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

SÉRGIO LUIS SARDI MERGEN

Casamento de Esquemas XML e Esquemas Relacionais

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

Dr. Carlos Alberto Heuser
Orientador

Porto Alegre, março de 2007

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Mergen, Sérgio Luis Sardi

Casamento de Esquemas XML e Esquemas Relacionais / Sérgio Luis Sardi Mergen. – Porto Alegre: PPGC da UFRGS, 2007.

82 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2007. Orientador: Carlos Alberto Heuser.

1. XML. 2. Casamento de Esquemas. 3. Bancos de Dados. 4. Integração de Informação. 5. Intercâmbio de Dados. I. Heuser, Carlos Alberto. II. Título.

AGRADECIMENTOS

Foram dois anos de mestrado. Nestes dois anos muitas pessoas estiveram presentes em minha vida. O relacionamento delas comigo se estende desde o nível pessoal ao nível acadêmico, e é difícil retratar sua importância durante esta etapa em tão poucas linhas. Assim, valho-me de uma analogia que me ajuda a expressar melhor meus sentimentos:

O mestrado foi uma caminhada, cujo trajeto foi repleto de obstáculos e desvios. No decorrer da caminhada encontrei pessoas, que a partir daí seguiam junto ao meu lado, prontas para ajudar no que fosse necessário.

Partiram comigo meu pai Ilceo, minha mãe Carla, meu irmão Cahê, além de Rosi e Yuri. Meu pai não caminhava bem ao meu lado, mas uns poucos metros a frente, vasculhando os arredores por perigos, preparando o terreno para que quando eu cruzasse, qualquer espécie de risco fosse o menor possível. Minha mãe bateu palmas o tempo todo, vibrando a cada passo, encorajando-me a perseguir na caminhada até o fim. Nos finais de semana em que eu descansava, lá estava meu irmão companheiro, pronto para uma partida de futebol ou um cafezinho em algum bar à beira-estrada.

No segundo ano de caminhada, contei com a ajuda do meu tio Douglas e minha tia Valéria. Além de fornecerem um cantinho em sua residência, sempre se mostraram solícitos quando deles precisava.

Outras pessoas encontrei durante o percurso, e elas se mostraram particularmente importantes quando eu esbarrava em obstáculos: São os meus colegas Adrovane, Alvaristo, André, Andrei, Carina, César, Eduardo, Felipe, Raquel, Renata e Vanessa. Agradecimento especial ao Adrovane, que contribuiu com material pertinente ao meu trabalho, à Carina, pela ajuda nos experimentos realizados, e à Vanessa, pela ajuda na definição do formalismo empregado na dissertação.

Agradeço também a meu orientador Heuser, pela sua ajuda constante e pelos seus sábios conselhos sobre qual caminho seguir quando me deparava com encruzilhadas.

Ao CNpQ, por ter apostado em mim e pago o "combustível"dessa minha jornada.

Agora, enquanto dou meus últimos passos, olho ao meu redor e vejo muitas outras pessoas, pessoas demais para caberem todas neste espaço. Mas saibam, vocês não foram esquecidos. Têm um lugar reservado em meu coração.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS 6

LISTA DE FIGURAS 7

LISTA DE TABELAS 8

RESUMO 9

ABSTRACT 10

1 INTRODUÇÃO 11

2 ESTADO DA ARTE 14
2.1 A operação de casamento (*Match*) 15
2.2 Cardinalidade de casamento 16
2.3 Abordagens lingüísticas de casamento 17
2.3.1 Comparação de cadeias pequenas: 18
2.3.2 Comparação de cadeias longas: 18
2.4 Casadores em nível de esquema 20
2.5 Abordagens com base em instâncias 20
2.6 Abordagens baseadas em restrições 21
2.7 Reuso de informações de esquemas 22
2.8 Sistemas de casamento de esquemas 22
2.8.1 Autoplex and Automatch 22
2.8.2 Clio 22
2.8.3 COMA 23
2.8.4 Cupid 23
2.8.5 LSD e GLUE 24
2.8.6 SemInt 24
2.8.7 Similarity Flooding (SF) 24
2.8.8 Considerações Finais 25

3 VISÃO GERAL DO PROCESSO DE CASAMENTO PROPOSTO . 26
3.1 Representação dos esquemas 27
3.1.1 Considerações gerais 31

4 CASADOR LINGÜÍSTICO CARLA 32
4.1 Métrica proposta 33
4.2 Tamanho dos termos 34
4.3 Algoritmo proposto 34
4.4 Considerações gerais 38

5 CASADORES ESTRUTURAIS 39
5.1 Casador de nomenclatura 39
5.2 Casador de relação 41
5.3 Casadores de atração 44
5.3.1 Casador de vizinhança 45
5.3.2 Casador de cardinalidade 46
5.4 Considerações Gerais 48

6 EXPERIMENTOS REALIZADOS 49
6.1 Experimentos com a técnica Carla 49
6.2 Experimentos com os casadores estruturais 51
6.2.1 Execução dos casadores 51
6.2.2 Combinação das similaridades 51
6.2.3 Filtro dos resultados 52
6.2.4 Domínios usados 52
6.3 Livros 53
6.4 Locadora de filmes 54
6.5 Locadora de imóveis 55
6.6 Considerações gerais 57

7 CONCLUSÃO 58

REFERÊNCIAS 61

APÊNDICE A EXPERIMENTOS REALIZADOS 65
A.1 Esquemas do domínio de livros 65
A.2 Esquemas do domínio de locadora de filmes 70
A.3 Esquemas do domínio de locação de imóveis 75

LISTA DE ABREVIATURAS E SIGLAS

DTD	Document Type Definition
W3C	World Wide Web Consortium
XML	Extensible Markup Language

LISTA DE FIGURAS

Figura 2.1: Classificação das abordagens de casamento de esquemas. 16
Figura 2.2: Exemplo de cardinalidade de casamento. 17

Figura 3.1: Visão geral do processo de casamento 26
Figura 3.2: Esquema canônico relacional 28
Figura 3.3: Esquema canônico DTD 29

Figura 4.1: Processo de divisão de cadeias em termos 33
Figura 4.2: Matrizes para os casos (nomeAutor, nomeAutor) e (sem rua, rua charrua) 35
Figura 4.3: Matrizes para o caso (PatyBeatriz, BeaPatricia) 36

Figura 5.1: Atribuição dos conjuntos de nós relacionados 41
Figura 5.2: Atribuição de fatores de relevância 42
Figura 5.3: Campos de atração entre dois conceitos 44
Figura 5.4: Exemplo de campos de atração sob a ótica dos nós vizinhos . . . 45
Figura 5.5: Exemplo de campos de atração sob a ótica dos nós multi-valorados 47

Figura 6.1: Conjunto de casamentos corretos e computados automaticamente 49
Figura 6.2: Gráficos de precisão e revocação dos casadores lingüísticos 50
Figura 6.3: Tempo de execução dos casadores lingüísticos 50
Figura 6.4: Seqüência de execução dos casadores 51
Figura 6.5: Gráficos de revocação e precisão para o domínio de livros 54
Figura 6.6: Gráficos de revocação e precisão para o domínio de locadora de filmes 55
Figura 6.7: Gráficos de revocação e precisão para o domínio de locadora de imóveis 56

LISTA DE TABELAS

Tabela 6.1: Experimentos no domínio de livros 53
Tabela 6.2: Experimentos no domínio de locadora de filmes 54
Tabela 6.3: Experimentos no domínio de locadora de imóveis 56

RESUMO

O casamento entre esquemas XML e esquemas relacionais é necessário em diversas aplicações, tais como integração de informação e intercâmbio de dados. Tipicamente o casamento de esquemas é um processo manual, talvez suportado por uma interface gráfica. No entanto, o casamento manual de esquemas muito grandes é um processo dispendioso e sujeito a erros. Disto surge a necessidade de técnicas (semi)-automáticas de casamento de esquemas que auxiliem o usuário fornecendo sugestões de casamento, dessa forma reduzindo o esforço manual aplicado nesta tarefa. Apesar deste tema já ter sido estudado na literatura, o casamento entre esquemas XML e esquemas relacionais é ainda um tema em aberto. Isto porque os trabalhos existentes ou se aplicam para esquemas definidos no mesmo modelo, ou são genéricos demais para o problema em questão. O objetivo desta dissertação é o desenvolvimento de técnicas específicas para o casamento de esquemas XML e esquemas relacionais. Tais técnicas exploram as particularidades existentes entre estes esquemas para inferir valores de similaridade entre eles. As técnicas propostas são avaliadas através de experimentos com esquemas do mundo real.

Palavras-chave: XML, Casamento de Esquemas, Bancos de Dados, Integração de Informação, Intercâmbio de Dados.

Matching of XML Schemas and Relational Schemas

ABSTRACT

The matching between XML schemas and relational schemas has many applications, such as information integration and data exchange. Typically, schema matching is done manually by domain experts, sometimes using a graphical tool. However, the matching of large schemas is a time consuming and error-prone task. The use of (semi-)automatic schema matching techniques can help the user in finding the correct matches, thereby reducing his labor. The schema matching problem has already been addressed in the literature. Nevertheless, the matching of XML schemas and relational schemas is still an open issue. This comes from the fact that the existing work is whether specific for schemas designed in the same model, or too generic for the problem in discussion. The mais goal of this dissertation is to develop specific techniques for the matching of XML schemas and relational schemas. Such techniques exploit the particularities found when analyzing the two schemas together, and use these cues to leverage the matching process. The techniques are evaluated by running experiments with real-world schemas.

Keywords: XML, Schema Matching, Databases, Information Integration, Data Exchange.

1 INTRODUÇÃO

O casamento de esquemas é fundamental em áreas que trabalham com dados de diferentes fontes, tais como integração de esquemas, data warehouses (BERNSTEIN; RAHM, 2000), comércio eletrônico e processamento de consultas semânticas (RISHE et al., 2000). O casamento de esquemas consiste em, dados dois esquemas, identificar as correspondências que existem entre os elementos de ambos os esquemas.

Neste trabalho destaca-se a necessidade de casamento de esquemas XML com esquemas relacionais. O uso deste tipo de casamento pode ser analisado em pelo menos duas aplicações: integração de informação e intercâmbio de informações.

O problema de integrar dados de diferentes fontes consiste basicamente no seguinte (BATINI; LENZERINI; NAVATHE, 1986): dado um conjunto de esquemas $S_1, \ldots, S_n$, deve-se construir uma visão global V_G. Normalmente, cada um dos esquemas S é desenvolvido de forma independente e possui estrutura e terminologia diferente dos demais. Naturalmente, é de se esperar que isso aconteça quando os esquemas pertencem a domínios diferentes (por exemplo, um boletim de notas de um aluno e uma nota fiscal de venda). Entretanto, mesmo esquemas pertencentes ao mesmo domínio de problema podem apresentar discrepâncias na escolha da terminologia, devido ao simples fato de que eles são elaborados por diferentes pessoas, em contextos distintos. No domínio de comércio eletrônico, por exemplo, o preço de um produto pode ser chamado de *preço* em um esquema e de *valor* em outro. Pode haver, ainda, diferenças na estrutura: o endereço de um cliente pode ser armazenado como uma única cadeia de caracteres em um esquema e como um conjunto de elementos {rua, número, cidade, cep} em outro.

A solução do problema de integração de esquemas envolve dois passos (MILLER et al., 2001): primeiro, identificar e caracterizar os relacionamentos intra-esquemas (ou seja, o processo de casamento de esquemas propriamente dito) e, depois, especificar o mapeamento entre os esquemas. Este mapeamento é um conjunto de expressões que especificam como os dados de uma fonte podem ser traduzidos para dados da outra fonte. Durante a última fase são criados programas ou consultas que permitem a tradução dos dados de cada um dos esquemas originais para o esquema integrado.

É importante salientar que casamento de esquemas (*schema matching*) e mapeamento de esquemas(*schema mapping*) são termos distintos (HERNANDEZ et al., June 2001). No casamento de esquemas, a preocupação é encontrar elementos nos dois esquemas que correspondam a mesma informação. Já no mapeamento de esquemas, a preocupação é utilizar os elementos casados para definir a forma como instâncias de um esquema podem ser transformadas em instâncias de outro esquema. Para exemplificar considera-se dois esquemas que modelam pacotes de turismo. Ambos

esquemas possuem um elemento chamado *valor*, que indica o preço de um determinado pacote. Apesar destes elementos armazenarem a mesma informação, cada um deles apresenta o preço utilizando uma moeda diferente. Assim, enquanto no casamento de esquemas ocorre a identificação de que os dois elementos cujo nome é *valor* correspondem a mesma informação, no mapeamento de esquemas deve ser realizada uma transformação (no caso em questão é uma transformação aritmética), que permite que os valores de preço de um dos esquemas possam ser traduzidos para valores de preço no outro esquema. Normalmente o mapeamento é unidirecional e é construído na forma de consultas (visões) sobre o esquema de origem.

Outra questão importante na área de integração de informação é a heterogeneidade de formato, onde o problema enfrentado é como usar uma interface única para acessar dados que estão dispostos em fontes de diversos modelos, como documentos XML, páginas HTML, arquivos texto ou bases relacionais (LEVY et al., 1995; CHAWATHE et al., 1994). No caso de interesse deste trabalho, considera-se o cenário onde o esquema mediado é definido nos moldes de esquemas relacionais e existem fontes descritas em formato XML. Disto surge a necessidade do casamento entre esquemas XML e esquemas relacional.

Também na área de intercâmbio de dados o casamento de esquemas XML e relacional torna-se necessário. O problema de intercâmbio de dados consiste em transportar dados armazenados em uma fonte para outra fonte, sendo ambas fontes construídas independentemente (FAGIN et al., 2002). O intercâmbio de dados é mais comumente requisitado em ambientes coorporativos, onde parceiros de negócio interagem e dados devem fluir entre as fontes de dados envolvidas.

Um dos obstáculos envolvidos no processo de intercâmbio está relacionado a heterogeneidade dos dados. No mundo coorporativo, geralmente os dados são armazenados em bases de dados relacionais. Como estas bases são modeladas de forma independente, é normal que bases distintas contenham diferenças estruturais e terminológicas, o que torna o processo de conversão mais elaborado.

Outro obstáculo diz respeito ao formato de dados intermediário que deve ser usado para transportar os dados de uma base até a outra. Este obstáculo pode ser sobreposto com a adoção de XML como um formato padronizado para o intercâmbio de informações.

Levando em consideração essas observações, e tendo-se o problema de intercâmbio de dados entre duas bases relacionais S_1 e S_2, uma solução possível envolve a conversão de dados de S_1 em um documento XML intermediário, e a futura conversão deste documento XML em dados de S_2. Mais um vez surge a necessidade do casamento entre esquemas relacionais e documentos XML.

Tipicamente o processo de casamento de esquemas é um processo manual, talvez suportado por uma interface gráfica (HERRING; MILOSEVIC, 2001). Isso pode até ser aceitável se os esquemas a serem integrados forem de pequeno tamanho, quando é mais fácil verificar a ocorrência de erros ou omissões. À medida em que o tamanho dos esquemas cresce, entretanto, fica muito mais difícil de se manter o controle sobre o processo de casamento.

