Um Método para Indexação de Formulários Web visando Consultas por Similaridade[1]

Willian Ventura Koerich, Ronaldo dos Santos Mello

Centro Tecnológico (CTC) - Departamento de Informática e Estatística (INE)
Universidade Federal de Santa Catarina (UFSC) – Florianópolis, SC - Brasil

{willian.vkoerich,ronaldo}@inf.ufsc.br

Abstract: *Search engines do not support specific searches for web forms found on Deep Web. Within this context, the WF-Sim project proposes a query-by-similarity system for Web Forms to deal with this lack. This paper presents an indexing technique for querying-by-similarity web forms as a WF-Sim system component. This technique is centered on suitable index structures to the main kinds of queries applied to web forms, as well as some optimizations in these structures to reduce the number of index entries. To evaluate the indexes' performance, we ran experiments on two persistence strategies: file system and database. The performance of accessing the database was higher. We also compare the performance of our indexes with the traditional keyword-based index, and the results were also satisfactory.*

Resumo: *Motores de busca atuais não possuem suporte à buscas específicas por formulários web relacionados à Deep Web. Neste contexto, o projeto WF-Sim propõe um processador de consultas por similaridade para formulários Web para lidar com esta limitação. Este artigo apresenta uma técnica de indexação para buscas por similaridade em formulários web, atuando como um componente do sistema WF-Sim. Esta técnica está centrada em estruturas de índice adequadas aos principais tipos de consulta aplicados a formulários web, bem como otimizações nestas estruturas para reduzir a quantidade de entradas no índice. Experimentos preliminares sobre duas estratégias de persistência de dados suportados pelo WF-Sim foram realizados: sistema de arquivos e banco de dados. O desempenho de acesso ao banco de dados foi superior. Comparou-se também o desempenho dos índices propostos contra o tradicional índice de palavras-chave, e o resultado também foi satisfatório.*

1. Introdução

Uma grande quantidade de serviços está atualmente à disposição das pessoas através da Web, como locação e vendas de veículos, reserva de hotéis, compra de livros, oferta de empregos, etc. Esses serviços disponibilizam diversos dados para consultas aos usuários como por exemplo, veículos de diversos fabricantes e modelos, no caso de um web *site* de uma concessionária. O acesso a esses bancos de dados é possível através de formulários existentes em páginas Web. Estes formulários exibem atributos do banco de dados sobre os quais o cliente especifica filtros e então submete consultas. Estes bancos de dados na Web são denominados banco de dados escondidos (*hidden databases* ou

[1] Este trabalho é parcialmente financiado pelo CNPq através do projeto WF-Sim (Nro. processo:481569/2010-3).

deep-Web)[2], uma vez que a sua estrutura e o seu conteúdo não estão completamente visíveis ao usuário. Somente alguns atributos (e alguns eventuais valores que permitem a definição de filtros) estão visíveis nos formulários [Madhavan et al. 2009].

O projeto WF-Sim visa desenvolver uma solução para esta problemática: um processador de consultas por similaridade para formulários Web [Gonçalves 2011]. Este processador caracteriza-se por ser um software responsável pela execução de todas as tarefas necessárias à geração de um resultado adequado a uma consulta por similaridade, como especificação de uma consulta por parte do usuário e métricas adequadas para definição do grau de similaridade entre os formulários. Este projeto propõe ainda um método de busca por campos dos formulários web, internamente chamados de *elementos* de formulários. A Figura 1 mostra um exemplo de formulário web no domínio de veículos. Cada atributo (elemento) de um formulário geralmente possui um rótulo e uma série de valores possíveis associados a ele, como por exemplo, "*Make"* e "*Model"*.

Buscas no WF-Sim ocorrem sobre elementos de formulários indexados. As estruturas de índice são definidas a partir de clusters gerados por um processo de *matching* de elementos de formulários, permitindo recuperação de formulários com elementos similares. Estas estruturas de índice foram devidamente projetadas para facilitar as consultas típicas por formulários web.

