UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

EDUARDO NUNES BORGES

MD-PROM: um Mecanismo de Deduplicação de Metadados e Rastreio da Proveniência

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Profa. Dra. Renata de Matos Galante
Orientadora

Porto Alegre, abril de 2008

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Borges, Eduardo Nunes

MD-PROM: um Mecanismo de Deduplicação de Metadados e Rastreio da Proveniência / Eduardo Nunes Borges. – Porto Alegre: PPGC da UFRGS, 2008.

73 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2008. Orientadora: Renata de Matos Galante.

1. Bibliotecas Digitais. 2. Metadados. 3. Similaridade. 4. Proveniência de Dados. 5. Deduplicação. I. Galante, Renata de Matos. II. Título.

“O desenvolvimento da capacidade geral de pensamento e livre-arbítrio sempre deveria ser colocado em primeiro lugar, e não a aquisição de conhecimento especializado. Se uma pessoa domina o fundamental no seu campo de estudo e aprendeu a pensar e a trabalhar livremente, ela certamente encontrará o seu caminho e será mais capaz de adaptar-se ao progresso e às mudanças.”

— ALBERT EINSTEIN

AGRADECIMENTOS

Inicialmente eu agradeço Ione Nunes, que além de minha mãe, é meu pai, minha mentora, meu orgulho e minha maior mestra. Seus ensinamentos me possibilitaram trilhar este caminho. A distância que nos separou me fez ver ainda mais que posso contar sempre com ela. Muito obrigado pelo apoio e incentivo, além das correções gramaticais realizadas neste trabalho.

À minha noiva Cristina Opazo pela ajuda, amor, conforto, carinho, companheirismo e muita paciência no decorrer destes 14 anos de convívio.

Aos meus colegas de laboratório Adrovane Kade, Alexander Vinson, André Geraldo, Cláudio Fuzitaki, Deise Saccol, Eduardo Piveta, Euler de Oliveira, Gabriel Simões, Gustavo Piltcher, Giseli Lopes, Marcos Nunes, Mariusa Warpechowski, Maurício Dias, Otávio Acosta, Sérgio Mergen e Thiago Alves. Eles foram ótimos companheiros de pesquisa, parceiros de churrasco e de cerveja. Em especial, ao Gabriel pela companhia nas disciplinas e viagens e a Alexander e Giseli pelas discussões sobre o trabalho, contribuições científicas e ajuda na execução dos experimentos. Agradeço também pelo envolvimento de Fabrício Andreis e Denise Giacomolli, responsáveis pela implementação de parte do MD-PROM.

Aos meus amigos Caetano Almeida e Carlos Cony, pelas discussões, frustrações, medos, confraternizações, muitas gargalhadas, companhia para atividades diversas e grande amizade desde o início da Graduação.

Aos professores Álvaro Moreira, Carlos Heuser, José Palazzo, Leandro Wives, Mirella Moro, Renata Galante e Viviane Orengo pela instrução, idéias e discussões no grupo de pesquisa. Em especial ao professor Palazzo pelo auxílio financeiro dos projetos dos quais é coordenador e à professora Viviane pela ajuda na definição dos experimentos e métodos para avaliá-los.

Agradecimentos especiais à professora Renata Galante, pela orientação, dedicação e confiança depositada em mim, fundamentais no desenvolvimento deste trabalho, pelo conhecimento e experiência transmitidos, viagens realizadas e pela oportunidade de cursar o mestrado.

Por fim, agradeço à UFRGS e ao Instituto de Informática pela infra-estrutura disponibilizada. Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e à Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul (FAPERGS) pelo apoio financeiro nos dois anos de curso, sem o qual o desenvolvimento deste trabalho com dedicação exclusiva seria impossível.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

ACM	Association for Computing Machinery
BDBComp	Biblioteca Digital Brasileira de Computação
CAiSE	Conference on Advanced Information Systems Engineering
CCSB	Collection of Computer Science Bibliographies
DBLP	Digital Bibliography & Library Project
DC	Dublin Core
DCMI	Dublin Core Metadata Initiative
DOI	Digital Object Identifier
DTD	Document Type Descriptor
EM	Expectation Maximization
ERBD	Escola Regional de Bancos de Dados
ESSW	Earth System Science Workbench
IDF	Inverse Document Frequency
IDO	Integrated Digital Object
IEEE	Institute of Electrical and Electronics Engineers
INTERACT	IFIP Conference on Human-Computer Interaction
MAV	Métricas para Valores Atômicos
MARLIN	Multiply Adaptative Record Linkage with Induction
MCV	Métricas para Valores Complexos
MD-PROM	Metadata Deduplication and Provenance Tracing Mechanism
OAI-PMH	Open Archives Initiative - Protocol for Metadata Harvesting
OCR	Reconhecimento Ótico de Caracteres
SAX	Simple API for XML
SBBD	Simpósio Brasileiro de Banco de Dados
SGBD	Sistema Gerenciador de Bancos de Dados
SIBGRAPI	Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens

SQL	Structured Query Language
SVM	Support Vector Machine
SVR	Symposium on Virtual and Augmented Reality
TFIDF	Term Frequency - Inverse Document Frequency
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WebMedia	Brazilian Symposium on Multimedia and the Web
WDL	Workshop on Digital Libraries
WTDBD	Workshop de Teses e Dissertações em Bancos de Dados
XML	Extensible Markup Language

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Bibliotecas digitais são repositórios de objetos digitais que oferecem serviços aos seus usuários como pesquisa e publicação desses objetos. Cada objeto digital é descrito por um conjunto de metadados que especifica a forma como esse objeto pode ser recuperado. Sistemas de integração de bibliotecas digitais indexam objetos digitais adquiridos de fontes diferentes, os quais podem estar representados através de vários padrões de metadados. Estes metadados são heterogêneos tanto em conteúdo quanto em estrutura. Conseqüentemente, os sistemas de integração de bibliotecas digitais não estão aptos a fornecer respostas livres de informação redundante que integrem as várias fontes de dados.

Quando um usuário realiza uma consulta sobre várias bibliotecas digitais, é interessante que sejam retornados metadados integrados das diversas fontes e a origem de cada informação recuperada, ou seja, a biblioteca digital que publicou aquela informação (metadado). O uso de proveniência de dados nas consultas a metadados em sistemas de integração de bibliotecas digitais, de modo a rastrear a origem das informações recuperadas, permite que usuários avaliem a qualidade das bibliotecas digitais.

Este trabalho apresenta o MD-PROM (*Metadata Deduplication and PROvenance tracing Mechanism*), um mecanismo de deduplicação de metadados e rastreio da proveniência. Este mecanismo identifica metadados de objetos digitais duplicados em bibliotecas digitais distintas, integra os metadados duplicados e recupera informações de proveniência dos metadados integrados.

A identificação de duplicatas é realizada através do casamento automático de esquemas dos metadados e da aplicação de funções de similaridade sobre os principais metadados que descrevem os objetos digitais. São propostas a função de similaridade de nomes próprios *IniSim*, o algoritmo de casamento de autores *NameMatch* e o algoritmo de casamento de objetos digitais *Digital Object Match* que identifica múltiplas representações dos metadados. Além dos algoritmos de similaridade, o MD-PROM especifica uma estrutura baseada em árvore para representar a proveniência de dados que identifica a origem dos metadados, bem como os valores dos quais os metadados foram derivados. Também é proposto um algoritmo de integração de metadados e rastreio da proveniência denominado *MetadataProv*.

A principal contribuição do trabalho é melhorar a qualidade da pesquisa do usuário de bibliotecas digitais. O MD-PROM fornece uma resposta única, livre de redundância e sem perda de informação relevante para consultas a metadados de objetos digitais oriundos de bibliotecas digitais distintas. Além disso, são recuperadas informações de proveniência que permitem ao usuário verificar a veracidade e confiabilidade dos metadados retornados pelas consultas em sistemas de integração de bibliotecas digitais. São apresentados também os resultados de diversos experimentos que avaliam a qualidade da deduplicação de objetos digitais comparando a técnica proposta com outras abordagens estudadas.

Palavras-chave: Bibliotecas Digitais, Metadados, Similaridade, Proveniência de Dados, Deduplicação.

MD-PROM: a mechanism for metadata deduplication and provenance tracing

ABSTRACT

Digital libraries are repositories of digital objects that provide services to their users such as search and publication of these objects. Each digital object is described by a set of metadata that specifies how this object can be retrieved. Integrated digital library systems index digital objects acquired from different sources, which can be represented through several metadata patterns. These metadata are heterogeneous both in content and in structure. Consequently, the integrated digital library systems are not able to provide answers free from redundant information that integrate the several data sources.

When a user performs a query on various digital libraries, it is interesting to return integrated metadata from several sources and the origin of each information retrieved, that is, the digital library which published that information (metadata). Using data provenance in metadata queries on integrated digital library systems, so as to trace the origin of the information retrieved, allows users to analyze the quality of digital libraries.

This work presents MD-PROM (*Metadata Deduplication and PROvenance tracing Me- chanism*), a mechanism for metadata deduplication and provenance tracing. This mechanism identifies duplicated digital objects metadata in different digital libraries, integrates duplicated metadata and retrieves provenance information of the integrated metadata.

The identification of duplicates is performed through automatic metadata schema matching and through similarity functions applied over main metadata that describe the digital objects. The surname similarity function *IniSim*, the authors matching algorithm *NameMatch* and digital objects matching algorithm *Digital Object Match*, which identifies multiple representations of metadata, have been proposed. Besides the similarity algorithms, MD-PROM specifies a tree-based structure to represent the data provenance that identifies the origin of metadata as well as the values from which the metadata were derived. An algorithm for the integration of metadata and provenance tracing, called *MetadataProv*, is also proposed.

The main contribution of this work is to improve the quality of the searches posed by the users of digital libraries. MD-PROM provides a single answer, free from redundancy and loss of relevant information related to queries on digital objects metadata from different digital libraries. In addition, provenance information is retrieved allowing the user to verify the accuracy and the reliability of the metadata returned by queries on integrated digital library systems. There are also reports on several experiments, which evaluate the quality of the deduplication of digital objects comparing the proposed technique with other approaches.

Keywords: Digital Libraries, Metadata, Similariry, Data Provenance, Deduplication.

1 INTRODUÇÃO

Bibliotecas digitais são compostas por coleções de objetos digitais, como, por exemplo, documentos, imagens, vídeos, mapas, etc. que oferecem serviços aos seus usuários como pesquisa e publicação desses objetos (FOX et al., 1995). Além dos objetos digitais, as bibliotecas digitais são compostas por um catálogo de metadados cuja função é descrever, organizar e especificar a forma como esses objetos podem ser manipulados e recuperados.

Uma característica determinante dos metadados é a descrição de informações relacionadas a alguma fonte específica. Por exemplo, o *Dublin Core* (DCMI, 2008) define um padrão para a representação, armazenamento e consulta de informações a respeito artigos científicos, periódicos e páginas *Web*. A *International DOI Foundation* define um *Digital Object Identifier* (DOI) como um identificador permanente de qualquer objeto de propriedade intelectual que, ao contrário de um *Uniform Resource Locator* (URL), não dependente da localização do objeto (WIKIPEDIA, 2007). Entretanto, não há um consenso na utilização dos padrões por todas as digitais existentes.

Considere o exemplo da Figura 1.1 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. Os elementos *creator* (linhas 03-04), *date* (linha 05) e *identifier* (linha 06) presentes nos metadados da BDBComp correspondem, respectivamente, aos elementos *author* (linhas 10-11), *year* (linha 14) e *ee* (linha 16) na DBLP. As estruturas dos metadados, apesar de diferentes, fazem referência à mesma informação. Ainda são identificados outros problemas na variação do conteúdo e da 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).

A variação na representação das informações referentes aos objetos digitais possui três principais causas: (i) os dados armazenados são digitados por diferentes usuários ou gerados por diferentes aplicações; (ii) a falta de padronização na representação de nomes de autores, referências bibliográficas, nomes de conferências e de periódicos; (iii) algumas bibliotecas digitais, como a *ACM Digital Library* (ACM, 2007), realizam o reconhecimento ótico de caracteres (OCR) em certos documentos para armazená-los digitalmente. É comum que o OCR falhe em alguns trechos dos documentos e não processe certos caracteres ou palavras.

Para o suporte à 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 envolvidas na consulta. Além disso, esta resposta deve ser livre de redundância. Para usuários avançados, é interessante que a origem das informações recuperadas seja apresentada.

BDBComp

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

DBLP

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

Figura 1.1: Heterogeneidade de metadados.

Nos últimos anos, várias abordagens para identificar registros duplicados foram propostas. Algumas propostas utilizam funções de similaridade sobre cada atributo dos registros e combinam os escores gerados pelas funções através de alguma métrica (GUHA et al., 2004; CHAUDHURI et al., 2003). Outras abordagens propõem métodos de identificação de duplicatas baseados em alguma técnica de aprendizado de máquina (DORNELES et al., 2007; CARVALHO et al., 2006; BILENKO; MOONEY, 2003; TEJADA; KNOBLOCK; MINTON, 2001). A maioria dos trabalhos é aplicada à integração de dados relacionais e à realização de consultas por similaridade em dados relacionais ou semi-estruturados. Poucas abordagens foram desenvolvidas no contexto das bibliotecas digitais. Objetos digitais têm como principais atributos os metadados que descrevem a autoria do objeto. Pode existir mais de um objeto de mesmo título com autoria diferente. Por exemplo, tanto Brioniaccyr Feverstein quanto Rick Greenwald e David Kreines publicaram livros intitulados “Oracle in a Nutshell”. Outro problema comum é a variação na representação dos nomes de autores em citações bibliográficas. Alguns exemplos destas variações são abreviações, inversões de nomes, grafias diferentes, hifenização e uso de sufixos como Júnior. Técnicas de deduplicação aplicadas ao contexto das bibliotecas digitais devem valorizar os atributos que se referem aos nomes dos autores para identificar corretamente objetos digitais duplicados. A grande maioria das abordagens estudadas não trata especificamente a similaridade de nomes próprios.

Conhecida também como linhagem de dados, a proveniência de dados é a descrição das origens de uma porção de dados e o processo pelo qual ela é obtida (BUNEMAN; KHANNA; TAN, 2001). O uso de proveniência de dados, nas consultas a metadados em sistemas de integração de bibliotecas digitais, de modo a rastrear a origem das informações recuperadas, permite que os usuários avaliem a qualidade e confiabilidade das bibliotecas digitais.

Atualmente, a pesquisa na área de proveniência de dados destina-se principalmente ao domínio da ciência. Diversos sistemas foram propostos com a finalidade de recuperar a linhagem de dados científicos (ZHAO et al., 2004; BHAGWAT et al., 2004; TAN, 2003; FOSTER et al., 2002; FREW; BOSE, 2001). Não foram encontrados na literatura trabalhos na área de bibliotecas digitais que tracem a linhagem dos metadados que descrevem uma publicação. Recuperar informações de proveniência em sistemas de integração de

bibliotecas digitais é um problema que não tem recebido muita atenção pela comunidade científica. Portanto, são necessárias contribuições específicas para o domínio das bibliotecas digitais, que identifiquem objetos digitais duplicados e recupere informações de proveniência dos metadados, facilitando as consultas aplicadas sobre bibliotecas distintas.

O objetivo deste trabalho é especificar um mecanismo que identifique metadados de objetos duplicados em sistemas de integração de bibliotecas digitais, integre os metadados duplicados, e armazene informações de proveniência de cada metadado. Consultas realizadas sobre o mecanismo recuperam os metadados integrados para cada objeto digital além das informações de origem dos objetos digitais. O mecanismo proposto, denominado *Metadata Deduplication and PROvenance tracing Mechanism* (MD-PROM), identifica objetos duplicados através do casamento automático de esquemas dos metadados e de funções de similaridade aplicadas sobre os metadados que descrevem os objetos digitais. O MD-PROM especifica uma estrutura de dados baseada em árvore que identifica a origem dos metadados, bem como os valores dos quais os metadados foram derivados. A qualidade da identificação de objetos duplicados é avaliada através de uma série de experimentos realizados os quais comparam a técnica proposta com outras abordagens estudadas.

As principais contribuições do MD-PROM são: o casamento de esquemas dos metadados sem a intervenção do usuário; a especificação da função de similaridade de nomes próprios *IniSim*; o algoritmo de casamento de autores *NameMatch*; o algoritmo de casamento de objetos digitais *Digital Object Match*; e o algoritmo de integração de metadados e rastreio da proveniência *MetadataProv*. O diferencial da proposta é a especificação de uma estrutura de dados capaz de identificar a origem de metadados do domínio das bibliotecas digitais e os valores dos quais estes metadados foram derivados.

O restante do texto está organizado da seguinte forma: o capítulo 2, Revisão Bibliográfica, apresenta o contexto no qual a dissertação está inserida, a saber, deduplicação e proveniência de dados, identificando as principais necessidades e desafios para realização de consultas em ambientes integrados de bibliotecas digitais. Cada tópico é finalizado com uma análise comparativa entre as propostas. O capítulo 3, MD-PROM, especifica detalhadamente o mecanismo proposto para deduplicar objetos digitais e recuperar a proveniência de metadados oriundos de bibliotecas digitais distintas. São apresentados os componentes do mecanismo e definido um conjunto de funções e algoritmos que especificam cada componente. O capítulo 4, Avaliação Experimental, apresenta uma série de experimentos onde a qualidade da deduplicação de objetos digitais realizada pelo MD-PROM é avaliada em relação a trabalhos relacionados. Por fim, o capítulo 5, Conclusões e Trabalhos Futuros, revisa as contribuições do trabalho realizado e apresenta possibilidades de trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

Este capítulo apresenta o contexto no qual a dissertação está inserida: deduplicação de objetos digitais e proveniência de dados. As áreas de pesquisa são apresentadas, identificando as principais necessidades e desafios visando delimitar o escopo do problema tratado. Dentro desse contexto são descritos os principais trabalhos relacionados com a solução proposta nesta dissertação. Estes trabalhos são comparados entre si e, posteriormente, com a abordagem proposta.

2.1 Deduplicação

A tarefa de identificar em um repositório de dados registros duplicados que se referem a mesma entidade do mundo real, incluindo variações de grafia e omissão de palavras, é denominada deduplicação (CARVALHO et al., 2006). Conhecida também como casamento de registros (*record linkage*), de objetos (*object matching*) ou de instâncias (*instance matching*), a deduplicação é a descoberta de registros correspondentes em uma ou mais fontes de dados.

Um dos trabalhos pioneiros na área da deduplicação foi proposto por Fellegi e Sunter (1969). Os autores formalizaram uma solução para o problema de reconhecer registros duplicados (pessoas, objetos, ou eventos idênticos) em dois arquivos de dados diferentes através de um modelo matemático. O modelo proposto exige a definição de dois valores de limiar. Uma função de similaridade, denominada *linkage rule*, é aplicada a um par de registros e o valor retornado é comparado com os dois limiares definidos anteriormente. Se o valor de similaridade for maior do que ambos os limiares, os registros são considerados duplicatas (*link*). Se o valor de similaridade for menor do que ambos os limiares, os registros são confirmados como diferentes, ou seja, não são réplicas (*non-link*). Os registros ainda podem ser classificados como possíveis casamentos (*possible link*) quando o valor de similaridade retornado pela função estiver entre os dois limiares. Neste caso, é necessária a intervenção humana para julgar a similaridade.

O trabalho de Fellegi e Sunter serviu como base para diversas alternativas propostas pela comunidade científica. De forma geral, estas alternativas podem ser classificadas sob dois aspectos principais: aquelas que apresentam propostas para combinar os valores de escores dos atributos através de alguma métrica de similaridade (CHAUDHURI et al., 2003; GUHA et al., 2004; DORNELES et al., 2004; CARVALHO; SILVA, 2003; CULOTTA; MCCALLUM, 2005); e aquelas que, além de combinar os escores, propõem métodos baseados em alguma técnica de aprendizado de máquina (TEJADA; KNOBLOCK; MINTON, 2001; COHEN; RICHMAN, 2002; BILENKO; MOONEY, 2003; BILENKO et al., 2003; DORNELES et al., 2007), conforme detalhado a seguir.

2.1.1 Deduplicação baseada na Combinação de Escores de Similaridade

Chaudhuri et al. (2003) trata os registros como vetores de palavras. É proposta uma função de similaridade que considera os pesos das palavras utilizando o método *Inverse Document Frequency* (IDF) (BAEZA-YATES; RIBEIRO-NETO, 1999). Um registro de entrada é comparado a um conjunto de registros de referência através da função de similaridade proposta. Também é proposto um índice tolerante a erros e um algoritmo probabilístico para recuperar de maneira eficiente os k registros de referência mais similares ao registro de entrada de acordo com a função de similaridade. Se mais de um candidato ao casamento for retornado, o usuário deve escolher o candidato mais próximo ao registro em questão. A técnica proposta é extensível, pois possibilita o uso de funções de pesos específicas para certos domínios de aplicação ao invés dos pesos gerados pelo método IDF.

A deduplicação é abordada por Guha et al. (2004) na combinação de *rankings* de similaridade. *Rankings* individuais são gerados para cada atributo de uma relação R, de acordo com uma consulta Q. Cada *ranking* individual é composto por um conjunto de tuplas formadas por um único atributo e são ordenados em função da similaridade com o atributo correspondente da consulta Q. É proposta uma função de fusão (*merging function*), denominada *footrule distance*, que combina os escores de cada atributo de uma tupla da relação R considerando também suas posições em cada um dos *rankings*. São selecionados os $top - k$ (CHAUDHURI; GRAVANO, 1999) registros mais relevantes. O objetivo da proposta é derivar um *ranking* final "ótimo" de tamanho k, definido pelo usuário.

Carvalho e Silva (2003) usam o modelo vetorial para calcular a similaridade entre objetos de múltiplas fontes. A abordagem pode ser utilizada para deduplicar objetos com estruturas complexas como documentos XML. É assumido que conjuntos de atributos equivalentes semanticamente são previamente identificados por uma determinada abordagem de casamento de esquemas (RAHM; BERNSTEIN, 2001). São propostas e avaliadas quatro estratégias distintas para atribuir pesos e combinar escores de similaridade para cada atributo:

1. cada vetor é composto de todos os termos de todos os atributos de um objeto. É utilizado o modelo vetorial para o cálculo da similaridade;

2. os vetores são compostos de todos os termos semanticamente equivalentes em ambos os objetos. É utilizado o modelo vetorial para o cálculo da similaridade;

3. um vetor é construído para cada atributo semanticamente equivalentes em ambos os objetos. A similaridade é calculada através do somatório das similaridades (modelo vetorial) entre cada par de vetores. O peso $\phi = 1$ é utilizado para cada atributo, ou seja, a importância de cada atributo para o cálculo da similaridade é a mesma;

4. a mesma estratégia do item anterior com a exceção de que o peso de cada atributo pode assumir um valor $\phi \geq 0$.

