CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
RECUPERAÇÃO DE INFORMAÇÃO - MÁQUINAS DE BUSCA NA WEB

Última alteração: 8 de abril de 2013 — 1º semestre de 2013
Professores: Berthier Ribeiro-Neto e Nivio Ziviani
Monitor: Aline Bessa
Trabalho Prático 1: 06/03/2013

Data de Entrega: 15/04/2013
Penalização por Atrazo: 1 ponto até 17/04/2013 mais 1 ponto por dia útil a seguir

---

Arquivos Invertidos

O objetivo deste trabalho é projetar e implementar um sistema de programas para recuperar eficientemente informação em grandes arquivos armazenados em memória secundária, utilizando um tipo de índice conhecido como arquivo invertido. As principal referências para este trabalho são Witten, Moffat e Bell (1999), Neubert (2000) e Moffat e Bell (1995). Nesta primeira parte do trabalho o seu programa deverá ser capaz de construir o índice sem necessidade de ser *in situ*, e sem necessidade de usar compressão.

O conceito de arquivo invertido será apresentado a seguir. Considere um conjunto de documentos. A cada documento é atribuído um conjunto de palavras-chave ou atributos. Um *arquivo invertido* é constituído de uma lista ordenada de palavras-chave, onde cada palavra-chave tem uma lista de apontadores para os documentos que contêm aquela palavra-chave. Este é o tipo de índice utilizado pela maioria dos sistemas para recuperação em arquivos constituídos de texto.

A utilização de arquivo invertido aumenta a eficiência de pesquisa em várias ordens de magnitude, característica importante para aplicações que utilizam grandes arquivos constituídos de texto. O custo para se ter essa eficiência é a necessidade de armazenar uma estrutura de dados que pode ocupar entre 2% e 100% do tamanho do texto original, dependendo da quantidade de informação armazenada no índice, mais a necessidade de atualização do índice toda vez que a coleção de documentos sofre alguma alteração.

O que fazer

A estrutura de dados a ser implementada deverá ser constituída do vocabulário do texto, incluindo o número de documentos associados com cada palavra-chave e uma lista de ocorrências da palavra na coleção de documentos. Cada entrada da lista indica o número do documento onde a palavra ocorreu e o número de ocorrências. O formato da lista invertida deve ser o formato apresentado em Moffat e Bell (1995) e Witten, Moffat e Bell (1999). Para cada termo t devem ser armazenados todos as triplas $< d, f_{d,t}, p >$, onde d é o número do documento onde t ocorre, $f_{d,t}$ é a freqüência do termo t no documento d e p é a posição onde t ocorre dentro de d.

Pontos Extras

As listas de números de documentos dentro da lista invertida contém números inteiros positivos em ordem ascendente. Existem métodos de compressão específicos para conjuntos de números em ordem ascendente que levam a boas taxas de compressão. três destes métodos são codificação unária, Elias-γ e Elias-δ, todos descritos nas referências citadas abaixo. Este trabalho não exige o uso de compressão do índice. Entretanto, quem decidir usar compressão no momento de armazenar a lista invertida no disco terá 10% de *pontos extras*.

Processamento de Consultas

O sistema de programas recebe do usuário uma ou mais palavras e imprime todos os documentos que satisfaçam a consulta. No caso deste trabalho a linguagem de consulta pode conter:

a) um conector lógico `and` entre duas palavras,

b) um conector lógico `or` entre duas palavras.

A saída será apenas a impressão dos documentos (seus identificadores) que satisfaçam a consulta.

Como fazer

1. A linguagem a ser utilizada é C++ (obrigatório).
2. Você pode utilizar um parser público. Um que pode ser utilizado está em: `http://sourceforge.net/projects/htmlcxx`.
3. Um pequeno exemplo da coleção pode ser usado imediatamente para implementar e testar seu trabalho em `http://garnize.latin.dcc.ufmg.br/colecaoRI/toyExample.tgz` A coleção é composta pelo arquivo de indice e os arquivos de texto, conforme abaixo:

   ```
   a) arquivo de indice:
   <url 0> <arquivo de texto 0> <offset de inicio> <offset de fim>
   ...
   <url i> <arquivo de texto k> <offset de inicio> <offset de fim>
   ...
   <url n> <arquivo de texto m> <offset de inicio> <offset de fim>

   b) arquivo de texto (contem o HTML das paginas):
   <pagina 1> ... <pagina k> ... <pagina n>

   Abaixo temos um exemplo de uma colecao com 5 urls.
   a) arquivo de indice:
   http://www.uol.com.br pagesRI0 0 2048
   http://www.globo.com pagesRI0 2049 3049
   http://www.google.com pagesRI1 3050 4200
   http://www.yahoo.com pagesRI1 4201 5500
   http://www.dcc.ufmg.br/~nivio pagesRI2 5501 6000

   b) arquivos de texto: conforme mostra o arquivo de indice temos tres
   arquivos de texto, pagesRI0, pagesRI1 e pagesRI2

   pagesRI0:
   <texto da pagina do UOL><texto da pagina da Globo>
   pagesRI1:
   <texto da pagina do Google><texto da pagina do Yahoo>
   pagesRI2:
   <texto da pagina do Nivio>
   ```

   Contudo, o trabalho deve ser testado usando a coleção completa, disponível em DVD no LATIN (sala 3023). Uma vez que os dados da coleção completa estão comprimidos você deve usar as classes para ler os dados disponibilizadas em: `http://garnize.latin.dcc.ufmg.br/colecaoRI/riCode.tgz`.
4. É proibido fazer tudo em memória principal, sendo necessário ir ao disco e fazer ordenação externa.

5. Preocupem com eficiência: façam medida do tempo de criação do índice. Experimentos em diferentes máquinas, ou mesmo comparando tecnologias de storage diferentes (como HD vs SSD) sao apreciados, ainda que não contem ponto extra.

6. Procure deixar o indexador genérico, pois provavelmente terão de indexar uma outra colecao um pouquinho maior para o TP2.

Avaliação

O trabalho será avaliado a partir das listagens dos programas, da documentação entregue, da análise de complexidade das rotinas e do resultado da execução.

Apresente uma boa documentação do trabalho, contendo pelo menos os seguintes itens: saída legível mostrando o funcionamento do código, código bem estruturado, comentários explicativos sobre os algoritmos e estruturas de dados, análise da complexidade, resultados experimentais medindo tempos e análise dos resultados.

Referências

Ian H. Witten, Alistair Moffat e Timothy C. Bell: *Managing Gigabytes: Compressing and Indexing Documents and Images*, Morgan Kaufmann Publishers, 1999, second edition.

Justin Zobel e Alistair Moffat: Inverted Files for Text Search Engines, *ACM Computing Surveys 38*(2), 2006.

Alistair Moffat e Timothy C. Bell: In Situ Generation of Compressed Inverted Files, *Journal of the American Society for Information Science 46*(7), 1995: 537–550.

Ricardo Baeza-Yates and Berthier Ribeiro-Neto: *Modern Information Retrieval*, Pearson, 2011 (second edition), 913 pages.