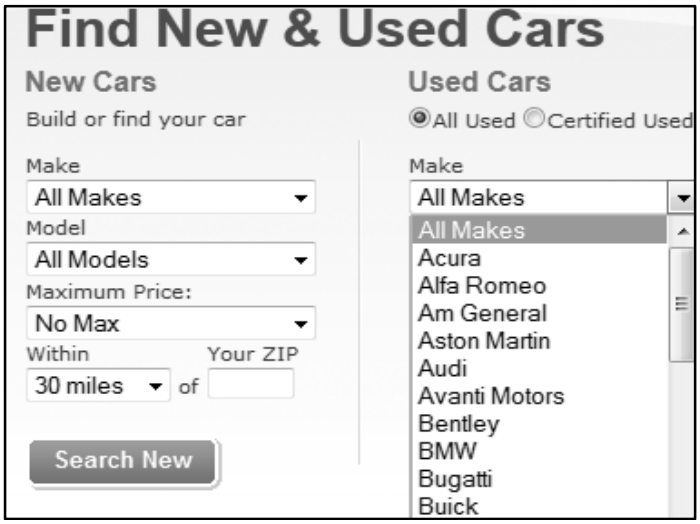

Figura 1. Exemplo de Formulário Web

Este artigo apresenta a estratégia de indexação por similaridade para formulários web desenvolvida para o WF-Sim, visando o acesso a dados mantidos em dois tipos de mecanismos de persistência: arquivos e banco de dados. Avalia-se aqui não apenas o desempenho das estruturas de índice para cada tipo de persistência, mas também quais estruturas de índice apresentaram melhor desempenho.

As demais seções detalham o desenvolvimento deste trabalho. A seção 2 aborda o projeto WF-Sim, com foco na atividade de indexação. A seção 3 detalha o módulo de indexação e as estruturas de índice propostas. A seção 4 descreve os experimentos realizados e a seção 5 é dedicada à conclusão.

[2]*http://brightplanet.com/wp-content/uploads/2012/03/12550176481-deepwebwhitepaper1.pdf*

2. WF-Sim

As técnicas de busca por formulários web atualmente utilizadas geralmente são baseadas no modelo de busca por palavra chave, que executa o *matching* entre a entrada com termos existentes nos formulários. O principal exemplo é a máquina de busca *DeepPeep*[3]. Visando adicionar capacidade de busca por similaridade a formulários web, o projeto WF-Sim, desenvolvido na Universidade Federal de Santa Catarina pelo grupo de Banco de dados (GBD/UFSC)[4], com financiamento do CNPq, se inspira no fato de que os dados disponíveis na *Deep Web* são relevantes para usuários que desejam encontrar formulários web que satisfaçam suas necessidades em termos de serviços online para os mais diversos fins.