Os experimentos realizados pelos autores indicam que a estratégia 4 obtém melhores resultados, mas a proposta não determina automaticamente o valor dos pesos ϕ associados a cada atributo. Esta tarefa fica por conta do usuário.

Culotta e McCallum (2005) definem um modelo de deduplicação de registros que captura dependências relacionais. É proposto um algoritmo de agrupamento aglomerativo

que explora estas dependências e identifica duplicatas de registros em bases de dados relacionais a partir da identificação de outros registros duplicados relacionados.

Dorneles et al. (2004) propõe uma série de métricas de similaridade que manipulam coleções de valores que ocorrem nos documentos XML. Através da combinação destas métricas e do uso de um limiar de similaridade, é possível identificar documentos duplicados em bases de dados XML. Considerando que os nodos de uma árvore XML podem ser atômicos (possuem valores únicos como pequenas seqüências de caracteres, datas, números, etc.) ou complexos (estruturas aninhadas que contêm outros nodos) são propostos dois tipos de métricas de similaridade: métricas para valores atômicos (MAV); e métricas para valores complexos (MCV). As MAV são dependentes do domínio de aplicação. Por exemplo, uma MAV pode ser aplicada a determinados domínios como datas, preços de produtos, nomes de pessoas, cidades, etc. As MCV são definidas da seguinte forma:

- $tuple$ - utilizada em elementos complexos cujos elementos filho possuem nomes diferentes (registros);
- $collection$ - utilizada em elementos complexos cujos elementos filho possuem nomes iguais. Quando a ordem de precedência dos elementos filho é importante, é utilizada a MCV $list$. Para conjuntos determinados por valores sem ordem definida é proposta a MCV set.

A MCV set é definida pela equação 2.1 onde ϵ_p, ϵ_d são nodos complexos, n é o número de nodos filho de ϵ_p, m é o número de nodos filho de ϵ_d, $(1 \leq i \leq n)$ e $(1 \leq j \leq m)$. A função $max()$ retorna o maior escore de similaridade entre os parâmetros. Cada elemento ϵ_p^i é comparado com todos os nodos filho $[\epsilon_d^1, \ldots \epsilon_d^m] \in \epsilon_d$. É calculado o valor do somatório das máximas similaridades recuperadas e este valor é dividido pelo maior número de nodos filho, ou seja, o tamanho do maior conjunto.

$$setSim(\epsilon_p, \epsilon_d) = \frac{\sum_{\epsilon_p^i.\eta=\epsilon_d^j.\eta}(max(sim(\epsilon_p^i, [\epsilon_d^1, \ldots \epsilon_d^m])))}{max(m, n)} \tag{2.1}$$

Qualquer função de similaridade entre *strings* pode ser utilizada como uma MAV para calcular o escore de similaridade de nodos atômicos. Já para calcular o valor de similaridade entre dois nodos complexos, a estratégia é combinar os diversos valores já calculados para os nodos filho destes nodos. Se os nodos filho são atômicos, MAV são combinadas. Se os nodos filho também são nodos complexos, uma MCV é usada recursivamente.

2.1.2 Deduplicação baseada em Técnicas de Aprendizado de Máquina

Os escores retornados por uma função de similaridade dependem do algoritmo que implementa a função e não têm um significado lógico para o usuário. Os escores retornados por diferentes funções possuem distribuições diferentes. A distribuição ainda pode variar quando a mesma função é aplicada a diferentes conjuntos de dados. Dorneles et al. (2007) estendem o trabalho anterior (DORNELES et al., 2004) de forma que ao invés de definir um limiar de similaridade em termos dos escores retornados por uma função de similaridade, o usuário possa especificar a precisão esperada do processo de casamento de registros. A precisão é uma medida de qualidade conhecida que possui clara interpretação do ponto de vista do usuário. A abordagem realiza o mapeamento entre os escores de similaridade e os valores de precisão através de um conjunto de treinamento. O resultado do treinamento é uma tabela para cada função que mapeia cada escore de similaridade em um escore ajustado. Para o processo de casamento, o usuário especifica um limiar de

precisão que é substituído pelo escore correspondente de acordo com a tabela de mapeamentos obtida no treinamento.

O Sistema Active Atlas (TEJADA; KNOBLOCK; MINTON, 2001) tem como objetivo efetuar o mapeamento entre objetos a fim de integrar fontes de dados. Inicialmente, o sistema efetua o cálculo do escore de similaridade para cada atributo de um objeto através de transformações nas cadeias de caracteres e funções de similaridade específicas para o domínio dos atributos. Após, regras de mapeamento entre os atributos são especificadas a partir de um processo de treinamento. Estas regras são geradas através da aplicação de aprendizado com árvores de decisão (QUINLAN, 1986). A idéia básica é usar técnicas da recuperação de informação (similaridade textual) para fornecer um mapeamento inicial e aplicar técnicas de aprendizagem de máquina para melhorar o mapeamento. O Active Atlas exige a participação de um usuário especialista que é requisitado para verificar o mapeamento inicial de alguns pares de objetos. A partir das verificações do especialista, o sistema tenta aprender regras novas, e seleciona pares de objetos adicionais para que o especialista verifique novamente.

A deduplicação de objetos é definida por Cohen e Richman (2002) como a seguinte tarefa: a partir de duas listas de nomes de entidades provenientes de duas fontes distintas, determinar pares de nomes que se referem a mesma entidade do mundo real. Os autores exploram a similaridade textual dos objetos em diferentes bases de dados, propondo uma técnica escalável e adaptativa para agrupar esses objetos. A técnica utiliza um conjunto de treinamento e um algoritmo de aprendizado. O algoritmo realiza o casamento entre pares ou conjuntos de *strings*. A técnica proposta é comparada a outras duas abordagens que utilizam a distância de edição (LEVENSHTEIN, 1966) e *Term Frequency - Inverse Document Frequency* (TFIDF) (BAEZA-YATES; RIBEIRO-NETO, 1999). Os experimentos realizados apresentam bons resultados para a qualidade da deduplicação utilizando as medidas de avaliação precisão, revocação e medida F.

O sistema MARLIN (*Multiply Adaptative Record Linkage with Induction*) (BILENKO; MOONEY, 2003; BILENKO et al., 2003) descreve um *framework* para identificação de registros duplicados que utiliza métricas de similaridade textual adaptativas aplicadas a cada atributo. Através de um processo de treinamento, estas métricas são capazes de se adaptar de acordo com o domínio de valores de cada atributo. São definidas duas métricas de similaridade entre cadeias de caracteres:

- uma variação da distância de edição (LEVENSHTEIN, 1966), denominada *Expectation Maximization* (EM). A métrica EM avalia um escore de similaridade s entre um par (a, b) de *strings* comparando caractere a caractere;
- uma técnica de aprendizado de máquina, que utiliza o algoritmo *Support Vector Machine* (SVM) (BOSER; GUYON; VAPNIK, 1992). Baseada no modelo vetorial, a métrica calcula a similaridade estimada s_e entre um par (a, b) de *strings* comparando palavra a palavra.

As funções de similaridade usadas para cada atributo da base de dados passam por um processo de treinamento. Funções consideradas treinadas são usadas para calcular a distância de cada atributo em um par de tuplas, que são posteriormente usadas por um classificador binário que identifica cada par de tuplas como duplicada ou não duplicada. A similaridade entre tuplas compostas por múltiplos atributos é calculada através da combinação das similaridades estimadas nos atributos individuais. Durante o processo de combinação, são atribuídos pesos aos atributos de acordo com sua contribuição à real similaridade entre as tuplas.

Recentemente, foi proposta uma abordagem baseada na programação genética para deduplicar objetos digitais (CARVALHO et al., 2006). A programação genética é uma técnica de aprendizado de máquina que auxilia na solução de problemas onde o espaço de busca é muito grande e quando há mais de um objetivo a ser cumprido. A abordagem proposta é capaz de gerar funções de similaridade automaticamente para identificar registros duplicados em um dado repositório. O processo de combinar evidências para criar uma função de similaridade de registros usando programação genética é dividido em duas fases:

- treinamento - nesta fase, as características dos registros similares são aprendidas;
- teste - as melhores árvores selecionadas no conjunto de treinamento são usadas para identificar réplicas no restante dos registros.

Os experimentos realizados sobre informações de autores e citações de artigos provenientes de bibliotecas digitais mostram que a abordagem proposta produz melhores resultados que os métodos propostos por Fellegi e Sunter (1969). Apesar do número de duplicatas identificadas corretamente ser quase o mesmo dos métodos de Fellegi e Sunter, pouquíssimos objetos foram identificados incorretamente, gerando índices de precisão significativamente maiores.

A deduplicação de citações bibliográficas também é abordada em (LAWRENCE; GILES; BOLLACKER, 1999), onde são propostos algoritmos de casamento de citações baseados na distância de edição (LEVENSHTEIN, 1966) e no casamento de palavras e frases.

2.2 Comparação entre as Propostas de Deduplicação

Esta seção apresenta uma análise comparativa entre as abordagens que tratam o problema deduplicação de objetos apresentadas na seção 2.1. Primeiramente, são apresentados os critérios utilizados para comparação. Em seguida, é feita a análise de cada critério, que é resumida na Tabela 2.1.

Os seguintes critérios foram utilizados para realizar a comparação dos algoritmos:

- combinação - indica como ocorre o processo de combinação dos valores de escores gerados para os atributos de um objeto. Alguns trabalhos propõem o uso de métricas de similaridade, enquanto outros propõem métodos que utilizam técnicas de aprendizado de máquina;
- função - mostra qual a função (ou conjunto de funções) de similaridade é usada para comparar os atributos de um objeto;
- área - indica em que contexto o trabalho é proposto (área de aplicação);
- atributos (A) - indica quais atributos são utilizados durante o casamento de objetos, ou seja, quais os atributos são comparados durante o processo. Podem haver duas alternativas: atributos comuns (C) aos dois objetos comparados ou todos (T) os atributos;
- pesos - algumas abordagens utilizam pesos para indicar a importância de cada atributo no processo de deduplicação, enquanto outras abordagens consideram que todos os atributos possuem a mesma importância;

- nomes - mostra quais das alternativas utilizam funções ou técnicas de similaridade específicas para a deduplicação de nomes próprios.

Tabela 2.1: Comparativo entre os trabalhos de deduplicação de objetos.

Trabalho	**Combinação**	**Função**	**Área**	**A**	**Pesos**	**Nomes**
(CHAUDHURI et al., 2003)	algoritmo fuzzy baseado na IDF	NA	*data cleaning*	T	V	X
(GUHA et al., 2004)	combinação de *rankings*	NA	*data cleaning*	C	NA	X
(CARVALHO; SILVA, 2003)	estratégias baseadas no modelo vetorial	NA	deduplicação de documentos XML	C	V	X
(CULOTTA; MCCALLUM, 2005)	dependências relacionais	NA	integração de dados	C	V	X
(DORNELES et al., 2004)	MCV	MAV	consultas aproximadas	T	NA	X
(DORNELES et al., 2007)	MCV e escore ajustado	MAV	consultas aproximadas	T	NA	X
Active Atlas (TEJADA; KNOBLOCK; MINTON, 2001)	árvores de decisão	transformações, funções específicas do domínio	integração de dados	C	X	X
(COHEN; RICHMAN, 2002)	algoritmo de aprendizado	funções específicas do domínio	integração de dados	C	V	X
Marlin (BILENKO; MOONEY, 2003)	classificador binário	EM e SVM	consultas aproximadas	C	V	X
(CARVALHO et al., 2006)	programação genética	funções geradas automaticamente	bibliotecas digitais	T	V	V

Legenda: C comuns T todos V utiliza X não utiliza NA não aplicável

Embora o processo de combinação dos valores de escores gerados para os atributos de um objeto seja único para cada trabalho, a maioria das propostas de deduplicação de objetos analisadas são de propósito geral, ou seja, foram desenvolvidas com o objetivo de deduplicar registros em bases de dados relacionais ou documentos XML. Somente a técnica proposta por Carvalho et al. (2006) identifica objetos duplicados em sistemas de bibliotecas digitais.

Apesar das técnicas serem desenvolvidas com o mesmo objetivo final, as abordagens são aplicadas em diferentes contextos: *data cleaning*, deduplicação de documentos XML, integração de dados de fontes distintas, realização de consultas aproximadas sobre bases de dados e identificação de réplicas em bibliotecas digitais.

As abordagens propostas por Chaudhuri et al. (2003), Dorneles et al. (2007; 2004) e Carvalho et al. (2006) utilizam todos os atributos dos objetos comparados para realizar o casamento. Esta característica pode fazer com que objetos que diferem muito no número de atributos não sejam identificados como réplicas.

Técnicas que realizam o casamento aproximado utilizando todos os atributos dos objetos devem possuir algum artifício que indique a importância de cada atributo no processo de deduplicação. Os trabalhos propostos por Chaudhuri et al. (2003), Carvalho e Silva (2003), Culotta e Mccallum (2005), Cohen e Richman (2002), Bilenko e Mooney (2003) e Carvalho et al. (2006) utilizam pesos para representar esta importância.

Para deduplicar objetos digitais corretamente, é muito importante identificar ambigüidades na autoria de objetos digitais. A maioria dos trabalhos apresentados nesta seção, apesar de apresentarem soluções que deduplicam registros corretamente, não tratam especificamente da deduplicação de nomes próprios. Dentre os trabalhos analisados pode-se observar que somente a abordagem proposta por Carvalho et al. (2006) propõe o uso de uma função de similaridade específica para a comparação de nomes de autores.

Identificar variações nos nomes de autores presentes em citações bibliográficas pode ser considerado um subproblema da deduplicação. Algoritmos específicos para a comparação de nomes e técnicas para a identificação de autoria e remoção de ambigüidades em bibliotecas digitais foram propostos nos últimos anos. A seção 2.3 apresenta algumas abordagens que tratam da solução deste subproblema.

2.3 Algoritmos de Casamento de Nomes

Nomes de autores sofrem problemas de variação causados por diversos motivos. Alguns exemplos são abreviações (Carlos Alberto Heuser ou Carlos A. Heuser), permutações (Renata Galante ou Galante, Renata), grafias diferentes (Borges ou Borjes), hifenização (Ribeiro-Neto ou Ribeiro Neto), composição de nomes (El-Masri ou Elmasri), uso de prefixos e sufixos (Sr., Jr., números, etc.), além do uso de letras maiúsculas e minúsculas (OLIVEIRA; LAENDER; GONÇALVES, 2005).

Christen (2006) sumariza uma série de algoritmos de similaridade entre palavras que são classificados de acordo com a técnica utilizada: baseado em fonemas, caracteres, ou híbridos. Algoritmos baseados em fonemas (Soundex, Phonex, Double-Metaphone, Fuzzy Soundex, etc.) e híbridos (Editex, *Syllable Alignment Distance*) são dependentes do idioma de origem dos nomes próprios comparados. Algoritmos baseados em caracteres (*Edit Distance*, *Bag Distance*, Smith-Waterman, *Longest Common Sub-string*, Q-Grams, Jaro, Jaro-Winkler) não dependem de idioma, mas podem não identificar corretamente alguns problemas de variação como abreviações e grafias diferentes.

Mais recentemente, outras abordagens foram propostas para remover a ambigüidade entre os nomes próprios identificando a autoria de objetos digitais em sistemas de bibliotecas digitais (SONG et al., 2007; COTA; GONÇALVES; LAENDER, 2007; MALIN, 2005; HAN; ZHA; GILES, 2005; HAN et al., 2004). Estas abordagens fazem uso de algoritmos de agrupamento complexos e técnicas de aprendizado que utilizam outras informações dos objetos, tais como, co-autores, título e nome da conferência ou periódico em que foi publicado.

O restante desta seção apresenta três algoritmos desenvolvidos especificamente para o casamento de nomes que podem ser utilizados com a finalidade de identificar corretamente a autoria de publicações digitais. A escolha destes algoritmos foi realizada porque todos são independentes de idioma e identificam grafias diferentes. Os algoritmos são comparados entre si e, posteriormente, com a abordagem proposta nesta dissertação para a deduplicação de nomes de autores.

2.3.1 Guth

O algoritmo $Guth$ (GUTH, 1976; LAIT; RANDELL, 1993) utiliza comparações baseadas em caracteres e por esse motivo possui a vantagem de ser independente de um determinado idioma. Considere um conjunto $P = \{x \mid x = [a-z|A-Z]*\}$ como o conjunto de todas as palavras formadas por quaisquer caracteres alfabéticos. O algoritmo recebe como parâmetro um par de palavras $a, b \in P$, as quais representam nomes próprios, e produz como saída um valor lógico (*boolean*). Quando os parâmetros correspondem ao mesmo nome próprio, ou seja, quando as duas representações podem expressar o mesmo objeto do mundo real, o algoritmo $Guth$ retorna verdadeiro. Caso contrário, é retornado falso.

Inicialmente, o algoritmo verifica se as duas palavras a, b são idênticas. Se o teste falha, o algoritmo prossegue comparando os nomes caractere a caractere. Quando são

encontrados caracteres diferentes na mesma posição em ambas palavras, caracteres correspondentes são procurados em outras posições. O padrão de comparação é apresentado na Tabela 2.2 onde o algoritmo também é exemplificado. São passadas como parâmetros as palavras $a = EDUARDO$ e $b = EDWARD$. O algoritmo encontra a *substring* ED no início de ambas palavras (teste 1). O terceiro caractere $U \in a$ não é encontrado na mesma posição da palavra b, portanto é buscado nas posições $x + 1$, $x + 2$ e $x - 1$ (testes 2-4). A seguir são realizadas outras comparações buscando em a o terceiro caractere $W \in b$ (testes 5-7). Ainda são realizados mais duas comparações buscando a equivalência entre os caracteres (testes 8-9). Se o algoritmo falha em encontrar quaisquer dos padrões apresentados, os nomes são considerados diferentes. Caso contrário, os nomes são considerados equivalentes. Para o exemplo apresentado, o valor falso é retornado.

Tabela 2.2: Padrão de comparação e exemplo de execução do Algoritmo Guth

Teste	**Posição em** a	**Posição em** b	**Exemplo**
1	X	X	**ED**UARDO, **ED**WARD
2	X	X + 1	ED**U**ARDO, EDW**A**RD
3	X	X + 2	ED**U**ARDO, EDWA**R**D
4	X	X - 1	ED**U**ARDO, E**D**WARD
5	X - 1	X	E**D**UARDO, ED**W**ARD
6	X + 1	X	EDU**A**RDO, ED**W**ARD
7	X + 2	X	EDUA**R**DO, ED**W**ARD
8	X + 1	X + 1	EDU**A**RDO, EDW**A**RD
9	X + 2	X + 2	EDUA**R**DO, EDWA**R**D

2.3.2 Acronyms

O algoritmo $Acronyms$ (LIMA, 2002) compara palavras com o objetivo de identificar possíveis acrônimos. Um acrônimo, também conhecido por sigla, é um agrupamento de letras de várias palavras, usualmente letras iniciais, que formam uma abreviação geralmente pronunciável. São exemplos de acrônimos as seguintes palavras: OTAN (Organização do Tratado do Atlântico Norte), GNR (Guarda Nacional Republicana) e JPEG (*Joint Photographic Experts Group*).

$Acronyms$ recebe como parâmetro duas cadeias de caracteres a, b. Primeiramente, é executada uma etapa de pré-processamento que visa padronizar as cadeias de caracteres para futura comparação. As tarefas realizadas durante esta etapa são:

1. substituir caracteres indesejados (. , ; - : ') por espaços em branco;

2. substituir números por espaços em branco;

3. retirar espaços em branco no início e no fim das cadeias de caracteres.

Após a etapa de pré-processamento, o algoritmo $Acronyms$ realiza as seguintes operações. Para cada cadeia de caracteres:

4. dividir a palavra em *tokens*, tendo como delimitadores os espaços em branco;

5. se existir somente um *token*, considerar a possibilidade de ser uma sigla. Senão, procurar por caracteres passíveis de formar siglas, tais como, caracteres no início de cada *token* e letras maiúsculas;

6. montar uma *string* com os caracteres encontrados.

Depois de montadas as duas strings - uma para cada parâmetro - elas são passadas para o algoritmo $Guth$ (seção 2.3.1), a fim de serem comparadas e um valor de similaridade é retornado. Este valor é baseado no valor de retorno do algoritmo $Guth$. Um exemplo de execução do algoritmo é apresentado na Tabela 2.3. No passo 2, os *tokens* da palavra a são identificados. Os caracteres iniciais dos *tokens* bem como os caracteres maiúsculos são selecionados no passo 3 e logo após concatenados em uma única *string* (passo 4). A palavra b é formada por um único token, por isso é passada diretamente ao passo 4. No passo 5, as strings encontradas são utilizadas como parâmetros do algoritmo $Guth$ que retorna o valor verdadeiro (passo 6).

Tabela 2.3: Exemplo de execução do algoritmo Acronyms

Passo	**Palavra** a			**Palavra** b
1	HyperText Transfer Protocol			HTTP
2	HyperText	Transfer	Protocol	
3	HT	T	P	
4	HTTP			HTTP
5	$Guth(HTTP, HTTP)$			
6	$True$			

2.3.3 Casamento por Fragmentos

Oliveira, Laender e Gonçalves (2005) definem uma estratégia para o tratamento de ambigüidades encontradas em campos referentes a nomes de autores em bibliotecas digitais. São utilizadas funções de casamento de padrão e técnicas de recuperação de informação, associadas a um algoritmo de agrupamento (*clustering*), que permitem a criação de índices unificados que mantêm a correspondência entre diferentes formas permissíveis de nomes de autores.