Do ponto de vista da integração de informação, há que se considerar também o constante crescimento no número de fontes que se quer integrar, bem como a diversidade de tipos de esquemas a serem integrados: esquemas de bancos de dados, XML, ontologias, etc. Quando se considera a *Web* como repositório de fontes de informação a serem integradas, percebe-se a complexidade desta tarefa. A situação

se torna ainda mais crítica quando a tarefa de casamento envolve a análise conjunta de informações do esquema e das instâncias de dados, tendo em vista o grande volume de informações que precisam ser manipuladas. Sem a existência de um mecanismo que permita a automatização — pelo menos parcial — será necessário um investimento elevado em tempo e dinheiro.

A área de casamento (semi)automático de esquemas visa suprir as deficiências encontradas no casamento puramente manual. O problema nesta área consiste em encontrar o casamento entre elementos dos dois esquemas, que normalmente são elaborados separadamente e que podem ou não utilizar terminologias diferentes para identificar dados semelhantes. Um casamento entre dois elementos pode ser estabelecido quando existe uma similaridade semântica entre eles, ou seja, quando as instâncias de dados que eles representam pertencem ao mesmo tipo de informação.

O problema de casamento (semi)automático de esquemas já foi bastante estudado na literatura (RAHM; BERNSTEIN, 2001). No entanto, a maior parte dos esforços foi dispendida ou no casamento de esquemas descritos no mesmo modelo (como por exemplo o casamento de esquemas relacionais (BERLIN; MOTRO, 2001; EMBLEY; JACKMAN; XU, 2001) e o casamento de ontologias (DOAN et al., 2002)) ou em abordagens genéricas demais (MADHAVAN; BERNSTEIN; RAHM, 2001; MELNIK; GARCIA-MOLINA; RAHM, 2002). O casamento entre esquemas XML e relacional ainda é uma área em aberto. Apesar de poderem ser aproveitados os esforços dedicados em abordagens genéricas, o desenvolvimento de técnicas próprias para o casamento de esquemas XML e esquemas relacionais pode explorar as particularidades existentes nesses esquemas para tornar o resultado do processo de casamento automático mais confiável.

O foco deste trabalho é na tarefa de casamento de esquemas XML com esquemas relacionais para fins de integração e/ou intercâmbio de dados. A principal meta é desenvolver formas de automatizar pelo menos parcialmente essa atividade, reduzindo a necessidade de trabalho manual. Para tanto, foram projetadas técnicas de casamento de esquemas específicas para o casamento de esquemas XML e esquemas relacionais. Por serem específicas para o caso em questão, tais técnicas podem ser mais capazes de descobrir os relacionamentos intra-esquemas do que as demais técnicas existentes.

O texto da dissertação está estruturado em seis capítulos, e um anexo, além das referências bibliográficas. O Capítulo 2 aborda o estado da arte no tocante à área de casamento de esquemas. O Capítulo 3 apresenta uma visão geral do processo de casamento adotado. O Capítulo 4 apresenta a técnica de casamento lingüístico proposta neste trabalho. O Capítulo 5 mostra as técnicas de casamento estruturais desenvolvidas. O Capítulo 5 também contém uma seção de experimentos que visam validar as soluções propostas. No capítulo 6 são mencionados os trabalhos futuros e demais considerações finais. O anexo I descreve os esquemas que foram usados nos experimentos realizados.

2 ESTADO DA ARTE

Há um interesse crescente na área de casamento de esquemas, conforme se pode observar pelos estudos de (MILO; ZOHAR, 1998; BERGHOLZ; FREYTAG, 1999; RAHM; BERNSTEIN, 2001; REYNAUD; SIROT; VODISLAV, 2001; DOAN, 2002; KURGAN; SWIERCZ; CIOS, 2002; CASTANO et al., 2002; DO; MELNIK; RAHM, 2002; DOAN et al., 2003; XU; EMBLEY, 2003; KANG; NAUGHTON, 2003; MADHAVAN et al., 2003), entre outros. Esse interesse decorre do alto custo exigido pela necessidade de se determinar os mapeamentos manualmente, que pode ser severamente reduzido por meio da automatização, mesmo que parcial.

Segundo (DOAN, 2002), as soluções propostas para contornar o alto custo gerado pelo mapeamento manual podem ser incluídas em dois grupos. No primeiro, estão os que tentam elaborar padrões (ou seja, vocabulários comuns) para um determinado domínio de problema. Uma vez estabelecido o padrão, todos teriam que estar de acordo com ele. Essa abordagem elimina a necessidade do casamento de esquemas. A sua aplicabilidade, entretanto, fica restrita a domínios pequenos e bem definidos, como determinadas áreas de negócios, por exemplo. A dificuldade advém de duas situações. Em primeiro lugar, para um determinado domínio de problema, normalmente são gerados muitos padrões que competem entre si. A escolha de um único padrão, se alcançada, virá apenas após um longo processo de negociação. Em segundo lugar, as organizações tendem a estender os padrões com elementos proprietários, no intuito de atender a necessidades próprias. Isso levará, inevitavelmente, à necessidade de um novo processo de padronização.

Já o segundo grupo, assumindo que a necessidade do casamento de esquemas não cessa de existir, investe em estratégias para automatizar o processo. Apesar dos inevitáveis avanços, as abordagens adotadas por esse grupo apresentam dois problemas. Em primeiro lugar, elas tipicamente empregam uma única estratégia de casamento, e exploram somente certos tipos de informação, que é freqüentemente otimizada para um único tipo de aplicação. Segundo, a maior parte das propostas conseguem descobrir somente casamentos 1:1, ou seja, elas não possuem mecanismos para derivar mapeamentos como `nome = concat(prim-nome, sobrenome)`.

Além disso, a maioria dessas abordagens possui ainda a limitação de considerar apenas o esquema dos dados para o processo de casamento, sem levar em conta informações que poderiam ser obtidas por meio da análise das instâncias de dados.

Dos autores citados acima, (RAHM; BERNSTEIN, 2001), em especial, apresenta um *survey* das técnicas de casamento de esquemas desenvolvidas no contexto de tradução e integração de esquemas, representação de conhecimento, aprendizado de máquina (*machine learning*) e recuperação de informações. Como resultado do estudo, o autor apresenta uma taxonomia em que explica as características comuns

destas técnicas.

Na seção a seguir, apresenta-se a operação básica necessária para o casamento de esquemas. A seguir, descreve-se a taxonomia proposta por (RAHM; BERNSTEIN, 2001).

2.1 A operação de casamento (*Match*)

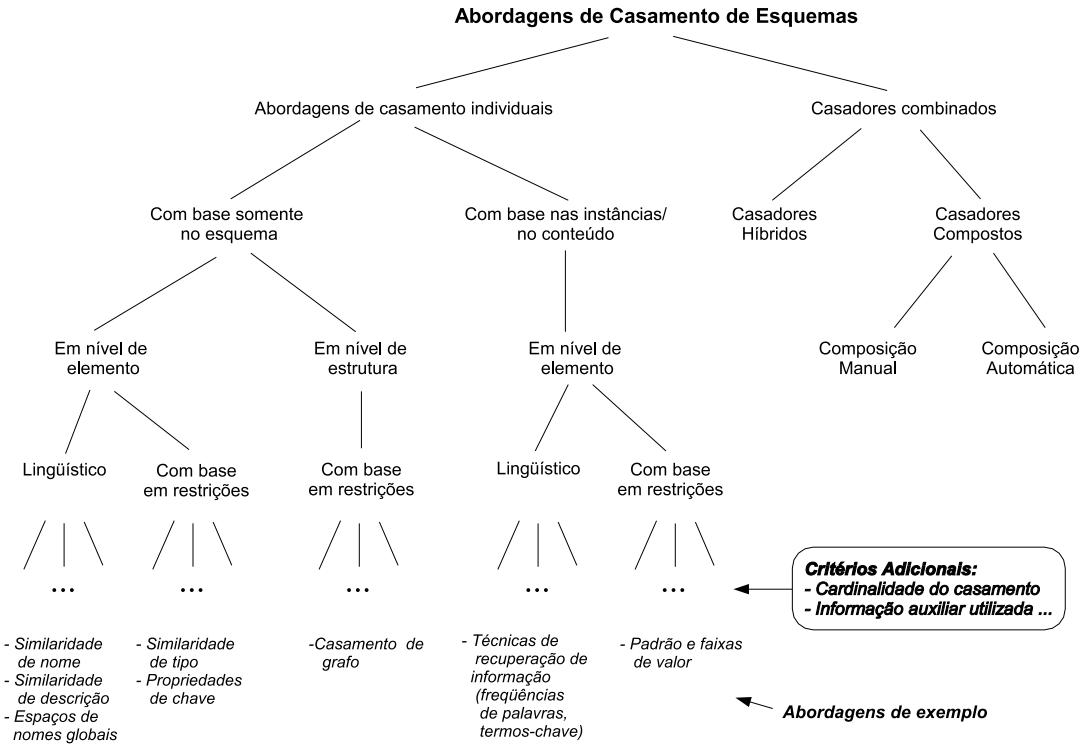
A tarefa de casar esquemas, independentemente da técnica utilizada, pode ser vista como uma generalização de uma operação fundamental, chamada casamento (*Match*). Essa operação recebe como argumentos dois esquemas e produz um conjunto de casamentos entre elementos desses esquemas[1] (RAHM; BERNSTEIN, 2001). Considerando-se, por exemplo a existência de dois esquemas, S e T, sendo S composto por um conjunto de elementos $s_1, s_2, \ldots, s_n$ e T composto por $t_1, t_2, \ldots, t_n$, um casamento $s_1 \leftrightarrow t_1$ é um valor que indica o grau de similaridade entre esses dois elementos. Normalmente este valor está contido no intervalo [0; 1], onde 1 representa o maior grau de similaridade possível.

Em (RAHM; BERNSTEIN, 2001), apresenta-se uma classificação das abordagens de casamento de esquemas em uma taxonomia, reproduzida na fig. 2.1. Para efeitos dessa classificação, considera-se que o casamento pode ser implementado de diferentes maneiras, envolvendo o uso de um conjunto de algoritmos de casamento ou casadores (*matchers*). A forma como os algoritmos são utilizados permite imaginar a solução do problema em dois níveis. No primeiro, são considerados os casadores individuais, ou seja, apenas um algoritmo é aplicado. No segundo, diversos algoritmos individuais são combinados, pelo uso de uma das seguintes alernativas: a) utilização de diferentes tipos de estratégias (por exemplo, busca por igualdade de nomes ou tipos) em um casador híbrido ou b) combinação dos resultados dos casadores individuais em um casador composto.

Os casadores individuais são classificados de acordo com o seguinte critério:

- *Instância versus esquema*: as abordagens de casamento podem considerar dados das instâncias ou somente dos esquemas.
- *Elemento versus estrutura*: os casamentos podem ser feitos considerando-se apenas elementos atômicos ou então estruturas complexas compreendendo diversos desses elementos.
- *Linguagem versus restrição*: um casador pode usar uma abordagem baseada em lingüística (nomes de atributos, por exemplo) ou em restrições, tais como chaves e relacionamentos.
- *Cardinalidade*: o processo de casamento pode resultar no casamento de um ou mais elementos de um esquema com um ou mais elementos do outro esquema.
- *Informações auxiliares*: além das informações contidas em cada esquema, algumas abordagens utilizam estruturas auxiliares, tais como dicionários, esquemas

[1] Perceba-se que não se está considerando aqui os diferentes formatos em que cada esquema pode ser representado, tais como um modelo E-R, um modelo orientado a objetos, um grafo dirigido ou um documento XML. Em cada caso, existe uma relação entre as construções do formato escolhido e os conceitos de *elemento* e *estrutura*. Por exemplo, entidades e relacionamentos em um modelo E-R e elementos, sub-elementos e ID-REFS em XML.



Fonte: (RAHM; BERNSTEIN, 2001).

Figura 2.1: Classificação das abordagens de casamento de esquemas.

globais, decisões tomadas em processos de casamento anteriores e intervenção do usuário.

A seguir são descritos alguns tópicos relacionados com os critérios de classificação dos casadores de esquema. São eles: cardinalidade de casamento, abordagens lingüísticas de casamento, casadores em nível de esquema, abordagens baseadas em instâncias, abordagens baseadas em restrições e reuso de informações de esquemas. O capítulo é fechado com a apresentação de alguns sistemas de casamento propostos na literatura.

2.2 Cardinalidade de casamento

No que diz respeito à cardinalidade, um determinado elemento pode aparecer em nenhum, um ou mais casamentos no resultado final do casamento de dois esquemas S e T. Isso pode ser generalizado para quatro situações de cardinalidade: 1:1, 1:n, n:1 e n:m. Diz-se ainda (RAHM; BERNSTEIN, 2001) que a cardinalidade pode ser *local*, quando se considera cada casamento individualmente, ou *global*, quando todos os casamentos são considerados em conjunto. A figura 2.2 apresenta exemplos de casamentos locais e globais. O casamento local 1:n significa a existência de um único casamento, onde cada instância do elemento *info* no esquema 2 equivale a combinação de instâncias de *nome* e *horario* no esquema 1. O casamento global 1:n significa a existência de dois casamentos separados, ou seja, cada professor representado no esquema 1 deve ser traduzido para dois elementos separados no esquema 2. Em ambos os casos percebe-se a necessidade de expressões de mapeamento (problema da área de mapeamento de esquemas) que permitam traduzir as instâncias

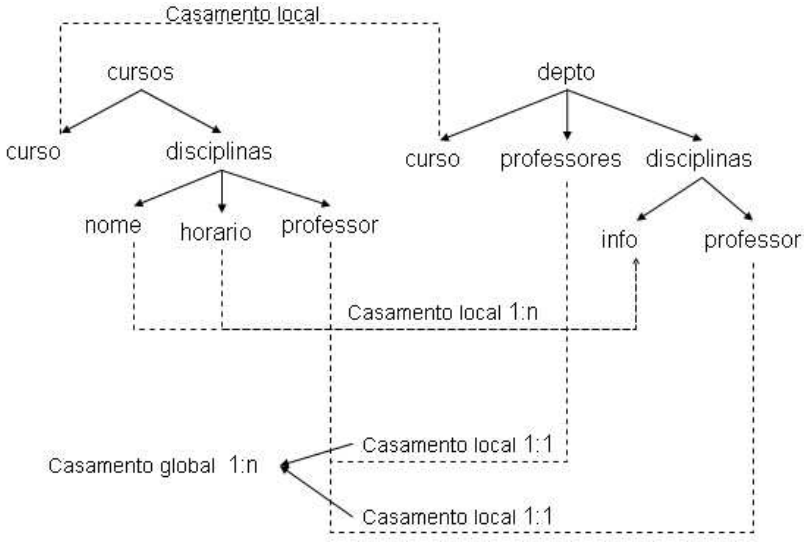


Figura 2.2: Exemplo de cardinalidade de casamento.

de um esquema para o outro.