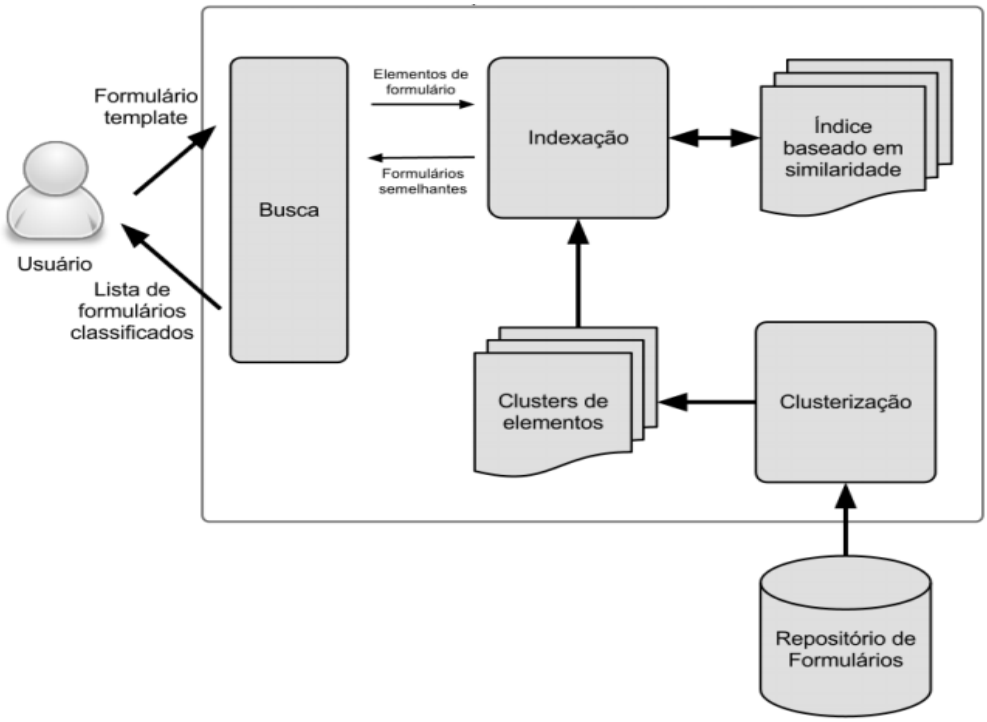

Figura 2. Arquitetura do Sistema WF-Sim

O WF-Sim é um processador de busca por similaridade para formulários web baseados nos seus elementos. A Figura 2 mostra a arquitetura do sistema. Com base em uma consulta de entrada, no caso,um conjunto de elementos denominado *formulário template*, o sistema retorna um conjunto de formulários web que possuem elementos similares. O sistema possui os módulos de busca, indexação e clusterização. O componente de clusterização foi alvo de um trabalho anterior [Silva & Mello 2012], estando, assim, fora do escopo deste artigo. Neste componente são aplicadas métricas de similaridade de modo a agrupar os formulários em clusters com elementos similares.

O módulo de indexação é o foco deste artigo e visa criar uma estrutura de índice, garantir o acesso a estas estruturas e a recuperar os formulários relevantes. Dado um *formulário template* de entrada, a busca por elementos similares acessa um dicionário de sinônimos que direciona elementos do *template* a elementos sinônimos ditos elementos centroides, ou seja, elementos representativos de um cluster de elementos

[3] *http://www.deeppeep.org/*
[4] *http://www.gbd.inf.ufsc.br*

similares. Um exemplo seria um cluster formado pelos elementos *"Make"*, *"Brand"*, *"Select a Make"*, sendo *"Make"* eleito como centróide. A próxima seção descreve o módulo de indexação.

3. Módulo de Indexação

Três estruturas de índice foram definidas para o acesso a dados de formulários Web: *palavra-chave*, *contexto* e *metadado*. A idéia destas estruturas foi obtida de um trabalho anterior do GBD/UFSC [Mello et. al. 2010] e adaptada à problemática de busca por similaridade em formulários Web. Estas estruturas, em particular os índices de contexto e metadado, são adequadas aos tipos de consulta mais frequentes sobre formulários web.

Consultas por contexto associam um rótulo aos seus respectivos valores presentes em um formulário. A idéia é recuperar formulários com base na contextualização de campos por domínio, ou seja, permite que o usuário recupere formulários com determinado tipo de conteúdo de campo que lhe interessa. Já consultas por metadado permitem indicar se um termo de interesse é um metadado do tipo rótulo ou valor.

