Um mecanismo automático para detectar versões de objetos XML provenientes de bibliotecas digitais

Eduardo Nunes Borges Renata de Matos Galante

Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970
Porto Alegre – RS – Brazil

+55 51 84080044 +55 51 81258008

{enborges, galante} @inf.ufrgs.br

RESUMO

Artigos científicos indexados por diferentes bibliotecas digitais podem estar representados de diferentes formas. Os metadados que descrevem estes artigos são heterogêneos. Visando melhorar a qualidade da pesquisa do usuário de bibliotecas digitais, fornecendo uma resposta precisa e livre de redundância, este trabalho apresenta um mecanismo que realiza a detecção automática de versões de objetos XML provenientes de diferentes bibliotecas digitais. São propostas duas funções de similaridade: nomesIni e simNomes. Essas funções comparam os nomes dos autores, que junto ao título do artigo, possuem papel fundamental na identificação de versões. Além disso, os autores são metadados que possuem inúmeras representações distintas. Adicionalmente, algoritmos clássicos são usados para calcular a similaridade entre os demais metadados. Por fim, foram realizados experimentos visando a validação da detecção de versões que demonstraram a eficácia das funções propostas tanto na detecção (através de medidas de precisão e revocação), quanto no tempo de processamento.

Categorias e Descritores de Assunto

H.3.7 [**Information Storage and Retrieval**]: Digital Libraries – systems issues.

Termos Gerais

Algorithms, Measurement, Experimentation.

Palavras-chave

XML, Similaridade, Bibliotecas Digitais, Versões.

1. INTRODUÇÃO

As bibliotecas digitais são compostas por coleções de objetos digitais, como, por exemplo, documentos e imagens, incluindo um catálogo de metadados cuja função é descrever, organizar e especificar a forma como esses objetos podem ser manipulados e recuperados [10]. O catálogo de metadados é geralmente armazenado junto com o repositório da biblioteca digital, fornecendo facilidades para o acesso e gerenciamento dos objetos digitais. Idealmente, cada objeto de uma biblioteca digital deveria possuir um registro correspondente no catálogo de metadados, sendo que esses registros deveriam apresentar uma estrutura específica definida por um esquema. Entretanto, a representação dos metadados é diferente para cada biblioteca digital, sendo que nem todos os objetos armazenados possuem um registro correspondente no catálogo de metadados.

Uma característica determinante dos metadados é a descrição de informações relacionadas a alguma fonte específica. Por exemplo, o Dublin Core [6] define um padrão para o armazenamento de informações a respeito artigos científicos, periódicos e páginas *Web*. Atualmente, muitos usuários realizam buscas na *Web* sobre informações de artigos científicos, os quais estão dispostos em várias bibliotecas digitais como Digital Bibliography & Library Project (DBLP) [7], ACM Digital Library [1], IEEE Computer Society Digital Library [11], Biblioteca Digital Brasileira de Computação (BDBComp) [13], entre outras. Embora alguns padrões tenham sido propostos [6], não há um consenso na utilização dos mesmos.

Quando um usuário submete uma consulta a várias bibliotecas digitais, a recuperação da informação requisitada pode ser facilitada pela replicação dos objetos na *Web*. Entretanto, para um mesmo artigo científico, referenciado por várias bibliotecas digitais, a representação desta referência é diferente para cada sistema. O nome dos autores de determinado artigo, por exemplo, pode estar armazenado de diversas formas. Os metadados associados ao autor do artigo são diferentes em cada biblioteca digital. Neste contexto, é necessário um processo de identificação destas múltiplas representações do mesmo objeto compartilhado, ou seja, do mesmo artigo científico indexado por diversas bibliotecas digitais.

Diversos problemas de heterogeneidade são encontrados em ambientes de integração de bibliotecas digitais. A representação da referência a um determinado objeto é diferente para cada biblioteca digital. Os metadados associados aos artigos científicos, freqüentemente representados através do formato XML, são heterogêneos tanto em estrutura quanto em terminologia. Ainda podem existir diferentes representações de conteúdo e variações na codificação de caracteres utilizada na descrição dos metadados.

Em grande parte das bibliotecas digitais, os metadados são armazenados no formato XML. Outros formatos como BibTex [19], por exemplo, podem ser facilmente convertidos para o formato XML. Nos últimos anos, vários algoritmos de similaridade entre documentos XML foram propostos [8, 9, 12, 14]. No entanto, a análise semântica da similaridade entre dois documentos XML – de modo a identificá-los como versões de um mesmo documento ou instâncias diferentes – ainda é um problema em aberto pela comunidade científica. Portanto, são necessárias contribuições específicas para o domínio das bibliotecas digitais, que facilitem as consultas aplicadas sobre diferentes bibliotecas.