2.3 Abordagens lingüísticas de casamento

As abordagens lingüísticas, ou baseadas em idiomas, são aplicáveis tanto aos casadores em nível de esquema quanto àqueles em nível de instância (RAHM; BERNSTEIN, 2001). Dadas duas cadeias de caracteres S_1 e S_2, o objetivo é encontrar um valor de similaridade entre essas cadeias. Em nível de esquema, elementos podem ser casados com base na semelhança dos seus nomes, ou ainda com base na semelhança dos seus descritores. Descritores são informações complementares que podem estar presentes nos esquemas sob a forma de comentários e/ou informações adicionais a respeito dos nomes dos elementos. Há várias formas de se avaliar e medir a semelhança entre os nomes, dentre as quais:

- similaridade de nomes: parte da suposição de que dois elementos possuem a mesma semântica se forem escritos de forma semelhante.
- igualdade de sinônimos: por exemplo, carro $\cong$ automóvel.
- igualdade de *hypernyms*: X é um *hypernym* de Y se Y é um tipo de X. Por exemplo, se livro é-uma publicação e artigo é-uma publicação, então livro $\cong$ publicação, artigo $\cong$ publicação, e livro $\cong$ artigo.

Algumas estratégias para avaliar a semelhança de nomes, tais como sinônimos e *hypernyms*, exigem a presença de um dicionário ou *thesaurus*. Se os esquemas a serem casados foram elaborados em idiomas diferentes, então esse dicionário deverá ser multi-idioma. Além disso, os esquemas podem usar termos que são próprios de uma área de negócios (medicina, por exemplo), e o mecanismo de casamento pode fazer uso de informações provindas de dicionários muito específicos, ou taxonomias da área em questão, que podem estar sob a forma de ontologias, por exemplo. Outro caso específico a ser considerado é a existência de homônimos, ou seja, palavras com a mesma grafia, mas com significados diferentes.

Quanto a similaridade de nomes, pode-se dividir os algoritmos em dois tipos de comparação: Comparação de cadeias pequenas e comparação de cadeias longas. Cada um desses tipos é analisado nas seções seguintes.

2.3.1 Comparação de cadeias pequenas:

Em cadeias pequenas, como palavras, todos os caracteres existentes em cada cadeia são representativos do conceito ao qual a cadeia se refere. O cálculo da similaridade se baseia, de modo geral, no cálculo do número de caracteres que as duas cadeias têm em comum. A seguir são descritas algumas técnicas usadas para o cálculo da similaridade de cadeias pequenas:

Edit Distance (BAEZZA-YATES; RIBEIRO-NETO, 1999) Essa técnica considera o custo envolvido em termos do menor número de operações de edição que aplicadas em S_1, acabam por converte-la em S_2. As operações podem ser executadas sobre caracteres individuais, e compreendem remoção, inserção, a realocação ou substituição dos mesmos. Esta métrica é largamente usada, não só para processamento textual como também para alinhamento de seqüencias biológicas, e muitas variações são possíveis. A forma mais simples da métrica é conhecida como distância *Levenstein*. Algumas variações bastante conhecidas são *NeedleMan-Wunsch* e *Smith-Waterman*.

Q-Grams (GRAVANO et al., 2001) Simplificadamente, essa técnica divide uma cadeias em termos de tamanho q, e verifica se esses termos podem ser encontrados em outra cadeia.

Soundex (ZOBEL; DART, 1996) O propósito desta técnica é comparar se duas palavras são foneticamente parecidas. De modo geral, as palavras são divididas em termos, onde cada termo corresponde a um fonema. A comparação dos fonemas em comum dá a similaridade das palavras.

Jaro (COHEN; RAVIKUMAR; FIENBERG, 2003) Esta métrica é baseada no número e ordem dos caracteres em comum em duas cadeias. Os caracteres $c_i \in S_1$ e $c_j \in S_2$ são considerados comuns se $c_i = c_j$ e $|i - j| < \frac{\min(|S_1|,|S_2|)}{2}$. Uma das variantes desta métrica, chamada de *Jaro Winkler*, usa a informação do mais longo prefixo em comum nas cadeias no cálculo da similaridade.

2.3.2 Comparação de cadeias longas:

Quando cadeias longas, como frases, são comparadas, dificilmente se encontrarão frases escritas da mesma forma, mesmo que elas possuam o mesmo significado. Para comparação de frases convém considerar apenas as palavras que possuem maior significado semântico. Geralmente essas palavras são os substantivos do texto. Assim, a comparação de frases pode vir acompanhada de um pré-processamento que determine os termos mais significativos. Nesta fase de pré-processamento podem ser aplicadas algumas operações de texto, como a eliminação de palavras que aparecem com muita freqüência (*stop-words*) e a extração das raízes gramaticais das palavras (*steeming*).

Após o pré-processamento, duas alternativas são possíveis: i) aplicar as técnicas tradicionais de casamento de cadeias curtas descritas anteriormente, como por exemplo *edit distance* e ii) usar abordagens baseada em vetores.

Nas abordagens baseadas em vetores, após o pré-processamento, cada cadeia é convertida em um conjunto de termos, onde cada termo equivale a uma das palavras da cadeia considerada. Cada conjunto da origem a um vetor de n dimensões, onde n equivale ao número de termos de cada conjunto. O cálculo da similaridade entre duas cadeias envolve técnicas de comparação dos vetores destas cadeias. Para fins de compreensão, os vetores (ou conjuntos) originados das cadeias S_1 e S_2 serão referenciados por V_1 e V_2. Abaixo são listadas algumas técnicas baseadas em vetores usadas na literatura:

Coeficiente Simples (VAN RIJSBERGEN, 1979) Esta é uma abordagem bastante simples que conta o número de termos(dimensões) onde ambos os vetores são não zero. Assim, dados os conjuntos de termos V_1 e V_2, o coeficiente de similaridade é $|V_1 \cap V_2|$. Isto pode ser visto como uma contagem de termos relacionados baseada em vetores.

Coeficiente Dice (VAN RIJSBERGEN, 1979) A medida de similaridade é definida como o dobro do número de termos em comum dividido pelo número total de termos em ambas cadeias. A similaridade é medida de 0 até 1, onde o valor 1 indica que os vetores são idênticos e o valor 0 indica que os vetores são ortogonais. A equação abaixo demonstra o calculo da similaridade:

$$sim = \frac{2 * |V_1 \cap V_2|}{|V_1| + |V_2|} \tag{2.1}$$

Jaccard (COHEN; RAVIKUMAR; FIENBERG, 2003) Esta medida é semelhante a métrica *dice*, sendo que a principal diferença é que a primeira penaliza mais do que a segunda quando as cadeias têm poucos termos em comum.

$$sim = \frac{|V_1 \cap V_2|}{|V_1 \cup V_2|} \tag{2.2}$$

Coeficiente Overlap (VAN RIJSBERGEN, 1979) Esta métrica procura valorizar situações onde um dos conjuntos é um sub-conjunto do outro. Quanto mais um sub-conjunto estiver contido dentro do outro, maior o coeficiente de similaridade.

$$sim = \frac{|V_1 \cap V_2|}{\min(|V_1|, |V_2|)} \tag{2.3}$$

TF/IDF (BAEZZA-YATES; RIBEIRO-NETO, 1999) esta métrica é tipicamente usada para computar a relevância de páginas em mecanismos de busca de páginas *Web*. Este problema pode ser generalizado como o problema de encontrar, dentro de uma coleção de documentos N, os documentos que mais correspondem a uma consulta que seja expressa como um conjunto de termos. Nesta abordagem, cada documento equivale a um vetor, sendo que os termos do vetor são normalizados com a aplicação de pesos. Dado um termo t em um documento d, o peso w deste termo está relacionado com a freqüência de ocorrência do termo no documento (tf) e com a importância do termo dentro da coleção inteira (idf). Abaixo é mostrada uma equação comumente usada para o calculo de idf

$$idf(t) = \log(\frac{N}{n_t}) \tag{2.4}$$

onde n_t é o número de documentos onde o termo t ocorreu. Conforme a equação, quanto mais raro o termo dentro do conjunto N, maior será o seu *idf*. A arquitetura baseada em TF-IDF envolve uma série de outras questões que não são abordadas neste trabalho. É importante salientar que o peso de um termo na consulta é proporcional ao seu *idf*, e que a noção de *idf*, apesar de ser mais comumente usada em mecanismos de busca, também pode ser empregada explicitamente como métrica de similaridade. Uma métrica deste tipo pode ser útil em esquemas que possuem comentários em linguagens natural para expressar o significado semântico dos elementos do esquema. A similaridade destes comentários pode ser usada para determinar a similaridade entre os elementos dos esquemas. Técnicas baseadas em pesos geralmente são melhores aos métodos insensíveis às freqüências individuais dos termos.

2.4 Casadores em nível de esquema

Os casadores em nível de esquema consideram para a tarefa de casamento apenas as informações contidas nos esquemas. Isso inclui informações sobre os nomes dos elementos, os tipos de dado de cada um (inteiro, data, etc.), a estrutura (relacionamentos pai-filho, irmãos, etc.) e restrições dos relacionamentos (é-um, parte de, etc.).

Uma questão a ser considerada na aplicação de casadores em nível de esquema diz respeito à granularidade do casamento. Com relação à granularidade, existem duas situações:

- em nível de elemento: apenas elementos no nível de granularidade mais baixo (ou seja, no nível *atômico*) são considerados.
- em nível de estrutura: neste caso, o processo de casamento considera elementos que podem aparecer agrupados em um determinado esquema. Podem ocorrer situações em que estruturas completas podem ser casadas ou, alternativamente, pode haver um casamento parcial entre duas estruturas.

2.5 Abordagens com base em instâncias

A aplicação das técnicas descritas na seção anterior pode ser suficiente, na maioria dos casos, para se obter conjuntos bastante completos de casamentos. Existem situações, entretanto, nas quais os esquemas fornecem apenas informações limitadas sobre os dados armazenados. Isso acontece, por exemplo, com os dados semi-estruturados, nos quais podem não haver informações sobre o tipo de dados (se todos os dados forem tratados como sendo do tipo texto) ou restrições. Quando isso acontece, as informações disponíveis no esquema podem não ser suficientes para se produzir um conjunto razoável de casamentos. A solução para isso é buscar informações complementares nas instâncias de dados.

Uma outra consideração a ser feita é que, mesmo nas situações nas quais o esquema possui informações detalhadas sobre os dados (esquemas de bancos de

dados relacionais, por exemplo), ainda assim pode-se utilizar a análise das instâncias para validar e completar os casamentos. Por exemplo, em situações nas quais existe ambigüidade entre dois casamentos, similaridades entre as instâncias pode ajudar a decidir qual deles é o melhor.

Abaixo são descritas duas técnicas usadas para o casamento de esquemas com base em instâncias:

- técnicas de recuperação de informação (*information retrieval*) podem ser aplicadas para obter uma caracterização lingüística de elementos de texto, como, por exemplo, a extração de palavras-chave com base na freqüência da ocorrência de determinados termos;

- para atributos mais estruturados, tais como números e datas, pode-se aplicar uma caracterização com base em restrições, identificando-se faixas de valores adequadas, médias ou padrões de caracteres.

As características das instâncias obtidas a partir da sua análise pode ser aplicada de duas formas:

- usar a caracterização para melhorar os casadores de nível de esquema;

- efetuar o casamento baseado em instâncias por si mesmo.

Na segunda situação, as instâncias do esquema S_1 são avaliadas para caracterizar o conteúdo dos elementos de S_1 e, então, as instâncias de S_2 são casadas uma por uma contra as caracterizações dos elementos de S_1. Os resultados do casamento por instância passam por um processo de agrupamento e abstração para o nível de esquema, a fim de gerar um *ranking* de candidatos a casamento em S_1 para cada elemento em nível de esquema em S_2. Várias abordagens têm sido propostas para fazer casamento por instâncias, tais como redes neurais (LI; CLIFTON, 1994, 2000a,b) e técnicas de aprendizado de máquina (BERLIN; MOTRO, 2001; DOAN; DOMINGOS; LEVY, 2000a; DOAN; DOMINGOS; HALEVY, 2001; KURGAN; SWIERCZ; CIOS, 2002).

2.6 Abordagens baseadas em restrições

Os esquemas normalmente contém informações sobre restrições, tais como o tipo dos dados, intervalos de valores, unicidade, opcionalidade, tipos de relacionamento, cardinalidades, etc. Caso ambos os esquemas a serem casados possuírem esse tipo de informação, ela pode ser usada pelo casador para determinar a similaridade de elementos do esquema. Não é recomendável, entretanto, usar restrições como a única fonte de informações para a tarefa de casamento, pois isso pode levar a mapeamentos n:m errôneos. Considerando-se, por exemplo, que um esquema possui um atributo `nome`, do tipo `varchar`, e o outro possui um atributo `cidade` e outro `endereço` do mesmo tipo. Caso seja utilizado somente o tipo dos campos como informação, o atributo do primeiro esquema será casado erroneamente com ambos os atributos do segundo esquema.

2.7 Reuso de informações de esquemas

O reuso de informações para o processo de casamento pode ocorrer de duas formas:

- por meio do uso de estruturas auxiliares, tais como dicionários, *thesauri*, etc.
- pelo armazenamento e recuperação de esquemas e casamentos anteriores.

Nesta situação, elabora-se um banco de dados contendo esquemas já utilizados e os casamentos resultantes entre os elementos destes esquemas. Isso é vantajoso quando existe probabilidade de que tarefas de casamento sejam semelhantes àquelas já efetuadas. Um exemplo de aplicação em que isso pode acontecer é o comércio eletrônico, em que estruturas são normalmente repetidas dentro de mensagens, tais como linhas de itens de produto dentro de um pedido. Um sistema dessa natureza, chamado *Mapping Knowledge Base* – MKB, é descrito em (MADHAVAN et al., 2003). Neste sistema, o conjunto de esquemas e informações de casamento – denominado *corpus* – é construído a partir de casamentos conhecidos e resultados de tarefas anteriores de casamento.

2.8 Sistemas de casamento de esquemas

Nesta seção são descritos alguns sistemas de casamento de esquemas propostos na literatura. Em (DO; MELNIK; RAHM, 2002), apresenta-se uma comparação de alguns desses sistemas.

2.8.1 Autoplex and Automatch

Automatch (BERLIN; MOTRO, 2002) é uma versão melhorada de Autoplex-(BERLIN; MOTRO, 2001). Ambos os sistemas apresentam uma solução de estratégia única baseada em técnicas de aprendizado de máquina. Um *learner* baseado em uma rede bayesiana explora características das instâncias para casar atributos de um esquema relacional (origem) para um esquema global previamente construído. Para cada atributo do esquema de origem, são calculadas as probabilidades de casamento e não-casamento com relação a cada elemento do esquema global. As probabilidades são normalizadas para somarem um, e a probabilidade de casamento é devolvida como a medida de *similaridade* entre os atributos da origem e do esquema global. As correspondências são filtradas para maximizar a soma das suas similaridades sob a condição de que nenhuma correspondência compartilhe um elemento comum.