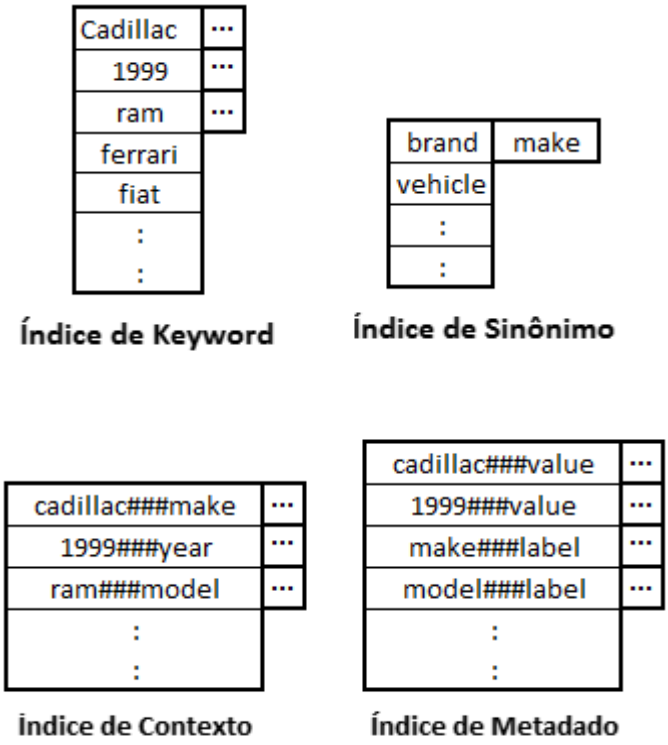

Figura 3. Estrutura dos Índices Propostos

A Figura 3 mostra exemplos dos tipos de índice desenvolvidos. O índice de palavra-chave (*keyword*) possui entradas tradicionais de termos para cada possível valor ou rótulo existente. O índice de contexto define entradas de índice para as combinações de rótulos com seus respectivos valores presentes em elementos, no formato *nome_valor###nome_rótulo*. O índice de metadado classifica o tipo de informação que se deseja recuperar(rótulo ou valor), com entradas no formato *nome_valor###VALUE* ou *nome_valor###LABEL*. Priorizou-se os termos cuja variação de conteúdo é maior (nomes de valores) no início da entrada do índice, para tornar a ordenação do índice mais adequada para fins de busca. Esta ordenação é relevante principalmente para

buscas por desigualdade ou por *range*, que requerem uma navegação seqüencial em parte da estrutura do índice.

O índice de sinônimos é uma estrutura auxiliar ao funcionamento dos índices de contexto, metadado e palavra-chave. No momento da indexação de um rótulo de um elemento de formulário Web, é feita uma consulta a um banco de dados de sinônimos para verificar se o rótulo é um sinônimo de um centróide. Em caso positivo, é adicionado ao índice de sinônimos uma entrada com o rótulo em questão e sua associação para o centróide sinônimo que está indexado nas outras estruturas de índice. Desta forma, quando o rótulo informado no *template* não estiver explicitamente indexado nas estruturas propostas, mas for um sinônimo de um centróide, é possível encontrar formulários similares através da procura por sinônimos, garantindo, assim, uma busca por similaridade. O índice de sinônimo é fundamental na estratégia de indexação proposta, visto permite que os índices de palavra-chave, contexto e metadado indexem apenas os rótulos dos elementos centróides de cada cluster, viabilizando, estruturas de índice com número reduzido de entradas.

3.1 Limpeza de Dados

O método de indexação é responsável também pela execução de procedimentos de limpeza nos dados para melhorar a indexação dos mesmos. Para tanto, esse módulo possui um analisador que faz a uniformização dos termos, passando as letras para o formato minúsculo e removendo variações indesejadas através do processo de *stemming* e remoção de *stop words*. Esses procedimentos garantem que campos de formulários representando uma mesma informação, mas com grafias diferentes, possam ser indexados de maneira uniforme. A remoção de *stop words* reduz o tamanho dos índices, pois evita a criação de entradas nos índices para termos existentes nos rótulos que não sejam relevantes para consultas. O processo de *stemming* também reduz a quantidade de entradas, pois apenas o radical dos termos dos elementos desejáveis dos rótulos é indexado. Um exemplo é mostrado na Figura 4. Um elemento com rótulo “Do ano” e outro elemento com rótulo “Ano” sem o processo de limpeza seriam indexados em posições diferentes no índice. Com a inclusão do analisador, os campos são indexados uma única vez, com letras minúsculas sem a *stop word* “Do”.

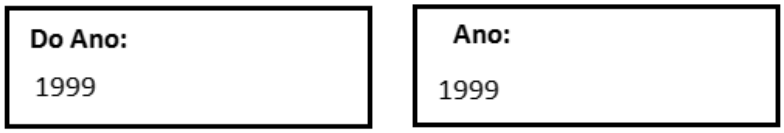

Figura 4. Exemplo de rótulos extraídos de formulários

O analisador também é utilizado no processo de busca por formulários web. Neste caso, os rótulos dos elementos do *template* passam igualmente por um processo de limpeza. Após, os termos resultantes são verificados nos índices.

3.2 Acesso aos Índices

O acesso a formulários web relevantes através dos índices se baseia no tradicional modelo booleano de recuperação de informação [Yates & Neto 1999]. Esse modelo possibilita a definição de filtros utilizando os operadores lógicos AND, OR e NOT.