Para o suporte a integração e consulta em diversas bibliotecas digitais é necessário retornar ao usuário uma resposta única, sem perda de informação relevante das diversas fontes de dados – bibliotecas digitais – envolvidas na consulta. Além disso, esta resposta deve ser livre de redundância. Dentro deste contexto, o objetivo deste artigo é apresentar um mecanismo para detectar ou identificar versões de objetos XML provenientes de diferentes bibliotecas digitais de forma automática, sem a intervenção do usuário. São propostas duas funções de similaridade aplicadas ao domínio das bibliotecas digitais: *nomesIni* e *simNomes*. Essas funções comparam os nomes dos autores, considerando o aspecto semântico relacionado às abreviaturas, que junto ao título do artigo, possuem papel fundamental na identificação de versões. Além disso, as funções propostas buscam reduzir o tempo do processo de comparação dos metadados. Para o cálculo da similaridade dos demais metadados, são utilizados algoritmos clássicos [5]. Foram realizados experimentos visando a validação da detecção de versões que demonstraram a eficácia do algoritmo tanto na detecção (através de medidas de precisão e revocação), quanto no tempo de processamento.

A principal contribuição deste trabalho é melhorar a qualidade da pesquisa do usuário de bibliotecas digitais, fornecendo uma resposta precisa e livre de redundância para consultas a metadados de publicações científicas obtidos através da *Web*.

O restante do texto está organizado da seguinte forma. Na seção 2 são apresentados os trabalhos relacionados incluindo técnicas de similaridade aplicadas a documentos XML. A seção 3 define o ambiente de detecção e consulta a objetos XML oriundos de bibliotecas digitais. Na seção 4 é descrito o mecanismo utilizado na detecção de versões. As funções de similaridade propostas, bem como os experimentos, são apresentados, respectivamente, nas seções 5 e 6. Por fim, na seção 7 são comentados as conclusões e os trabalhos futuros.

2. TRABALHOS RELACIONADOS

A *Collection of Computer Science Bibliographies* (CCSB) [20] é uma biblioteca digital formada por uma coleção de bibliografias na área da ciência da computação. Ela integra literatura de várias fontes, incluindo outras bibliotecas digitais como a BDBComp e a DBLP. Atualmente, conta com mais de 2,3 GBytes de dados, totalizando mais de dois milhões de referências a artigos científicos e relatórios técnicos. Os metadados são coletados e armazenados no formato BibTex [19]. Um dos serviços suportados pela CCSB é a identificação de duplicatas, ou seja, duas ou mais referências que apontam para o mesmo artigo científico. Esta identificação é realizada de uma forma muito simples. São consideradas duplicatas artigos que contenham o mesmo título e os mesmos últimos nomes de autores. Não são considerados outros metadados disponíveis como a data de publicação, a conferência ou o livro onde o artigo foi publicado. Além disso, são exibidas todas as versões de metadados (uma para cada fonte) nas respostas às consultas realizadas pelo usuário, o que polui a interface do sistema com a redundância dos metadados armazenados.

Além do BibTex, o formato XML vem sendo bastante utilizado na representação de metadados de objetos digitais [7, 13]. O processo de detecção de versões de documentos XML envolve a comparação destes documentos e a análise da similaridade entre os mesmos. Algumas técnicas de similaridade entre documentos XML utilizam algoritmos de detecção de diferenças entre documentos XML [16, 17], também conhecidos como algoritmos *diff* [4]. Grande parte destes algoritmos retorna um *script* (*delta*) contendo as operações básicas necessárias para transformar um documento em outro. O conjunto de diferenças fornecido como resposta destes algoritmos não é suficiente para a análise semântica da similaridade entre os documentos analisados. É necessário um coeficiente de similaridade entre os documentos XML que possa distinguir as versões das instâncias diferentes. Para calcular a similaridade, algumas propostas consideram somente o conteúdo textual dos documentos [2]. Toda informação estrutural é descartada. Outras abordagens mais recentes analisam também a estrutura dos documentos [8, 9, 12, 14].

Apesar de várias abordagens terem sido propostas, a análise semântica do grau de similaridade entre dois documentos XML – de modo a identificá-los como versões de um mesmo documento ou instâncias diferentes – ainda é um problema em aberto pela comunidade científica. Além disso, a detecção de versões de metadados referentes a artigos científicos provenientes de diferentes bibliotecas digitais é um problema pouco abordado. O diferencial da nossa proposta é a especificação de um mecanismo para detectar versões de metadados que descrevem publicações científicas, a fim de prover ao usuário uma resposta simples, única, sem perda de informação relevante e livre de redundância para consultas realizadas a diversas bibliotecas digitais.