A partir de um repositório em formato XML, é realizada uma análise sintática e uma normalização dos dados, criando um índice de autoria. O índice de autoria é uma estrutura de dados que reúne, para cada autor, as informações bibliográficas contidas no repositório em uma seqüência de tuplas. Caracteres especiais, letras maiúsculas e letras acentuadas são substituídas nas cadeias de caracteres que compõem o índice de autoria por, respectivamente, espaços em branco, letras minúsculas e caracteres não acentuados. No caso de nomes de autores, ainda é alterada a ordem dos nomes na ocorrência de vírgulas e expandidas, através de uma lista de correspondências, algumas abreviaturas comuns. Por exemplo, “Silva Jr., José Geraldo” se torna “jose geraldo silva junior”.

O índice de autoria é a entrada para um algoritmo de agrupamento, que é dividido em duas fases: pré-avaliação e cálculo da similaridade. Este algoritmo gera como saída um índice unificado. A partir de um índice unificado é possível identificar quais são as obras contidas no repositório de uma biblioteca digital que são de autoria de um mesmo autor independentemente da forma como o seu nome está armazenado.

A etapa de pré-avaliação consiste em uma função para casamento de padrão que compara os nomes dos autores contidos no índice de autoria. Quando ocorre um casamento entre dois nomes de autores, estes são considerados candidatos a representarem o mesmo autor. Sendo o resultado superior a um determinado limiar, o par avaliado é associado, formando um grupo.

O algoritmo proposto para o casamento de padrão denomina-se *Comparação por Fragmentos*. O algoritmo compara individualmente, através de distância de edição (LEVENSHTEIN, 1966), cada fragmento delimitado por espaço em branco nas cadeias de caracteres normalizadas, considerando também o casamento entre iniciais e nomes por extenso e ignorando a ordem em que os fragmentos aparecem nos nomes.

O algoritmo de casamento *Comparação por Fragmentos* recebe como parâmetros duas cadeias de caracteres c_1 *e* c_2 com nomes de autores a serem comparados e um limiar inteiro $lim \in I$ utilizado pela distância de edição. Inicialmente, o algoritmo encarrega-se de descartar comparações entre nomes incompatíveis, verificando o número de fragmentos em cada cadeia de caracteres c_1 *e* c_2 e as iniciais dos fragmentos. Após, são comparados os fragmentos de c_1 e c_2, e os fragmentos considerados compatíveis em cada laço do algoritmo são marcados. Ao final da execução, se em pelo menos uma das cadeias de caracteres comparadas todos os fragmentos estiverem marcados, então os nomes são considerados compatíveis e o algoritmo retorna verdadeiro. Caso contrário, são considerados incompatíveis retornando falso.

Por fim, são utilizadas diversas técnicas da área de recuperação de informações para determinar o grau de similaridade entre os autores. São utilizadas técnicas como medida do cosseno (BAEZA-YATES; RIBEIRO-NETO, 1999), *bag of words* (CHAKRABARTI, 2002) e coeficiente de Jaccard (SALTON, 1988) sobre os outros metadados disponíveis no índice de autoria. Uma descrição mais detalhada da etapa de cálculo do grau de similaridade do algoritmo de agrupamento pode ser encontrada na dissertação de mestrado de Oliveira (2005).

Cota, Gonçalves e Laender (2007) utilizam o mesmo algoritmo de casamento para determinar a similaridade entre nomes de autores. O algoritmo aliado a uma heurística sobre os co-autores cria um conjunto inicial de *clusters* de co-autoria. Outras evidências, baseadas no título das publicações, são utilizadas para fundir os *clusters* de maneira hierárquica.

2.4 Comparação entre os Algoritmos de Casamento de Nomes

Esta seção apresenta uma análise comparativa entre os algoritmos de comparação de nomes apresentados nas seções 2.3.1 a 2.3.3. Primeiramente, são apresentados os critérios utilizados para comparação. Em seguida, é feita a análise de cada critério, que é resumida na Tabela 2.4.

Os seguintes critérios foram utilizados para realizar a comparação dos algoritmos:

- independência de idioma - para melhorar os resultados dos algoritmos de casamento de nomes podem ser utilizadas técnicas como comparação fonética (MANNING; RAGHAVAN; SCHÜTZE, 2008), por exemplo. Estas técnicas tornam os algoritmos dependentes do idioma de origem dos nomes comparados. Esta característica não é interessante na avaliação da autoria de objetos digitais, pois as bibliotecas digitais contam com publicações de autores de todo o mundo;

- variações de grafia - o processo de aquisição de dados utilizado pelas bibliotecas digitais pode ser realizado de diversas formas: auto-arquivamento (LAW, 2004), coleta de metadados através do protocolo Open Archives Initiative - Protocol for Metadata Harvesting (OAI-PMH) (SOMPEL et al., 2004), ou de outros artifícios que as bibliotecas digitais disponibilizem. Esta diversidade de mecanismos de aquisição pode ocasionar o armazenamento incorreto de alguns metadados. Citações bibliográficas com variações de grafia são comuns em sistemas de bibliotecas digitais, e devem ser consideradas pelos algoritmos de comparação de nomes;
- baseado em *token* - indica que o algoritmo de casamento de nomes utiliza *tokens* ou fragmentos como elementos de comparação;
- baseado em caractere - indica que o algoritmo utiliza caracteres como elementos de comparação;
- suporte a iniciais - nomes de autores podem ser abreviados e representados parcialmente pelas iniciais. Por exemplo, o autor *Eduardo Nunes Borges* corresponde ao autor *E. N. Borges*. Também é comum a ocorrência de abreviações nos nomes próprios referenciados por bibliotecas digitais. É necessário que os algoritmos de casamento de nomes próprios suportem a comparação das iniciais dos nomes;
- suporte a inversões - exitem diversas formas de citar os autores de um objeto digital. Por exemplo, o autor *Eduardo Borges* corresponde ao autor *Borges, Eduardo*. É comum a ocorrência de inversões nos nomes próprios referenciados por bibliotecas digitais. Esta característica é importante para que os algoritmos de casamento de nomes próprios identifiquem corretamente os autores de determinada publicação;
- complexidade - refere-se à complexidade computacional do algoritmo.

A Tabela 2.4 resume as características apresentadas nessa seção para cada algoritmo de comparação de nomes.

Tabela 2.4: Comparação dos algoritmos de casamento de nomes.

Características	Guth	Acronyms	Fragmentos
independência de idioma	V	V	V
variações de grafia	limitado	V	V
token	X	V	V
caractere	V	V	V
iniciais	X	limitado	limitado
inversões	X	X	parcial
complexidade	linear	linear	quadrática

Legenda: V sim X não

Os três algoritmos analisados são independentes de idioma e suportam variações de grafia. Guth (GUTH, 1976) compara os caracteres dos nomes nos testes executados pelo algoritmo, por isso pode ser classificado como algoritmo baseado em caractere. Acronyms (LIMA, 2002) utiliza o algoritmo Guth para verificar as variações de grafia, enquanto o algoritmo Comparação por Fragmentos utiliza a distância de edição (LEVENSHTEIN, 1966). Ambos são baseados tanto em *token* quanto em caractere, pois

dividem as *strings* passadas como parâmetros em *tokens* e os utilizam como elemento de comparação nos algoritmos baseados em caractere (Guth e Levenshtein).

Acronyms suporta a comparação das iniciais dos nomes, mas possui uma limitação quanto à presença de letras maiúsculas que não sejam iniciais. Por exemplo, *Acronyms ("Ramez El-Masri", "Ramez Elmasri")* retorna falso. Comparação por Fragmentos também suporta a comparação das iniciais. Embora a estratégia de comparação seja aplicável a nomes de autores, ela não cobre todos os casos. Por exemplo, *Comparação por Fragmentos ("Eduardo Nunes Borges", "E.N.Borges")* retorna verdadeiro, mas *Comparação por Fragmentos ("Eduardo N. Borges", "E. Borges")* retorna falso.

Somente o algoritmo Comparação por Fragmentos possui suporte a inversões. Embora resolva a maioria dos casos, existem determinadas situações que não são previstas pelo algoritmo. Por exemplo, *Comparação por Fragmentos ("Eduardo Borges", "Borges, Eduardo")* retorna verdadeiro, mas *Comparação por Fragmentos ("E. Nunes Borges", "Borges, Eduardo")* retorna falso.

Guth e Acronyms possuem complexidade linear em função do tamanho dos nomes passados como parâmetros, embora Guth seja mais rápido. Já o algoritmo Comparação por Fragmentos possui complexidade quadrática tanto em função do número quanto do tamanho dos fragmentos.

Para identificar ambigüidades na autoria de objetos digitais, é muito importante que sejam consideradas características como abreviações (iniciais) e inversões. Os algoritmos apresentados nesta seção, apesar de apresentarem soluções que abordam estas características, apenas solucionam parcialmente o problema, visto que não cobrem todos os casos possíveis. É necessário que os algoritmos de casamento de nomes identifiquem corretamente nomes próprios, independente das variações de grafia que os nomes apresentem.

2.5 Proveniência de Dados

Conhecida também como *data pedigree* ou *data lineage*, a proveniência de dados é a descrição das origens de uma porção de dados e o processo pelo qual ela é obtida (BUNEMAN; KHANNA; TAN, 2001). Greenwood et al. (2003) estende este conceito dizendo que a proveniência de dados é caracterizada por metadados que descrevem os processos de *workflows* e anotações sobre experimentos.

Duas principais características compõem a proveniência de dados: *where-provenance* e *why-provenance* (BUNEMAN; KHANNA; TAN, 2001). O termo *where-provenance* especifica a localização dos dados, ou seja, a origem de uma porção de dados (de onde eles são obtidos). O termo *why-provenance* especifica por que esta porção de dados está em uma determinada base de dados. Uma taxonomia das técnicas de proveniência é definida por Simmhan, Plale e Gannon (2005), na qual definem o conceito de produto de dados como uma porção de dados que é alvo da proveniência.

As técnicas existentes para rastrear a proveniência de dados são divididas em duas principais abordagens (TAN, 2004):

- imediata (*eager*) - armazena a proveniência conforme os dados vão sendo transformados . Seja D uma determinada base de dados. A cada passo de derivação sofrido por um produto de dados d pertencente a D, metadados ou anotações são armazenados em uma base de proveniência P. Anotações são informações que descrevem um determinado resultado de um passo de derivação. Os metadados e as anotações descrevem a origem de d e o processo pelo qual d é obtido e derivado. A maior vantagem desta abordagem é que o armazenamento dos metadados e das anotações

é pré-computado. O tempo de processamento necessário para acessar informações de proveniência é apenas o tempo de realizar uma consulta $Q(d)$ sobre o índice de P. A desvantagem diz respeito ao espaço necessário para armazenar P. Dependendo da granularidade de d, a base de proveniência P pode atingir um tamanho muito maior que a base de dados D. Deve-se considerar também se a atualização das fontes de origem dos dados implica na atualização da proveniência. Para refletir o estado original de D é necessário computar periodicamente a proveniência. Técnicas que utilizam esta abordagem incluem (BHAGWAT et al., 2004; TAN, 2003).

- postergada (*lazy*) - computa a proveniência somente quando necessário. Um determinado produto de dados d é gerado através da realização de uma consulta Q sobre uma base de dados D. Através de técnicas de inversão de consultas, é possível determinar uma consulta $Q'(d, D, Q)$ que identifica a proveniência de d a partir de D e Q. Q' inverte o processo de derivação a fim de determinar os dados originais. A principal vantagem desta abordagem é que a proveniência é computada somente quando um usuário realiza uma consulta. Não é necessário armazenar nenhuma informação sobre as derivações de um produto de dados. Além disso, mesmo que a fonte de origem seja atualizada, as respostas das consultas sempre refletem o estado original de D. A desvantagem diz respeito ao tempo de resposta das consultas. Como a proveniência é processada durante a consulta Q', o tempo de resposta é definido pelo algoritmo de inversão utilizado. Este tempo é inversamente proporcional ao número de derivações que um produto de dados sofreu. Técnicas que utilizam esta abordagem incluem (CUI; WIDOM, 2001; BUNEMAN; KHANNA; TAN, 2001).

Atualmente, a pesquisa na área de proveniência de dados destina-se principalmente ao domínio da ciência. Diversos sistemas foram propostos com a finalidade de recuperar a linhagem de dados científicos. *Earth System Science Workbench* (ESSW) (FREW; BOSE, 2001) é um sistema de armazenamento e gerenciamento de dados e imagens de satélites. A proveniência armazenada permite a descoberta de erros nos processos de derivação dos dados, determinando a qualidade das bases de dados geradas. MyGrid (ZHAO et al., 2004) fornece um *middleware* para o suporte de experimentos na área da biologia. Um registro de proveniência é automaticamente armazenado durante a execução de um *workflow*, contendo os serviços invocados, o tempo de início e de fim, descrições de ontologias e os produtos de dados usados e derivados. Chimera (FOSTER et al., 2002) é um sistema que gerencia a derivação e a análise de produtos de dados físicos e astronômicos em ambientes de colaboração. Através de um modelo orientado a processo, o usuário constrói *workflows* denominados grafos de derivação usando uma linguagem de dados virtual.

Ao contrário dos sistemas anteriormente citados, Trio (WIDOM, 2005) é um sistema de bancos de dados livre de domínio de aplicação. Este sistema possui características nativas de proveniência e precisão. Os valores dos dados podem ser incorretos, incompletos ou até aproximados. O sistema utiliza o modelo de inversão (abordagem *lazy*) para determinar automaticamente a origem dos dados para novas tuplas criadas, bem como a precisão da derivação destes dados. A proveniência é recuperada através de uma linguagem de consulta própria.

2.6 Deduplicação e Proveniência de Dados em Bibliotecas Digitais

Buneman, Khanna, e Tan (2000) abordam o assunto proveniência na citação de dados, especificamente na citação de documentos em bibliotecas digitais. Enquanto tuplas são identificadas de forma única nos bancos de dados (através do uso de chaves), porções de documentos não possuem identificadores únicos. Para documentos XML é possível identificar um elemento através do identificador ID presente em um *Document Type Descriptor* (DTD). Mas o atributo ID não define a unicidade em relação à estrutura do documento, somente a definição de um identificador pelo usuário. Portanto, técnicas de recuperação de proveniência de dados utilizadas em bases relacionais não podem ser diretamente utilizadas em bases XML.

Além dos objetos digitais (documentos, imagens, vídeos, etc.), as bibliotecas digitais armazenam um conjunto de metadados que descrevem informações relacionadas aos objetos e especificam a maneira como eles são recuperados. Por exemplo, o *Dublin Core* (DCMI, 2008) define um padrão de metadados a respeito de artigos científicos, periódicos e páginas *Web*. O *Dublin Core Metadata Initiative* (DCMI) propôs um termo denominado *provenance* que tem como objetivo documentar mudanças na propriedade e custódia dos recursos que são importantes para a autenticidade, integridade ou interpretação de um documento digital (DCMI, 2004). A semântica associada a este metadado pode não ser processável automaticamente, ou seja, interpretada por um computador. O principal padrão de publicação de artigos científicos utilizado nas bibliotecas digitais ainda não permite o rastreio automático da proveniência.

A *Collection of Computer Science Bibliographies* (CCSB) (ALF-CHRISTIAN ACHILLES, 2007) 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 Biblioteca Digital Brasileira de Computação (BDBComp) (LAENDER; GONÇALVES; ROBERTO, 2004), *Digital Bibliography & Library Project* (DBLP) (UNIVERSITY OF TRIER, 2007) e CiteSeer (GILES; BOLLACKER; LAWRENCE, 1998). Atualmente, a CCSB 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 (LAMPORT, 1986). A Figura 2.1 mostra um exemplo de artigo científico consultado no sistema. É identificado que o objeto digital de título "*UXQuery: Building Updatable XML Views over Relational Databases*" possui três duplicatas. As bibliotecas digitais de origem de cada réplica (DBLP, BDBComp e CiteSeer) são informadas pela interface do sistema.

A granularidade da proveniência adotada pela CCSB é de uma publicação. A partir do link BibTEX, exibido nas respostas às consultas do usuário, são disponibilizados os metadados principais da publicação (Figura 2.2), provenientes de somente uma das fontes. O critério de escolha desta fonte não é fornecido no sistema.

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 (Figura 2.3).

100: Vanessa P. Braganholo and Susan B. Davidson and Carlos A. Heuser
UXQuery: Building Updatable XML Views over Relational Databases
[FIND SIMILAR] [TRY GOOGLE]

SBBD, pp. 26-40, UFAM, 2003.

BibTeX (3 dupl. with Abstract and URL)

DBLP (2003)

Some entries might have been ommited on the list above if author/title combination has already occurred. If in doubt check the duplicate list linked on the right of some citations.

Top ten bibliographies for this result page:

1 match: DBLP (2003)
1 match: BDBComp Archive
1 match: CiteSeer (2003_08)

Figura 2.1: Informações de proveniência recuperadas em uma consulta à CCSB. São exibidas três fontes distintas (DBLP, BDBComp e CiteSeer).

From DBLP (2003):

```
@InProceedings{conf/sbbd/BraganholoDH03,
  title =        "{UXQ}uery: Building Updatable {XML} Views over
                  Relational Databases",
  author =       "Vanessa P. Braganholo and Susan B. Davidson and Carlos
                  A. Heuser",
  bibdate =      "2003-10-06",
  bibsource =    "DBLP,
                  http://dblp.uni-trier.de/db/conf/sbbd/sbbd2003.html#BraganholoDH03",
  booktitle =    "SBBD",
  booktitle =    "{XVIII} Simp{\'o}sio Brasileiro de Bancos de Dados,
                  6-8 de Outubro, Manaus, Amazonas, Brasil,
                  Anais/Proceedings",
  publisher =    "UFAM",
  year =         "2003",
  editor =       "Alberto H. F. Laender",
  pages =        "26--40",
}
```

Figura 2.2: Metadados no formato BibTEX provenientes de uma única fonte.

The Collection of Computer Science Bibliographies

Home | About | FAQ | Browse | Add | Statistics

All entries that share the same title and author(s) last names:

Hint: *Consider searching listed here subcollections for other publications in the subject of your interest.*

From DBLP (2003):

```
@InProceedings{conf/sbbd/BraganholoDH03,
  title =        "{UXQ}uery: Building Updatable {XML} Views over
                  Relational Databases",
  author =       "Vanessa P. Braganholo and Susan B. Davidson and Carlos
                  A. Heuser",
  bibdate =      "2003-10-06",
  bibsource =    "DBLP,
                  http://dblp.uni-trier.de/db/conf/sbbd/sbbd2003.html#BraganholoDH03",
  booktitle =    "SBBD",
  booktitle =    "{XVIII} Simp{\'o}sio Brasileiro de Bancos de Dados,
                  6-8 de Outubro, Manaus, Amazonas, Brasil,
                  Anais/Proceedings",
  publisher =    "UFAM",
  year =         "2003",
  editor =       "Alberto H. F. Laender",
  pages =        "26--40",
}
```

From CiteSeer (2003_08):

CiteSeer

```
@Misc{oai:CiteSeerPSU:675314,
  title =        "{UXQ}uery: Building Updatable {XML} Views over
                  Relational Databases",
  author =       "Vanessa P. Braganholo and Susan B. Davidson and Carlos
                  A. Heuser",
  year =         "2003",
  month =        aug # "~15",
  abstract =     "XML has become an important medium for data exchange,
                  and is frequently used as an interface to -- i.e. a
                  view of -- a relational database. Although much
                  attention has been paid to the problem of querying
                  relational databases through XML views, the problem of
                  updating relational databases through XML views has not
                  been addressed. In this paper we investigate how a
                  subset of XQuery can be used to build updatable XML
                  views, so that an update to the view can be
                  unambiguously translated to a set of updates on the
                  underlying relational database, assuming that certain
                  key and foreign key constraints hold. In particular, we
                  show how views defined in this subset of XQuery can be
                  mapped to a set of relational views, thus transforming
                  the problem of updating relational databases through
                  XML views into a classical problem of updating
                  relational databases through relational views.",
  citeseer-references = "oai:CiteSeerPSU:583241; oai:CiteSeerPSU:583820;
                  oai:CiteSeerPSU:570271; oai:CiteSeerPSU:600331;
                  oai:CiteSeerPSU:114249; oai:CiteSeerPSU:603512;
                  oai:CiteSeerPSU:651122; oai:CiteSeerPSU:603731;
                  oai:CiteSeerPSU:588278",
  annote =       "Carlos A. Heuser (Instituto de Informtica -
                  Universidade Federal do Rio Grande do Sul - Brazil;
                  Department of Computer and Information Science -
                  University of Pennsylvania);",
  bibsource =    "OAI-PMH server at cs1.ist.psu.edu",
  language =     "en",
  oai =          "oai:CiteSeerPSU:675314",
  rights =       "unrestricted",
  URL =          "http://citeseer.ist.psu.edu/675314.html;
                  http://www.inf.ufrgs.br/~vanessa/artigos/sbbd2003.pdf",
}
```

From BDBComp Archive:

PDF

```
@Article{sbbd2003meta003,
  title =        "{UXQ}uery: building updatable {XML} views over
                  relational databases",
  author =       "Vanessa P. Braganholo and Susan B. Davidson and Carlos
                  A. Heuser",
  year =         "2003",
  abstract =     "XML has become an important medium for data exchange,
                  and is frequently used as an interface to - i.e. a view
                  of - a relational database. Although much attention has
                  been paid to the problem of querying relational
                  databases through XML views, the problem of updating
                  relational databases through XML views has not been
                  addressed. In this paper we investigate how a subset of
                  XQuery can be used to build updatable XML views, so
                  that an update to the view can be unambiguously
                  translated to a set of updates on the underlying
                  relational database, assuming that certain key and
                  foreign key constraints hold. In particular, we show
                  how views defined in this subset of XQuery can be
                  mapped to a set of relational views, thus transforming
                  the problem of updating relational databases through
                  XML views into a classical problem of updating
                  relational databases through relational views.",
  bibsource =    "OAI-PMH server at www.lbd.dcc.ufmg.br",
  coverage =     "Manaus, AM, Brasil",
  identifier =   "sbbd2003article003",
  language =     "por",
  rights =       "Sociedade Brasileira de Computa{\c c}{\~a}o",
  source =       "sbbd2003",
  URL =          "http://www.lbd.dcc.ufmg.br/sbbd2003/artigos/paper003.pdf",
}
```

Figura 2.3: Interface do sistema poluída com a redundância dos metadados armazenados.