Para ilustrar a sua utilização, suponha que um usuário necessita encontrar formulários que possuam veículos da marca (*brand*) GM e rótulo ano (*year*), ou seja, um formulário *template* com esses 2 elementos[5]. O módulo de Busca gera então a seguinte consulta: *GM###brand AND year###LABEL*. Cada um dos filtros gerados é então passado para o módulo de Indexação. Considerando o filtro *"GM###brand"*, o módulo de Indexação o caracteriza como sendo um filtro de contexto (formado por 2 termos – rótulo e um possível valor para ele), verifica a necessidade de limpeza de dados para o rótulo e então acessa o índice de sinônimos. Supondo que o termo *"brand"* tenha apenas um (1) centróide como sinônimo (*"make"*, por exemplo), o filtro é convertido para *"GM###make"* e a entrada correspondente a este filtro e então acessada no índice de contexto. Caso haja mais de um termo sinônimo para *"brand"*, as entradas do índice correspondentes a todos os sinônimos são acessadas e é feita uma união das URLs dos formulários web presentes em cada entrada. O mesmo raciocínio se aplica ao filtro *"year##LABEL"* e o conjunto de formulários web resultante de cada filtro retornado pelo módulo de Indexação é processado posteriormente pelo módulo de Busca conforme os operadores lógicos definidos na consulta.

3.3. Implementação

A ferramenta Lucene[6] foi utilizada para a implementação das estruturas de indexação propostas [Hatcher & Gospodnetic 2005]. Lucene é uma biblioteca de software para a recuperação de informação, sendo responsável pela indexação e da informação indexada. Essa ferramenta possibilita a criação de índices invertidos, abordagem típica para recuperação de informação na web [Yates & Neto 1999], [Elmasri & Shamkant 2011]. O índice invertido é uma estrutura de dados que mapeia um conteúdo para os documentos que o contém.

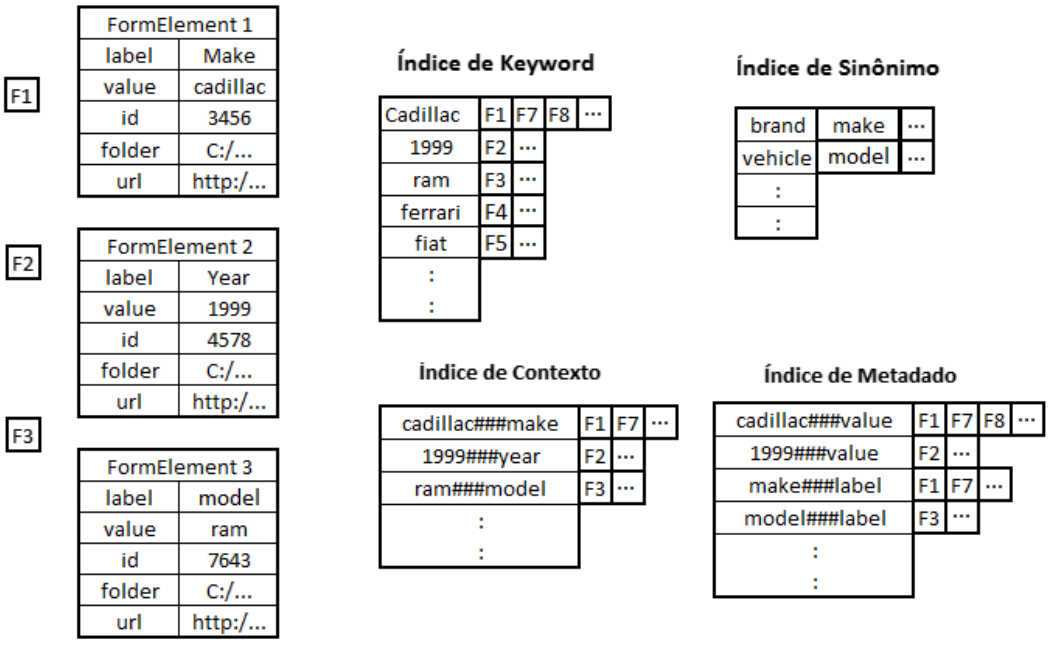

Figura 5. Informações sobre elementos considerados nos índices

[5] Maiores detalhes sobre como o módulo de busca processa templates e gera filtros de consulta para o módulo de indexação estão fora do escopo deste artigo.