2.8.2 Clio

Clio (MILLER; HASS; HERNáNDEZ, 2000) é um sistema originalmente proposto para resolver o problema de mapear bancos de dados legados para um novo esquema. Nesse sentido, o sistema gera um conjunto de definições de visões que permitem que as aplicações acessem diretamente as fontes de dados, por meio de um mecanismo de consulta *middleware*. Clio produz as consultas SQL para o usuário, fornecendo exemplos de dados e outros tipos de *feedback* para permitir que eles compreendam os mapeamentos que foram produzidos. A base do sistema é o conceito de *correspondência de valor* (*value correspondence*), que é um par, consistindo de (1) uma função que define como um valor (ou combinação de valores) de um banco

de dados de origem pode ser usado para formar um valor no destino e (2) um filtro, que indica quais valores de origem podem ser usados. O resultado desse processo é uma consulta SQL que faz o mapeamento entre os bancos de dados de origem e de destino. Posteriormente (POPA et al., 2002), Clio foi estendido para suportar, além de esquemas relacionais, também esquemas XML. O mecanismo de integração, entretanto, continua sendo o mesmo.

2.8.3 COMA

COMA (DO; RAHM, 2002) segue uma abordagem de composição, que fornece uma biblioteca extensível de diferentes casadores e suporta diversas maneiras de combinar os resultados do casamento. O sistema COMA foi desenvolvido para combinar algoritmos de casamento de uma forma flexível, de onde vem a origem do nome: *COmbining MAtching*. O processo de casamento se baseia na extração de informações do esquema, que pode ser complementada por um casador específico para reutilizar os resultados de operações de casamento anteriores. Esses resultados ficam armazenados em um repositório baseado em banco de dados relacional. Os esquemas – que podem ser tabelas relacionais ou documentos XML, por exemplo – são representados por grafos acíclicos dirigidos, nos quais os nós representam elementos do esquema, que são conectados por arcos dirigidos que representam diferentes tipos de relacionamentos. Os esquemas são importados de fontes externas, como um banco de dados relacional, por exemplo, e convertidos para o formato interno, sobre o qual são aplicados os operadores de casamento. O resultado é um conjunto de casamentos, compostos por elementos do esquema, juntamente com um valor de semelhança entre zero (forte dessemelhança) e um (forte semelhança). A biblioteca de casadores inclui um conjunto de algoritmos simples, tais como *q-gram*, *EditDistance* e *DataType* (similaridade relacionada com os tipos dos dados), que podem ser agrupados de forma fixa para compor casadores híbridos.

2.8.4 Cupid

Cupid (MADHAVAN; BERNSTEIN; RAHM, 2001) apresenta uma abordagem híbrida, combinando um casador de nome com um casador de estrutura. O casador de nomes é aplicado primeiro, utilizando uma abordagem lingüística que considera o nome do elemento e, em segundo plano, informações restritas sobre o tipo e a estrutura do esquema. O casador lingüístico funciona em três passos: *normalização*, onde um *thesaurus* é utilizado para selecionar palavras com radicais semelhantes; *categorização*, onde os elementos de cada esquema são separados em categorias, com base nos tipos de dados, na hierarquia do esquema ou no conteúdo lingüístico dos seus nomes; e *comparação*, onde coeficientes de similaridade (*linguistic similarity coefficients – lsim*) são computados, pela comparação dos termos extraídos dos seus nomes, usando um *thesaurus* com relacionamentos de sinonímia e hipernímia. O resultado da fase lingüística é uma tabela de coeficientes *lsim* entre elementos dos dois esquemas, cujos valores estão na faixa [0, 1], na qual o valor "1" indica um casamento lingüístico perfeito. Na fase de casamento estrutural, um algoritmo calcula a semelhança estrutural (*structural similarity – ssim*), que é uma medida de semelhança dos contextos nos quais os elementos ocorrem nos dois esquemas. A partir da *ssim* e da *lsim* calculada anteriormente, calcula-se a semelhança ponderada (*weighted similarity – wsim*).

2.8.5 LSD e GLUE

O sistema LSD (*Learning Source Descriptions*) (DOAN; DOMINGOS; LEVY, 2000b) e a sua extensão GLUE (DOAN et al., 2002) usam uma abordagem composta baseada em aprendizado de máquina para combinar diferentes casadores. LSD extrai elementos de um novo esquema e os mapeia para um esquema mediado previamente elaborado, em duas etapas: *treinamento* e *classificação*. Na etapa de treinamento, os *learners* são treinados por meio do processamento de um conjunto de instâncias-exemplo. Após, é iniciada a etapa de classificação, na qual os *learners* individuais utilizam o conhecimento aprendido na etapa anterior para reconhecer os elementos do novo esquema. Uma vez encerrado esse processo, um *meta-learner* combina as indicações de todos os *learners* para formar uma previsão final.

O sistema GLUE é uma extensão do LSD, e seu objetivo é buscar correspondências entre as taxonomias de duas ontologias. GLUE possui três módulos principais: o *Distributor Estimator*, o *Similarity Estimator* e o *Relaxation Labeler*. O primeiro, *Distributor Estimator*, recebe duas taxonomias como entrada, juntamente com as suas instâncias e aplica técnicas de aprendizado de máquina para calcular a distribuição de probabilidade para cada par de conceitos. A seguir, os resultados dessa etapa são encaminhados para o *Similarity Estimator*, que aplica uma função de similaridade fornecida pelo usuário para calcular um valor de similaridade para cada par de conceitos. Como resultado, é criada uma *matriz de similaridade* entre os conceitos nas duas taxonomias. A seguir, o *Relaxation Labeler* utiliza a matriz de similaridade e restrições de domínio para buscar a combinação de casamentos que melhor satisfaz as restrições de domínio.

2.8.6 SemInt

SemInt (LI; CLIFTON, 2000a) apresenta uma abordagem híbrida, que explora tanto informações do esquema quanto das instâncias para identificar atributos correspondentes entre esquemas relacionais. As restrições em nível de esquema, tais como tipos de dados e restrições de chave, são extraídas do catálogo do banco de dados. Os dados das instâncias são explorados para obter informações adicionais, tais como distribuições de valores, médias numéricas, etc. Para cada atributo, SemInt atribui uma assinatura, que consiste de valores no intervalo [0,1] para todos os critérios de casamento envolvidos. Essas assinaturas são usadas para agrupar (*clustering*) atributos semelhantes do primeiro esquema e, após, encontrar o melhor agrupamento para atributos do segundo esquema. Os processos de agrupamento e classificação são executados por redes neurais com treinamento automático, o que reduz o esforço necessário antes do casamento propriamente dito. O resultado consiste de agrupamentos de atributos similares de ambos os esquemas.

2.8.7 Similarity Flooding (SF)

SF (MELNIK; GARCIA-MOLINA; RAHM, 2002) utiliza uma abordagem baseada em grafos para estabelecer casamentos entre dois esquemas, que podem estar representados em SQL DDL, XML ou RDF. Os casamentos são determinados pela aplicação de um algoritmo que utiliza um cálculo de ponto fixo, tendo como base um valor de semelhança determinado por um casador que analisa os nomes dos elementos. Esse valor é reavaliado em seguidas iterações, até que um determinado resíduo seja alcançado. Ao contrário de outras abordagens, SF não explora os rela-

cionamentos terminológicos em um dicionário externo, restringindo o seu cálculo de similaridade aos nomes dos elementos. No último passo, diversos filtros podem ser especificados para selecionar subconjuntos relevantes de resultados de casamentos produzidos pelo casador estrutural.

2.8.8 Considerações Finais

A efetividade dos sistemas atualmente disponíveis de casamento automático de esquemas não é clara (RAHM; BERNSTEIN, 2001), porque as avaliações disponíveis foram conduzidas de diferentes formas, tornando difícil essa verificação. Além disso, todas as avaliações demonstraram forte degradação dos resultados com esquemas grandes.

Quanto aos tipos de modelos casados, nenhum sistema se compromete ao casamento entre esquemas XML e esquemas relacionais. Na verdade, abordagens genéricas como (MADHAVAN; BERNSTEIN; RAHM, 2001; MELNIK; GARCIA-MOLINA; RAHM, 2002) podem ser usadas para esse fim, mas elas não utilizam as particularidades que podem existir nos casamentos entre esquemas XML e relacional. Com o uso de abordagens específicas para o casamento entre estes dois modelos, muito mais recursos podem ser usados para encontrar as similaridades entre esquemas. Já o sistema COMA (DO; RAHM, 2002) diferencia-se dos demais por ter como objetivo principal criar um ambiente que permita combinar diferentes casadores. Assim, COMA pode ser usado para o problema de casamento entre esquemas XML e relacionais, desde que os algoritmos criados sejam compatíveis com o ambiente interno do sistema.

3 VISÃO GERAL DO PROCESSO DE CASAMENTO PROPOSTO

Neste capítulo é apresentado o modelo de execução que rege o processo de casamento de esquemas adotado neste trabalho. Este modelo cumpre duas funções importantes: i) define como os esquemas a serem casados interagem com os algoritmos de casamento e ii) estabelece critérios a serem usados no processo de casamento propriamente dito.

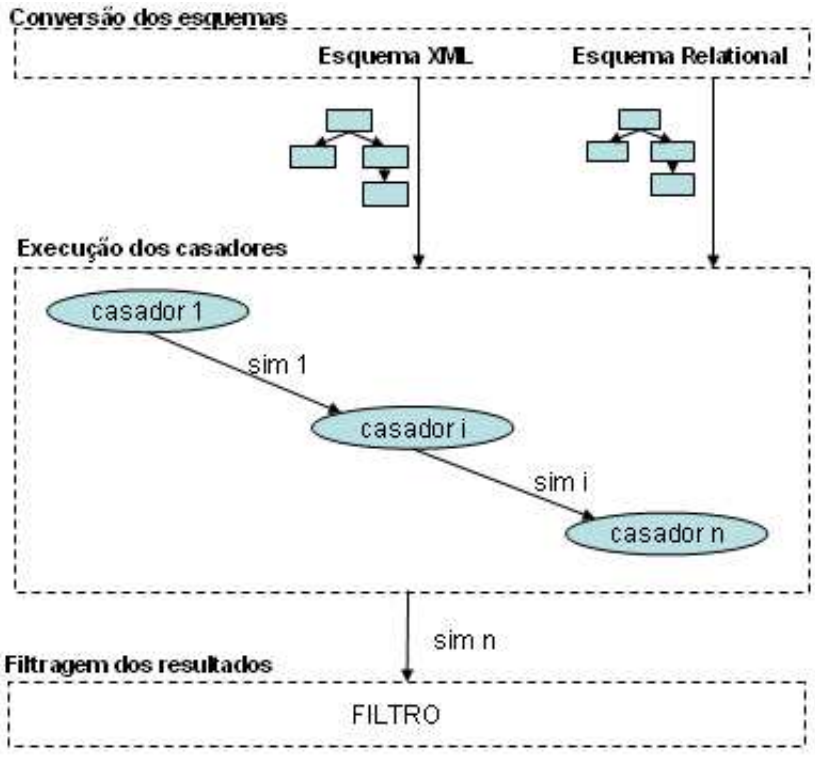


Figura 3.1: Visão geral do processo de casamento

Estas funções estão representadas na figura 3.1. Esta figura demonstra uma visão geral dos processos envolvidos na tarefa de casamento de esquemas XML e relacional. A descrição de cada um dos processos é dada abaixo:

Conversão dos esquemas Nesta primeira etapa o usuário interage com o sistema fornecendo os esquemas que ele deseja que sejam casados. Estes esquemas estão em seu formato original, e precisam ser transformados em uma repre-

sentação comum para que os casadores possam compreendê-lo. Esta representação comum, ou esquema canônico, é composto por um conjunto de nós e arcos conectando estes nós. Maiores detalhes sobre a transformação de esquemas XML e relacional em um esquema canônico são encontrados na seção 3.1.1.

Execução dos casadores O próximo passo é alimentar os algoritmos de casamento de esquema com os dois esquemas canônicos provindos da etapa anterior. O objetivo de cada algoritmo é inferir um valor de similaridade entre cada par de nós, sendo que cada par é composto por um nó XML e um nó relacional. É importante salientar uma restrição adotada neste trabalho. Somente podem ser casados os nós atômicos. Esta restrição está relacionada com o fato de que somente os nós atômicos podem conter informações. Os nós atômicos XML correspondem a elemento PCDATA e atributos, enquanto nós atômicos relacionais correspondem a colunas de tabela. Os demais nós dos esquemas tem função puramente estrutural. O resultado da execução de cada casador é um conjunto de triplas (n_i, n_j, sim_{ij}), onde n_i é um nó atômico XML, n_j é um nó atômico relacional e sim_{ij} é a similaridade obtida para o par. Os casadores são executados em seqüência, sendo que o resultado da execução de um casador é uma combinação das similaridades por ele computadas e das similaridades computadas pelo casador anterior. A forma como as similaridades são combinadas é discutida na seção de experimentos do capítulo 5.

Filtro dos resultados Esta etapa tem por objetivo eliminar casamentos com base em alguma evidência que os caracteriza como casamentos incorretos. Um dos filtros utilizados envolve restrições de cardinalidade. Considera-se neste trabalho que os casamentos entre os esquemas possuem cardinalidades local e global 1:1, ou seja, um nó de um esquema pode ser casado com apenas um nó de outro esquema. A restrição de cardinalidade 1:1 funciona da seguinte forma: dando-se as opções de casamento (n_1, n_2, sim_{12}) e (n_1, n_6, sim_{16}), pode-se eliminar a opção que apresentar a menor similaridade para n_1. Os demais filtros usados são apresentados na seção de experimentos dos capítulo 5.

3.1 Representação dos esquemas

Os esquemas XML e relacional são representados neste trabalho como grafos acíclicos. Os *casadores* usam estes grafos para realizar as comparações entre os nós de ambas representações. Nesta seção é descrito o formalismo envolvido na conversão destes esquemas para uma representação em grafos.

A figura 3.2 mostra a descrição de um esquema relacional, juntamente com o grafo gerado para ele. Um esquema relacional pode ser descrito conforme a definição abaixo:

Definição 1 *(Esquema Relacional) Considere que* $T = \{t_1, ..., t_n\}$ *seja um conjunto de tabelas de um esquema relacional RS. Cada tabela* t_i *tem um conjunto de colunas* $tc_i = \{c_1, ..., c_n\}$ *. Além disso, considere que* $C = \{c_i\}$ *seja um conjunto contendo todas as colunas do esquema relacional. A tupla* $k = (t_i, t_j)$ *representa uma relação de chave estrangeira entre duas tabelas, onde* t_j *é a tabela referenciada por* t_i*. Ademais, o conjunto* FK *contém todas as colunas de RS que são chaves estrangeiras.*

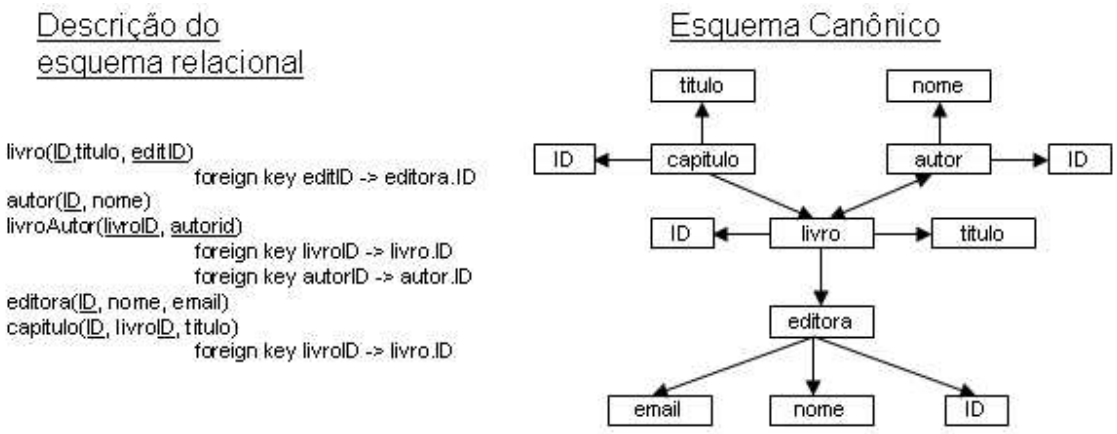


Figura 3.2: Esquema canônico relacional

A definição 1 considera uma versão simplificada de um esquema relacional, composto por tabelas, atributos e relações de chave estrangeira. Os demais componentes de um esquema relacional não são tratados por não serem relevantes ao problema de casamento de esquemas.