2.7 Considerações Finais

Neste capítulo, foram apresentados trabalhos relacionados ao contexto da dissertação que envolvem as áreas de deduplicação e proveniência de dados. Também foram apresentados algoritmos de similaridade de nomes, que são utilizados nos experimentos comparando com a proposta desta dissertação (capítulo 4).

A partir da análise realizada neste capítulo é possível identificar algumas características desejáveis que não são abordadas nestes trabalhos:

- casamento de esquemas sem a intervenção do usuário: nenhum dos trabalhos estudados realiza o casamento entre os esquemas dos objetos. Todos partem do princípio que o casamento de esquemas é previamente realizado, seja por um usuário especialista ou por uma técnica semi-automática de casamento;

- deduplicação de objetos digitais: somente o trabalho proposto por Carvalho et al. (2006) e a CCSB (ALF-CHRISTIAN ACHILLES, 2007) consideram deduplicação de objetos em bibliotecas digitais, sendo que a CCSB utiliza uma abordagem muito simples (são considerados duplicatas os objetos que possuem títulos e últimos nomes de autores iguais);

- proveniência de objetos digitais: o domínio de aplicação da maioria dos trabalhos que tratam a proveniência de dados diz respeito a dados de experimentos científicos (*e-science*). Somente a CCSB (ALF-CHRISTIAN ACHILLES, 2007) trata o domínio das bibliotecas digitais. Neste trabalho, o produto de dados (alvo da proveniência) é uma publicação. Não foram encontrados trabalhos que tracem a linhagem dos metadados que descrevem uma publicação;

- facilidade de disseminação da proveniência: garantir que a disseminação da proveniência seja realizada através de qualquer linguagem de consulta, sem necessidade de utilizar uma linguagem ou interface específica;

- determinar quais atributos possuem maior importância no casamento: embora muitas abordagens utilizem pesos para indicar a importância de cada atributo no processo de deduplicação, a escolha desses pesos é de responsabilidade do usuário;

- deduplicação de nomes próprios: para identificar a autoria de objetos é necessário utilizar técnicas específicas para a deduplicação de nomes próprios. A maioria dos trabalhos não possui esta característica;

- suporte a abreviações e inversões: algoritmos de comparação de nomes próprios devem suportar abreviações e inversões dos nomes. Somente um dos trabalhos analisados suporta inversões de nomes, e de forma parcial;

As características citadas motivam o desenvolvimento deste trabalho, onde é definido um mecanismo de deduplicação de metadados e rastreio da proveniência. Este mecanismo identifica objetos duplicados em sistemas de bibliotecas digitais e armazena informações de proveniência de cada metadado. O casamento entre os esquemas dos objetos digitais é realizado sem a intervenção do usuário. O mecanismo proposto para solucionar as carências acima citadas é especificado detalhadamente no próximo capítulo.

3 MD-PROM

Este capítulo apresenta o mecanismo MD-PROM (*Metadata Deduplication and PROvenance tracing Mechanism*) proposto para deduplicar objetos e recuperar a proveniência de metadados oriundos de bibliotecas digitais. A principal contribuição do MD-PROM é melhorar a qualidade da pesquisa do usuário de bibliotecas digitais, fornecendo uma resposta única e livre de informação redundante.

Este capítulo está organizado da seguinte forma. Primeiramente, é apresentada uma visão geral do MD-PROM e as definições formais necessárias para a especificação de suas propriedades. Em seguida, é exposta a abordagem para deduplicar objetos digitais incluindo o mapeamento entre os formatos e o casamento do conteúdo dos metadados. A estrutura de dados que representa a proveniência também é apresentada, bem como detalhes do processamento da integração dos metadados. Por fim, são expostos as considerações finais e um comparativo do mecanismo com os trabalhos relacionados.

3.1 Visão Geral do MD-PROM

A Figura 3.1 mostra a arquitetura do MD-PROM. As bases de dados A, B e C representam diferentes bibliotecas digitais. Um catálogo de metadados XML, que referenciam os objetos digitais, está associado a cada biblioteca digital. A base de dados ABC representa um sistema de integração de bibliotecas digitais, cujo catálogo de metadados é formado a partir da totalidade de metadados das bibliotecas digitais integradas. Duas ou mais bibliotecas digitais podem indexar um mesmo objeto digital, o que gera a replicação de metadados que fazem referência a este objeto. O mecanismo proposto é formado por três componentes principais:

- Identificação - responsável por coletar metadados XML referentes a sistemas de integração de bibliotecas digitais. Esta coleta é realizada através do protocolo OAI-PMH (SOMPEL et al., 2004) ou de outros artifícios que as bibliotecas digitais disponibilizem. Além disso, possui a função de identificar duplicatas dos metadados referentes aos artigos científicos através de técnicas de similaridade, considerando o conteúdo e a estrutura. O casamento de esquemas é realizado automaticamente, sem a intervenção do usuário (detalhes na seção 3.2);

- Armazenamento - é composto por duas bases de dados: Duplicatas e Proveniência. A primeira armazena os resultados do processamento da deduplicação, ou seja, as diferentes representações dos metadados. Na segunda base são armazenadas informações resultantes do processamento da proveniência, ou seja, a origem dos metadados (bibliotecas digitais de onde foram coletados) e os valores dos quais foram derivados;

- Proveniência - responsável por rastrear as informações de proveniência dos metadados. Este componente gerencia o armazenamento das anotações que descrevem a proveniência de dados e realiza a integração dos metadados replicados (detalhes na seção 3.3).

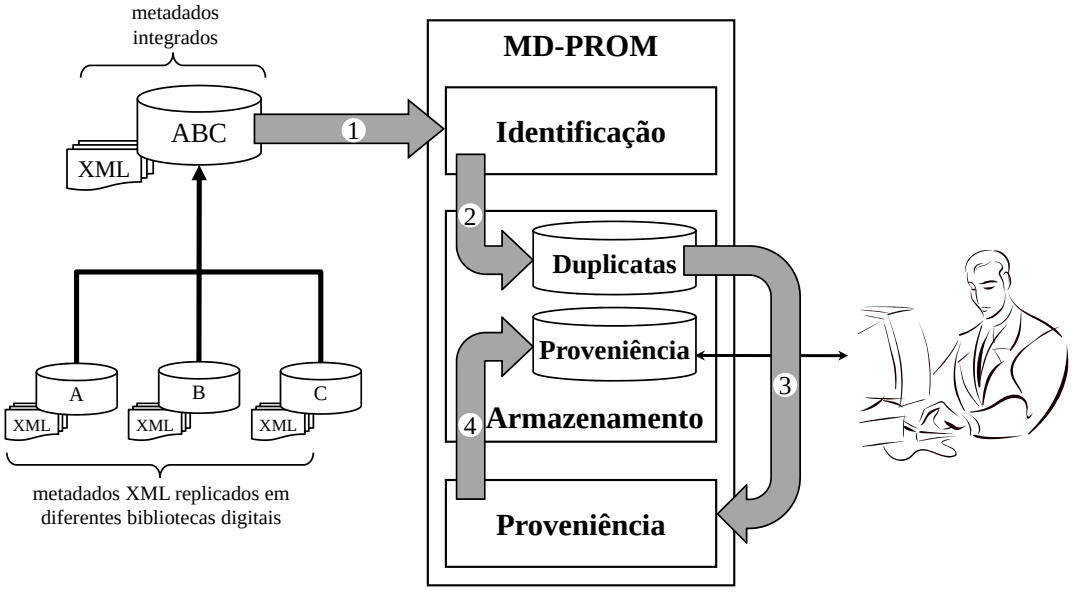

Figura 3.1: Arquitetura do MD-PROM.

Inicialmente, o mecanismo coleta os metadados a partir das bibliotecas digitais (1). Os metadados são comparados através de um algoritmo de casamento visando identificar objetos duplicados. As duplicatas de metadados de um mesmo objeto indexado por diferentes bibliotecas digitais, detectadas pelo componente *Identificação*, são armazenadas pelo componente *Armazenamento* na base de dados *Duplicatas* (2).

Tanto a coleta quanto a identificação de réplicas 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 componente *Identificação* coleta os novos metadados disponibilizados e compara-os com os objetos previamente identificados e armazenados a fim de detectar novas réplicas.

O componente *Proveniência* utiliza os objetos duplicados armazenados na base *Duplicatas* (3) para rastrear a linhagem dos metadados. Nesta etapa, as duplicatas são integradas evitando a redundância de dados. O resultado do processamento da proveniência é armazenado pelo componente *Armazenamento* na base de dados *Proveniência* (4). Quando um usuário submete uma consulta ao sistema, são buscadas as informações sobre os objetos digitais no componente *Armazenamento*.

As próximas seções apresentam como os requisitos aqui identificados para a deduplicação e para o processamento da proveniência são alcançados.

3.2 Identificação de Duplicatas

Esta seção define o processo de deduplicação de objetos digitais. A estrutura dos objetos é comparada através do casamento entre os esquemas dos metadados. O conteúdo é comparado utilizando funções de similaridade propostas para o domínio de valores de cada metadado.

Dois objetos digitais (a, b) são identificados como duplicatas quando possuem equivalência semântica, ou seja, quando tanto os metadados do objeto a quanto os metadados do objeto b descrevem a mesma publicação indexada por diferentes bibliotecas digitais.

3.2.1 Casamento de Esquemas

O casamento de esquemas é a tarefa de identificar diferentes estruturas que possuem equivalência semântica. Técnicas de casamento de esquemas fornecem uma lista de casamentos candidatos e as partes dos esquemas que não possuem correspondência, sendo necessária a intervenção do usuário especialista na etapa final do casamento.

Os metadados de bibliotecas digitais são representados freqüentemente pelos formatos BibTEX (LAMPORT, 1986) ou OAI Dublin Core (DCMI, 2008). Como ambos os formatos descrevem metadados de objetos digitais e possuem equivalências, o casamento entre os diferentes esquemas é realizado através da correspondência direta entre os rótulos dos metadados de cada formato, sem que seja necessária a intervenção do usuário. A relação de correspondência entre os rótulos dos metadados presentes em ambos os formatos é definida da seguinte forma:

- pares de metadados com rótulos idênticos formam um casamento;
- os pares de metadados cujos rótulos pertencem ao conjunto { (*year*, *date*), (*author*, *creator*), (*copyright*, *rights*), (*url*, *identifier*), (*location*, *coverage*), (*keywords*, *subject*), (*abstract*, *description*), (*booktitle*, *source*) } formam casamentos.

Algumas bibliotecas digitais, como a DBLP (UNIVERSITY OF TRIER, 2007), utilizam um esquema próprio para representar os metadados. Para bibliotecas digitais que não utilizam os formatos BibTEX ou OAI Dublin Core, é necessária a intervenção humana para definir os casamentos correspondentes. Por exemplo, a Tabela 3.1 apresenta a correspondência dos rótulos dos metadados utilizados pela DBLP em relação aos esquemas BibTEX e OAI Dublin Core realizada por um usuário especialista. Os metadados mais freqüentes em referências bibliográficas são destacados em negrito.

3.2.2 Casamento de Dados

Para realizar o casamento de dados são especificadas três funções de similaridade aplicadas ao domínio de bibliotecas digitais, denominadas $YearSim$, $IniSim$ e $NameMatch$. Estas funções comparam as datas de publicação e os nomes dos autores visando identificar diferentes representações da autoria de um objeto digital.

3.2.2.1 ***YearSim***

A função de similaridade $YearSim : \{\mathbb{N}\} \rightarrow \mathbb{N}_S$, sendo $\mathbb{N}$ o conjunto dos números naturais e $\mathbb{N}_S = \{x \in \mathbb{N} \mid x = 0 \vee x = 1\}$, compara as datas em que dois objetos digitais foram publicados. A Equação 3.1 define a função,

$$YearSim(year_1, year_2, t_Y) = \begin{cases} 1, & \text{se } |year_1 - year_2| \leq t_Y \\ 0, & \text{caso contrário} \end{cases} \tag{3.1}$$

onde $year_i \in \mathbb{N}$ é o ano de publicação de um objeto digital e $t_Y \in \mathbb{N}$ representa a máxima diferença entre os anos de publicação.

Se o valor absoluto da diferença entre os anos de publicação dos objetos for menor ou igual ao limiar t_Y, a função retorna o escore de similaridade $s \in \mathbb{N}_S | s = 1$. Caso

contrário, é retornado o escore $s = 0$. Por exemplo, $YearSim$(2005, 2004, 1) = 1 e $YearSim$(1995, 1999, 3) = 0.

Tabela 3.1: Casamento entre os rótulos dos metadados presentes nos esquemas BibTEX, OAI Dublin Core Qualificado (omitindo refinamentos) e DBLP.

Semântica associada ao metadado	**BibTEX**	**OAI Dublin Core**	**DBLP**
afiliação	affiliation		
ano de publicação	**year**	**date**	**year**
anotação	annote		note
autor	**author**	**creator**	**author**
cdrom			cdrom
contribuidor		contributor	
direitos autorais	**copyright**	**rights**	
edição	edition		
editor	editor		editor
editora	**publisher**	**publisher**	**publisher**
endereço da editora	address		address
endereço web	**url**	**identifier**	**ee, url**
formato (MIME Media Types)		format	
idioma	**language**	**language**	
índice de conteúdo	contents		
informação extra	note		
instituição de ensino	school		school
instituição envolvida na publicação	institution		
ISBN	isbn		isbn
ISSN	issn		
journal ou revista	journal		journal
LCCN	lccn		
localização geográfica	**location**	**coverage**	
método de publicação	howpublished		
mês de publicação	month		month
mudanças		provenance	
número de capítulo	chapter		chapter
número de journal ou revista	number		number
números de página	**pages**		**pages**
número de revisão matemática	mrnumber		
ordem dos metadados	key		
palavras-chave ou assunto	**keywords**	**subject**	
patrocinador de uma conferência	organization		
público alvo		audience	
quem detém direitos		rightsholder	
referência			ref
referência cruzada	crossref		crossref
resumo	**abstract**	**description**	
série de livros	series		series
tamanho	size		
tipo	type	type	
título	**title**	**title**	**title**
título de livro, periódico ou anais	**booktitle**	**source**	**booktitle**
trabalho relacionado		relation	cite
valor	price		
volume de um journal ou livro	volume		volume

3.2.2.2 ***Initials***

A função $Initials : \{C\} \rightarrow C$, sendo C o conjunto composto por qualquer cadeia de caracteres, extrai as iniciais do nome de um autor. Esta função é chamada pela função $IniSim$ (seção 3.2.2.3), a qual calcula a similaridade entre as iniciais dos nomes dos autores. $Initials$ é definida pelo algoritmo 1,

Algoritmo 1 *Initials*

INITIALS$(author)$
1 **for** *all* $token \in author$
2 **do** $initials =$ CONCAT$(initials,$ FIRSTCHAR$(token))$;
3 **return** $initials$;

onde:

- $author \in C$ é uma cadeia de caracteres que representa o nome próprio de um autor;
- $token \in C$ é uma *substring* de $author$;
- FIRSTCHAR é uma função que retorna o primeiro caractere de uma palavra;
- $initials \in C$ é uma palavra formada pelas iniciais do nome próprio de um autor;
- CONCAT é uma função que retorna a concatenação de duas palavras.

Por exemplo, $Initials$(*Eduardo Nunes Borges*) = *ENB* e $Initials$(*Galante, Renata*) = *GR*. O primeiro caractere de cada nome é extraído e os caracteres são concatenados (linha 2). A função retorna uma palavra que representa as iniciais do nome passado como parâmetro (linha 3).

3.2.2.3 ***IniSim***

A função de similaridade $IniSim$ identifica variações na representação do nome de um autor considerando grafias diferentes, inversões e abreviações. São realizadas comparações somente entre as iniciais dos nomes de autores, o que torna possível a implementação da função através de um algoritmo de complexidade linear.

$IniSim : \{C\} \rightarrow \mathbb{N}_S$, sendo C o conjunto composto por qualquer cadeia de caracteres, $\mathbb{N}$ o conjunto dos números naturais e $\mathbb{N}_S = \{x \in \mathbb{N} \mid x = 0 \vee x = 1\}$, é uma função que calcula a similaridade entre as iniciais dos nomes de autores extraídas a partir da função $Initials$ (seção 3.2.2.2). A função $IniSim$ verifica se as iniciais $(a, b) \in C$ representam o nome de um mesmo autor. São realizadas comparações entre a primeira, a segunda e a última inicial. A função retorna um escore de similaridade $s \in \mathbb{N}_S$. Quando as iniciais correspondem ao mesmo autor, ou seja, quando as duas representações podem expressar o mesmo objeto do mundo real, a função $IniSim$ retorna um escore de similaridade $s = 1$. Caso contrário, é retornado o escore $s = 0$. $IniSim$ é definida pela Equação 3.2,

$$IniSim(a,b) = \begin{cases} 1, & se \quad \begin{array}{l}(a_1 = b_1 \wedge a_m = b_n) \vee \\ (a_1 = b_2 \wedge a_m = b_1) \vee \\ (a_1 = b_1 \wedge a_2 = b_2) \vee \\ (a_1 = b_n \wedge a_2 = b_1)\end{array} \\ 0, & \text{caso contrário} \end{cases} \tag{3.2}$$

onde:

- a, b são as iniciais dos nomes de autores;
- a_i é a i-ésima letra da palavra a, ou seja a i-ésima inicial do nome;
- b_i é a i-ésima letra da palavra b, ou seja a i-ésima inicial do nome;
- m é o tamanho da palavra a;
- n é o tamanho da palavra b.

Por exemplo, $IniSim(Initials$(*Eduardo Nunes Borges*), $Initials$(*Borges, Eduardo*)) = 1 e $IniSim(Initials$(*Eduardo Nunes*), $Initials$(*Borges, Eduardo*)) = 0. As condições impostas pela função partem do princípio que o nome de um autor pode estar representado de diversas maneiras como mostra a Tabela 3.2.

Tabela 3.2: Possíveis representações de Eduardo Nunes Borges.

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

3.2.2.4 ***NameMatch***

$NameMatch$ é uma função de similaridade que compara todos os autores de um objeto digital com os autores de outro objeto. O objetivo da função é identificar diferentes representações da autoria de um objeto digital.

A função $NameMatch : \{C, \mathbb{R}_S\} \rightarrow \mathbb{N}_S$, sendo C o conjunto composto por qualquer cadeia de caracteres, $\mathbb{R}$ o conjunto dos números reais, $\mathbb{R}_S = \{x \in \mathbb{R} \mid 0 \leq x \leq 1\}$, $\mathbb{N}$ o conjunto dos números naturais e $\mathbb{N}_S = \{x \in \mathbb{N} \mid x = 0 \vee x = 1\}$, é definida pelo Algoritmo 2,

Algoritmo 2 *NameMatch*

```
NAMEMATCH(K, L, t_N)
 1  m ← LENGTH(K);
 2  n ← LENGTH(L);
 3  for i ← 1 to m
 4      do for j ← 1 to n
 5             do if INISIM(K_i, Li) = 1
 6                    then counter ← counter + 1;
 7                         L_j ← Null;
 8  if counter/MAX(m, n) < t_N
 9    then return 0;
10    else  return 1;
```

onde:

- K, L são listas de iniciais dos nomes dos autores dos objetos digitais comparados;
- $K_i \in C$ é o i-ésimo elemento da lista K, ou seja as inicias no i-ésimo autor do primeiro objeto;
- $L_j \in C$ é o j-ésimo elemento da lista L, ou seja as inicias no j-ésimo autor do segundo objeto;
- $t_N \in \mathbb{R}_S$ é um valor de limiar (*threshold*) de similaridade aplicado a função;
- LENGTH é uma função que retorna o tamanho de uma lista;
- m é o tamanho da lista K;
- n é o tamanho da lista L;
- INISIM (seção 3.2.2.3) compara as iniciais dos nomes dos autores;
- $counter$ acumula o número de casamentos encontrados pela função $IniSim$;
- MAX é uma função que retorna o tamanho da maior lista de palavras.

O algoritmo recebe como parâmetros duas listas compostas pelas iniciais dos nomes dos autores (K, L) e o limiar mínimo de casamento entre os autores t_N. Este limiar tem como objetivo deixar a função flexível a ajustes. $NameMatch$ gera um escore de similaridade $s \in \mathbb{N}_S$.

As duas listas K, L são percorridas (linhas 3-4) a fim de encontrar o casamento entre os nomes dos autores, o qual é realizado através da função $IniSim$ (seção 3.2.2.3). Quando ocorre o casamento entre dois autores (linha 5), é atribuído um valor nulo $(Null)$ a uma palavra da lista L para que esta não seja utilizada em futuras comparações (linha 7). A variável $counter$ conta o número de casamentos verificados (linha 6). Quando o limiar de casamento mínimo é atingido (linha 8), a função $NameMatch$ retorna $s = 1$, caso contrário retorna $s = 0$.

Quando ocorrem erros no processo automático de aquisição dos dados pelas bibliotecas digitais, ou seja, quando a lista de autores de um determinada publicação não está

completa, o limiar t_N passado como parâmetro ajusta a função para que ocorra a identificação da duplicação.

Por exemplo, considere a omissão do nome de um autor no segundo parâmetro da função $NameMatch$ ((*AB*, *CD*, *EF*), (*FE*, *BA*), *0.6*). São atribuídos os valores 3 e 2 às variáveis m e n respectivamente. A lista (*AB*, *CD*, *EF*) é percorrida e cada elemento é comparado a todos os elementos da lista (*FE*, *BA*). A função $IniSim$ retorna 1 para os parâmetros (AB, BA) e (EF, FE), sendo acumulado o valor 2 na variável $counter$. Como a razão entre o número de casamentos encontrados ($counter = 2$) e o retorno da função Max ($Max(m, n) = 3$) é igual a 66,6%, o limiar mínimo de 60% passado como parâmetro é atingido e a função $NameMatch$ retorna 1. O limiar de similaridade adotado, neste caso, faz com que dois objetos digitais com diferentes números de autores sejam possíveis candidatos ao casamento[1].

3.2.2.5 ***Digital Object Match***

A função $Digital\ Object\ Match$ realiza o casamento de dois objetos digitais. O algoritmo tem como objetivo deduplicar objetos digitais provenientes de bibliotecas digitais distintas.