3. O AMBIENTE DE IDENTIFICAÇÃO E CONSULTA DE VERSÕES

Esta seção apresenta uma visão geral do mecanismo proposto para detectar versões de objetos XML provenientes de bibliotecas digitais. Considere um exemplo em que um usuário submete uma consulta por nomes de autores: "Edleno Silva de Moura, Altigran Soares da Silva" para as bibliotecas digitais BDBComp e DBLP. A Figura 1 mostra os metadados de um artigo recuperado nesta consulta provenientes, respectivamente, de cada uma das bibliotecas digitais acima citadas. Os elementos *creator* (linhas 03-04) e *date* (linha 05) presentes nos metadados da BDBComp correspondem, respectivamente, aos elementos *author* (linhas 10-11) e *year* (linha 14) na DBLP. As representações, apesar de diferentes, fazem referência à mesma informação. Ainda são identificados outros problemas como diferentes representações de conteúdo e variações na codificação de caracteres. O metadado *title* assume o valor "Detecção de `Sítios Replicados` Utilizando Conteúdo e Estrutura" na BDBComp (linha 02), enquanto "Detecção de `Réplicas` Utilizando Conteúdo e Estrutura" na DBLP (linha 12).

Para que seja retornada ao usuário uma resposta única, livre de redundância e sem perda de informação relevante, torna-se necessário um mecanismo de detecção destas versões. A Figura 2 mostra a arquitetura do ambiente de detecção de versões proposto em trabalho prévio [3]. Este ambiente é composto de três módulos principais:

- Identificação – responsável por coletar metadados XML referentes a sistemas de integração de bibliotecas digitais. Esta coleta é realizada através do protocolo OAI-PMH [21] ou de outros artifícios que as bibliotecas digitais disponibilizem. Além disso, possui a função de detectar automaticamente versões dos objetos XML referentes aos artigos científicos através de técnicas de similaridade, considerando o conteúdo e a estrutura.

BDBComp

```
01 <oaidc:dc>
02     <title>Detec&#231;&#227;o de Sitios Replicados Utilizando
           Conte&#250;do e Estrutura</title>
03     <creator>Edleno Silva de Moura</creator>
04     <creator>Altigran Soares da Silva</creator>
05     <date>2005</date>
06     <identifier>http://www.sbbd-sbes2005.ufu.br/arquivos/artigo-
           02-novo_Carvalho.pdf</identifier>
07     <language>por</language>
08 </oaidc:dc>
```

DBLP

```
09 <inproceedings>
10     <author>Edleno Silva de Moura</author>
11     <author>Altigran Soares da Silva</author>
12     <title>Detec&amp;ccedil;&amp;atilde;o de
           R&amp;eacute;plicas Utilizando Conte&amp;uacute;do
           e Estrutura.</title>
13     <pages>25-39</pages>
14     <year>2005</year>
15     <booktitle>SBBD</booktitle>
16     <ee>http://www.sbbd-sbes2005.ufu.br/arquivos/artigo-02-
           novo_Carvalho.pdf</ee>
17 </inproceedings>
```

Figura 1. Heterogeneidade de metadados.

- Proveniência – responsável por rastrear as informações de proveniência dos atributos dos objetos XML. Este módulo inclui o módulo Armazenamento, que especifica um modelo de versões para o armazenamento das versões dos objetos XML. São armazenadas informações sobre a origem dos metadados, ou seja, as bibliotecas digitais de onde foram coletados. O módulo de proveniência gerencia, portanto, o armazenamento das anotações que descrevem a proveniência de dados e os resultados do processamento da detecção de versões[1];
- Consulta – permite ao usuário final realizar consultas recuperando uma resposta única para os artigos científicos pesquisados (versões integradas, ou seja, a totalidade de informação relevante contida nas versões).

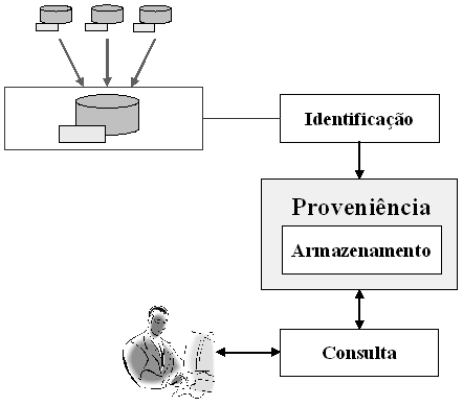

Figura 2. Arquitetura do ambiente.

[1] A especificação do modelo de proveniência, bem como resultados mais concretos sobre este módulo serão apresentados em trabalhos futuros, pois o foco deste artigo é na detecção de versões.