Definição 2 *(Grafo Relacional) Considere que a tripla* $RG =< N, E, name >$ *seja um grafo gerado para um RS, onde* $N = T \cup (C - FK)$ *são os nós do grafo,* $E \subseteq T \times (T \cup (C - FK))$ *são os arcos, e* $name : N \rightarrow \eta$ *(onde* η *é um conjunto de nomes), é a função de nome.*

Segundo a definição 2, todo t_i é transformado em um nó tabela. Da mesma forma, toda coluna que não é chave estrangeira é transformada em um nó coluna. O nome dos nós é determinado pela função *name*.

Os arcos conectam tabelas com colunas e tabelas com tabelas, e podem ser direcionados ou bi-direcionados. Os arcos conectando tabelas com colunas são descritos na definição 3.

Definição 3 *(Arcos de Contenção) Considere que* $ed(t_j \rightarrow c_i)$ *seja um arco direcionado conectando um nó tabela com um nó coluna. Assim,* $\forall\ c_i \in tc_j\ |c_i \notin FK, (\exists\ ed(t_j \rightarrow c_i))$.

Os arcos direcionados são chamados arcos de contenção. Na verdade, esta definição também é usada em (MADHAVAN; BERNSTEIN; RAHM, 2001). Arcos de contenção (*Containment edges*) são arcos direcionados que indicam que um nó está contido dentro de outro. Uma propriedade que caracteriza estes arcos é que um dado nó só pode estar contido dentro de um único nó pai. Em outras palavras, não é possível que dois arcos tenham como destino o mesmo nó.

Segundo a definição 4, os arcos de relação conectam nós tabela com nós tabela de acordo com as relações de chave estrangeira existentes entre elas.

Definição 4 *(Arcos de Relação) Considere que* $ed(t_i \rightarrow t_j)$ *seja um arco direcionado conectando dois nós tabelas. Então,* $\forall\ k = (t_i, t_j),\ \exists\ ed(t_i \rightarrow t_j)$.

A semântica dos arcos de contenção é diferente da semântica dos arcos de relação. Os algoritmos que utilizarem essa representação devem estar cientes dessas diferenças para que a interpretação dos grafos seja adequada.

Percebe-se pela definição 2 que as chaves estrangeiras não são representadas como nós, já que elas são na verdade apontadores para outras tabelas, e esses apontadores já são representados como arcos de relação (definição 4). Podem existir casos onde as chaves estrangeiras ajudem no processo de cálculo da similaridade. No entanto, o tratamento de chaves estrangeiras é um tópico não considerado nesta dissertação.

A definição 5 indica como a conversão deve proceder no caso das tabelas associativas. Tabelas associativas são aquelas que substituem os relacionamentos n:m entre tabelas.

Definição 5 *(Regra das Tabelas Associativas) Considere que* $ed(t_i \leftrightarrow t_j)$ *seja um arco bi-direcionado conectando dois nós tabela. Além disso,* te_i *é o conjunto de arcos de relação direcionados para* t_i*. Assim,* $\forall\ (t_j \in te_i, t_k \in te_i), (\exists\ ed(t_j \leftrightarrow t_k))$*. Ainda, se* $tc_i \subseteq FK$ *então* $(N = N - t_i)$*.*

Para o caso de uma tabela associativa cujos todos atributos sejam chaves estrangeiras, esta tabela é eliminada do grafo e são inseridos arcos de relação bi-direcionados entre as tabelas que estavam previamente conectadas a ela. Um exemplo de aplicação desta regra ocorre com a tabela **livroAutor**. O nó tabela correspondente a esta tabela foi removido e foi criado um arco bi-direcionado conectando os nós tabela **livro** e **autor**. Vale frisar que uma tabela associativa só é removida caso todos os seus atributos sejam do tipo chave estrangeira.

A seguir são apresentadas as definições usadas na conversão de esquemas XML em grafos. Considera-se que a estrutura dos documentos XML seja descrita no formato DTD. Formatos mais complexos, como XML Schema não são abordados neste trabalho. XML Schema é um padrão mais recente reconhecido pela W3C, de modo que atualmente poucos documentos XML são construídos com base neste padrão. Trabalhos futuros podem estudar formas de casar esquemas descritos neste formato. A figura 3.3 mostra a descrição de um esquema XML juntamente com o grafo gerado para ele. Para a descrição de DTDs foi adotada uma variação das definições apresentadas em (ARENAS; LIBKIN, 2002).

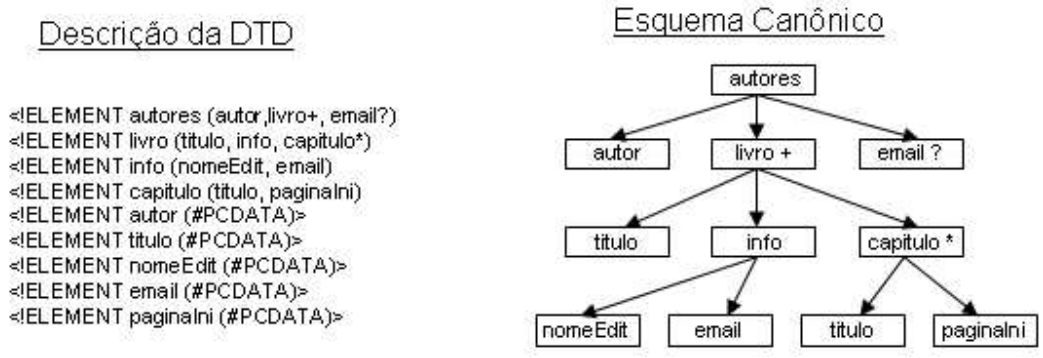


Figura 3.3: Esquema canônico DTD

Definição 6 *(DTD) Uma DTD D é uma tupla (E, A, G,* M_E*,*M_G *,*M_A*), onde*

- *E é o conjunto de elementos;*
- *A é o conjunto de atributos;*

- *G é o conjunto de grupos;*
- *M_E é o mapeamento de E para uma definição do tipo de elemento. Dado $e \in E$, $M_E(e) = S \mid \varepsilon$, ou $M_E(e)$ é uma expressão regular α definida como:*

$$\alpha \ ::= \alpha, \alpha \mid e' \mid \alpha^* \mid \alpha^+ \mid \alpha^? \mid g$$

onde $e' \in E$, $g \in G$, "," indica seqüencia de elementos, e "", "+"e "?" indicam cardinalidades ("zero ou mais", "um ou mais"e "zero ou um", respectivamente).*

- *M_G é o mapeamento de G para um sub-conjunto de E. Dado $g \in G$, $M_G(g)$ é definido como:*

$$M_G(g) \ ::= (\alpha) \mid (\alpha)^? \mid (\alpha)^+ \mid (\alpha)^*$$

- *M_A é o mapeamento de E para o conjunto de A. Se $@a \in M_A(e)$, diz-se que $@a$ está definido para e.*

Se $M_E(e) = S$, então o conteúdo de e é textual (`#PCDATA`). Se $M_E(e) = \varepsilon$, então o conteúdo de e é vazio (`EMPTY`). Um grupo corresponde a um conjunto de elementos circundados por parênteses. Nota-se que na definição acima não é permitido conteúdo misto ou DTDs com disjunções (operador de escolha).

Definição 7 *(Grafo DTD) Considere que a tripla $DG =< N, Ed, name >$ seja o grafo gerado para um D, onde $N = E \cup G \cup A$ são os nós do grafo, $Ed \subseteq ((E \times E \times G) \cup A$ são os arcos, e $name : N \rightarrow \eta$(onde η é um conjunto de nomes) é a função de nome.*

Em suma, todos elementos, atributos e grupos são transformados em nós. O nome dos nós é determinado pela função *name*. Os nós são conectados de acordo com a definição 8.

Definição 8 *(Arcos DTD) Considere que $ed(n_i \rightarrow n_j)$ seja um arco direcionado conectando dois nós. Então, $\forall\ w \in M_E(e_j), \exists\ ed(e_j \rightarrow w)$. Ainda, $\forall\ e_i \in M_G(g_j)$, $\exists\ ed(g_j \rightarrow e_i)$. Para concluir, $\forall\ @a \in M_A(e_j), \exists\ ed(e_j \rightarrow @a)$*

A relação entre pais e filhos na DTD é transformada em arcos direcionados, partindo do nó pai em direção ao nó filho. Existem arcos entre elementos, arcos entre um grupo e um elemento e arcos partindo de um elemento em direção a um atributo. Os arcos de uma DTD são todos arcos de contenção. Como esses arcos são caracterizados por não permitirem que um mesmo nó seja filho de pais diferentes, elementos desse tipo na DTD devem ser duplicados. Observa-se que na figura 3.3 os elementos **email** e **titulo** aparecem como nós filho duas vezes. A duplicação dos nós preserva o contexto hierárquico dos elementos. Essa forma de conversão é importante por dois motivos. Em primeiro lugar, nós com hierarquia diferente representam conceitos diferentes e podem ser casados individualmente com elementos de um outro esquema. Em segundo lugar, conforme (WANG et al., 2003), algoritmos de

casamento podem explorar as informações de hierarquia de um elemento afim de obter um casamento adequado para ele.

Um elemento poderia ter mais de um pai, desde que os atributos do tipo ID / IDREF fossem analisados de forma mais elaborada. Esses operadores representam relacionamentos cruzados entre elementos. No entanto, a descoberta desses relacionamentos envolveria análise de instâncias XML, o que torna a sua aplicação menos viável, já que nem sempre haverão instâncias XML disponíveis para análise. Além disso, tais operadores são pouco usados, de forma que o seu estudo não foi contemplado neste trabalho. A propósito, DTDs sem esses identificadores podem ser vistas como árvores, o que simplifica o processo de casamento de esquemas. Na verdade, como em (REYNAUD; SIROT; VODISLAV, 2001), uma DTD pode ser vista como uma floresta de árvores, onde cada nó raiz corresponde a um elemento sem pai.

3.1.1 Considerações gerais

Cada nó do esquema canônico possui um valor que o identifica exclusivamente dentro do grafo a que ele pertence. Os nós XML são unicamente identificados através de uma expressão *xpath* que contém todos os elementos que fazem parte da hierarquia do nó em questão, até a raiz. Por exemplo, os dois nós **email** são identificados por *</autores/email>* e por *</autores/livro/info/email>*. Os nós tabela de um esquema relacional são identificados simplesmente pelo seu nome, por exemplo *<book>*. Já os nós atributos do esquema relacional são identificados pela concatenação dos seus nomes com nome da tabela a que eles pertencem. Assim, os dois atributos **titulo** são identificados por *<livro.titulo>* e por *<capitulo.titulo>*

A cardinalidade múltipla dos elementos("*","+") e dos grupos é uma informação interna dos nós do grafo do esquema XML. Esta informação enriquece a semântica do grafo e pode ser usada no processo de casamento de esquemas (LARSON; NAVATHE; ELMASRI, 1989). No grafo do esquema relacional, os nós não contém informações de cardinalidade múltipla. Esta informação está implícita nos arcos de relação existentes entre os nós tabela. Por exemplo, no grafo da figura 3.2, o arco entre *<livro>* e *<editora>* indica que para uma editora podem estar relacionados vários livros. A cardinalidade opcional não é tratada neste trabalho, mas tal informação pode ser capturada nos nós atributo, para aqueles atributos que são definidos como nulos, e para elementos DTD definidos com cardinalidades opcional "?".

4 CASADOR LINGÜÍSTICO CARLA

Neste capítulo é apresentado o casador lingüístico desenvolvido nesta dissertação, chamado *Carla*. Fazem parte deste casador um algoritmo de reconhecimento dos caracteres em comum entre duas cadeias, e uma métrica que permite computar a similaridade com base nos caracteres em comum.

A técnica projetada neste trabalho para o casamento lingüístico visa combater uma deficiência encontrada nas demais técnicas existentes. Essa deficiência advém de casos onde as cadeias comparadas são compostas por termos, e a maior diferença entre elas é a ordem em que os seus termos aparecem. Além disso, os termos em cada cadeia não são separados por espaços em branco. Esse tipo de cadeia pode ser encontrada tanto em esquemas XML como em esquemas relacionais. Em ambos esquemas, os nomes dos conceitos não podem conter espaços em branco.

Para exemplificar, considera-se as cadeias $S_1 = nomeAutor$ e $S_2 = autorNome$. Tais cadeias são compostas por dois termos *("nome", "autor")*, sendo que a única diferença entre elas é a ordem dos termos[1]. Intuitivamente, pode-se concluir que trata-se da mesma informação. No entanto, técnicas tradicionais de comparação de nomes retornam resultados abaixo do esperado. Por exemplo, se as técnicas de casamento *Edit Distance* e *q-grams* com $q = 2$ fossem aplicadas para esse caso, o retorno seria respectivamente 0.11 e 0.7. O ideal para esse tipo de caso, onde ocorre uma inversão de sub-cadeias nos nomes de conceitos que semanticamente são equivalentes, seria que a similaridade fosse total (1.0).

Existem técnicas que possibilitam inferir a similaridade entre duas cadeias verificando os termos que elas têm em comum, onde um termo é uma seqüencia de caracteres adjacentes. São as técnicas baseadas em termos. Sabendo quais são os termos das duas cadeias, pode-se utilizar diversas métricas que computem a similaridade com base nos termos em comum (*Dice*, *Overlap*, *Jaccard*, ...).

Nessas técnicas, a identificação dos termos é trivial: cada conjunto de caracteres separado por espaço - ou seja - uma palavra - é um termo. No entanto, nas cadeias do exemplo acima não existe espaço entre os caracteres, o que torna difícil a transformação da cadeia em uma série de termos.

As cadeias podem ser quebradas de diversas formas, como por exemplo, a ilustrada na figura 4.1. Neste caso, o raciocínio empregado para a divisão é a formação dos maiores termos possíveis que possuem correspondência com termos da outra cadeia. Por exemplo, a cadeia *nomeAutor* é quebrada em dois termos ("*nome*", "*autor*"), sendo que ambos termos possuem correspondência na outra cadeia. Outra opção seria quebrar a cadeia em mais termos, como ("*no*","*me*","*au*","*tor*"), e ainda assim

[1] Não é feita distinção entre letras maiúsculas e minúsculas.

manter a correspondência com termos na outra cadeia.