A função $DigitalObjectMatch : \{M, \mathbb{N}_S, \mathbb{R}_S\} \rightarrow \mathbb{N}_S$, sendo M o conjunto de metadados de um objeto digital, $\mathbb{R}$ o conjunto dos números reais, $\mathbb{R}_S = \{x \in \mathbb{R} \mid 0 \leq x \leq 1\}$, $\mathbb{N}$ o conjunto dos números naturais e $\mathbb{N}_S = \{x \in \mathbb{N} \mid x = 0 \vee x = 1\}$, é definida pelo Algoritmo 3,

Algoritmo 3 *Digital Object Match*

DIGITAL OBJECT MATCH(a, b, t_Y, t_N, t_L)
1 **if** YEARSIM(YEAR(a), YEAR$(b), t_Y) = 1$
2 **then for** *all* AUTHOR(a)
3 **do** ADD$(iniList_a$, INITIALS(AUTHOR$(a)))$;
4 **for** *all* AUTHOR(b)
5 **do** ADD$(iniList_b$, INITIALS(AUTHOR$(b)))$;
6 **if** NAMEMATCH$(iniList_a, iniList_b, t_N) = 1$
7 **then if** LEVENSHTEIN(TITLE(a), TITLE$(b)) > t_L$
8 **then return** $true$;
9 **return** $false$;

onde:

- (a, b) são metadados de dois objetos digitais;
- $t_Y \in \mathbb{N}$ é a máxima diferença entre os anos de publicação;
- $t_N \in \mathbb{R}_S$ é o limiar mínimo de casamento entre os autores;
- $t_L \in \mathbb{R}_S$ é o limiar mínimo de similaridade entre os títulos das publicações;
- YEAR é uma função que retorna o ano de publicação de um objeto digital;

[1]Definições adequadas de valores de limiar são discutidas na literatura (STASIU; HEUSER; SILVA, 2005) e não fazem parte do escopo do trabalho apresentado nesta dissertação.

- YEARSIM (seção 3.2.2.1) compara os anos de publicação;
- AUTHOR é uma função que retorna os autores de um objeto digital;
- INITIALS (seção 3.2.2.2) retorna as iniciais de um autor;
- $iniList_i$ é uma lista das iniciais dos nomes dos autores do objeto i;
- ADD é uma função que adiciona as iniciais de um autor a uma lista;
- NAMEMATCH (seção 3.2.2.4) realiza o casamento entre os autores;
- TITLE é uma função que retorna o título de um objeto digital;
- LEVENSHTEIN calcula a similaridade entre os títulos dos objetos.

As funções de similaridade definidas nas seções 3.2.2.1 a 3.2.2.4 são aplicadas aos principais metadados de cada objeto. A Tabela 3.3 lista estas funções e os metadados em que são aplicadas.

Tabela 3.3: Funções de similaridade aplicadas a cada metadado.

Metadado	**Domínio**	**Função**
autor	nomes próprios	$Initials$, $IniSim$, $NameMatch$
título	títulos de publicações	$Levenshtein$
ano de publicação	datas	$YearSim$

A escolha da função $Levenshtein$ (LEVENSHTEIN, 1966) para computar a similaridade entre os títulos dos objetos digitais foi definida com base no domínio de aplicação do metadado, em experimentos previamente executados (BORGES; DORNELES, 2006), e pelo fato da função preservar a ordem em que as palavras e caracteres aparecem nas *strings*. Assim, títulos com várias palavras iguais, mas que a ordem das palavras é diferente, geram escores de similaridade próximos de zero. Por exemplo, para as *strings* "qualidade da função edit distance" e "edit distance em função da qualidade", o escore gerado pela função $levenshtein$ é 36%. Discussões e avaliações de várias funções podem ser encontradas na literatura (COHEN; RAVIKUMAR; FIENBERG, 2003; SILVA et al., 2007).

O algoritmo $DigitalObjectMatch$ inicia checando se as datas de publicação dos objetos digitais são compatíveis (linha 1). Logo após, são extraídas as iniciais dos nomes dos autores (linhas 2-5) e adicionadas as listas $iniList_i$. É realizado o casamento entre os autores utilizando a função $NameMatch$ (linha 6). Somente os objetos digitais que atenderem ao limiar de casamento têm seus títulos comparados pela função $Levenshtein$. O algoritmo retorna verdadeiro (linha 8) se as condições impostas pelas funções $YearSim$, $NameMatch$ e $Levenshtein$ forem atendidas. A qualquer momento, se uma das condições não for atendida, o algoritmo retorna falso (linha 9).

A função $NameMatch$ proposta tende a obter a revocação máxima, ou seja, entre os casamentos de objetos digitais recuperados estarão todos os casamentos relevantes. A precisão é mínima, visto que o número de casamentos recuperados pode ser muito maior do que o número de casamentos relevantes. O motivo deste comportamento é explicado pela função $IniSim$. Considere os nomes de autores "Eduardo Borges" e "Batista, Erídio". Quando aplicada a função $IniSim$ sobre as iniciais destes autores (EB e BE), é

retornado o valor de similaridade $s = 1$ indicando a correspondência entre as iniciais, embora o nome dos autores não representem o mesmo objeto. A precisão da deduplicação atinge valores aceitáveis quando as instâncias que satisfizerem as condições impostas pela métrica de similaridade $NameMatch$ forem avaliadas pela função $Levenshtein$. O metadado que representa o título do objeto digital só será comparado para os pares de publicações em que a função $NameMatch$ não retorne zero. Isto reduz o tempo de processamento no processo de deduplicação, pois o algoritmo proposto limita as comparações do restante dos metadados.

3.2.3 Um Exemplo de Deduplicação

Para exemplificar o processo de deduplicação proposto, esta seção apresenta, passo a passo, a computação realizada para todo o processo de casamento de dados. A Figura 3.2 mostra os dois objetos alvo da deduplicação provenientes de duas bibliotecas digitais distintas.

Objeto Digital A

```
01 <title>Deduplicação de Objetos em Bibliotecas Digitais</title>
02 <creator>Eduardo N. Borges</creator>
03 <creator>Galante, Renata</creator>
04 <creator>Heuser, C. A.</creator>
05 <date>2008</date>
06 <source>Simpósio Brasileiro de Bancos de Dados</source>
```

Objeto Digital B

```
07 <author>Renata de Mattos Galante</author>
08 <author>Borges, Eduardo</author>
09 <author>Heuser, Carlos</author>
10 <title>Deduplicando Objetos em Bibliotecas Digitais</title>
11 <booktitle>Simp. Brasileiro de Bancos de Dados</booktitle>
12 <year>2008</year>
```

Figura 3.2: Objetos alvo da dedupicação.

Foram utilizados os valores de limiares de similaridade:

- $t_Y = 0$ para a função $YearSim$, para que sejam comparados somente objetos que possuam o mesmo ano de publicação;
- $t_N = 100\%$ para a função $NameMatch$, para que sejam comparados objetos de mesmo número de autores;
- $t_L = 75\%$ para a função $Levenshtein$, para que sejam retornados objetos que difiram pouco nos títulos.

Através da técnica descrita na seção 3.2.1, são gerados os seguintes mapeamentos:

- A.title $\Leftrightarrow$ B.title
- A.creator $\Leftrightarrow$ B.author
- A.date $\Leftrightarrow$ B.year

O algoritmo $DigtalObjectMatch$ (seção 3.2.2.5) é aplicado aos objetos e inicialmente verifica a compatibilidade das datas de publicação dos objetos digitais através da função $YearSim$(A.date, B.year, 0). Como ambas as datas são iguais (2008), a diferença entre os anos de publicação é zero, respeitando o limiar $t_Y = 0$. Deste modo, o algoritmo prossegue sua execução.

A seguir são extraídas as iniciais dos nomes dos autores do objeto A, através da função $Initials$ (seção 3.2.2.2), e inseridas em uma lista. O mesmo ocorre para o objeto B gerando as seguintes listas:

- $iniList_A$ - (ENB, GR, HCA)
- $iniList_B$ - (RdMG, BE, HC)

As listas são utilizadas como parâmetros da função $NameMatch$ (seção 3.2.2.4) junto com o limiar $t_N = 100\%$ indicando que todos os autores de um objeto devem corresponder aos autores do outro. As listas são percorridas e é aplicada a função $IniSim$ (seçaõ 3.2.2.3) aos pares de iniciais formados por elementos de ambas as listas. Durante a execução do algoritmo $NameMatch$ é verificado que ENB corresponde a BE $(a_1 = b_2 \wedge a_m = b_1)$, GR corresponde a RdMG $(a_1 = b_n \wedge a_2 = b_1)$ e HCA corresponde a HC $(a_1 = b_1 \wedge a_2 = b_2)$. A cada casamento de iniciais encontrado é incrementado um contador. A razão entre o contador e o número de elementos da maior lista, neste caso $3/3 = 1$ define a porcentagem de casamentos encontrados. A função $NameMatch$ retorna 1, pois foi atendido o limiar de 100% de casamentos.

Após é realizado o cálculo da similaridade entre os títulos dos objetos digitais através da função $Levenshtein$ (LEVENSHTEIN, 1966). A distância de edição entre os parâmetros "*Deduplicação de Objetos em Bibliotecas Digitais*" e "*Deduplicando Objetos em Bibliotecas Digitais*" assume o valor 5. Conseqüentemente, a similaridade retornada é $1 - 5/47 = 89.36\%$ (47 é o número de caracteres da maior palavra). Como o valor retornado é maior que o limiar $t_L = 75\%$, o algoritmo finaliza a execução retornando verdadeiro.

3.3 Proveniência de Metadados

Esta seção define o processo de integração dos metadados e rastreio da proveniência. A proveniência é representada através de uma estrutura baseada em árvore. É especificado um algoritmo de integração que realiza a fusão dos metadados de objetos replicados e recupera a linhagem de cada metadado.

A proveniência modelada é orientada a dados, com granularidade fina, ou seja, o produto de dados em questão é um metadado qualquer que descreve um objeto digital, como por exemplo, *author*. É utilizada a abordagem imediata (*eager*) para representar a proveniência dos metadados. A proveniência é rastreada quando um dado é derivado (transformado), ou seja, quando ocorre a integração dos metadados das diferentes bibliotecas digitais. Somente um passo de derivação é suportado. Não é possível rastrear a proveniência de um metadado que derive de outro sistema de integração de bibliotecas digitais. A proveniência é computada periodicamente.

3.3.1 A Estrutura para Representação da Proveniência

Um *objeto digital integrado* representa a fusão dos metadados de objetos digitais replicados através de uma árvore de metadados e proveniências. A Figura 3.3 mostra a estrutura proposta de um objeto digital integrado, onde:

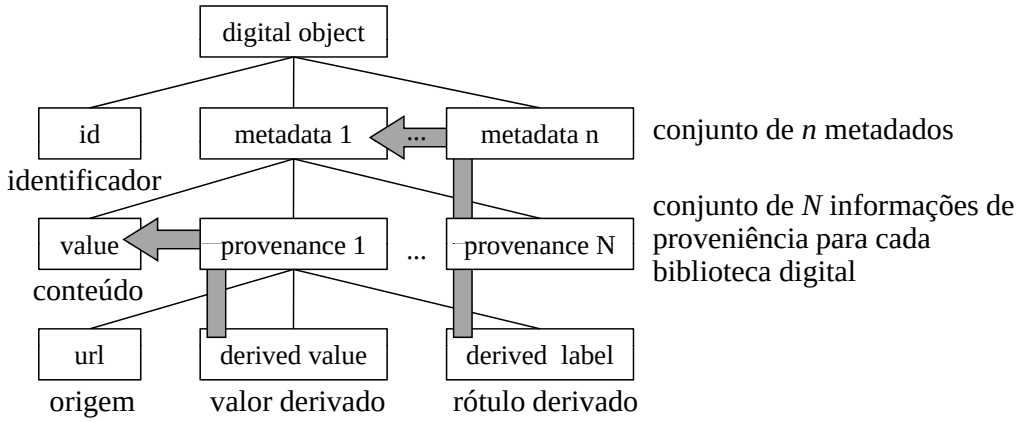

Figura 3.3: Estrutura de um objeto digital integrado.

- o nodo raiz possui rótulo $digitalobject$;
- o segundo nível da árvore é composto por um nodo de rótulo id e um conjunto de n outros nodos cujos rótulos identificam os metadados que descrevem o objeto digital;
- o nodo de rótulo id possui valor igual ao identificador único do objeto digital no sistema de integração de bibliotecas digitais;
- todos os nodos do conjunto de n metadados formam subárvores que possuem como descendentes diretos da raiz um nodo de rótulo $value$ e um conjunto de N outros nodos cujos rótulos são $provenance$;
- todos os nodos do conjunto de N informações de proveniência (uma para cada biblioteca digital que referencia o objeto) formam subárvores que possuem como descendentes diretos da raiz três nodos de rótulo url, $derivedlabel$ e $derivedvalue$;
- o nodo de rótulo url possui valor igual a URL da biblioteca digital de onde foi adquirido o metadado, ou seja, o endereço da biblioteca digital responsável pelas informações de proveniência definidas nos nodos $derivedvalue$ e $derivedlabel$;
- o nodo de rótulo $derivedvalue$ possui valor igual ao valor original do metadado, o qual foi derivado durante o processo de integração;
- o nodo de rótulo $derivedlabel$ possui valor igual ao rótulo original do metadado, o qual foi derivado durante o processo de integração;
- o nodo de rótulo $value$ possui valor igual ao valor mais freqüente dos nodos de rótulo $derivedvalue$.

Os valores dos nodos de rótulo $derivedvalue$ e $derivedlabel$ indicam os estados originais do rótulo ($value$) e valor ($metadata_i$) do metadado integrado. Por exemplo, a Figura 3.4 mostra um objeto digital integrado formado por um conjunto de metadados que inclui $publication\ year$, $title$ e $abstract$. O metadado $publication\ year$ possui três conjuntos de informações de proveniência. O ano de publicação do objeto digital é representado pelo rótulo $date$ e possui o valor 2005 na biblioteca digital 1. Na biblioteca digital 2, o mesmo metadado é representado pelo rótulo $year$ e também possui valor 2005. Já para a

biblioteca digital 3, o valor do metadado $year$ é 2004. A partir das informações de proveniência, o metadado $value$ assume o valor 2005 que é o valor mais freqüente entre os nodos $derivedvalue$.

```
<digital object>
 <publication year>
   <value>2005</value>
    <provenance>
      <url>http://biblioteca1.com</url>
      <derived value>2005</derived label>
      <derived label>date</derived label>
    </provenance>
    <provenance>
       < url>http://biblioteca2.com</url>
       <derived value>2005</derived label>
       <derived label>year</derived label>
    </provenance>
    <provenance>
       < url>http://biblioteca3.com</url>
       <derived value>2004</derived label>
       <derived label>year</derived label>
    </provenance>
 <publication year>

 <title> ... </title>
 ...
 <abstract> ... </abstract>
 ...
</digitalobject>
```

Figura 3.4: Exemplo de um objeto digital integrado.

3.3.2 O Algoritmo MetadataProv

Durante o processo de integração dos metadados replicados, informações de proveniência são recuperadas para a composição da árvore de metadados e proveniências. A integração dos metadados de um objeto digital deduplicado é realizada através do algoritmo $MetadataProv$, que é definido pelo Algoritmo 4,

Algoritmo 4 *MetadataProv*

```
METADATAPROV(L, IDO)
 1  INSERT(IDO, digitalobject, null);
 2  ident ← GENID();
 3  LEAF(id, ident);
 4  INSERT(IDO, id, digitalobject);
 5  for i ← 1 to LENGTH(L)
 6      do for all ϵ ∈ L_i
 7             do label ← MAPPING(ϵ.η);
 8                if SEARCH(IDO, label) = null
 9                  then INSERT(IDO, label, digitalobject);
10                INSERT(IDO, provenance, label);
11                LEAF(url, URL(L_i));
12                INSERT(IDO, url, provenance);
13                LEAF(derivedlabel, ϵ.η);
14                INSERT(IDO, derivedlabel, provenance);
15                LEAF(derivedvalue, ϵ.ν);
16                INSERT(IDO, derivedvalue, provenance);
17  for all CHILD(digitalobject)
18      do LEAF(value, MOSTFREQUENT(
19            SEARCH(CHILD(digitalobject), derivedvalue).ν));
20         INSERT(IDO, value, CHILD(digitalobject));
21  return IDO;
```

onde:

- o parâmetro L é uma lista de objetos digitais replicados;
- o parâmetro IDO é uma árvore, inicialmente vazia, que representa um objeto digital integrado;
- INSERT(T, η_d, η) insere um nodo de rótulo η_d na árvore T como descendente do nodo de rótulo η (quando η é nulo, trata-se da exceção inclusão da raiz);
- GENID é uma função que gera um identificador único para cada objeto digital integrado. Este identificador é armazenado na variável $ident$;
- LEAF(ν, η) cria um nodo folha de rótulo η e atribui o valor ν a este nodo;
- LENGTH é uma função que retorna o tamanho de uma lista;
- $\epsilon \in L_i$ é um metadado do i-ésimo objeto digital da lista L;
- $\epsilon.\eta$ e $\epsilon.\nu$ são, respectivamente, o rótulo e o valor do metadado ϵ;

- MAPPING é uma função que mapeia o rótulo de um metadado para o rótulo correspondente no esquema BibTEX (através da técnica apresentada na seção 3.2.1). O rótulo mapeado é armazenado na variável $label$;
- SEARCH(T, η) busca por nodos de rótulo η na árvore T e retorna apontadores para estes nodos;
- URL é uma função que retorna a URL da biblioteca digital de origem de um objeto digital;
- CHILD(η) retorna um apontador para um nodo descendente do nodo de rótulo η;
- MOSTFREQUENT é uma função que retorna o valor mais freqüente entre uma lista de valores.

O algoritmo $MetadataProv$ recebe como parâmetros uma lista de objetos digitais duplicados L e uma árvore vazia IDO (*Integrated Digital Object*). O algoritmo constrói a árvore IDO com a integração dos metadados e as respectivas proveniências e retorna um apontador para esta árvore.

Inicialmente, a raiz da árvore é definida por um nodo cujo rótulo é $digitalobject$. É atribuído ao objeto digital integrado um identificador único através da função GENID (linha 2). Um nodo de rótulo id e de valor igual ao identificador gerado é inserido como descendente da raiz (linhas 3-4). O laço compreendido entre as linhas 5 e 16 percorre a lista de objetos digitais duplicados. Para cada objeto, o laço compreendido entre as linhas 6 e 16 recupera os metadados ϵ do objeto L_i que são processados a fim de compor a árvore de proveniência e realizar a integração.

O rótulo do metadado $\epsilon.\eta$ mapeado para o formato BibTEX é atribuído à variável $label$ (linha 7). Se o metadado ainda não existir no objeto digital integrado (linha 8), o respectivo nodo é adicionado como descendente da raiz (linha 9). Após, são inseridas as informações de proveniência compostas pela subárvore $provenance$ (linhas 10-16). A biblioteca digital de origem do objeto L_i, retornada pela função URL, é atribuída ao valor de um nodo que é inserido como descendente de $provenance$ (linhas 11-12). Outros dois nodos são inseridos, os quais atribuem, respectivamente, o rótulo η e o valor ν do metadado ϵ aos valores dos nodos de rótulos $derivedlabel$ e $devivedvalue$ (13-16).

Por fim, o laço compreendido entre as linhas 17 e 20 insere nodos de rótulo $value$ no terceiro nível da árvore. Para cada descendente da raiz, são buscados os valores dos nodos de rótulo $derivedvalue$ pertencentes à subárvore (linha 19). O valor mais freqüente entre os valores derivados das diversas bibliotecas digitais, retornado pela função MOSTFREQUENT, é atribuído aos nodos de rótulo $value$ (linha 18) que são adicionados à arvore (linha 20).

O algoritmo $MetadataProv$ finaliza a execução retornando um apontador para a árvore IDO (linha 21) que contém a integração dos metadados do objeto digital deduplicado com as respectivas proveniências.

3.3.3 Um Exemplo de Integração e Rastreio da Proveniência

A fim de exemplificar a estrutura de um documento digital integrado e do algoritmo $MetadataProv$, considere que os três objetos digitais presentes na Figura 3.5 tenham passado pelo processo de deduplicação (seção 3.2) e foram identificados como réplicas. As diferentes representações do mesmo objeto digital são provenientes de três bibliotecas digitais distintas.

BDBComp

```
1  <title>A Computer Vision Framework for Remote Eye Gaze Tracking</title>
2  <creator>Carlos H. Morimoto</creator>
3  <source>sibgrapi2003</source>
```

DBLP

```
4  <title>A Computer Vision Framework for Eye Gaze Tracking</title>
5  <author>Carlos Hitoshi Morimoto</author>
6  <booktitle>SIBGRAPI</booktitle>
```

IEEE Xplore

```
7  <title>A computer vision framework for eye gaze tracking</title>
8  <author>Morimoto, C.H.</author>
9  <pages>406</pages>
```

Figura 3.5: Objeto digital deduplicado.

O algoritmo $MetadataProv$ recebe como parâmetros uma lista composta pelos três objetos duplicados e uma árvore vazia. A partir dos objetos replicados, é realizada a integração dos metadados e o rastreio da proveniência. A árvore gerada é apresentada como um documento XML apresentado na Figura 3.6.

Inicialmente, é definida a raiz da árvore com o rótulo *digitalobject* e o identificador do objeto *00123* é inserido como descendente da raiz (Figura 3.6, linhas 01-02).

O primeiro objeto digital, proveniente da BDBComp (Figura 3.5, linha 1), possui *title* como primeiro metadado. Um nodo de mesmo rótulo (*title* é mapeado como *title*) é inserido como descendente da raiz (Figura 3.6, linha 03). Uma subárvore de proveniência é inserida como descendente do nodo de rótulo *title* (Figura 3.6, linha 04). A biblioteca digital de origem do objeto é inserida no nodo de rótulo *url* (Figura 3.6, linha 05). O rótulo e o valor do metadado são atribuídos aos nodos *derivedlabel* e *derivedvalue* (Figura 3.6, linhas 06-07). O mesmo procedimento é realizado para os metadados *creator* e *source* (Figura 3.5, linhas 2-3) gerando os nodos *author*, *booktitle* e as subárvores de proveniência (Figura 3.6, linhas 20, 37, 21-25 e 38-42).

