual.* Programas semelhantes receberão nota 0.
2. Seja cuidadoso com sua programação (correção, documentação, apresentação, clareza do código, etc), dando especial atenção a suas estruturas de dados. A correção será feita levando isso em conta.
3. Comparem entre vocês o desempenho de seus programas.
4. Entregue seu EP no Paca.
5. Não deixe de incluir em seu código um *relatório* para discutir seu EP: discuta as estruturas de dados usadas, os algoritmos usados, etc. *Se você escrever claramente como funciona seu EP, o monitor terá pouca dificuldade em corrigi-lo, e assim você terá uma nota mais alta.* (Se o monitor sofrer para entender seu código, sua nota será baixa.)

Observação final. Enviem dúvidas para a lista de discussão da disciplina.

Instituto de Matemática e Estatística, Universidade de São Paulo, Rua do Matão 1010, 05508–090 São Paulo, SP

Endereço eletrônico: `yoshi@ime.usp.br`