O processo de coleta dos metadados e da identificação de versões ocorre de maneira incremental. Através do protocolo OAI-PMH, é possível recuperar somente os novos metadados disponibilizados pelas bibliotecas digitais a partir de certa data. A cada período de tempo *t*, o módulo *Identificação* compara estes novos metadados a fim de identificar novas versões. Estas versões de metadados de um mesmo artigo indexado por diferentes bibliotecas digitais, previamente detectadas pelo módulo *Identificação*, são armazenadas através de um modelo de versões pelo módulo *Armazenamento*.

Quando um usuário submete uma consulta ao sistema, o módulo *Consulta* busca as informações sobre os artigos científicos no módulo *Armazenamento*. Estas informações são então exibidas ao usuário por uma interface de conexão. A Figura 3 apresenta os resultados da mesma consulta apresentada no início desta seção, aplicada ao ambiente de identificação de versões. Este resultado integra as informações relevantes de ambas as versões além de eliminar dados redundantes. As próximas seções apresentam como os requisitos aqui identificados para a detecção de versões são alcançados e os experimentos realizados que comprovam a aplicabilidade da proposta.

BDBComp + DBLP

```
01 <metadata>
02     <title>Detecção de Réplicas Utilizando Conteúdo e
           Estrutura</title>
03     <author>Edleno Silva de Moura</author>
04     <author>Altigran Soares da Silva</author>
05     <year>2005</year>
06     <text>http://www.sbbd-sbes2005.ufu.br/arquivos/artigo-02-
           novo_Carvalho.pdf</text>
07     <language>Portuguese</language>
08     <booktitle>SBBD</booktitle>
09     <pages>25-39</pages>
10 </metadata>
```

Figura 3. Integração de versões.

4. DETECÇÃO DE VERSÕES

Esta seção descreve em detalhes a técnica implementada pelo módulo *Identificação*, ou seja, o mecanismo utilizado na detecção de versões. Após coletar os metadados XML que descrevem os artigos científicos, estes metadados são comparados, considerando o seu conteúdo e a sua estrutura, a fim de identificar as versões de um mesmo artigo.

A Figura 4 mostra um exemplo de detecção de versões. Os objetos XML representam os metadados referentes a determinadas publicações em duas bibliotecas digitais *A* e *B*. Inicialmente todos os metadados descrevem publicações distintas. É realizado o casamento entre os esquemas das bibliotecas digitais e então os metadados são comparados utilizando as métricas de similaridade definidas na seção 5. Durante o processo de detecção de versões, é identificado que os objetos *1* e *4*, pertencentes a biblioteca digital *A*, representam respectivamente, os objetos *7* e *8* de *B*. Ao final do processo de identificação, os identificadores dos artigos são atualizados, de modo que um identificador da biblioteca digital é adicionado ao identificador da publicação. Dados oito objetos XML, são detectados seis artigos científicos distintos.

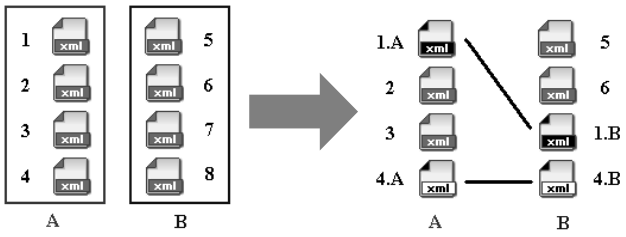

Figura 4. Identificação de versões de objetos XML.

4.1 Casamento de esquemas

Para realizar o processo de identificar versões de artigos científicos é necessário mapear os diferentes metadados provenientes de cada biblioteca digital que compõe o ambiente. Esta tarefa é realizada em parte através do casamento entre os diferentes esquemas dos metadados. Atualmente, o casamento de esquemas é realizado de forma semi-automática [15]. Técnicas de casamento de esquemas fornecem uma lista de casamentos candidatos e as partes do esquema que não possuem correspondência, sendo necessária a intervenção do usuário especialista na etapa final do casamento.

Na abordagem proposta neste trabalho, o casamento entre esquemas XML é realizado através da correspondência direta entre os rótulos dos metadados como mostra a Tabela 1, sem que seja necessária a intervenção do usuário. Esta escolha deve-se ao fato de que o módulo de identificação proposto é específico para o domínio das bibliotecas digitais e tem como escopo metadados relativos a artigos científicos, os quais são representados freqüentemente pelos formatos OAI Dublin Core [6], BibTex [19] ou o esquema utilizado pela DBLP.

Após a etapa de casamento, é necessário comparar cada par de referências disponível nos arquivos XML, utilizando técnicas de similaridade de conteúdo, no intuito de identificar as diferentes representações de um mesmo documento (artigo científico publicado).