O segundo objeto digital, proveniente da DBLP, é avaliado. Nodos de rótulos iguais ao mapeamento dos metadados *title*, *author*, e *booktitle* (Figura 3.5, linhas 4-6) já existem na árvore do objeto digital integrado (Figura 3.6, linhas 3, 20, 37), portanto não há necessidade de criá-los. Uma subárvore de proveniência é criada para cada metadado (Figura 3.6, linhas 09-13, 26-30, 43-47).

O terceiro objeto digital, proveniente da IEEE Xplore, é avaliado. Ocorre o mesmo procedimento do segundo objeto para os metadados *title* e *author* (Figura 3.5, linhas 7-8) gerando as subárvores de proveniência (Figura 3.6, linhas 14-18 e 31-35). Como não existe um nodo de rótulo igual ao mapeamento do metadado *pages*, este nodo é inserido como descendente da raiz (Figura 3.6, linha 49). Uma subárvore de proveniência é criada para o metadado *pages* (Figura 3.6, linhas 50-54).

Por fim, é inserido um nodo de rótulo *value* como descendente de cada metadado (Figura 3.6, linhas 03, 20, 37, 49). Os valores dos nodos são definidos com base nos valores derivados (*derivedvalue*). É escolhida a representação mais freqüente das fontes de origem. Por exemplo, é atribuído o valor *A Computer Vision Framework for Eye Gaze Tracking* (Figura 3.6, linha 03), pois duas das três fontes possuem nodos de rótulo *derivedvalue* com este conteúdo (Figura 3.6, linhas 07, 12, 17). Quando todas as representações são diferentes, uma delas é escolhida aleatoriamente.

Omitindo-se as subárvores cujo rótulo do nodo raiz é *provenance* da resposta de uma consulta, a proveniência é omitida e é exibida a integração dos metadados disponíveis nas diversas fontes livre de informação redundante e sem perda de informação relevante.

```
<digitalobject>
  <id>00123</id>
  <title> <value>A Computer Vision Framework for Eye Gaze Tracking</value>
      <provenance>
          <url>http://www.lbd.dcc.ufmg.br/bdbcomp</url>
          <derivedlabel>title</derivedlabel>
          <derivedvalue>A Computer Vision Framework for Remote Eye Gaze Tracking</derivedvalue>
      </provenance>
      <provenance>
          <url>http://dblp.uni-trier.de</url>
          <derivedlabel>title</derivedlabel>
          <derivedvalue>A Computer Vision Framework for Eye Gaze Tracking</derivedvalue>
      </provenance>
      <provenance>
          <url>http://ieeexplore.ieee.org</url>
          <derivedlabel>title</derivedlabel>
          <derivedvalue>A computer vision framework for eye gaze tracking</derivedvalue>
      </provenance>
  </title>
  <author> <value>Carlos Hitoshi Morimoto</value>
      <provenance>
          <url>http://www.lbd.dcc.ufmg.br/bdbcomp</url>
          <derivedlabel>creator</derivedlabel>
          <derivedvalue>Carlos H. Morimoto</derivedvalue>
      </provenance>
      <provenance>
          <url>http://dblp.uni-trier.de</url>
          <derivedlabel>author</derivedlabel>
          <derivedvalue>Carlos Hitoshi Morimoto</derivedvalue>
      </provenance>
      <provenance>
          <url>http://ieeexplore.ieee.org</url>
          <derivedlabel>author</derivedlabel>
          <derivedvalue>Morimoto, C.H.</derivedvalue>
      </provenance>
  </author>
  <booktitle> <value>SIBGRAPI</value>
      <provenance>
          <url>http://www.lbd.dcc.ufmg.br/bdbcomp</url>
          <derivedlabel>source</derivedlabel>
          <derivedvalue>sibgrapi2003</derivedvalue>
      </provenance>
      <provenance>
          <url>http://dblp.uni-trier.de</url>
          <derivedlabel>booktitle</derivedlabel>
          <derivedvalue>SIBGRAPI</derivedvalue>
      </provenance>
  </booktitle>
  <pages> <value>406</value>
      <provenance>
          <url>http://ieeexplore.ieee.org</url>
          <derivedlabel>pages</derivedlabel>
          <derivedvalue>406</derivedvalue>
      </provenance>
  </pages>
</digitalobject>
```

Figura 3.6: Proveniência do objeto digital processada.

3.4 Considerações Finais

Este capítulo apresentou o mecanismo MD-PROM proposto para deduplicar objetos e recuperar a proveniência de metadados oriundos de bibliotecas digitais. Foram apresentadas a visão geral do MD-PROM e a especificação de suas propriedades, que incluem a abordagem para deduplicar objetos digitais (casamento de esquemas e o conteúdo dos metadados) e os detalhes do processamento da proveniência.

A principal contribuição do trabalho apresentado nesta dissertação é melhorar a qualidade da pesquisa do usuário de bibliotecas digitais, fornecendo uma resposta única e livre de redundância para consultas a metadados de publicações científicas obtidos através da *Web*. O diferencial da nossa proposta é a especificação de uma estrutura de dados capaz de identificar a origem de metadados do domínio das bibliotecas digitais e os valores dos quais estes metadados foram derivados.

A proposta apresentada neste capítulo traz contribuições tanto na área de deduplicação quanto na área de proveniência de dados. É proposto um casamento mínimo entre os diferentes esquemas dos metadados através da correspondência direta entre rótulos nos formatos OAI Dublin Core e BibTEX. São especificadas várias funções de similaridade que identificam variações na representação da autoria de um objeto digital. É proposto um algoritmo de casamento de objetos digitais que deduplica objetos digitais provenientes de bibliotecas digitais distintas. São definidos uma estrutura de dados e um algoritmo de integração que funde os metadados replicados e recupera informações de proveniência para cada metadado integrado.

As tabelas 3.4 e 3.5 apresentam uma análise comparativa do trabalho proposto frente os trabalhos apresentados no capítulo 2.

Tabela 3.4: Comparativo MD-PROM e trabalhos relacionados: deduplicação de objetos.

Trabalho	**Combinação**	**Função**	**Área**	**A**	**Pesos**	**Nomes**
MD-PROM	algoritmo *digital object match*	funções específicas do domínio	bibliotecas digitais	C	NA	V
(CHAUDHURI et al., 2003)	algoritmo fuzzy baseado na IDF	NA	*data cleaning*	T	V	X
(GUHA et al., 2004)	combinação de *rankings*	NA	*data cleaning*	C	NA	X
(CARVALHO; SILVA, 2003)	estratégias baseadas no modelo vetorial	NA	deduplicação de documentos XML	C	V	X
(CULOTTA; MCCALLUM, 2005)	dependências relacionais	NA	integração de dados	C	V	X
(DORNELES et al., 2004)	MCV	MAV	consultas aproximadas	T	NA	X
(DORNELES et al., 2007)	MCV e escore ajustado	MAV	consultas aproximadas	T	NA	X
Active Atlas (TEJADA; KNOBLOCK; MINTON, 2001)	árvores de decisão	transformações, funções específicas do domínio	integração de dados	C	X	X
(COHEN; RICHMAN, 2002)	algoritmo de aprendizado	funções específicas do domínio	integração de dados	C	V	X
Marlin (BILENKO; MOONEY, 2003)	classificador binário	EM e SVM	consultas aproximadas	C	V	X
(CARVALHO et al., 2006)	programação genética	funções geradas automaticamente	bibliotecas digitais	T	V	V

Legenda: C comuns T todos V utiliza X não utiliza NA não aplicável

Tabela 3.5: Comparativo IniSim e trabalhos relacionados: deduplicação de nomes próprios.

Características	IniSim	Guth	Acronyms	Fragmentos
independência de idioma	V	V	V	V
variações de grafia	V	limitado	V	V
token	V	X	V	V
caractere	V	V	V	V
iniciais	V	X	limitado	limitado
inversões	V	X	X	parcial
complexidade	linear	linear	linear	quadrática

Legenda: V sim X não

Quando comparado a trabalhos na área de deduplicação de objetos, o MD-PROM difere de todas as propostas porque não utiliza pesos para avaliar a importância de cada atributo, comparando somente atributos específicos (ano, título e autores), os quais possuem grande importância na deduplicação de objetos digitais. Estes atributos são comparados utilizando as funções propostas $YearSim$, $IniSim$ e $NameMatch$ e a função $Levenshtein$ que são específicas do domínio das bibliotecas digitais. O processo de combinação dos escores de similaridade gerados pelas funções é definido pelo algoritmo *Digital Object Match*.

A função proposta $IniSim$ possui complexidade linear e suporta variações de grafia, como a maioria dos trabalhos de casamento de nomes analisados. No entanto, o MD-PROM soluciona o suporte a iniciais que é inexistente ou limitado nas abordagens estudadas. Além disso, o suporte a inversões de nomes, que é parcial para o algoritmo $Fragmentos$, é resolvido pela função $IniSim$.

Quanto à proveniência, o domínio de aplicação não diz respeito a dados de experimentos científicos (*e-science*), e sim ao domínio das bibliotecas digitais. A granularidade da proveniência adotada pela CCSB (ALF-CHRISTIAN ACHILLES, 2007) é de uma publicação. Na maioria dos sistemas a granularidade são arquivos. Com o MD-PROM, é possível recuperar a linhagem de cada metadado para cada objeto digital (granularidade fina). Além de rastrear a proveniência dos metadados, o algoritmo proposto $Metadataprov$ realiza a integração dos metadados oriundos de bibliotecas digitais distintas e não polui a interface do sistema com a redundância dos metadados.

4 AVALIAÇÃO EXPERIMENTAL

Este capítulo apresenta os experimentos realizados que visam determinar a qualidade das funções e algoritmos propostos para a identificação de objetos digitais duplicados. Os algoritmos propostos são comparados aos algoritmos estudados (*Guth*, *Acronyms*, *Fragments* e *MCV Set*) apresentados na revisão bibliográfica. Os metadados utilizados nos experimentos são provenientes das bibliotecas digitais BDBComp e DBLP.

Este capítulo está organizado da seguinte forma: a seção 4.1 apresenta as métricas de avaliação dos experimentos. A plataforma de Hardware e Software utilizada é definida na seção 4.2. A seção 4.3 descreve a Base de Dados utilizada. Os experimentos realizados e os resultados alcançados são apresentados na seção 4.4. O capítulo é finalizado na seção 4.5 com uma análise dos resultados.

4.1 Métricas de Avaliação

Os experimentos realizados utilizam as seguintes métricas para avaliação dos resultados:

- precisão (*precision*) (MANNING; RAGHAVAN; SCHÜTZE, 2008) - mensura a taxa de acerto do componente Identificação do MD-PROM. A precisão define, portanto, a porcentagem de pares de objetos digitais identificados corretamente em relação ao número de pares retornados;

- revocação (*recall*) (MANNING; RAGHAVAN; SCHÜTZE, 2008) - mensura fração das respostas relevantes retornadas pelo MD-PROM. A revocação define, portanto, a porcentagem de pares de objetos digitais identificados corretamente em relação ao número de objetos digitais duplicados existentes;

- medida F balanceada (*balanced F-measure*) (MANNING; RAGHAVAN; SCHÜTZE, 2008) - mensura a relação entre a precisão e a revocação. É a média harmônica ponderada da precisão e da revocação, a qual atribui pesos iguais à precisão e revocação;

- tempo de processamento - indica o tempo em segundos necessário para a deduplicação dos objetos digitais;

- teste T (*Student's t-test*) (WIKIPEDIA, 2008) - verifica se o desempenho de dois algoritmos é estatisticamente significativo. O teste T avalia se as médias de duas distribuições normais de valores são estatisticamente diferentes. Segundo Hull (1993), o teste T apresenta bons resultados mesmo quando as distribuições não são perfeitamente normais. Foi utilizado o limiar de significância estatística $\alpha = 0,05$. Quando

o valor P bi-caudal calculado é menor que α, existe uma diferença significativa entre os desempenhos dos dois algoritmos analisados. O melhor desempenho é do algoritmo com a maior média de distribuição.

4.2 Plataforma de Trabalho

Os experimentos foram executados em um micro computador com placa mãe ASUS A7N8X-Deluxe, processador AMD Athlon XP 2700 (2.17 GHz), 1 GB de memória DDR2 400 MHz (2x 512 MB *dual channel*) e disco rígido IDE 5400 RPM de 40GB.

Todas as funções e algoritmos utilizados nos experimentos foram implementados na linguagem PL/pgSQL[1]. Os experimentos utilizaram o SGBD PostgreSQL 8.3[2], rodando sobre o sistema operacional Linux Ubuntu 7.10[3], para o armazenamento das bases de dados e a realização das consultas.

4.3 Base de Dados

Os metadados utilizados nos experimentos são provenientes das bibliotecas digitais BDBComp e DBLP. 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, 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.

O MD-PROM realiza um pré-processamento dos metadados XML com o objetivo de carregar em um banco de dados relacional somente os metadados utilizados pelo mecanismo, obtendo como saída um *script* SQL gerado para popular a base relacional. O mapeamento entre os esquemas da BDBComp e DBLP é realizado durante esta etapa através de um componente desenvolvido na linguagem JAVA[4] utilizando a API SAX[5].

A base de teste é composta por três tabelas principais: *bdbcomp*, *dblp* e *match*. O esquema das relações é especificado na Figura 4.1. As relações *bdbcomp* e *dblp* possuem esquemas similares, onde id corresponde ao identificador único de cada objeto digital. Este campo é gerado através de um contador durante o pré-processamento dos metadados XML, o qual é incrementado a cada elemento *oaidc:dc* ou *inproceedings* encontrados, respectivamente, em *bdbcomp.xml* ou *dblp.xml*. O atributo $title$ corresponde ao metadado *title* presente nas duas bibliotecas digitais. Uma lista composta pelos elementos *creator* (BDBComp) ou *author* (DBLP) corresponde ao campo $authors$ das tabelas. $year$ corresponde aos metadados *date* (BDBComp) ou *year* (DBLP). Por fim, o atributo $booktitle$ corresponde aos metadados *source* (BDBComp), com a omissão do ano de publicação, ou *booktitle* (DBLP). A relação *match* possui 2 atributos bid, did que referenciam os identificadores das tabelas *bdbcomp* e *dblp*. Estes pares de identificadores representam os

[1] http://www.postgresql.org/docs/8.3/static/plpgsql.html
[2] http://www.postgresql.org
[3] http://www.ubuntu.com
[4] http://java.sun.com
[5] http://www.saxproject.org

casamentos identificados por um usuário especialista, ou seja, as correspondências entre os objetos digitais provenientes da BDBComp e da DBLP, totalizando 415 ocorrências.

```
CREATE TABLE bdbcomp (
    id integer NOT NULL,
    title text,
    authors text,
    "year" integer,
    booktitle text,
    CONSTRAINT bdbcomp_pkey PRIMARY KEY (id)
)

CREATE INDEX bdbcomp_year
    ON bdbcomp USING btree (year);

CREATE TABLE match (
    bid integer NOT NULL,
    did integer NOT NULL,
    CONSTRAINT match_pkey PRIMARY KEY (bid, did),
    CONSTRAINT match_bid_fkey FOREIGN KEY (bid)
        REFERENCES bdbcomp (id),
    CONSTRAINT match_did_fkey FOREIGN KEY (did)
        REFERENCES dblp (id)
)
```

```
CREATE TABLE dblp (
    id integer NOT NULL,
    title text,
    authors text,
    "year" integer,
    booktitle text,
    CONSTRAINT dblp_pkey PRIMARY KEY (id)
)

CREATE INDEX dblp_year
    ON dblp USING btree (year);
```

Figura 4.1: Esquema da base de dados relacional.

4.4 Experimentos Realizados

Para realização dos experimentos foram selecionados os metadados que descrevem os artigos científicos publicados nas conferências Escola Regional de Bancos de Dados (ERBD), Simpósio Brasileiro de Bancos de Dados (SBBD), *Web Information and Data Management* (WIDM), *International Conference on Conceptual Modeling* (ER), *Conference on Advanced Information Systems Engineering* (CAiSE), Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens (SIBGRAPI), *Computer Graphics International* (CGI), *Symposium on Virtual and Augmented Reality* (SVR), *IFIP Conference on Human-Computer Interaction* (INTERACT) e Simpósio Brasileiro de Inteligência Artificial (SBIA).

A escolha destas conferências é baseada no fato de que alguns pesquisadores brasileiros que publicam trabalhos nas conferências nacionais, estendem os artigos publicados e os submetem para conferências internacionais de mesma área do conhecimento ou áreas correlatas. A Tabela 4.1 indica o número de objetos em cada biblioteca digital para cada conferência, bem como o intervalo de anos de publicação selecionado.

Tabela 4.1: Objetos digitais utilizados nos experimentos.

Conferência	Intervalo	BDBComp	DBLP	Área do Conhecimento	Abrangência
ERBD	2005	18		Bancos de Dados	Nacional
SBBD	2001-2005	122	143	Bancos de Dados	Nacional
WIDM	2001-2004		76	Bancos de Dados	Internacional
ER	2001-2004		214	Sistemas de Informação	Internacional
CAiSE	2001-2004		192	Sistemas de Informação	Internacional
SIBGRAPI	2001-2004	242	274	Computação Gráfica	Nacional
CGI	2001-2004		196	Computação Gráfica	Internacional
SVR	2001-2004	107		Realidade Virtual	Nacional
INTERACT	2003		215	Interação Humano-Computador	Internacional
SBIA	2002-2004	93	93	Inteligência Artificial	Nacional

As conferências SBBD, SBIA e SIBGRAPI são indexadas pelas duas bibliotecas digitais, logo existem metadados duplicados. Os experimentos realizados têm como objetivo

específico deduplicar os 415 objetos duplicados destas três conferências.

Diversas consultas foram realizadas sobre a base de dados visando identificar os objetos duplicados nas bibliotecas digitais BDBComp e DBLP. A Figura 4.2 apresenta a estrutura das consultas. Cada consulta retorna um par de identificadores $b.id, d.id$ que representam um par de objetos duplicados nas tabelas $bdbcomp$ e $dblp$. São comparados objetos digitais que variem o ano de publicação em t_Y anos. O algoritmo de casamento e a respectiva função de comparação de nomes é aplicada aos autores de cada par de objetos. Quando aplicável, é utilizado um limiar t_A. Os objeto digitais que satisfaçam as condições impostas pelo algoritmo de casamento ainda têm os títulos comparados pela função $levenshtein$ com o limiar t_L.

```
SELECT b.id, d.id
FROM dblp d, bdbcomp b
WHERE b.year BETWEEN d.year-tY AND d.year+tY
AND algorithm (b.authors, d. authors, tA) = 1
AND levenshtein (b.title, d.title) >= tL;
```

Figura 4.2: Estrutura das consultas realizadas sobre a base de dados.

Os experimentos 1-9 variam os seguintes parâmetros:

- t_L - limiar de similaridade aplicado ao algoritmo $Levenshtein$ que opera os títulos dos objetos digitais.
- t_Y - Indica a variação no ano de publicação. Consultas que não utilizam este limiar comparam por igualdade os anos de publicação dos objetos digitais.

Os resultados desses experimentos são sumarizados nas tabelas 4.2-4.10 onde C é o identificador da consulta e $Pares$ mostra o número de objetos duplicados identificados.

4.4.1 Experimento 1 - Qualidade da função *Guth* utilizando a *MCV Set*

O experimento 1 analisa a qualidade da função $Guth$ (seção 2.3.1) utilizando a *MCV Set* (seção 2.1.1, equação 2.1) para a combinação dos escores gerados pela função.

O objetivo do experimento é calcular a precisão, revocação, medida F e o tempo de processamento da deduplicação dos objetos digitais utilizando $Guth$ e *MCV Set*.

Os resultados do experimento 1 são sumarizados na tabela 4.2.

Tabela 4.2: Qualidade e tempo de processamento do experimento 1.

C	Alg. Nomes	Alg. Casamento	t_Y	t_L	Pares	Precisão	Revocação	Medida F	Tempo(s)
1	Guth	MCV Set		0,5	99	98,99%	23,61%	38,13%	9,6
2	Guth	MCV Set		0,7	93	100,00%	22,41%	36,61%	9,5
3	Guth	MCV Set	1	0,5	99	98,99%	23,61%	38,13%¨	23,1
4	Guth	MCV Set	1	0,7	93	100,00%	22,41%	36,61%	22,9

4.4.2 Experimento 2 - Qualidade da função *Acronyms* utilizando a *MCV Set*

O experimento 2 analisa a qualidade da função $Acronyms$ (seção 2.3.2) utilizando a *MCV Set* (seção 2.1.1, equação 2.1) para a combinação dos escores gerados pela função.

O objetivo do experimento é calcular a precisão, revocação, medida F e o tempo de processamento da deduplicação dos objetos digitais utilizando $Acronyms$ e *MCV Set*.

Os resultados do experimento 2 são sumarizados na tabela 4.3.

Tabela 4.3: Qualidade e tempo de processamento do experimento 2.

C	Alg. Nomes	Alg. Casamento	t_Y	t_L	Pares	Precisão	Revocação	Medida F	Tempo(s)
5	Acronyms	MCV Set		0,5	267	99,25%	63,86%	77,71%	9,8
6	Acronyms	MCV Set		0,7	249	100,00%	60,00%	75,00%	9,6
7	Acronyms	MCV Set	1	0,5	269	98,51%	63,86%	77,49%¨	23,2
8	Acronyms	MCV Set	1	0,7	249	100,00%	60,00%	75,00%	23,1

4.4.3 Experimento 3 - Qualidade da função *Fragments* utilizando a *MCV Set*

O experimento 3 analisa a qualidade da função $Fragments$ (seção 2.3.3) utilizando a *MCV Set* (seção 2.1.1, equação 2.1) para a combinação dos escores gerados pela função.

O objetivo do experimento é calcular a precisão, revocação, medida F e o tempo de processamento da deduplicação dos objetos digitais utilizando $Fragments$ e *MCV Set*.

Além dos limiares t_L e t_Y, o experimento varia o parâmetro t_F - limiar de similaridade aplicado ao algoritmo $Fragments$.

Os resultados do experimento 3 são sumarizados na tabela 4.4.

Tabela 4.4: Qualidade e tempo de processamento do experimento 3.

