Indexação e Apache Lucene

Bruna T. Silva[1], Gabriel G. Conejo[2]

[1]Departamento de Ciência da Computação – Universidade do Estado de Santa Catarina
Centro de Ciências Tecnológicas – Caixa Postal 15.064 – Joinville – SC – Brasil

{gabrielgconejo,silvatavares.bruna}@gmail.com

Resumo. *O Apache Lucene é uma biblioteca construída para indexação e busca de documentos, utilizando o processo de stemming e com índices invertidos. Este artigo visa trazer um entendimento básico sobre o Lucene e seus componentes, e trás também um resumo sobre os testes usando o Ferret (Lucene para linguagem Ruby).*

1. O Apache Lucene

O Apache Lucene é um projeto de código aberto que desenvolve biblioteca de pesquisa pesquisa. Criado em 2000 por Doug Cutting, possui duas funcionalidades principais: indexação e pesquisa em texto. Com uma indexação de alta performance, a API Lucene está presente em vários aplicativos desktop e web [Hatcher and Gospodnetic 2004]. —a. O que é o Apache Lucene

2. Funcionamento

A indexação de um documento, têm sua primeira etapa instanciando o IndexWriter, adiciona-se documentos a base, e indexando o documento com o processo chamado de índice Invertido. O índice invertido, ao contrário dos índices comuns, cria uma estrutura de termos que referenciam os documentos indexados (com uma chave para o documento). Porém antes de indexar um documento ele é analisado.

A análise possui vários processos de conversão, onde o objetivo é converter o dado em um termo, durante a análise os dados passam por um processo de transformação em token, onde há extração das palavras, remoção de palavras comuns, redução de palavras para o formato raíz, etc. [IBM 2009]

```
// Cria o analyzador
StandardAnalyzer analyzer = new StandardAnalyzer();

// Diretório virtual para o índice
Directory indexDirectory = new RAMDirectory();

// Cria o arquivo com tamanho ilimitado.
IndexWriter w = new IndexWriter(indexDirectory, analyzer, true,
    IndexWriter.MaxFieldLength.UNLIMITED);

// Adiciona 4 documentos.
addDoc(w, "Lucene in Action");
addDoc(w, "Lucene for Dummies");
addDoc(w, "Managing Gigabytes");
addDoc(w, "The Art of Computer Science");

// Fecha o arquivo.
w.close();
```

3. Stemming

Stemming é um algoritmo que reduz palavras a sua forma comum, ao seu radical, através de um processo chamado conflação, combinar formas variantes de um termo. O stemming reduz uma palavra ao seu radical, através de exclusão de sufixos e/ou prefixos.

As vantagens no uso do stemming incluem: reduzir o tamanho do índice, aumentar a chance de recuperar um documento. E suas desvantagens são principalmente a perda de precisão na recuperação de documentos e a perda de termos diferentes, quando são diferentes em contexto, mas possuem radicais iguais ou parecidos, acabam sendo fundidos.

3.1. Stemming em Português

Existem diversos algoritmos de stemming para português, dentre eles os mais conhecidos são: a versão para português do algoritmo de Porter, o Removedor de Sufixos da Lingua Portuguesa (RSLP) e o algoritmo de STEMBR. O RSLP possui arquitetura similar ao de Porter, aplicando sucessivas remoções de sufixos através de regras, com um diferencial de possuir um dicionário de exceções. As regras podem ser declaradas com: sufixo. um tamanho mínimo para o stem resultante após a remoção do sufixo. um sufixo de substituição (opcional) e uma lista de exceções (opcional).

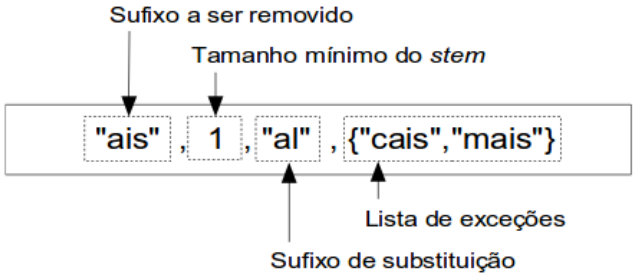

Figura 1. Exemplo de declaração de regra para o RSP. [Coelho 2007]

4. Melhorias do Algoritmo

A alternativa mais viável e consideravelmente mais efetiva, para uma melhoria do algoritmo em português, seria criar um dicionário auxiliar de dados, e a cada stem criado para um termo, esse termo e seu respectivo stem seriam salvos no dicionário, e a cada processo de stem seria consultado antes o dicionário, onde esse dicionário poderia usar o processo de índice invertido.

5. Remoção de Documento

Remoção de documentos. A remoção de um documento ocorre nos seguintes cenários: conteúdos já indexados são atualizados, ou índices tornaram-se grandes demais (em questão de tamanho). A exclusão de documento utiliza o IndexWriter, uma busca para exclusão.[Lucene 2006]

6. Eficiência com o tempo

Eficiência de um índice com o passar do tempo. O índice é eficiente com o tempo, mas se houverem muitas remoções e adições de documentos, é indicado criar os índices novamente para toda a biblioteca de documento, a fim de melhorar a performance.

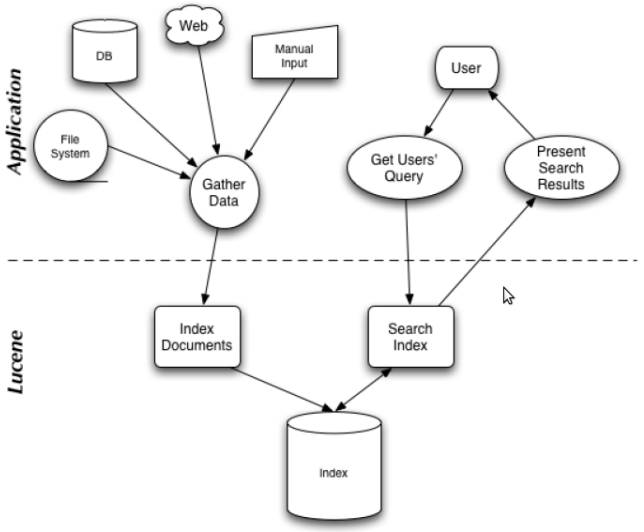

Figura 2. Um exemplo típico de implementação integrado ao Lucene [Hatcher and Gospodnetic 2004]

7. Implementação

A implementação do algoritmo para criação de índice, e busca de informações, foi feita em linguagem Ruby. Dividida em dois programas principais, o primeiro com o download da página web e criação do índice da referida página, e o segundo com a pesquisa nos índices.

8. Conclusão

O framework Lucene é uma ferramenta de busca e criação de índices rápida e eficiente, mas sua principal utilidade é a portabilidade, como está implementado em várias linguagens possui poder de implementação em diferentes aplicações e diferentes ramos. Usado em sites conhecidos como o Wikipedia, Sourceforge ou a CNET, incluindo suas aplicações que vão de simples programas de busca, como o implementado neste trabalho, a poderosos bancos de dados GeoEspacial em NoSql (usando o Lucene 4 Spatial3 para construir o ındice espacial), o Lucene é um framework de alta performance que pode ser implementado em vários domínios, sendo suas buscas simples e rápidas.

Referências

Coelho, A. R. (2007). Stemming para a língua portuguesa: estudo, análise e melhoria do algoritmo rslp.

Hatcher, E. and Gospodnetic, O. (2004). Lucene in action.

IBM, d. (2009). Usando o apache lucene para procura de texto.

Lucene, A. (2006). Apache lucene core.