Tabela 1. Correspondência entre formatos de metadados

OAI Dublin Core	DBLP	BibTeX
title	title	title
description		abstract
date	year	year
source	booktitle	booktitle
language		language
coverage		location
rights		copyright
identifier	ee	URL
	pages	pages
creator	author	author

5. *SimNomes* - O ALGORITMO DE SIMILARIDADE PROPOSTO

Métricas de similaridade tradicionais [5], além de possuírem alto custo computacional, não expressam significativamente a similaridade de nomes próprios. Portanto, foram propostas duas métricas de similaridade aplicadas ao domínio de bibliotecas digitais, denominadas *nomesIni* e *simNomes*. Elas comparam os nomes dos autores, que junto ao título de um artigo, possuem papel fundamental na identificação de versões. Adicionalmente, algoritmos de similaridade clássicos, como a distância de edição normalizada, são utilizados para calcular a similaridade entre demais metadados.

Considere um conjunto $I_S = \{x \in I \mid x = 0 \vee x = 1\}$ como o conjunto dos valores inteiros em I_S no intervalo [0,1], e um conjunto $P = \{y \mid y = [A\text{-}Z]^*\}$ como o conjunto de todas as palavras formadas por quaisquer caracteres alfabéticos. A função de similaridade $nomesIni : P \rightarrow I_S$ recebe como parâmetro um par de palavras $a, b \in P$, as quais representam as iniciais dos nomes de autores, e gera um valor de similaridade $s \in I_S$. A Figura 5 define a função de similaridade *nomesIni*, onde a_i é a i-ésima letra da palavra a, b_i é i-ésima letra da palavra b, m é o tamanho da palavra a, e n é o tamanho da palavra b. Quando as iniciais correspondem ao mesmo autor, ou seja, quando as duas representações podem expressar o mesmo objeto do mundo real, a função *nomesIni* retorna um valor de similaridade $s = 1$ (um). Caso contrário, é retornado o valor $s = 0$ (zero).

$$nomesIni\ (a, b) = \begin{cases} 1, & (a_1 = b_1 \wedge a_m = b_n) \vee \\ & (a_1 = b_2 \wedge a_m = b_1) \vee \\ & (a_1 = b_1 \wedge a_2 = b_2) \vee \\ & (a_1 = b_n \wedge a_2 = b_1) \\ 0, & \text{caso contrário} \end{cases}$$

Figura 5. Definição da função *nomesIni*.

As condições impostas pela função partem do princípio que o nome de um autor pode estar representado de diversas maneiras como mostra a Tabela 2.

Tabela 2. Possíveis representações de Eduardo Nunes Borges

Nome	Iniciais
Eduardo N. Borges	ENB
E. Borjes	EB
Borges, Edward	BE
Borjes, E. Nuñes	BEN

Considere um conjunto $R_S = \{x \in R \mid x \geq 0 \wedge x \leq 1\}$ como o conjunto de todos os valores reais em R_S no intervalo [0,1]. A função de similaridade $simNomes : \{R_S, P\} \rightarrow I_S$ recebe como parâmetro duas listas de palavras K e L que representam cada uma a lista de iniciais dos nomes dos autores de um determinado artigo – onde cada elemento das listas K_i , $L_i \in P$. Além das listas, a função recebe um terceiro parâmetro real que é um valor de limiar (*threshold*) de similaridade $t \in R_S$. Este limiar corresponde ao

limiar mínimo de casamento entre as iniciais dos autores e tem como objetivo deixar a função flexível a ajustes, variando no intervalo fechado [0,1]. A função *simNomes* gera um valor de similaridade $s \in I_S$.

A Figura 6 define a função de similaridade *simNomes* através de um algoritmo, onde *i* é a i-ésima palavra da lista *K*, *j* é a j-ésima palavra da lista *L*, *m* é o tamanho da lista *K*, e *n* é o tamanho da lista *L*. *max(i, j)* é uma função que retorna o tamanho da maior lista de palavras. As duas listas, *K* e *L*, são percorridas (linhas 08-09) a fim de encontrar o casamento entre as palavras, o qual é realizado através da função *nomesIni* definida anteriormente. Quando ocorre o casamento entre duas palavras (linha 10), é atribuído um valor *Nulo* a uma palavra da lista *L* para que esta não seja utilizada em futuras comparações. Quando o limiar de casamento mínimo é atingido (linha 17), o algoritmo retorna *s = 1* (um), caso contrário retorna *s = 0* (zero).

```
01  Algoritmo simNomes
02  Entradas: K, L, t
03  Saída: s
04  Variáveis: m, n, i, j, contador: inteiro
05  Início
06      m ← tamanho (K)
07      n ← tamanho (L)
08      Para i de 1 até m faça
09          Para j de 1 até n faça
10              Se nomesIni (K_i , L_j) = 1
11                  contador ← contador + 1
12                  L_j ← Nulo
13                  Pare
14              Fim Se
15          Fim Para
16      Fim Para
17      Se contador / max (i , j) < t
18        Retorne s = 0
19      Senão
20        Retorne s = 1
21  Fim
```

Figura 6. Definição do algoritmo *simNomes*.