C	Alg. Nomes	Alg. Casamento	t_Y	t_F	t_L	Pares	Precisão	Revocação	Medida F	Tempo(s)
9	Fragments	MCV Set		2	0,5	197	99,49%	47,23%	64,05%	9,2
10	Fragments	MCV Set		2	0,7	186	100,00%	44,82%	61,90%	9,0
11	Fragments	MCV Set		4	0,5	214	99,53%	51,33%	67,73%	9,2
12	Fragments	MCV Set		4	0,7	203	100,00%	48,92%	65,70%	10,2
13	Fragments	MCV Set	1	2	0,5	198	98,99%	47,23%	63,95%	24,5
14	Fragments	MCV Set	1	2	0,7	186	100,00%	44,82%	61,90%	22,6
15	Fragments	MCV Set	1	4	0,5	215	99,07%	51,33%	67,62%	26,2
16	Fragments	MCV Set	1	4	0,7	203	100,00%	48,92%	65,70%	25,6

4.4.4 Experimento 4 - Qualidade da função *IniSim* utilizando a *MCV Set*

O experimento 4 analisa a qualidade da função **proposta** $IniSim$ (seção 3.2.2.1) utilizando a *MCV Set* (seção 2.1.1, equação 2.1) para a combinação dos escores gerados pela função.

O objetivo do experimento é calcular a precisão, revocação, medida F e o tempo de processamento da deduplicação dos objetos digitais utilizando $IniSim$ e *MCV Set*.

Os resultados do experimento 4 são sumarizados na tabela 4.5 onde C é o identificador da consulta e $Pares$ mostra o número de objetos duplicados identificados.

Tabela 4.5: Qualidade e tempo de processamento do experimento 4.

C	Alg. Nomes	Alg. Casamento	t_Y	t_L	Pares	Precisão	Revocação	Medida F	Tempo(s)
17	IniSim	MCV Set		0,5	394	99,49%	94,46%	96,91%	11,0
18	IniSim	MCV Set		0,7	366	100,00%	88,19%	93,73%	10,6
19	IniSim	MCV Set	1	0,5	397	98,74%	94,46%	96,55%	24,4
20	IniSim	MCV Set	1	0,7	366	100,00%	88,19%	93,73%	26,4

4.4.5 Experimento 5 - Qualidade da função *Guth* utilizando *NameMatch*

O experimento 5 analisa a qualidade da função $Guth$ (seção 2.3.1) utilizando o algoritmo **proposto** $NameMatch$ (seção 3.2.2.4) para a combinação dos escores gerados pela função.

O objetivo do experimento é calcular a precisão, revocação, medida F e o tempo de processamento da deduplicação dos objetos digitais utilizando $Guth$ e $NameMatch$.

Além dos limiares t_L e t_Y, o experimento varia o parâmetro t_N - limiar de similaridade aplicado ao algoritmo $NameMatch$.

Os resultados do experimento 5 são sumarizados na tabela 4.6.

Tabela 4.6: Qualidade e tempo de processamento do experimento 5.

C	Alg. Nomes	Alg. Casamento	t_Y	t_N	t_L	Pares	Precisão	Revocação	Medida F	Tempo(s)
21	Guth	NameMatch		1	0,5	99	98,99%	23,61%	38,13%	11,4
22	Guth	NameMatch		1	0,7	93	100,00%	22,41%	36,61%	9,4
23	Guth	NameMatch		0,75	0,5	117	99,15%	27,95%	43,61%	10,6
24	Guth	NameMatch		0,75	0,7	110	100,00%	26,51%	41,90%	9,3
25	Guth	NameMatch	1	1	0,5	99	98,99%	23,61%	38,13%	25,1
26	Guth	NameMatch	1	1	0,7	93	100,00%	22,41%	36,61%	23,0
27	Guth	NameMatch	1	0,75	0,5	117	99,15%	27,95%	43,61%	23,1
28	Guth	NameMatch	1	0,75	0,7	110	100,00%	26,51%	41,90%	23,2

4.4.6 Experimento 6 - Qualidade da função *Acronyms* utilizando *NameMatch*

O experimento 6 analisa a qualidade da função $Acronyms$ (seção 2.3.2) utilizando o algoritmo **proposto** $NameMatch$ (seção 3.2.2.4) para a combinação dos escores gerados pela função.

O objetivo do experimento é calcular a precisão, revocação, medida F e o tempo de processamento da deduplicação dos objetos digitais utilizando $Acronyms$ e $NameMatch$.

Além dos limiares t_L e t_Y, o experimento varia o parâmetro t_N - limiar de similaridade aplicado ao algoritmo $NameMatch$.

Os resultados do experimento 6 são sumarizados na tabela 4.7.

Tabela 4.7: Qualidade e tempo de processamento do experimento 6.

C	Alg. Nomes	Alg. Casamento	t_Y	t_N	t_L	Pares	Precisão	Revocação	Medida F	Tempo(s)
29	Acronyms	NameMatch		1	0,5	148	99,32%	35,42%	52,22%	10,7
30	Acronyms	NameMatch		1	0,7	140	100,00%	33,73%	50,45%	10,5
31	Acronyms	NameMatch		0,75	0,5	188	98,94%	44,82%	61,69%	9,6
32	Acronyms	NameMatch		0,75	0,7	176	100,00%	42,41%	59,56%	9,5
33	Acronyms	NameMatch	1	1	0,5	149	98,66%	35,42%	52,13%	23,2
34	Acronyms	NameMatch	1	1	0,7	140	100,00%	33,73%	50,45%	23,0
35	Acronyms	NameMatch	1	0,75	0,5	190	97,89%	44,82%	61,49%	24,2
36	Acronyms	NameMatch	1	0,75	0,7	176	100,00%	42,41%	59,56%	23,0

4.4.7 Experimento 7 - Qualidade da função *Fragments* utilizando *NameMatch*

O experimento 7 analisa a qualidade da função $Fragments$ (seção 2.3.3) utilizando o algoritmo **proposto** $NameMatch$ (seção 3.2.2.4) para a combinação dos escores gerados pela função.

O objetivo do experimento é calcular a precisão, revocação, medida F e o tempo de processamento da deduplicação dos objetos digitais utilizando $Fragments$ e $NameMatch$.

Além dos limiares t_L e t_Y, o experimento varia os parâmetros t_N - limiar de similaridade aplicado ao algoritmo $NameMatch$ - e t_F - limiar de similaridade aplicado ao algoritmo $Fragments$.

Os resultados do experimento 7 são sumarizados na tabela 4.8.

Tabela 4.8: Qualidade e tempo de processamento do experimento 7.

C	Alg. Nomes	Alg. Casamento	t_Y	t_N	t_F	t_L	Pares	Precisão	Revocação	Medida F	Tempo(s)
37	Fragments	NameMatch		1	2	0,5	197	99,49%	47,23%	64,05%	9,0
38	Fragments	NameMatch		1	2	0,7	186	100,00%	44,82%	61,90%	9,0
39	Fragments	NameMatch		1	4	0,5	209	99,52%	50,12%	66,67%	9,0
40	Fragments	NameMatch		1	4	0,7	198	100,00%	47,71%	64,60%	9,0
41	Fragments	NameMatch		0,75	2	0,5	236	99,58%	56,63%	72,20%	9,1
42	Fragments	NameMatch		0,75	2	0,7	223	100,00%	53,73%	69,91%	9,0
43	Fragments	NameMatch		0,75	4	0,5	249	99,20%	59,52%	74,40%	9,1
44	Fragments	NameMatch		0,75	4	0,7	234	100,00%	56,39%	72,11%	9,0
45	Fragments	NameMatch	1	1	2	0,5	198	98,99%	47,23%	63,95%	2,3
46	Fragments	NameMatch	1	1	2	0,7	186	100,00%	44,82%	61,90%	22,6
47	Fragments	NameMatch	1	1	4	0,5	210	99,05%	50,12%	66,56%	22,6
48	Fragments	NameMatch	1	1	4	0,7	198	100,00%	47,71%	64,60%	22,5
49	Fragments	NameMatch	1	0,75	2	0,5	237	99,16%	56,63%	72,09%	25,6
50	Fragments	NameMatch	1	0,75	2	0,7	223	100,00%	53,73%	69,91%	23,3
51	Fragments	NameMatch	1	0,75	4	0,5	251	98,41%	59,52%	74,17%	23,4
52	Fragments	NameMatch	1	0,75	4	0,7	234	100,00%	56,39%	72,11%	23,4

4.4.8 Experimento 8 - Qualidade da função *IniSim* utilizando *NameMatch*

O experimento 8 analisa a qualidade da função **proposta** $IniSim$ (seção 2.3.2) utilizando o algoritmo **proposto** $NameMatch$ (seção 3.2.2.4) para a combinação dos escores gerados pela função.

O objetivo do experimento é calcular a precisão, revocação, medida F e o tempo de processamento da deduplicação dos objetos digitais utilizando $IniSim$ e $NameMatch$.

Além dos limiares t_L e t_Y, o experimento varia o parâmetro t_N - limiar de similaridade aplicado ao algoritmo $NameMatch$.

Os resultados do experimento 8 são sumarizados na tabela 4.9.

Tabela 4.9: Qualidade e tempo de processamento do experimento 8.

C	Alg. Nomes	Alg. Casamento	t_Y	t_N	t_L	Pares	Precisão	Revocação	Medida F	Tempo(s)
53	IniSim	NameMatch		1	0,5	389	99,49%	93,25%	96,27%	9,1
54	IniSim	NameMatch		1	0,7	361	100,00%	86,99%	93,04%	9,0
55	IniSim	NameMatch		0,75	0,5	397	99,50%	95,18%	97,29%	9,2
56	IniSim	NameMatch		0,75	0,7	369	100,00%	88,92%	94,13%	9,0
57	IniSim	NameMatch	1	1	0,5	392	98,72%	93,25%	95,91%	23,3
58	IniSim	NameMatch	1	1	0,7	361	100,00%	86,99%	93,04%	23,2
59	IniSim	NameMatch	1	0,75	0,5	400	98,75%	95,18%	96,93%	23,2
60	IniSim	NameMatch	1	0,75	0,7	369	100,00%	88,92%	94,13%	22,5

4.4.9 Experimento 9 - Qualidade do algoritmo *Digital Object Match*

O experimento 9 analisa a qualidade do algoritmo **proposto** $Digital\ Object\ Match$ (seção 3.2.2.5).

O objetivo do experimento é calcular a precisão, revocação, medida F e o tempo de processamento da deduplicação dos objetos digitais utilizando $Digital\ Object\ Match$.

Além dos limiares t_L e t_Y, o experimento varia o parâmetro t_N - limiar de similaridade aplicado ao algoritmo $NameMatch$.

Os resultados do experimento 9 são sumarizados na tabela 4.10.

Tabela 4.10: Qualidade e tempo de processamento do experimento 9.

C	Algoritmo	t_Y	t_N	t_L	Pares	Precisão	Revocação	Medida F	Tempo(s)
61	Digital Object Match	1	1	0,5	392	98,72%	93,25%	95,91%	36,7
62	Digital Object Match	1	1	0,7	361	100,00%	86,99%	93,04%	36,6
63	Digital Object Match	1	0,75	0,5	400	98,75%	95,18%	96,93%	36,7
64	Digital Object Match	1	0,75	0,7	369	100,00%	88,92%	94,13%	36,6

4.4.10 Experimento 10 - Diferença de desempenho de *IniSim* e *Guth* utilizando a *MCV Set*

O experimento 10 verifica se a diferença de desempenho das funções $Guth$ (seção 2.3.1 e $IniSim$ (seção 3.2.2.3, utilizando a *MCV Set* (seção 2.1.1, equação 2.1) para a combinação dos escores gerados pelas funções, é estatisticamente significativa.

O objetivo do experimento é calcular o teste T sobre distribuições de resultados das consultas 1 (Tabela 4.2) e 17 (Tabela 4.5). Estas consultas foram escolhidas pois obtiveram os melhores resultados para a medida F.

Os resultados do experimento 10 são sumarizados na tabela 4.11 onde P(T<=t) bi-caudal = 8,52E-106 indica que o desempenho de $IniSim$ é estatisticamente melhor que o desempenho de $Guth$.

Tabela 4.11: Teste T do experimento 10.

Teste-t: duas amostras em par para médias		
	IniSim	Guth
Média	0,930120482	0,231325301
Variância	0,065153367	0,178243408
Observações	415	415
P(T<=t) bi-caudal	8,52E-106	

4.4.11 Experimento 11 - Diferença de desempenho de *IniSim* e *Acronyms* utilizando a *MCV Set*

O experimento 11 verifica se a diferença de desempenho das funções $Acronyms$ (seção 2.3.2 e $IniSim$ (seção 3.2.2.3, utilizando a *MCV Set* (seção 2.1.1, equação 2.1) para a combinação dos escores gerados pelas funções, é estatisticamente significativa.

O objetivo do experimento é calcular o teste T sobre distribuições de resultados das consultas 5 (Tabela 4.3) e 17 (Tabela 4.5). Estas consultas foram escolhidas pois obtiveram os melhores resultados para a medida F.

Os resultados do experimento 11 são sumarizados na tabela 4.12 onde P(T<=t) bi-caudal = 4,47667E-29 indica que o desempenho de $IniSim$ é estatisticamente melhor que o desempenho de $Acronyms$.

Tabela 4.12: Teste T do experimento 11.

Teste-t: duas amostras em par para médias		
	IniSim	Acronyms
Média	0,930120482	0,638554217
Variância	0,065153367	0,231360224
Observações	415	415
P(T<=t) bi-caudal	4,47667E-29	

4.4.12 Experimento 12 - Diferença de desempenho de *IniSim* e *Fragments* utilizando a *MCV Set*

O experimento 12 verifica se a diferença de desempenho das funções $Fragments$ (seção 2.3.3 e $IniSim$ (seção 3.2.2.3, utilizando a *MCV Set* (seção 2.1.1, equação 2.1) para a combinação dos escores gerados pelas funções, é estatisticamente significativa.

O objetivo do experimento é calcular o teste T sobre distribuições de resultados das consultas 11 (Tabela 4.4) e 17 (Tabela 4.5). Estas consultas foram escolhidas pois obtiveram os melhores resultados para a medida F.

Os resultados do experimento 12 são sumarizados na tabela 4.13 onde P(T<=t) bi-caudal = 1,76612E-45 indica que o desempenho de $IniSim$ é estatisticamente melhor que o desempenho de $Fragments$.

Tabela 4.13: Teste T do experimento 12.

Teste-t: duas amostras em par para médias		
	IniSim	Fragments
Média	0,930120482	0,510843373
Variância	0,065153367	0,250486002
Observações	415	415
P(T<=t) bi-caudal	1,76612E-45	

4.4.13 Experimento 13 - Diferença de desempenho de *IniSim* e *Guth* utilizando *NameMatch*

O experimento 13 verifica se a diferença de desempenho das funções $Guth$ (seção 2.3.1 e $IniSim$ (seção 3.2.2.3, utilizando o algoritmo $NameMatch$ (seção 3.2.2.4) para a combinação dos escores gerados pelas funções, é estatisticamente significativa.

O objetivo do experimento é calcular o teste T sobre distribuições de resultados das consultas 23 (Tabela 4.6) e 55 (Tabela 4.9). Estas consultas foram escolhidas pois obtiveram os melhores resultados para a medida F.

Os resultados do experimento 13 são sumarizados na tabela 4.14 onde P(T<=t) bi-caudal = 8,52E-106 indica que o desempenho de $IniSim$ é estatisticamente melhor que o desempenho de $Guth$ mesmo quando o algoritmo $NameMatch$ é utilizado.

Tabela 4.14: Teste T do experimento 13.

Teste-t: duas amostras em par para médias		
	IniSim	Guth
Média	0,930120482	0,231325301
Variância	0,065153367	0,178243408
Observações	415	415
P(T<=t) bi-caudal	8,52E-106	

4.4.14 Experimento 14 - Diferença de desempenho de *IniSim* e *Acronyms* utilizando *NameMatch*

O experimento 14 verifica se a diferença de desempenho das funções $Acronyms$ (seção 2.3.2 e $IniSim$ (seção 3.2.2.3, utilizando o algoritmo $NameMatch$ (seção 3.2.2.4) para a combinação dos escores gerados pelas funções, é estatisticamente significativa.

O objetivo do experimento é calcular o teste T sobre distribuições de resultados das consultas 31 (Tabela 4.7) e 55 (Tabela 4.9). Estas consultas foram escolhidas pois obtiveram os melhores resultados para a medida F.

Os resultados do experimento 14 são sumarizados na tabela 4.15 onde P(T<=t) bi-caudal = 2,47121E-58 indica que o desempenho de $IniSim$ é estatisticamente melhor que o desempenho de $Acronyms$ mesmo quando o algoritmo $NameMatch$ é utilizado.

Tabela 4.15: Teste T do experimento 14.

Teste-t: duas amostras em par para médias		
	IniSim	Acronyms
Média	0,930120482	0,445783133
Variância	0,065153367	0,247657296
Observações	415	415
P(T<=t) bi-caudal	2,47121E-58	

4.4.15 Experimento 15 - Diferença de desempenho de *IniSim* e *Fragments* utilizando *NameMatch*

O experimento 15 verifica se a diferença de desempenho das funções $Fragments$ (seção 2.3.3 e $IniSim$ (seção 3.2.2.3, utilizando o algoritmo $NameMatch$ (seção 3.2.2.4) para a combinação dos escores gerados pelas funções, é estatisticamente significativa.

O objetivo do experimento é calcular o teste T sobre distribuições de resultados das consultas 43 (Tabela 4.8) e 55 (Tabela 4.9). Estas consultas foram escolhidas pois obtiveram os melhores resultados para a medida F.

Os resultados do experimento 15 são sumarizados na tabela 4.16 onde P(T<=t) bi-caudal = 1,00543E-34 indica que o desempenho de $IniSim$ é estatisticamente melhor que o desempenho de $Fragments$ mesmo quando o algoritmo $NameMatch$ é utilizado.

Tabela 4.16: Teste T do experimento 15.

Teste-t: duas amostras em par para médias		
	IniSim	Fragments
Média	0,930120482	0,585542169
Variância	0,065153367	0,243268727
Observações	415	415
P(T<=t) bi-caudal	1,00543E-34	

4.5 Análise dos Resultados

Cada consulta realizada sobre a base de dados retornou um conjunto de pares de objetos digitais identificados como duplicatas. A partir destes pares selecionados e dos pares contidos na relação *match* (Figura 4.1) é possível calcular a precisão, revocação e medida F balanceada de cada consulta. O tempo de processamento é medido através do comando *explain analyse* do PostgreSQL.

Todos as consultas obtiveram precisão maior que 98,4%, ou seja, todos os algoritmos testados identificam corretamente objetos duplicados. Entretanto, a maioria das consultas obteve revocação menor que 60%. No contexto em que o MD-PROM foi desenvolvido, é necessário identificar o máximo possível de objetos replicados. Os experimentos 4,8,9 que utilizam a função proposta IniSim verificaram revocação entre 86,99 e 95,18%.

A qualidade geral dos algoritmos pode ser analisada pela medida F. O algoritmo $Fragments$ combinado com o algoritmo proposto $NameMatch$ (experimento 7) apresentou medida F superior a 70%. Percebe-se que o algoritmo $Fragments$ comporta-se adequadamente com valores de limiares $t_F = 4$ e $t_N = 0,75$. O algoritmo $Acronyms$ obteve índices entre 50,45% e 77,71%, mas os melhores resultados foram quando combinado com a *MCV Set*. O algoritmo $Guth$ não obteve índices aceitáveis de medida F. A função $IniSim$ apresentou medida F superior a 93,04% em todos os testes.

Quanto ao tempo de processamento das consultas percebe-se que todos os algoritmos foram executados em tempos aproximados. As consultas que comparam os anos de publicação por igualdade obtiveram tempos próximos a 10 segundos, enquanto as que utilizam o limiar $t_Y = 1$ marcaram em torno de 23 segundos. O tempo de processamento foi maior para todas as consultas que envolvem variação no ano de publicação pois é necessário um maior número de comparações entre os objetos digitais. O índice aplicado ao atributo *year* das tabelas (Figura 4.1) filtra o número de comparações entre os objetos das duas relações. O número de comparações é reduzido consideravelmente quando utilizada a comparação por igualdade entre os anos. As consultas do experimento 9 marcaram em torno de 36 segundos pois o algoritmo $Digital\ Object\ Match$ utiliza a função $YearSim$ e esta função não permite o uso do índice sobre o atributo *year*.

Quanto ao desempenho dos algoritmos testados, os experimentos 10-15 comprovam que a função proposta $IniSim$ possui desempenho estatisticamente superior em relação a $Guth$, $Acronyms$ e $Fragments$. Este comportamento permanece mesmo quando substituído o algoritmo de casamento que combina os escores gerados pelas funções (*MCV Set* e $NameMatch$).

Analisando a consulta 54 do experimento 8, que entre as consultas que utilizam as funções e algoritmos propostos foi uma das que obtiveram menor revocação (86,99%), pode-se observar que os pares relevantes não retornados apresentam os seguintes problemas:

- codificação de caracteres diferente - caracteres acentuados são representados na DBLP como entidades HTML que possuem comprimento muito maior que um caractere, portanto a função $Levenshtein$ não identifica a similaridade entre os títulos corretamente. Por exemplo, o título "*Recuperação de informação médica interlínguas*" possui 45 caracteres enquanto "*Recuperação de Informação Médica Interlínguas.*" possui 88. Este problema ocorreu em 38% dos pares de objetos relevantes não retornados, e pode ser facilmente corrigido com o uso de uma função de normalização de palavras que substitua as entidades HTML pelos respectivos caracteres acentuados;

- omissão de palavras no título - a função $Levenshtein$ não identifica a similaridade entre os títulos corretamente. Por exemplo, "*A Geographic Knowledge Base for Text Processing*" e "*A Geographic Knowledge Base for Semantic Web Applications*" diferem nas palavras *Text*, *Semantic* e *Web*. Este problema ocorreu em 10% dos pares de objetos relevantes não retornados;

- divergência no número de autores - o algoritmo $NameMatch$, quando utiliza um limiar de 100%, não admite dois objetos digitais com número de autores diferentes. Este problema ocorreu em 21% dos pares de objetos relevantes não retornados. 75% destes pares são detectados corretamente na consulta 56 que utiliza o limiar $t_N = 0.75$;

- omissão dos sobrenomes Júnior (ou Jr.), Filho e Neto - a função $IniSim$ considera a última inicial dos nomes dos autores. Portanto, "*Caetano Traina Jr.*" e "*Caetano Traina*", por exemplo, não são identificados como o mesmo autor. Este problema ocorre em 22% dos pares de objetos relevantes não retornados e pode ser parcialmente corrigido com a remoção dos sobrenomes Júnior (ou Jr.), Filho e Neto;

- omissão do primeiro ou último nome - a função $IniSim$ considera a primeira e última inicial dos nomes dos autores. Portanto, "*Gabriel P. Lopes*" e "*José Gabriel Pereira Lopes*", por exemplo, não são identificados como o mesmo autor. Este problema ocorre em 9% dos pares de objetos relevantes não retornados.