[6] *http://lucene.apache.org/core/*

No contexto deste trabalho, os índices invertidos mapeiam um termo, como por exemplo, um valor de um campo, para os formulários que contenham esse determinado termo. Como pode ser visto na Figura 5, o WF-Sim persiste um elemento de formulário com as seguintes propriedades: rótulo, valores que lhe podem ser atribuídos, endereço do formulário web original (URL) e um identificador junto ao banco de dados. Neste caso, os índices definidos no Lucene apontam para a localização destes elementos persistidos em arquivos ou banco de dados.

Para a criação de um banco de dados de sinônimos foi adotada a ferramenta *WordNet*[7] é um banco de dados léxico bastante utilizado como um dicionário de suporte para análise de textos e aplicações envolvendo inteligência artificial. O *WordNet* agrupa palavras da língua inglesa em conjuntos de sinônimos e outros tipos de relações semânticas, além de fornecer definições gerais das mesmas.

Nesse trabalho foi utilizada a biblioteca *Java WordNet Interface (JWI)*[8], que fornece acesso ao banco de dados de informações do *WordNet*. Sua utilização foi necessária para determinar a similaridade entre elementos e construir o índice de sinônimos.

4. Experimentos

A implementação do módulo de indexação foi avaliada através de experimentos preliminares com o objetivo de medir o desempenho do processamento de filtros de consulta acessando as estruturas de índice desenvolvidas. Foram testados índices para elementos de formulários persistidos em arquivos de dados serializados em disco e elementos de formulários persistidos em um banco de dados relacional.

Para os experimentos foi utilizado um computador equipado com um processador Athlon II X2 de 2.9 GHz, memória de 4 GB, disco rígido de 500 GB, sistema operacional Windows 7 Home Basic SP1 de 64 bits e banco de dados MySQL. A amostra de formulários Web disponível para teste compreende 1090 formulários que possuem na totalidade 6157 elementos.

A Figura 6 mostra a média geral de tempo de execução de um mesmo conjunto de filtros de consulta, específico para cada tipo de índice, acessando arquivos em disco e o banco de dados. Já a Figura 7 mostra resultados de experimentos com uma quantidade incremental de filtros no domínio de veículos, ou seja, sobre um, dois e cinco elementos, visando o acesso ao índice de contexto. O índice de contexto é o índice mais acessado na prática, pois indexa filtros de consulta mais usuais no contexto de formulários web.

Os testes mostrados na Figura 6 registraram uma grande diferença de desempenho de acesso ao banco de dados frente aos arquivos serializados, ou seja, as consultas gastaram muito mais tempo (8x mais lento, em média) no acesso aos arquivos. Essa superioridade se deve ao fato de que o acesso a arquivos em disco não conta com as otimizações características dos sistemas gerenciadores de banco de dados. Além disso, não se considerou testes com os dados totalmente na *cache* do MySQL, pois a intenção

[7] http://wordnet.princeton.edu/

[8] http://projects.csail.mit.edu/jwi/

é lidar futuramente um volume bem maior de dados (o volume de dados da *Deep Web* é realmente muito grande!) e isso seria inviável de ser mantido na íntegra em uma *cache*.