Por exemplo, *simNomes((ENB, NI, OCM), (IFN, BE, CO), 1.0)* retorna *1*, pois *ENB* corresponde a *BE*, *NI* corresponde a *IFN* e *OCM* corresponde a *CO*. Nesse caso, foi atendido o limiar de 100% de casamentos. Já *simNomes((ENB, NI, OCM), (IFN, BE, CYQ), 0.75)* retorna *0*, pois *ENB* corresponde a *BE*, *NI* corresponde a *IFN*, *OCM* e *CYQ* não possuem correspondência. Portanto, somente 66,6% das ocorrências casaram, não atingindo o limiar mínimo de 75% passado como parâmetro. O limiar de similaridade de 75% adotado, neste caso, faz com que um artigo possível candidato ao casamento fique fora da comparação do restante dos metadados. Quando ocorrem erros no processo automático de aquisição dos dados pelas bibliotecas digitais, ou seja, quando a lista de autores de um determinado artigo não está completa, o limiar passado como parâmetro ajusta a função para que ocorra a detecção da versão. Definições adequadas de valores de limiar são discutidas na literatura [18] e não fazem parte do escopo do trabalho apresentado neste artigo. Além disso, outros mecanismos como algoritmos de classificação podem ser utilizados para realizar o casamento das instâncias evitando o uso de *threshold* [17].

O algoritmo de similaridade *simNomes* proposto tende a obter a revocação (*recall*) máxima, ou seja, entre os casamentos de artigos recuperados estarão todos os casamentos relevantes. A precisão (*precision*) é mínima, visto que o número de casamentos recuperados é muito maior do que o número de casamentos relevantes. A precisão atinge valores aceitáveis quando as instâncias que satisfizerem as condições impostas por esta métrica de similaridade são avaliadas na totalidade. Esta avaliação total é realizada com qualquer outra função de similaridade [5] aplicada sobre outros metadados além dos nomes dos autores. Metadados como título, conferência e data de publicação só serão comparados para os pares de artigos em que a função *simNomes* não retorne zero. Isto reduz o tempo de processamento no processo de identificação de versões de metadados, pois o algoritmo proposto limita as comparações do restante dos metadados.

6. EXPERIMENTOS

Esta seção apresenta os resultados do experimento que demonstra que as funções propostas apresentam bons resultados, tanto em termos de precisão/revocação quanto em tempo de processamento. Os objetivos do experimento são os seguintes: (i) avaliar o mecanismo de detecção de versões proposto, através de medidas de precisão e revocação; e (ii) avaliar eficácia do algoritmo *simNomes* através do tempo de transação da detecção de versões. As bases de dados utilizadas foram a totalidade de metadados disponíveis pela BDBComp e DBLP. Foram utilizados os parâmetros – *thresholds* – de 100 e 75% para as funções *simNomes* e *levenshtein*, respectivamente. O experimento foi executado em um computador Pentium IV HT 3.0 GHz com 1 GB de memória DDR2. Através da função *simNomes* demonstrou-se ser possível reduzir em 94% o tempo do cálculo da similaridade entre os títulos dos artigos. Além disso, os testes mostram que a identificação das versões ocorre com 99,53 e 90,63% de precisão e revocação, respectivamente.

A BDBComp possui cerca de 4 mil referências para artigos científicos publicados no Brasil. Os metadados que descrevem os artigos científicos são disponibilizados através do protocolo OAI-PMH [21], no padrão Dublin Core. Foi realizada a coleta dos metadados (*metadata harvesting*) os quais foram agrupados em um único arquivo XML de 3,63 MB denominado *bdbcomp.xml*. Já a DBLP conta com mais de 800 mil referências para artigos científicos publicados em diversos países. Os metadados são disponibilizados no *web site* principal da biblioteca digital em um único arquivo XML de 350 MB denominado *dblp.xml*. Também é fornecida uma DTD contendo um simples esquema.