5 CONCLUSÕES E TRABALHOS FUTUROS

O presente trabalho apresenta o MD-PROM, um mecanismo automático de deduplicação de metadados e rastreio da proveniência dos objetos digitais. As consultas realizadas sobre o mecanismo recuperaram os metadados integrados para cada objeto digital além das informações de origem dos objetos digitais. É proposta uma série de funções e algoritmos que aplicadas sobre os metadados provenientes de bibliotecas digitais distintas, identificam com alta precisão e revocação os objetos digitais duplicados. O modelo de dados proposto armazena, além dos metadados dos objetos, informações de proveniência para cada metadado. Estas informações descrevem a origem, o conteúdo e a terminologia utilizada para representar o metadado em cada biblioteca digital que referencia o objeto.

As principais contribuições do MD-PROM são:

- mapeamento Dublin Core x BibTEX - é proposta a correspondência direta entre rótulos dos metadados em ambos os formatos;

- função de similaridade de nomes próprios - a função $IniSim$ identifica variações na representação do nome de um autor baseada nas iniciais desses nome. As varições suportadas incluem grafias diferentes, inversões e abreviações. Enquanto o suporte a iniciais é inexistente ou limitado nos algoritmos estudados, o MD-PROM soluciona o problema de forma satisfatória;

- algoritmo de casamento de autores - o algoritmo de similaridade $NameMatch$ compara todos os autores de um objeto digital com os autores de outro objeto, utilizando qualquer função de similaridade de *strings*. Através de um limiar se similaridade, o algoritmo verifica a compatibilidade no número de autores de cada objeto digital e evita a realização de comparações desnecessárias;

- algoritmo de casamento de objetos digitais - o algoritmo de similaridade *Digital Object Match* deduplica objetos digitais provenientes de bibliotecas digitais distintas. O algoritmo aplica as funções propostas $YearSim$ e $IniSim$ e o algoritmo $NameMatch$ sobre os principais metadados de cada objeto;

- representação da proveniência - o MD-PROM define uma estrutura de dados que modela a proveniência dos metadados de objetos digitais. Um objeto digital integrado é representado através de uma árvore de metadados e proveniências. As informações de proveniência permitem ao usuário verificar a origem, veracidade e confiabilidade dos metadados que descrevem os objetos digitais recuperados em uma consulta sobre um sistema de integração de bibliotecas digitais;

- algoritmo de integração de metadados e rastreio da proveniência - os metadados dos objetos digitais duplicados são recuperados pelo algoritmo $MetadataProv$, o qual rastreia a proveniência das diversas fontes de dados e integra os metadados dos objetos digitais deduplicados. A integração dos metadados e as informações de proveniência são armazenadas no modelo de proveniência proposto para futura consulta dos usuários;

- melhora na qualidade da pesquisa do usuário de bibliotecas digitais - além de fornecer uma resposta única, livre de redundância, sem perda de informação relevante, o diferencial da proposta é recuperar informações de proveniência para consultas a metadados de objetos digitais.

A deduplicação de objetos foi avaliada através de vários experimentos que mensuram a qualidade das funções propostas e as comparam com algoritmos apresentados na revisão bibliográfica. Os resultados indicam que o desempenho na deduplicação de objetos digitais da função $IniSim$ e do algoritmo $NameMatch$ propostos é significativamente superior aos desempenhos dos algoritmos comparados.

Como produção científica foram publicados os seguintes artigos:

- (BORGES; GALANTE, 2007a) Um mecanismo para identificação, representação e consulta de versões de objetos XML oriundos de bibliotecas digitais. In: WORKSHOP DE TESES E DISSERTAÇÕES EM BANCOS DE DADOS, WTDBD, 6., SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, SBBD, 22., 2007. Anais... [S.l.: s.n.], 2007. p.45–50. Este artigo apresenta a visão geral do MD-PROM;

- (BORGES; GALANTE, 2007b) Um mecanismo automático para detectar versões de objetos XML provenientes de bibliotecas digitais. In: WORKSHOP ON DIGITAL LIBRARIES, WDL, 3., BRAZILIAN SYMPOSIUM ON MULTIMEDIA AND THE WEB, WEBMEDIA, 13., 2007. Anais... [S.l.: s.n.], 2007. p.129–135. Este artigo apresenta os detalhes do processo de deduplicação;

- (SILVA; BORGES; GALANTE, 2008) XSimilarity : uma ferramenta para consultas por similaridade embutidas na linguagem xquery. In: ESCOLA REGIONAL DE BANCOS DE DADOS, ERBD, 4., 2008. Anais... [S.l.: s.n.], 2008. Este artigo apresenta a ferramenta utilizada na disseminação da proveniência dos metadados.

Dois trabalhos foram desenvolvidos em paralelo com esta dissertação:

- (GIACOMOLLI, 2007) **METADATAPROV - uma interface web para a consulta da proveniência de metadados de bibliotecas digitais**. 2007. Monografia (Graduação em Ciência da Computação) - Instituto de Informática, UFRGS, Porto Alegre. Desenvolvida paralelamente à este trabalho, esta monografia descreve a implementação da ferramenta de consulta MetadataProv, a qual foi co-orientada pelo autor desta dissertação;

- (SILVA, 2007) XSimilarity : uma ferramenta para consultas por similaridade embutidas na linguagem xquery. 2007. Monografia (Graduação em Ciência da Computação) - Instituto de Informática, UFRGS, Porto Alegre. Desenvolvida paralelamente à este trabalho, esta monografia descreve a implementação da ferramenta XSimilarity, a qual foi co-orientada pelo autor desta dissertação.

Ao longo do desenvolvimento deste trabalho, alguns aspectos do MD-PROM foram identificados como passíveis de melhorias ou extensões. As diferentes codificações de caracteres utilizadas pelas bibliotecas digitais para representar caracteres acentuados dificultam o cálculo da similaridade pela função $Levenshtein$. Funções de pré-processamento de strings aplicadas aos títulos dos objetos digitais podem aumentar a qualidade da identificação de duplicatas. Outros mecanismos como algoritmos de classificação podem ser utilizados para realizar o casamento das instâncias evitando o uso de *threshold* (SACCOL et al., 2007). Disponibilizar o acesso ao MD-PROM para a comunidade através da *Web* permite que muitos usuários de bibliotecas digitais avaliem qualitativamente o mecanismo e usufruam das informações de proveniência para avaliar a qualidade das bibliotecas digitais. Também é interessante a implementação de um serviço *Web* para coleta automática dos metadados de várias bibliotecas digitais. Por fim, destaca-se como trabalho futuro a avaliação experimental da proveniência de metadados.

REFERÊNCIAS

ACM. **Association for Computing Machinery (ACM) Digital Library**. Disponível em: <http://portal.acm.org/dl.cfm>. Acesso em: 10 jul. 2007.

ALF-CHRISTIAN ACHILLES. **The Collection of Computer Science Bibliographies (CCSB)**. Disponível em: <http://liinwww.ira.uka.de/bibliography>. Acesso em: 20 nov. 2007.

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Modern Information Retrieval**. [S.l.]: ACM Press / Addison-Wesley, 1999.

BHAGWAT, D. et al. An Annotation Management System for Relational Databases. In: VLDB, 2004. **Proceedings...** [S.l.]: Morgan Kaufmann, 2004. p.900–911.

BILENKO, M. et al. Adaptive Name Matching in Information Integration. **IEEE Intelligent Systems**, Piscataway, NJ, USA, v.18, n.5, p.16–23, 2003.

BILENKO, M.; MOONEY, R. J. Adaptive duplicate detection using learnable string similarity measures. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, KDD, 9., 2003. **Proceedings...** New York: ACM, 2003. p.39–48.

BORGES, E. N.; DORNELES, C. F. PgSimilar: uma ferramenta open source para suporte a consultas por similaridade no postgresql. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, SBBD, 21., 2006. **Anais da III Sessão de Demos.** Florianópolis: SBC, 2006.

BORGES, E. N.; 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 EM BANCOS DE DADOS, WTDBD, 6.; SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, SBBD, 22., 2007. **Anais...** [S.l.: s.n.], 2007. p.45–50.

BORGES, E. N.; GALANTE, R. M. Um mecanismo automático para detectar versões de objetos XML provenientes de bibliotecas digitais. In: WORKSHOP ON DIGITAL LIBRARIES, WDL, 3.; BRAZILIAN SYMPOSIUM ON MULTIMEDIA AND THE WEB, WEBMEDIA, 13., 2007. **Anais...** [S.l.: s.n.], 2007. p.129–135.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: ANNUAL WORKSHOP ON COMPUTATIONAL LEARNING THEORY, COLT, 5., 1992. **Proceedings...** New York: ACM, 1992. p.144–152.

BUNEMAN, P.; KHANNA, S.; TAN, W. C. Data Provenance: some basic issues. In: FSTTCS, 2000. **Proceedings...** Berlin: Springer, 2000. p.87–93. (Lecture Notes in Computer Science, v.1974).

BUNEMAN, P.; KHANNA, S.; TAN, W. C. Why and Where: a characterization of data provenance. In: ICDT, 2001. **Proceedings...** Berlin: Springer, 2001. p.316–330. (Lecture Notes in Computer Science, v.1973).

CARVALHO, J. C. P.; SILVA, A. S. da. Finding similar identities among objects from multiple web sources. In: ACM INTERNATIONAL WORKSHOP ON WEB INFORMATION AND DATA MANAGEMENT, WIDM, 5., 2003. **Proceedings...** New York: ACM, 2003. p.90–93.

CARVALHO, M. G. et al. Learning to deduplicate. In: JOINT CONFERENCE ON DIGITAL LIBRARIES, JCDL, 6., 2006. **Proceedings...** New York: ACM, 2006. p.41–50.

CHAKRABARTI, S. **Mining the Web**: discovering knowledge from hypertext data. [S.l.]: Morgan-Kauffman, 2002.

CHAUDHURI, S. et al. Robust and efficient fuzzy match for online data cleaning. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 2003. **Proceedings...** New York: ACM, 2003. p.313–324.

CHAUDHURI, S.; GRAVANO, L. Evaluating Top-k Selection Queries. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 25., 1999. **Proceedings...** San Francisco, CA: Morgan Kaufmann Publishers, 1999. p.397–410.

CHRISTEN, P. A Comparison of Personal Name Matching: techniques and practical issues. In: IEEE INTERNATIONAL CONFERENCE ON DATA MINING WORKSHOPS, ICDMW, 6., 2006, Hong Kong, China. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.290–294.

COHEN, W. W.; RAVIKUMAR, P.; FIENBERG, S. E. A Comparison of String Distance Metrics for Name-Matching Tasks. In: WORKSHOP ON INFORMATION INTEGRATION, IJCAI, 2003. **Proceedings...** [S.l.: s.n.], 2003. p.73–78.

COHEN, W. W.; RICHMAN, J. Learning to match and cluster large high-dimensional data sets for data integration. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, KDD, 8., 2002. **Proceedings...** New York: ACM, 2002. p.475–480.

COTA, R.; GONÇALVES, M. A.; LAENDER, A. H. F. A Heuristic-based Hierarchical Clustering Method for Author Name Disambiguation in Digital Libraries. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, SBBD, 2007. **Anais...** [S.l.: s.n.], 2007.

CUI, Y.; WIDOM, J. Lineage Tracing for General Data Warehouse Transformations. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 27., 2001. **Proceedings...** San Francisco, CA: Morgan Kaufmann Publishers, 2001. p.471–480.

CULOTTA, A.; MCCALLUM, A. Joint deduplication of multiple record types in relational data. In: ACM INTERNATIONAL CONFERENCE ON INFORMATION AND

KNOWLEDGE MANAGEMENT, CIKM, 14., 2005. **Proceedings...** New York: ACM, 2005. p.257–258.

DCMI. **DCMI Collection Description, Proposed Term** : provenance. Disponível em: <http://www.ukoln.ac.uk/metadata/dcmi/collection-provenance>. Acesso em: 29 jan. 2008.

DCMI. **DCMI Metadata Terms**. Disponível em: <http://dublincore.org/documents/dcmi-terms>. Acesso em: 25 jan. 2008.

DORNELES, C. F.; HEUSER, C. A.; LIMA, A. E. N.; SILVA, A. S. da; MOURA, E. S. de. Measuring similarity between collection of values. In: ANNUAL ACM INTERNATIONAL WORKSHOP ON WEB INFORMATION AND DATA MANAGEMENT, WIDM, 6., 2004. **Proceedings...** New York: ACM, 2004. p.56–63.

DORNELES, C. F.; HEUSER, C. A.; ORENGO, V. M.; SILVA, A. S. da; MOURA, E. S. de. A strategy for allowing meaningful and comparable scores in approximate matching. In: ACM CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT, CIKM, 16., 2007. **Proceedings...** New York: ACM, 2007. p.303–312.

FELLEGI, I. P.; SUNTER, A. B. A Theory for Record Linkage. **Journal of the American Statistical Association**, [S.l.], v.64, n.328, p.1183–1210, December 1969.

FOSTER, I. T. et al. Chimera: avirtual data system for representing, querying, and automating data derivation. In: INTERNATIONAL CONFERENCE ON SCIENTIFIC AND STATISTICAL DATABASE MANAGEMENT, SSDBM, 14., 2002. **Proceedings...** Washington, DC: IEEE Computer Society, 2002. p.37–46.

FOX, E. A. et al. Digital libraries. **Commun. ACM**, New York, NY, USA, v.38, n.4, p.22–28, 1995.

FREW, J.; BOSE, R. Earth System Science Workbench: a data management infrastructure for earth science products. In: INTERNATIONAL CONFERENCE ON SCIENTIFIC AND STATISTICAL DATABASE MANAGEMENT, SSDBM, 13., 2001. **Proceedings...** Washington: DC: IEEE Computer Society, 2001. p.180.

GIACOMOLLI, D. M. **METADATAPROV - uma interface web para a consulta da proveniência de metadados de bibliotecas digitais**. 2007. Monografia (Graduação em Ciência da Computação) — Instituto de Informática, UFRGS, Porto Alegre.

GILES, C. L.; BOLLACKER, K. D.; LAWRENCE, S. CiteSeer: an automatic citation indexing system. In: ACM CONFERENCE ON DIGITAL LIBRARIES, DL, 3., 1998. **Proceedings...** New York: ACM, 1998. p.89–98.

GREENWOOD, M. et al. Provenance of e-Science Experiments - experience from Bioinformatics. In: UK OST E-SCIENCE ALL HANDS MEETING, AHM, 2., 2003. **Proceedings...** [S.l.: s.n.], 2003. p.223–226.

GUHA, S. et al. Merging the results of approximate match operations. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 30., 2004. **Proceedings...** [S.l.: s.n.], 2004. p.636–647.

GUTH, G. J. A. Surname Spellings and Computerized Record Linkage. **Historical Methods Newsletter**, [S.l.], v.10, n.1, p.10–19, December 1976.

HAN, H. et al. Two supervised learning approaches for name disambiguation in author citations. In: ACM/IEEE-CS JOINT CONFERENCE ON DIGITAL LIBRARIES, JCDL, 4., 2004. **Proceedings...** New York: ACM, 2004. p.296–305.

HAN, H.; ZHA, H.; GILES, C. L. Name disambiguation in author citations using a K-way spectral clustering method. In: ACM/IEEE-CS JOINT CONFERENCE ON DIGITAL LIBRARIES, JCDL, 5., 2005. **Proceedings...** New York: ACM, 2005. p.334–343.

HULL, D. Using statistical testing in the evaluation of retrieval experiments. In: ANNUAL INTERNATIONAL ACM SIGIR CONFERENCE ON RESEARCH AND DEVELOPMENT IN INFORMATION RETRIEVAL, SIGIR, 16., 1993. **Proceedings...** New York: ACM, 1993. p.329–338.

LAENDER, A. H. F.; GONÇALVES, M. A.; ROBERTO, P. A. BDBComp: building a digital library for the brazilian computer science community. In: ACM/IEEE-CS JOINT CONFERENCE ON DIGITAL LIBRARIES, JCDL, 4., 2004. **Proceedings...** New York: ACM, 2004. p.23–24.

LAIT, A.; RANDELL, B. **An assessment of name matching algorithms**. [S.l.: s.n.], 1993.

LAMPORT, L. **LaTeX**: a document preparation system. [S.l.]: Addison-Wesley, 1986.

LAW, D. G. **Digital Libraries**: policy, planning, and practice. [S.l.]: Ashgate Publishing, 2004.

LAWRENCE, S.; GILES, C. L.; BOLLACKER, K. Digital Libraries and Autonomous Citation Indexing. **Computer**, Los Alamitos, CA, USA, v.32, n.6, p.67–71, 1999.

LEVENSHTEIN, V. I. Binary codes capable of correcting deletions, insertions and reversals. **Soviet Physics Doklady.**, [S.l.], v.10, n.8, p.707–710, February 1966.

LIMA, A. E. N. **Pesquisa de similaridade em XML**. 2002. Monografia (Graduação em Ciência da Computação) — Instituto de Informática, UFRGS, Porto Alegre.

MALIN, B. Unsupervised name disambiguation via social network similarity. In: WORKSHOP ON LINK ANALYSIS, COUNTERTERRORISM, AND SECURITY IN CONJUNCTION WITH THE SIAM INTERNATIONAL CONFERENCE ON DATA MINING, 2005. **Proceedings...** [S.l.: s.n.], 2005. p.93–102.

MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H. **Introduction to Information Retrieval**. [S.l.]: Cambridge University Press, 2008.

OLIVEIRA, J. W. A. **Uma Estratégia para Remoção de Ambigüidades na Identificação de Autoria de Objetos Bibliográficos**. 2005. Tese (Doutorado em Ciência da Computação) — Departamento de Ciência da Computação, UFMG, Belo Horizonte.

OLIVEIRA, J. W. A.; LAENDER, A. H. F.; GONÇALVES, M. A. Remoção de Ambigüidades na Identificação de Autoria de Objetos Bibliográficos. In: SBBD, 2005. **Anais...** UFU, 2005. p.205–219.

QUINLAN, J. R. Induction of Decision Trees. **Mach. Learn.**, Hingham, MA, USA, v.1, n.1, p.81–106, 1986.

RAHM, E.; BERNSTEIN, P. A. A survey of approaches to automatic schema matching. **The VLDB Journal**, Secaucus, NJ, USA, v.10, n.4, p.334–350, 2001.

SACCOL, D. B.; EDELWEISS, N.; GALANTE, R. M.; ZANIOLO, C. XML version detection. In: ACM SYMPOSIUM ON DOCUMENT ENGINEERING, DOCENG, 2007. **Proceedings...** New York: ACM, 2007. p.79–88.

SALTON, G. (Ed.). **Automatic text processing**. Boston, MA, USA: Addison-Wesley Longman, 1988.

SILVA, M. E. V. **XSimilarity** : uma ferramenta para consultas por similaridade embutidas na linguagem xquery. 2007. Monografia (Graduação em Ciência da Computação) — Instituto de Informática, UFRGS, Porto Alegre.

SILVA, M. E. V.; BORGES, E. N.; GALANTE, R. M. XSimilarity : uma ferramenta para consultas por similaridade embutidas na linguagem xquery. In: ESCOLA REGIONAL DE BANCOS DE DADOS, ERBD, 4., 2008. **Anais...** [S.l.: s.n.], 2008. p.148–157.

SILVA, R. da; STASIU, R.; ORENGO, V. M.; HEUSER, C. A. Measuring quality of similarity functions in approximate data matching. **Journal of Informetrics**, [S.l.], v.1, n.1, p.35–46, 2007.

SIMMHAN, Y. L.; PLALE, B.; GANNON, D. A survey of data provenance in e-science. **SIGMOD Rec.**, New York, NY, USA, v.34, n.3, p.31–36, 2005.

SOMPEL, H. Van de et al. Resource Harvesting within the OAI-PMH Framework. **D-Lib Magazine**, [S.l.], v.10, n.12, December 2004.

SONG, Y. et al. Efficient topic-based unsupervised name disambiguation. In: JOINT CONFERENCE ON DIGITAL LIBRARIES, JCDL, 2007. **Proceedings...** New York: ACM, 2007. p.342–351.

STASIU, R. K.; HEUSER, C. A.; SILVA, R. da. Estimating Recall and Precision for Vague Queries in Databases. In: CAISE, 2005. **Proceedings...** Berlin: Springer, 2005. p.187–200. (Lecture Notes in Computer Science, v.3520).

TAN, W. C. Containment of Relational Queries with Annotation Propagation. In: DBPL, 2003. **Proceedings...** Berlin: Springer, 2003. p.37–53. (Lecture Notes in Computer Science, v.2921).

TAN, W. C. Research Problems in Data Provenance. **IEEE Data Eng. Bull.**, [S.l.], v.27, n.4, p.45–52, 2004.

TEJADA, S.; KNOBLOCK, C. A.; MINTON, S. Learning object identification rules for information integration. **Inf. Syst.**, Oxford, UK, v.26, n.8, p.607–633, 2001.

UNIVERSITY OF TRIER. **Digital Bibliography & Library Project (DBLP)**. Disponível em: <http://dblp.uni-trier.de>. Acesso em: 10 nov. 2007.

WIDOM, J. Trio: a system for integrated management of data, accuracy, and lineage. In: CIDR, 2005. **Proceedings...** [S.l.: s.n.], 2005. p.262–276.

WIKIPEDIA. **Digital Object Identifier**. Disponível em: <http://en.wikipedia.org/wiki/Digital_object_identifier>. Acesso em: 16 nov. 2007.

WIKIPEDIA. **Student's t-test**. Disponível em: <http://en.wikipedia.org/wiki/Student's_t-test>. Acesso em: 12 mar. 2008.

ZHAO, J. et al. Semantically Linking and Browsing Provenance Logs for Escience. In: ICSNW, 2004. **Proceedings...** [S.l.: s.n.], 2004.