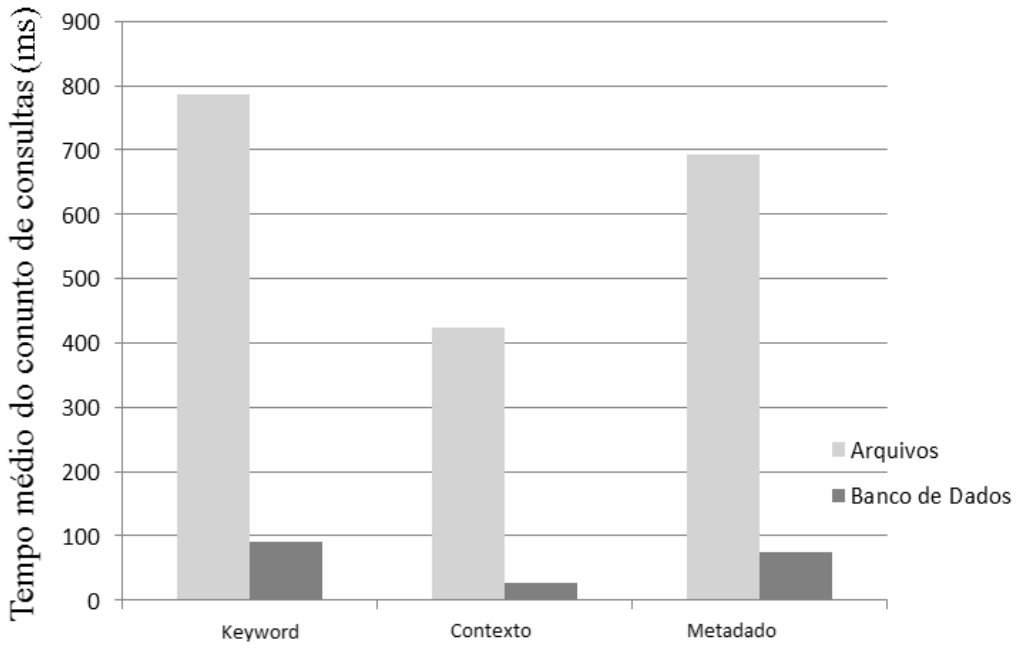

Figura 6. Média dos tempos de processamento dos filtros de consulta para um conjunto de testes

Já com relação aos testes mostrados na Figura 7, percebe-se, para o acesso ao banco de dados, que o tempo de processamento dobrou a cada variação de quantidade de filtros conjuntivos, enquanto que o aumento de tempo foi bem menor no caso do acesso aos arquivos, inclusive com redução na passagem de 2 para 5 filtros. Essa situação precisa ser melhor avaliada com uma bateria mais ampla de testes sobre volumes maiores de dados. Mesmo assim, a diferença de tempo no acesso a arquivos e banco de dados continuou sendo bastante acentuada, sendo o acesso ao banco de dados de 10x a 30x (aproximadamente) mais rápido para filtros conjuntivos.

CONTEXT	Qtde	Arquivos Tempo	Banco de Dados Tempo
jeep###make	355	381 ms	11 ms
corolla###model	155	118 ms	10 ms
jeep###make AND corolla###model	86	486 ms	19 ms
jeep###make OR corolla###model	424	484 ms	21 ms
jeep###make AND NOT corolla###model	269	484 ms	20 ms
acura###make AND audi###make AND chevrolet###make AND 1000###price AND maverick###vehicle"	0	459 ms	36 ms
acura###make OR audi###make OR bentley###make OR 1964###year OR 1999###year	464	490 ms	49 ms
acura###make OR audi###make AND bentley###make OR 1964###year AND NOT 1999###year	355	488 ms	53 ms

Figura 7. Conjunto de filtros de consulta submetidos ao índice de contexto

Uma importante contribuição deste trabalho, no contexto de consultas sobre formulários web, foram os menores tempos de busca para as consultas sobre os índices de contexto e metadados, se comparados com os tradicionais índices por palavras-chave. Este melhor desempenho está associado ao fato de que os dados de elementos passam, durante a construção destes índices, por um processo que gera automaticamente filtros por contexto e por metadado que ficam armazenados diretamente nestes índices. Esses tipos de estruturas são muito adequadas para consultas posteriores sobre formulários Web visto que garantem um mapeamento direto do filtro para uma entrada nestes índices. O uso do índice de sinônimo, visando garantir buscas por similaridade, contribuiu para uma redução no número de entradas nesses dois índices, uma vez que somente um dos termos de um conjunto de sinônimos é indexado, permanecendo os demais apenas no índice de sinônimos. Desta forma, diminuiu-se o tamanho das estruturas e o *overhead* de acesso.