Foi realizado um pré-processamento dos arquivos XML com o objetivo de carregar em um banco de dados relacional somente os metadados utilizados pelo mecanismo proposto. Um software desenvolvido em JAVA realiza o *parsing* dos arquivos XML e obtém como saída um script SQL gerado para popular uma base de dados do SGBD PostgreSQL. A base de teste é composta por duas tabelas principais, uma para cada biblioteca digital. O esquema da relação *bdbcomp* é especificado na Figura 7, onde *id* (linha 02) corresponde ao identificador único de cada artigo. Este campo é gerado através de um contador no *parser* do software JAVA, o qual é incrementado a cada elemento *oaidc:dc* encontrado em *bdbcomp.xml* (Figura 1). O atributo *title* (linha 02) corresponde ao metadado *title*. Uma lista composta pelos

elementos *creator* corresponde ao campo *authors* (linha 02) da tabela. *initilas* (linha 3) é formado pela lista das iniciais dos nomes dos autores, que é gerada a partir do atributo *authors*. Os campos *abstract*, *year*, *fultext*, *language*, *location* e *rigths* (linhas 03-04) correspondem, respectivamente, aos metadados *description*, *date*, *identifier*, *language*, *coverage* e *rights* da BDBComp. Por fim, o atributo *booktitle* (linha 04) é formado através do metadado *source* com a omissão do ano de publicação. A tabela *dblp* foi criada de maneira análoga.

```
01  CREATE TABLE bdbcomp (
02  id integer NOT NULL, title text, authors text,
03  initials text, abstract text, "year" smallint, fulltext text,
04  "language" text, booktitle text, location text, rights text );

05  ALTER TABLE ONLY bdbcomp
06      ADD CONSTRAINT bdbcomp_pkey
07          PRIMARY KEY (id);

08  CREATE INDEX bdbcomp_year ON bdbcomp
09      USING btree ("year");
```

Figura 7. Definição da tabela bdbcomp.

Além do *parser*, foram implementadas três funções de similaridade, as quais foram anexadas ao SGBD PostgreSQL. As funções *nomesIni* e *simNomes*, apresentadas na seção 5, e a função *levenshtein* (*normalized edit distance*) [5]. Após a execução do script SQL gerado, as tabelas *bdbcomp* e *dblp* contavam, respectivamente, com 3.976 e 529.202 instâncias. A Figura 8 mostra a consulta realizada sobre a base de dados a qual identifica os pares de instâncias (uma de cada biblioteca digital) que representam versões de uma mesma publicação.

```
01  SELECT b.id, d.id
02  FROM dblp_full d, bdbcomp_full b
03  WHERE b.year = d.year
04  AND simNomes(b.initials, d.initials, 1) = 1
05  AND levenshtein (b.title, d.title) >= 0.75
06  ORDER BY b.id, d.id;
```

Figura 8. Consulta que identifica as versões.

As duas tabelas são percorridas e ordenadas em função dos atributos *year*, em aproximadamente 2,1 minutos. Então é realizado a junção (*merge join*) das tabelas com a condição dos atributos *year* serem iguais e são aplicadas as funções *simNomes* e *levenshtein*. A consulta retorna 861 pares *b.id*, *d.id*. Por fim, são ordenados os resultados em função dos identificadores. O tempo total de transação é de 8 minutos.

A função *simNomes* limita a aplicação da função *levenshtein*, que não precisa ser executada para as tuplas onde a função *simNomes* retornar zero. Para realizar uma consulta semelhante a da Figura 8, apenas com a diferença de omitir-se a linha 04, o tempo de transação sobe para 2,3 horas. Portanto, a função *simNomes* reduz em 94,2% o tempo do cálculo da similaridade entre os títulos dos artigos.

Foram detectados 861 pares de metadados que podem corresponder a versões do mesmo artigo científico, utilizando um limiar de 75% de similaridade para a função *levenshtein*. Dentre os 861 pares detectados, apenas 4 não são versões do mesmo artigo científico, ou seja, não são pares relevantes para a consulta. Portanto, a precisão da consulta é de 99,53%. Mas o total de versões disponibilizadas pelas bibliotecas digitais é de 950. Portanto, a revocação foi de 90,63%.

7. CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho propõe um mecanismo para detectar ou identificar versões de objetos XML provenientes de diferentes bibliotecas digitais de forma automática, sem a intervenção do usuário. São propostas duas funções de similaridade específicas para o domínio das bibliotecas digitais: *nomesIni* e *simNomes*. Elas comparam os nomes dos autores, que compõem um dos metadados mais importantes na identificação de versões. Experimentos realizados indicam bons resultados em relação à qualidade da identificação de versões, tanto em termos de precisão/revocação quanto em tempo de processamento.

A principal contribuição deste trabalho é melhorar a qualidade da pesquisa do usuário de bibliotecas digitais, fornecendo uma resposta simples, única, sem perda de informação relevante e livre de redundância para consultas realizadas a diversas bibliotecas digitais.