Figura 4.1: Processo de divisão de cadeias em termos

Com base nestas observações, propõe-se o algoritmo *Carla* para o cálculo da similaridade entre duas cadeias. O objetivo deste algoritmo é implementar uma formação de termos que considera o maior número de sobreposições possíveis, desde que cada termo formado possua um tamanho mínimo de caracteres. O tamanho mínimo de caracteres permitido em um termo é definido através do parâmetro *tokenSize*. Uma fase posterior envolve a aplicação de métricas baseadas em termos para calcular a similaridade.

A seguir são apresentados os detalhes relacionados com a técnica proposta. Primeiramente é apresentada a métrica proposta para o problema em questão. Em seguida são levantadas questões pertinentes ao tamanho de *tokenSize* adotado neste trabalho. Por fim o algoritmo propriamente dito é explicado, e o capítulo é encerrado com considerações gerais sobre a técnica proposta.

4.1 Métrica proposta

A métrica a ser utilizada deve ser baseada em termos, uma vez que o objetivo do algoritmo é fornecer termos em comum entre duas cadeias. No entanto, as métricas existentes apresentam problemas quando aplicadas ao problema em questão, principalmente pela forma como os termos são formados. Isso pode ser comprovado aplicando as métricas existentes nos exemplos descritos na figura 4.1. Aqui será considerada apenas a métrica *Dice*, visto que as demais métricas trazem resultados semelhantes. Dado que $V_i = \{t_1, .., t_n\}$ sejam os termos de uma cadeia S_i, a métrica *Dice* segue a equação:

$$sim = \frac{2 * |V_1 \cap V_2|}{|V_1| + |V_2|}. \tag{4.1}$$

De acordo com a divisão de termos adotada na figura 4.1, para o caso $S_1 = nomeAutor$ e $S_2 = autorNome$ esta métrica retorna 1.0. Já no caso $S_1 = Machado\ de\ Assis$ e $S_2 = Assis, Machado$ o resultado é 0.66, o que é um valor muito baixo tendo em vista o número de caracteres em comum am ambas cadeias.

A solução encontrada foi adaptar as métricas baseadas em termos para o algoritmo proposto, de forma que a métrica utilize como base do cálculo caracteres individuais ao invés de termos. Na equação adaptada, o dividendo passa a ser formado pelo número de caracteres em comum entre duas cadeias e não pelo número de termos em comum. A este conjunto de caracteres em comum dá-se o nome C. Fazem parte deste conjunto todos os caracteres que pertençam a termos que sejam

comuns em ambas as cadeias. Já o divisor passa a ser composto pelo número de caracteres que compõe as cadeias. De acordo com essa nova concepção, a similaridade é relativa a quantidade de caracteres que duas cadeias têm em comum.

Adotando esta estratégia, a similaridade obtida pela métrica *Dice* adaptada continua 1.0 para o primeiro caso da figura e 0.88 para o segundo caso, o que demonstra uma melhora em comparação ao método puramente baseado no número de termos em comum.

Esta adaptação pode ser aplicada para todas as métricas baseadas em termos existentes. Neste trabalho, optou-se pela definição de uma métrica genérica, que utiliza parâmetros para variar o seu comportamento, conforme demonstrado abaixo:

$$sim = \frac{|C|}{|S_1| * \alpha + |S_2| * \beta} \tag{4.2}$$

sendo $\alpha + \beta = 1.0$.

As constantes α e β atribuem um fator de importância para cada uma das cadeias comparadas. Caso ambas cadeias recebam o mesmo grau de importância ($\alpha = \beta = 0.5$), a métrica se comporta como *Dice*. Caso uma das constantes seja nula ($\alpha = 1.0$ e $\beta = 0.0$), existem duas possibilidades. Se α estiver associado a cadeia de menor tamanho, a métrica funciona como *Overlap*, isto é, ela é adequada para identificar casos onde uma cadeia está contida dentro de outra. Caso α estiver associada a cadeia de maior tamanho, então a métrica penaliza cadeias que possuem tamanhos muito distintos.

De forma geral a contribuição deste métrica está na sua flexibilidade, já que ela pode ser configurada para atuar em diversos sentidos, dependendo do tipo de comparação que se deseja efetuar.

4.2 Tamanho dos termos

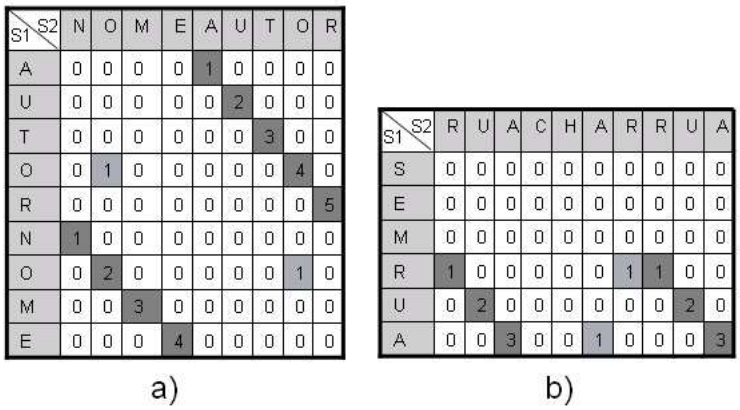
Quanto maior for o *tokenSize*, menos termos em comum são possíveis, e conseqüentemente, mais restritiva é a técnica. Ou seja, as probabilidades de que duas cadeias quaisquer sejam consideradas similares diminui. Já valores menores para termos são mais tolerantes a cadeias com grande fragmentação. No entanto, valores muito baixos podem considerar similares cadeias que representem conceitos diferentes. Por exemplo, comparando as cadeias $S_1 = caneta$ e $S_2 = nectar$ com a métrica proposta e usando o tamanho de termo igual a 2, a métrica retornaria o valor 0.66, que é considerado relativamente alto em se tratando de conceitos diferentes. Com termos de tamanho 1 a similaridade salta para 0.83. A escolha do melhor tamanho mínimo para termos depende do domínio dos dados comparados. Neste trabalho será usado um valor igual a 3, que, para os exemplos sugeridos mostrou-se bastante adequado. De fato, para o caso anterior, usando termos de tamanho 3, a similaridade cai para zero.

4.3 Algoritmo proposto

Esta seção descreve os detalhes referentes ao algoritmo proposto. O objetivo principal do algoritmo é inferir um valor que represente o número de caracteres que duas cadeias têm em comum (C). Para o casamento de duas cadeias S_1 e S_2, o algoritmo utiliza uma matriz $n\ x\ m$, sendo n igual ao número de caracteres de S_1 e

m igual ao número de caracteres de S_2.

Primeiramente, as posições da matriz $pos[x, y]$ são preenchidas com uma informação que indica se o caractere x da cadeia S_1 é o mesmo do caractere y da cadeia S_2. Caso sejam diferentes, o valor é zero ($pos[x, y] = 0$). Caso contrário, o valor equivale a soma do valor da diagonal superior esquerda mais 1 ($pos[x, y] = pos[x-1, y-1]+1$).



S1 \ S2	N	O	M	E	A	U	T	O	R
A	0	0	0	0	1	0	0	0	0
U	0	0	0	0	0	2	0	0	0
T	0	0	0	0	0	0	3	0	0
O	0	1	0	0	0	0	0	4	0
R	0	0	0	0	0	0	0	0	5
N	1	0	0	0	0	0	0	0	0
O	0	2	0	0	0	0	0	1	0
M	0	0	3	0	0	0	0	0	0
E	0	0	0	4	0	0	0	0	0

a)

S1 \ S2	R	U	A	C	H	A	R	R	U	A
S	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0	0	0	0
M	0	0	0	0	0	0	0	0	0	0
R	1	0	0	0	0	0	1	1	0	0
U	0	2	0	0	0	0	0	0	2	0
A	0	0	3	0	0	1	0	0	0	3

b)

Figura 4.2: Matrizes para os casos (nomeAutor, nomeAutor) e (sem rua, rua char-rua)

A matriz a) da figura 4.2 foi preenchida considerando $S_1 = nomeAutor$ e $S_2 = autorNome$. Na matriz foram destacadas em um tom de cinza mais escuro as células que representam os termos em comum (considera-se $tokenSize = 3$). Os termos incompletos, ou seja, que não possuem o tamanho mínimo exigido, são destacados com um tom de cinza mais claro. A partir desta matriz pode-se estabelecer o valor de C. À variável C é atribuída a soma de todos os caracteres presentes nos termos completos. Abaixo é descrito o cálculo da similaridade aplicado para o exemplo da matriz a). Nesse caso, C assume o valor 9 e a similaridade atinge o seu valor máximo (1.0).

```
sim = (4 + 5) / (9*0.5) + (9*0.5)    =
sim = (9) / (4.5 + 4.5)              =
sim = (9 / 9)                        = 1.0
```

O valor de C pode ser obtido na mesma varredura que atribui os valores dos elementos da matriz. Para tanto, toda vez que uma posição da matriz for analisada, é feito o seguinte processamento:

```
Se pos[x,y] = tokenSize
    C = C + 3;
Se pos[x,y] > tokenSize
    C = C + 1;
```

Apenas termos com no mínimo $tokenSize$ caracteres são incrementados na variável C. A grande vantagem dessa abordagem está no seu processamento, visto que apenas uma varredura é usada no cálculo. A desvantagem reside no fato de que pode haver sobreposição de termos. Ou seja, um mesmo termo de S_1 pode ser usado para identificar mais de um termo de S_2, o que pode levar a resultados indesejados, como no exemplo b) da figura 4.2. O termo "rua"de S_1 é usado duas vezes como termo

comum em S_2, o que leva a uma similaridade de 0.75 (conforme indicado abaixo), o que é considerado um valor muito elevado em se tratando de conceitos diferentes:

```
sim = (3+3) / ((6*0.5) + (10*0.5))  =
sim = (6) / (3+5)                   =
sim = (6 / 8)                       = 0.75
```

Para evitar problemas de sobreposição, é necessário assegurar que um dado termo seja usado apenas uma vez, tanto em S_1 como em S_2. Para o caso acima, é necessário selecionar um termo de S_2 e descartar o outro, o que acabaria diminuindo o valor de C, e conseqüentemente, a similaridade entre as cadeias.

Em alguns casos, podem ocorrer diversos termos sobrepostos dentro da matriz. Nesses casos, deve ser estabelecido um critério para a escolha de quais termos devem ser mantidos e quais devem ser descartados. Dependendo das escolhas, o valor de C pode ser maior ou menor.

Esta situação é ilustrada na matriz a) da figura 4.3. As duas cadeias contém o nome de duas pessoas, sendo que ambos os nomes são compostos. Nos dois casos, trata-se da mesma pessoa, no entanto, numa das cadeias, o primeiro nome aparece abreviado, e na outra cadeia, o segundo nome aparece abreviado. Além disso, a ordem dos nomes aparece invertida nas duas cadeias.

S1 \ S2	B	E	A	P	A	T	R	I	C	I	A
P	0	0	0	1	0	0	0	0	0	0	0
A	0	0	1	0	2	0	0	0	0	0	1
T	0	0	0	0	0	3	0	0	0	0	0
Y	0	0	0	0	0	0	0	0	0	0	0
B	1	0	0	0	0	0	0	0	0	0	0
E	0	2	0	0	0	0	0	0	0	0	0
A	0	0	3	0	1	0	0	0	0	0	1
T	0	0	0	0	0	2	0	0	0	0	0
R	0	0	0	0	0	0	3	0	0	0	0
I	0	0	0	0	0	0	0	4	0	1	0
Z	0	0	0	0	0	0	0	0	0	0	0

a)

S1 \ S2	B	E	A	P	A	T	R	I	C	I	A	max_x
P	0	0	0	1	0	0	0	0	0	0	0	1
A	0	0	1	0	2	0	0	0	0	0	1	1
T	0	0	0	0	0	3	0	0	0	0	0	1
Y	0	0	0	0	0	0	0	0	0	0	0	0
B	1	0	0	0	0	0	0	0	0	0	0	1
E	0	2	0	0	0	0	0	0	0	0	0	1
A	0	0	3	0	1	0	0	0	0	0	1	2
T	0	0	0	0	0	2	0	0	0	0	0	1
R	0	0	0	0	0	0	3	0	0	0	0	1
I	0	0	0	0	0	0	0	4	0	0	0	1
Z	0	0	0	0	0	0	0	0	0	0	0	0
max_y	1	1	1	1	2	2	1	1	0	0	0	0

b)

Figura 4.3: Matrizes para o caso (PatyBeatriz, BeaPatricia)

Caso o algoritmo escolha o maior termo em comum existente (*atri*), os demais termos teriam que ser descartados pela regra da sobreposição por deixarem de ter o tamanho mínimo exigido. Nesse caso, o valor de C corresponde a 4.

Entretanto, caso o maior termo seja refutado em benefício dos dois termos menores (*bea* e *pat*), o valor de C passa a ser 6, o que acarreta em um aumento na similaridade.

Como regra geral, é interessante que o valor de C reflita o maior número de caracteres em comum possível, sem que haja sobreposição de termos. O problema de maximizar o valor de C exige uma análise recursiva para cada um dos possíveis valores de termo, o que torna a implementação desta solução inviável. Neste trabalho é proposta uma heurística que calcula um valor aproximado para C, utilizando apenas duas varreduras na matriz.

Na primeira varredura as posições da matriz são preenchidas, conforme já indicado anteriormente. Além disso, na primeira varredura é realizado um processamento adicional que permite a posterior otimização no cálculo de C. Parte do processamento envolve zerar todas as posições que correspondem aos termos incompletos, como é mostrado a seguir:

```
if pos[x, y] = 0 then
  if pos[x − 1, y − 1] < tokenSize then
    steps = Pos[x − 1, y − 1]
    for i = 1 to steps do
      Pos[x-i][y-i] = 0;
    end for
  end if
end if
```

Além disso, é incluída uma linha e uma coluna no final da matriz, que são usadas para contabilizar, em cada linha ou em cada coluna, o número de posições que estão preenchidas com um valor diferente de zero:

```
if pos[x, y] = tokenSize then
  pos[max_x, y] = tokenSize
  pos[x, max_y] = tokenSize
else
  pos[max_x, y] = pos[max_x, y] + 1
  pos[x, max_y] = pos[x, max_y] + 1
end if
```

A figura 4.3 b) mostra como fica a matriz após a primeira varredura.