5. Conclusão

Este trabalho apresenta e valida uma estratégia de indexação visando buscas por similaridade para dados de formulários Web no contexto do projeto WF-Sim. Esta estratégia introduz mecanismos de refinamento para o contexto de formulários web. Esses mecanismos consideram a indexação de informações sobre elementos de formulários em estruturas de índice especificadas por contexto, metadado, além da tradicional busca por palavra-chave. As duas primeiras estruturas garantem otimizações para o módulo de busca uma vez que as estruturas já indexam os filtros de consultas mais freqüentes para formulários. Buscas tradicionais considerando apenas palavras-chave teriam que, no caso de uma busca por contexto, por exemplo, recuperar informações primeiro sobre o rótulo, depois sobre o valor desejado, e após computar uma intersecção dos formulários recuperados. Este *overhead* é desnecessário com a introdução do índice de contexto, e o mesmo vale para o índice de metadado.

O trabalho de [Mello et. al. 2010] foi a base escolhida para a construção das estruturas de índice aqui apresentadas. [Mello et. al. 2010] define índices de contexto e de metadado. Entretanto, o escopo do trabalho não é específico para o contexto de buscas por similaridade em formulários web, que é o objetivo do projeto WF-Sim. Este artigo aplica e estende essas idéias para o propósito do projeto. Nenhum outro trabalho relacionado na literatura se propõe a definir estruturas de índice para buscas por similaridade sobre formulários web.

O próximo passo no contexto deste trabalho é avaliar o desempenho do módulo de indexação com um repositório maior de formulários web. A amostra de testes do projeto conta com um número aproximado de 1090 formulários. Essa base é composta de dados públicos de formulários oriundos de *sites* de serviços diversos. Eles foram coletados para a utilização no projeto WF-Sim, não estando disponível ainda ao público. Futuramente, com a disponibilização do WF-Sim como uma aplicação web, esses dados serão passiveis de consulta. Através de uma parceria do GBD/UFSC com a Universidade de Utah, uma amostra de aproximadamente 40000 formulários está sendo disponibilizada para novos experimentos. Uma vez que a natureza dos dados coletados nessas fontes de dados é da língua inglesa, o *WordNet* cumpre seu papel de maneira satisfatória. Entretanto, levando em consideração futuras adições de dados na língua portuguesa (formulários web de sites nacionais), será necessário utilizar dicionários de suporte para o Português.

Outra atividade futura é considerar consultas que testam dependências entre elementos de formulários Web, como por exemplo, um campo *"Make"* cujos valores determinam os valores de um campo *"Model"*, supondo um domínio de veículos. A intenção é considerar filtros que testem a existência de tais dependências e definir estruturas de indexação que facilitem buscas por similaridade neste contexto. Percebe-se que este tipo de consulta é relevante para casos em que o usuário deseja acessar formulários que implementam automaticamente uma cadeia de dependências entre campos, facilitando, assim, a sua intenção de busca por alguma informação.

Referências

Baeza-Yates, R.; Ribeiro-Neto, R. Modern Information Retrieval. (1999). ACM Press / Addison-Wesley.

Elmasri, R.; Shamkant B. (2011). "Sistemas de Banco de Dados"; tradução Daniel Vieira, revisão técnica Enzo Seraphim e Thatyana de Faria Piola Seraphim; 6ª ed. São Paulo: Pearson Addison Wesley, 2011.

Gonçalves, R. et. al. (2011). "A Similarity Search Approach for *Web forms*". In: Proceedings of the IADIS International Conference IADIS WWW/Internet.

Hatcher, E.; Gospodnetic, O. (2005)."Lucene in Action".Greenwich: Manning Publications Co, 2005.

Madhavan, J.et al. (2009) "Harnessing the Deep Web: Present and Future". In:4th Biennial Conference on Innovative Data Systems Research (CIDR 2009).

Mello, R.S., Pinnamaneni, R., Freire, J., (2010) Indexing Web Form Constraints.,In: Journal of Information and Data Management (JIDM), Vol. 5, nº. 3, p.348-358.

Silva, F. R.; Mello, R. S. (2012). "Estratégias de Persistência de Clusters em uma Técnica de Casamento por Similaridade para Web Forms". In: VIII Escola Regional de Banco de Dados (ERBD 2012).