Com o aumento significativo de bibliotecas digitais públicas, torna-se necessário determinar a qualidade dos dados publicados. Um histórico detalhado dos dados permite aos usuários avaliar se estes dados são aceitáveis e confiáveis. Portanto, como trabalhos futuros destacam-se: (i) permitir ao usuário o rastreio da proveniência dos metadados XML oriundos de diferentes bibliotecas digitais; (ii) elaborar um modelo de proveniência de dados capaz de identificar a fonte e o processo de modificação dos metadados.

8. AGRADECIMENTOS

Este trabalho é parcialmente financiado por CNPq (processo 481516/2004-2), FAPERGS (processos 0408933 e 0412264), e DIGITEX – CTInfo (processo 550.845/2005-4).

9. REFERÊNCIAS

[1] Association for Computing Machinery (ACM) Digital Library. http://portal.acm.org/dl.cfm, July, 2007.

[2] Baeza-Yates R. and Ribeiro-Neto, B. *Modern Information Retrieval*, ACM Press Series/Addison Wesley, New York, 1999.

[3] Borges, E. N. and Galante , R. M. Um mecanismo para identificação, representação e consulta de versões de objetos XML oriundos de bibliotecas digitais. *In: Workshop de Teses e Dissertações - Simpósio Brasileiro de Bancos de Dados*, 2007, João Pessoa.

[4] Cobena, G., Abdessalem, T. and Hinnach, Y. *A comparative study of XML diff tools*. Verso report, INRIA, 2004.

[5] Cohen, W., Ravikumar, P. and Fienberg S. A Comparison of String Distance Metrics for Name-Matching Tasks in *IIWeb 2003: 73-78*.

[6] DCMI Metadata Terms, http://dublincore.org/documents/dcmi-terms, July, 2007.

[7] Digital Bibliography & Library Project (DBLP), http://dblp.uni-trier.de, July, 2007.

[8] Dorneles, C., Heuser, C, Lima, A, Silva A. and Moura, E. Measuring similarity between collection of values. *WIDM 2004*, p. 56-63.

[9] Flesca, S., Manco, G., Masciari, E., Pontieri, L. and Pugliese, A. Fast detection of XML structural similarity, *Knowledge and Data Engineering, IEEE Transactions* on, vol. 17, no. 2, p. 160- 175, 2005.

[10] Fox, E. A., Akscyn, R. M., Furuta, R. K. and Leggett, J. J. Digital Libraries. *Communications of the ACM*, *38(4)*, pp. 22-28, April, 1995.

[11] Institute of Electrical and Electronics Engineers (IEEE) Computer Society Digital Library. http://www.computer.org/portal/site/csdl. July, 2007.

[12] Joshi, S., Agrawal, N., Krishnapuram, R. and Negi, S. A bag of paths model for measuring structural similarity in Web documents, *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, August, 2003, p. 24-27, Washington, D.C.

[13] Laender, A., Gonçalves, M. and Roberto, P. BDBComp: Building a Digital Library for the Brazilian Computer Science Community. *In: Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, Tuscon, AZ, USA, 2004, pp. 23-24.

[14] Nierman, A., and Jagadish, H. Evaluating Structural Similarity in XML Documents. In: *Int'l Workshop on the Web and Databases (WebDB)*, Madison, WI, Jun, 2002.

[15] Rahm, E. and Bernstein, P. A survey of approaches to automatic schema matching, *Very Large Database J.*, 2001, 10(4):334–350.

[16] Saccol, D. B., Edelweiss, N. and Galante, R. M. Detecting, Managing and Querying Replicas and Versions in a Peer-to-Peer Environment. In: The First IEEE TCSC Doctoral Symposium, in conjunction with the 7th IEEE International Symposium on Cluster Computing and the Grid (CCGrid), Rio de Janeiro, Brazil. *Proceedings of CCGrid 2007 and IEEE Digital Library*, 2007. Pages: 881-886.

[17] Saccol, D. B., Edelweiss, N., Galante, R. M. and Zaniolo, C. XML Version Detection. In: ACM Symposium on Document Engineering, Winnipeg, Canada. *Proceedings of the 2007 ACM Symposium on Document Engineering (to appear)*, 2007. (p 79-88).

[18] Stasiu, R., Heuser, C. and Silva, R. Estimating Recall and Precision for Vague Queries in Databases. *CAiSE 2005*: 187-200.

[19] The BibTex Format. http://www.ecst.csuchico.edu/~jacobsd/bib/formats/bibtex.html, July, 2007.

[20] The Collection of Computer Science Bibliographies. http://liinwww.ira.uka.de/bibliography/, July, 2007.

[21] Van de Sompel, H, Nelson, M., Lagoze, C. and Warner, S. Resource Harvesting within the OAI-PMH Framework, *D-Lib Magazine*, December 2004, 10(12).