O próximo passo é executar uma segunda varredura na matriz de acordo com o algoritmo abaixo. O algoritmo percorre a matriz analisando as posições cujo valor é diferente de zero, o que indica que a posição pertence a um termo em comum e tenta atualizar uma variável chamada *sobreposicao*. Essa variável armazena o número de vezes em que ocorreram sobreposições na matriz. As sobreposições são identificadas por um valor superior a 1 na última linha ou na última coluna referente a posição da matriz sendo analisada. A variável sobreposição é incrementada a medida que forem identificadas sobreposições dentro de uma mesma linha (sobreposição horizontal) ou de uma mesma coluna (sobreposição vertical).

```
for x = 1 to size(S1) do
  for y = 1 to size(S2) do
    if pos[x, y] <> 0 then
      if (pos[max_x, y] > pos[x, max_y]) then
        sobreposicao = sobreposicao + pos[max_x, y] − 1
        pos[max_x, y] = 0
        for i = x to size(S1) do
          pos[i, y] = 0
        end for
      else
        sobreposicao = sobreposicao + pos[x, max_y] − 1
        pos[x, max_y] = 0
        for j = ytosize(S2) do
          pos[x, j] = 0
        end for
      end if
```

end if
end for
end for

Caso, para uma dada posição na matriz, existirem sobreposições tanto na vertical como na horizontal, será contado apenas o ramo que possuir o maior número de sobreposições. Caso uma posição da matriz já tenha participado de uma sobreposição, seu valor é zerado.

O novo valor de C passa a ser a diferença entre o valor antigo de C, que é a soma dos caracteres de todos termos em comum, e o número de sobreposições identificadas ($C = C - sobreposicao$).

Para o caso estudado, com 10 caracteres em comum e 2 sobreposições, o valor final de C sem sobreposições é 8. Na verdade, o valor ideal seria 6, como um indicador da soma dos caracteres dos termos "pat"e "bea". Todavia, levando em conta que a similaridade entre cadeias onde haja sobreposição de termos é subjetiva e depende do entendimento de cada pessoa, os resultados obtidos por essa heurística ainda podem representar uma boa aproximação da real similaridade entre as cadeias.

Ainda, em alguns casos, o erro introduzido pelo algoritmo pode tornar a similaridade ainda mais próxima do valor real. Por exemplo, no último caso estudado, com o valor de 6 para C, a similaridade resultante equivale a 0,54. Considerando C igual a 8, a similaridade passa a valer 0.72.

4.4 Considerações gerais

A técnica proposta é adequada para casos onde duas cadeias possuem muitos termos em comum, sendo que os termos podem aparecer em qualquer lugar dentro das cadeias. Apesar desta técnica apresentar bons resultados na comparação de diversas cadeias, em alguns casos os resultados não são desejáveis. É o caso onde uma inversão na ordem dos termos altera a semântica de uma cadeia. Assim, duas cadeias com significados diferentes seriam consideradas equivalentes pela técnica *Carla*. O exemplo "*Mario escreveu para Roberta*" mostra que a troca do termo *Mario* por *Roberta* altera o sentido da frase[2]. Não se pretende aqui discutir a probabilidade de ocorrência deste tipo de cadeias em domínios reais, mas sim apresentar um caso onde o uso da técnica não é aconselhável. No capítulo 5 são mostrados alguns experimentos realizados com esta técnica usando o domínio de instituições. Os resultados dos experimentos são gráficos que comparam a eficácia da técnica proposta com algumas outras técnicas existentes na literatura.

[2] O exemplo contém espaços em branco, mas o mesmo argumento é válido para cadeias sem espaços

5 CASADORES ESTRUTURAIS

Neste capítulo são discutidos os casadores estruturais desenvolvidos para o casamento de esquemas XML e esquemas relacionais. Foram criados quatro casadores estruturais, que são apresentados de acordo com sua ordem cronológica de desenvolvimento. São eles o casador de nomenclatura, o casador de relação, o casador de vizinhança e o casador de cardinalidade. Os dois últimos casadores são discutidos em uma única seção, tendo em vista que o embasamento teórico usado em ambos é o mesmo. No final do capítulo, são apresentados experimentos que mostram como os casadores de estrutura podem ser combinados de forma que a similaridade final seja um resultado mais aprimorado.

5.1 Casador de nomenclatura

Uma das formas mais tradicionais de inferir a similaridade entre dois elementos é através da comparação dos seus nomes. Na verdade, essa etapa é fundamental no processo de casamento de esquemas, e geralmente precede a execução de etapas mais elaboradas, que utilizam essa similaridade inicial para computar novos valores de similaridade.

Nesta seção, é apresentado um casador que procura estender o processo de casamento baseado em nomes através do uso de padrões de nomenclatura que podem existir entre os nós dos esquemas a serem casados.

Para definir os padrões de nomenclatura partiu-se da suposição de que os esquemas XML são criados como sub-conjuntos do esquema relacional. Ou seja, o projetista cria esquemas XML como se fossem visões de um banco relacional. Como esses esquemas XML estão relacionados com os esquemas relacionais, é normal que eles sejam escritos de forma semelhante ao esquema do banco, mantendo-se os nomes dos conceitos na medida do possível.

Dentro do contexto do casamento entre elementos atômicos XML com colunas de tabelas, é particularmente interessante estudar as relações que os nomes de elementos atômicos têm com as colunas de tabelas. Para traduzir as colunas do banco em elementos XML, foram concebidas três alternativas distintas:

Padrão EQUIV Os nomes dos elementos XML coincidem com os nomes das colunas a qual eles se referem. Essa situação de fato ocorre para o elemento **titulo** <*/autores/livro/titulo*>, descrito na figura 3.3 e a coluna **titulo**<*livro.titulo*>, descrita na figura 3.2. Ambos nomes representam o mesmo conceito e são escritos exatamente da mesma forma.

Padrão TABELA Outra alternativa de projeto considera que os nomes de elementos atômicos assumem nomes de tabelas do banco. Essa situação pode ocorrer quando se está modelando um elemento XML que represente a coluna mais representativa de uma tabela (se existir uma coluna mais representativa). Tal modelagem ocorre para o elemento **autor**<*/autores/autor*>, que equivale a coluna *nome* da tabela *autor*.

Padrão CONCAT O nome de um elemento atômico é uma concatenação do nome da coluna que ele representa com o nome da tabela desta coluna. O elemento **nomeEdit**<*/autores/livro/info/nomeEdit*> é um exemplo desse tipo de modelagem, onde o nome da coluna(*nome*) e uma parte do nome da tabela(*edit*) foram usados para formar o nome do elemento.

Convém ressaltar que podem existir mais formas de determinar como os nomes dos elementos XML são formados a partir de um esquema relacional. O estudo sobre formas alternativas de nomenclatura pode ser alvo de trabalhos futuros.

Dado um par de elementos que se deseja casar, verifica-se a ocorrência de um dos três padrões de nomenclatura expostos acima. Essa verificação envolve o uso de técnicas de casamento lingüístico que retornam um valor de similaridade para cada um dos padrões analisados. O padrão com o maior valor de similaridade é escolhido vencedor, e esse valor de similaridade é atribuído ao par de elementos casado.

A escolha de qual técnica lingüística usar no processo acima é difícil, pois em alguns casos alguma técnica pode ser mais adequada do que outra. Por exemplo, em casos onde os nomes de dois elementos diferem apenas por um erro de digitação ("cidade"e "ciade"), a técnica *Edit Distance* é bastante adequada. Para casos onde os nomes diferem pela ordem dos seus termos ("pubName"e "namePub"), a técnica *Carla* pode ser aproveitada. Enfim, a técnica de casamento escolhida para cada padrão pode depender do domínio dos dados analisados.

Assim, para tornar o algoritmo de casamento aqui descrito genérico, admite-se que várias técnicas lingüísticas possam ser aplicadas. O agrupamento dos resultados de cada técnica pode seguir duas abordagens distintas (DO; RAHM, 2002): a abordagem otimista e a pessimista:

Na abordagem otimista, o maior valor de similaridade entre as técnicas é escolhido, e os demais são descartados. Esta abordagem é considerada otimista pois ela procura maximizar o valor de similaridade. Já na abordagem pessimista o menor valor de similaridade entre as técnicas é escolhido.

A principal diferença entre as abordagens otimista e pessimista é que com o uso da primeira alternativa os casamentos possuem graus de similaridade sempre maiores, ou pelo menos iguais, aos casamentos obtidos com o uso da segunda alternativa. Uma conseqüência disso é que a primeira abordagem tende a apresentar similaridades altas para casamentos incorretos. Por outro lado, a abordagem otimista pode descartar casamentos corretos que não atingiram um valor alto de similaridade.

Para o trabalho proposto, considera-se a abordagem otimista. A principal justificativa para tal escolha está no fato de que os casamentos incorretos que porventura sejam bem classificados pelo casador podem ser descartados com a aplicação de casadores adicionais.

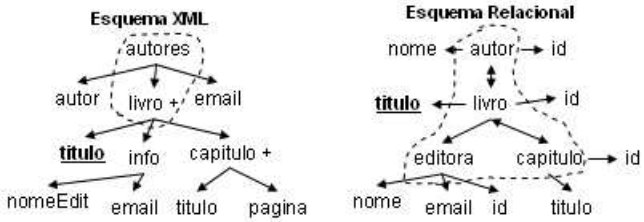


Figura 5.1: Atribuição dos conjuntos de nós relacionados

5.2 Casador de relação

Este foi o segundo casador estrutural desenvolvido neste trabalho. Ele faz parte de um trabalho inicial na exploração das estruturas dos esquemas para a inferência de similaridades nos nós atômicos. Este casador utiliza a noção de nós relacionados: todo nó atômico n_i possui um conjunto não vazio de nós relacionados. Os nós relacionados a n_i são todos aqueles que são direta ou indiretamente conectados a ele.

Este casador parte da suposição de que a similaridade entre um par(n_i, n_j) é proporcional à similaridade entre os nós relacionados de n_i e os nós relacionados de n_j.

O exemplo descrito abaixo é usado como ponto de partida para analisar o comportamento deste casador.

Exemplo 1 *Dada a figura 5.1, obter um valor de similaridade para os nós* </autores/livro/titulo> *e* <livro.titulo>.

Nos dois esquemas da figura estão sublinhados os nós que se deseja casar. Os conjuntos de nós relacionados de cada nó sublinhado estão demarcados por uma linha tracejada. Note que no caso do esquema XML, os nós relacionados equivalem a todos os ascendentes de um nó atômico. Já no esquema relacional, os nós relacionados equivalem a um conjunto de tabelas.

Dados os dois conjuntos de nós relacionados, deve-se casar todos os membros de um conjunto com os membros do outro conjunto. O cálculo da similaridade é efetuado usando a técnica lingüística *Carla*, desenvolvida nesta dissertação.

Observando os nós relacionados, percebe-se que existem similaridades entre os nós *(</autores>, <autor>)* e *(</autores/livro>, <livro>)*. As similaridades entre os demais nós relacionados é baixa demais para ser levada em consideração. Disto surge a necessidade de um limiar (*threshold*) para desconsiderar similaridades muito baixas.

Dados os pares destacados acima, o cálculo da similaridade final é obtido através da equação abaixo:

$$simFinal = \frac{\sum_1^n sim_i}{n} \tag{5.1}$$

O valor sim_i é uma das similaridades entre nós relacionados que permaneceram acima do limiar estabelecido. No caso da figura 5.1, aplicando-se a técnica *Carla* e um limiar igual a 0.7, tem-se:

```
similaridade entre autor e autores(sim1) = 0.83
similaridade entre livro e livro(sim2) = 1.0
```

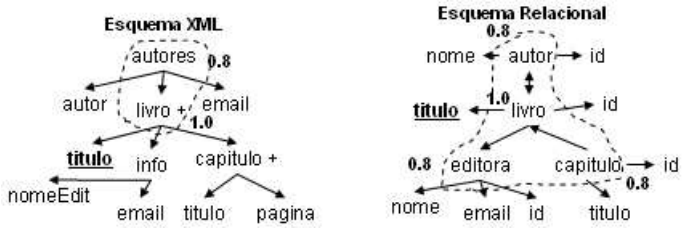


Figura 5.2: Atribuição de fatores de relevância

```
similaridade final = (sim1 + sim2) / 2  = (1.0 + 0.83) / 2 = 0.915
```

Perceba que esta técnica não utiliza as informações dos nós atômicos propriamente ditos, e ainda assim alcança um valor alto de similaridade para casamentos corretos.

Apesar do resultado satisfatório, a técnica pode retornar valores altos de similaridade para casamentos incorretos. Essa situação ocorre no exemplo descrito abaixo:

Exemplo 2 *Considerando os mesmos esquemas da figura 5.1, obter um valor de similaridade para os nós* </autores/livro/titulo> *e* <capitulo.titulo>.

Os dois conjuntos de nós relacionados não sofrem nenhuma alteração com relação aos conjuntos usados no exemplo 1. Dessa forma, o valor de similaridade do par considerado no exemplo 2 *(</autores/livro/titulo>, <capitulo.titulo)>)* assume o mesmo valor de similaridade do par considerado no exemplo 1 *(</autores/livro/titulo>, <livro.titulo)>)*, mesmo sendo o par do exemplo 2 um casamento incorreto.

A forma encontrada de combater esse problema é a atribuição de fatores de relevância (fr) para cada um dos nós relacionados a um nó atômico. Quanto maior for o fator de relevância, mais o nó contribui para o cálculo da similaridade final.

O fator de relevância de um nó n_j para um nó atômico n_i é inversamente proporcional a distância[1] que separa os nós. O calculo de fr é demonstrado na equação abaixo:

$$fr(n_i \rightarrow n_j) = 1 - ((arcos(n_i \rightarrow n_j) - 1) * frd) \tag{5.2}$$

A notação $n_i \rightarrow n_j$ indica que os nós n_i e n_j contem arco(s) entre eles. A função $arcos(n_i \rightarrow n_j)$ retorna o número de arcos entre n_i e n_j. O valor de frd é uma constante usada para que o fator de relevância diminua a uma taxa fixa, a medida que n_j se afasta de n_i.

Utilizando os fatores de relevância para o casamento entre *</autores/livro/titulo>* e *<livro.titulo>*, tem-se a figura 5.2. Nota-se que o maior fator de relevância possível (1.0) pertence ao nó relacionado mais próximo do nó atômico. Os demais valores de similaridade diminuem gradativamente. Fatores de similaridade abaixo de 0 caracterizam nós que estão muito distantes para fazerem parte do cálculo da similaridade.

Com a introdução dos fatores de relevância, o cálculo da similaridade dos nós relacionados utiliza a equação abaixo:

[1] número de arcos que separa os nós.

$$sim_{(i,j)} = \frac{ling(n_i, n_j) + rel(n_i, n_j)}{2} \tag{5.3}$$

A função $ling(n_i, n_j)$ retorna um valor de similaridade entre n_i e n_j com base na técnica lingüística *Carla*. A função $rel(n_i, n_j)$ retorna a média aritmética dos fatores de relevância dos nós n_i e n_j. Com a aplicação desta equação para o cálculo da similaridade entre *</autores/livro/titulo>* e *<livro.titulo>*, tem-se:

```
similaridade entre $/autores$ e $autor$
    ling = 0.83
    rel = (0.8 + 0.8) / 2 = 0.8
    sim1 = (ling + rel) / 2 = (0.83 + 0.8) / 2 = 0.815

similaridade entre $/autores/livro$ e $livro$
    ling = 1.0
    rel = (1.0 + 1.0) / 2 = 1.0
    sim2 = (ling + rel) / 2 = (1.0 + 1.0) / 2 = 1.0

similaridade final = (sim1 + sim2) / 2 =  (0.815 + 1.0) / 2 = 0.9075
```

A similaridade resultante deste cálculo é um pouco menor do que a similaridade computada quando não se utiliza fatores de relevância. Entretanto, o uso dos fatores compensa porque os casamentos incorretos são mais penalizados do que os casamentos corretos. Isso pode ser observado utilizando fatores de relevância para resolver o problema do exemplo 2, conforme indicado abaixo:

```
similaridade entre $/autores$ e $autor$
    ling = 0.83
    rel = (0.6 + 0.8) / 2 = 0.7
    sim1 = (ling + rel) / 2 = (0.83 + 0.7) / 2 = 0.765

similaridade entre $/autores/livro$ e $livro$
    ling = 1.0
    rel = (0.8 + 1.0) / 2 = 0.9
    sim2 = (ling + rel) / 2 = (1.0 + 0.9) / 2 = 0.95

similaridade final = (sim1 + sim2) / 2 =  (0.765 + 0.95) / 2 = 0.8575
```

Mesmo que a similaridade entre *</autores/livro/titulo>* e *<capitulo.titulo>* continue sendo alta, ainda assim ela é menor do que a similaridade entre *</autores/livro/titulo>* e *<livro.titulo>*, de forma a possibilitar que o casamento correto seja escolhido.

Como este casador não analisa as propriedades dos nós atômicos propriamente ditos, em alguns casos podem ocorrer incongruências. Por exemplo, da aplicação desta técnica resulta que a similaridade entre o par *(/autores/livro/capitulo/titulo, capitulo.titulo)* e *(/autores/livro/capitulo/pagina, capitulo.titulo)* é idêntica. Porém, esses efeitos são previsíveis e até mesmo esperados. Recomenda-se que este casador não seja usado de forma isolada, mas em combinação com outro(s) casador(s), principalmente casador(s) que analise(m) as propriedades dos nós atômicos.

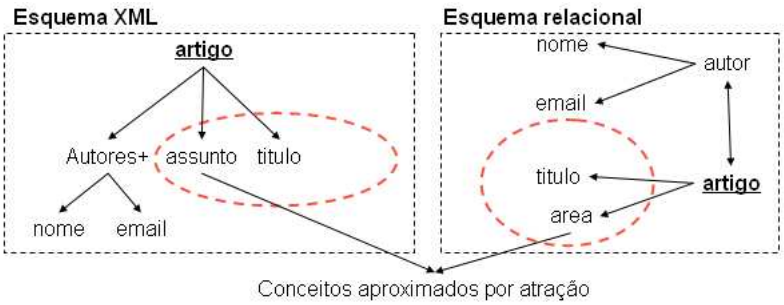


Figura 5.3: Campos de atração entre dois conceitos

5.3 Casadores de atração

Nesta seção são descritos os dois últimos algoritmos de casamento estruturais propostos nesta dissertação. Eles são chamados casadores de atração porque foram concebidos tendo em mente o conceito de atração entre pares de casamentos.

O conceito de atração considera que para todo par de nós casados, sendo cada um deles provenientes de um esquema, são gerados dois campos de atração, um em cada esquema. Todos os nós contidos em cada campo são chamados de nós internos, e denomina-se ao conjunto desses nós por σ. Os nós que geraram os campos de atração são chamados de nós origem, e são referenciados por η.

Os campos de atração atuam no sentido de atribuir similaridades entre os nós internos de um campo e os nós internos do outro campo. Em outras palavras, parte-se da suposição de que os nós internos possuem um grau de similaridade entre si semelhante à similaridade dos nós que geraram os campos.

Cada campo de atração possui uma intensidade, que é equivalente às similaridades dos nós origem. Ou seja, quanto maior for a similaridade entre os nós origem, maior é a intensidade do campo. Considera-se que essa similaridade já tenha sido computada por um outro casador. A cobertura do campo, isto é, os nós que fazem parte de σ, depende de heurísticas próprias de cada casador de atração.

A figura 5.3 ilustra um situação onde foi gerado um campo de atração entre os nós η_1 *</artigo>* e η_2 *<artigo>*. Cada campo possui dois nós internos. Partindo da suposição de que nós internos são similares, e considerando que já exista um valor alto de similaridade entre os nós *</artigo/titulo>* e *<artigo.titulo>*, deduz-se que os nós $n_1 \in \sigma_1$ *</artigo/assunto>* e $n_2 \in \sigma 2$ *<artigo.area>* são similares, ou pelo menos mais similares do que qualquer outra combinação que envolva algum destes nós.

Na figura 5.3 foi mostrada a atuação de apenas um par de campos de atração. Todavia, podem existir situações em que mais de um par de campos de atração exerça influência sobre os nós dos esquemas. Vale ressaltar que cada campo de atração de um esquema tem exatamente um campo correspondente no outro esquema, e a suposição de que os nós internos são similares serve apenas para nós pertencentes a dois campos correspondentes.

O cálculo da similaridade dos nós internos ocorre da seguinte forma: Toda vez que um determinado par de nós internos $n_1 \in \sigma_1$ e $n_2 \in \sigma_2$ estiver sob a influência de um par de campos de atração, é computada uma similaridade individual baseada na intensidade daquele par de campos específico. A similaridade final entre n_1 e n_2 é a média aritmética das similaridades individuais computadas em todos os casos onde

tanto n_1 e n_2 estiverem sob a influência de campos de atração correspondentes. Para evitar que a similaridade entre nós seja degradada por pares de campo de atração de intensidade muito baixa, pode-se aplicar um limiar. Campos de atração com intensidade inferior ao limiar não são considerados.

No exemplo da figura, os campos de atração gerados tem função meramente ilustrativa. Nas seções seguintes são descritas as heurísticas que foram utilizadas na geração dos campos de atração. O objetivo de cada heurística é, dados dois nós origem η_1 e η_2, obter os campos de atração, ou seja, os conjuntos σ_1 e σ_2. Independentemente da heurística utilizada, a similaridade entre os nós internos equivale a similaridade dos nós que geraram os campos de atração.

Neste trabalho, considera-se que apenas os nós atômicos podem se tornar nós origem. Essa restrição deve-se ao fato de que os campos só podem ser gerados através de similaridades pré-computadas, e sabe-se que os casadores executados anteriormente computaram similaridades somente entre os nós atômicos. Em versões futuras esta restrição pode ser relaxada, de modo a aceitar qualquer nó como nó origem. Outrossim, considera-se que os conjuntos σ também são compostos apenas por nós atômicos. Esta restrição faz sentido já que neste trabalho o interesse é em casar apenas os nós atômicos, de modo que não há necessidade de que nós não-atômicos façam parte de um conjunto σ.

5.3.1 Casador de vizinhança

Este casador atribui valores de similaridade considerando a noção de vizinhança entre os nós. Dado um nó origem η, o conjunto de nós internos σ é dado por todos os nós atômicos que são vizinhos de η. Para fins de compreensão, pode-se considerar que dois nós são vizinhos se eles pertencem a uma mesma tupla Γ, onde Γ é uma tupla na primeira forma normal.

Para um esquema XML, uma tupla Γ corresponde a um elemento e todos os seus descendentes que possuírem cardinalidades simples. Isto pode ser formalmente definido como:

Definição 9 *(Nós vizinhos DTD) Dois nós n_i e n_j são vizinhos se eles possuírem um ancestral comum α de forma que o caminho dos nós n_i e n_j até o ancestral α seja composto apenas por cardinalidade simples.*

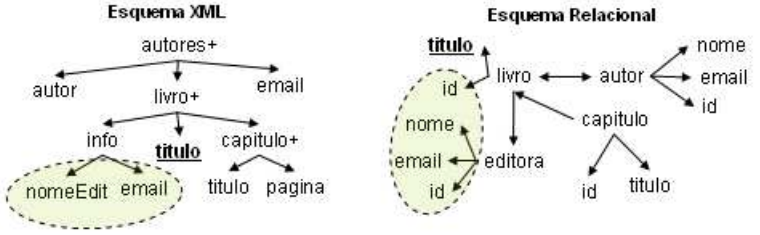


Figura 5.4: Exemplo de campos de atração sob a ótica dos nós vizinhos

No esquema XML da figura 5.4, o nó sublinhado indica o nó origem η_1 *</autores/livro/titulo>*. Neste caso, o nó α é *</autores/livro>*. Observa-se que, ao subir na árvore a partir de qualquer um dos nós internos até o nó α, somente cardinalidades simples são encontradas. Vale ressaltar que somente nós atômicos fazem parte

de σ, de acordo com a restrição explicada anteriormente neste capítulo. Uma forma de computar o conjunto σ a partir de um nó η é dada através dos passos abaixo:

1. Procurar o nó α, que é o primeiro ascendente de η que tem cardinalidade múltipla.

2. Inserir em σ todos os descendentes de α que sejam nós atômicos e que cujo caminho até α seja composto apenas por cardinalidades simples.

Já em esquemas relacionais, uma tupla Γ é definida em função de uma tabela τ, que contém o atributo que corresponde ao nó origem η. Dada a tabela τ, a tupla Γ corresponde a própria tabela τ, e um conjunto de tabelas ϕ. Para determinar o conjunto ϕ usa-se a definição abaixo:

Definição 10 *(Conjunto ϕ) Considere que $\{T_1, .., T_n\}$ seja uma lista de tabelas de um esquema relacional, e A_i seja a coleção de atributos de T_i. Também considere que $DF : T_1 \rightarrow T_2$ indique que a tabela T_1 possui uma dependência funcional com T_2. Assim, dada a tabela τ, tem-se $\forall\ T_i \mid DF : \tau \rightarrow T_i, T_i \in \phi$.*

Em outras palavras, todas tabelas que possuem dependência funcional[2] com a tabela τ fazem parte de ϕ. Por fim, o conjunto σ é formado pelos atributos de τ juntamente com os atributos das tabelas de ϕ.

No esquema relacional da figura 5.4 tem-se *<livro.titulo>* como nó origem η. Disto segue que a tabela τ é *livro*. O conjunto ϕ é dado por *<editora>*, já que apenas essa tabela tem dependência funcional com livro. O conjunto σ é dado por {*<livro.id>*, *<editora.id>*, *<editora.nome>*, *<editora.email>*}, o que termina por originar o campo de atração demonstrado na figura.

Depois de encontrados os campos de atração, o cálculo da similaridade dos nós internos ocorre como demonstrado no início do capítulo, ou seja, a similaridade dos nós internos equivale a similaridade dos nós que originaram os campos de atração.

5.3.2 Casador de cardinalidade

Este casador atribui valores de similaridade considerando a noção de dependência multivalorada entre os nós. Dado um nó origem η, o conjunto de nós internos σ é dado por todos os nós atômicos que têm dependência multivalorada com η. A notação aqui usada para indicar que um nó n_i tem uma dependência multivalorada com n_j é dada por $DMV : n_i \twoheadrightarrow n_j$. Em outras palavras, podem existir várias instâncias de n_j relacionadas a uma única instância de n_i. Em modelos relacionais essa notação é usada para identificar dependências multivaloradas entre atributos de uma tabela. Neste trabalho estende-se o conceito de DMV, de forma que ele possa ser aplicado ao problema em questão.

Dentro de um esquema relacional, as DMV podem ser diretamente obtidas através da análise das relações de chave estrangeira entre as tabelas. Essas dependências estão indicadas através de arcos conectando duas tabelas. O grafo pode ser interpretado da seguinte forma: Sempre que existir um arco entre dois nós tabela n_i e n_j, e esse arco for direcionado para n_j, então existe uma dependência multivalorada de n_i para n_j ($DMV : n_i \twoheadrightarrow n_j$), ou seja, uma única tupla da tabela

[2]Neste contexto, as dependências funcionais assemelham-se às dependências funcionais do modelo relacional, com a diferença de que são consideradas dependências entre tabelas ao invés de atributos.

correspondente a n_i pode estar relacionada com várias tuplas da tabela correspondente a n_j.

Dado um nó origem η em um esquema relacional, a geração do conjunto σ pode ser obtida executando-se os seguintes passos:

1. Inserir no conjunto α todos nós tabela que possuem DMV com a tabela que contém o atributo correspondente a η.

2. Inserir no conjunto σ todos os atributos das tabelas pertencentes a α.

Para o caso da figura 5.5, dado o nó origem *<livro.titulo>*, o conjunto α equivale a *{<autor>, <capitulo>}*. O conjunto σ, e conseqüentemente o campo de atração, equivale aos atributos das tabelas *autor* e *capitulo*.

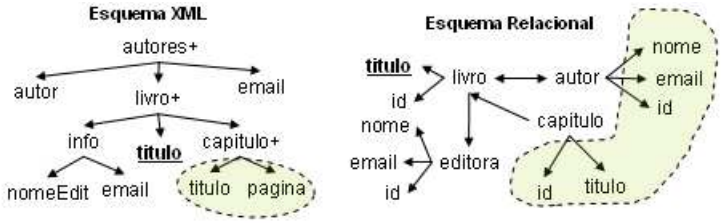


Figura 5.5: Exemplo de campos de atração sob a ótica dos nós multi-valorados

A geração do conjunto σ no esquema XML segue uma abordagem diferente. Isso ocorre porque os arcos no esquema relacional tem uma conotação diferente dos arcos no esquema XML. Neste último caso, as DMV são obtidas através de uma heurística que analisa as cardinalidades dos nós. A obtenção do conjunto σ é realizada através da execução das seguintes etapas:

1. Obter o primeiro nó α que seja ascendente de η e que tenha cardinalidade múltipla.

2. Percorrer todos os ramos filhos de α, inserindo no conjunto χ o primeiro nó de cada ramo que possua cardinalidade múltipla.

3. Percorrer todos os nós pertencentes a χ, inserindo no conjunto σ todos os nós atômicos alcançados no caminho.

No esquema XML da figura 5.5 está demonstrado o campo de atração gerado para o nó *</autores/livro/titulo>*. Obedecendo as etapas descritas acima, primeiramente obtém-se o nó α, que equivale a *</autores/livro>*. Na segunda etapa, obtém-se o conjunto χ, que no caso é constituído por *{</autores/livro/capitulo>}*. Na última etapa são obtidos os nós internos, que equivalem aos nós atômicos descendentes de *</autores/livro/capitulo>*, ou seja *{</autores/livro/capitulo/titulo>, </autores/livro/capitulo/pagina>}*. Assim como no casador de vizinhança, a similaridade dos nós internos equivale a similaridade dos nós que geraram os campos de atração.

Com base nesta heurística, pode-se concluir que no esquema XML da figura 5.5 existe uma DMV entre *autor* e *livro*, ou seja, um autor pode ter escrito vários livros. A situação inversa, de que um livro pode ser escrito por diversos autores, não

pode ser deduzida pelo esquema, apesar de ser uma DMV que um especialista consideraria coerente. Esta limitação, decorrente da impossibilidade de explicitamente declarar dependências funcionais no esquema XML, pode eventualmente acarretar em perda da semântica na análise das dependências multivaloradas. Com essa perda alguns casamentos podem não ser capturados pelo algoritmo. Todavia, como existem muitos casos em que o algoritmo é válido e considerando que os casos não contemplados podem ser tratados por outros casadores, o uso desta técnica contribui para que o resultado final do casamento de esquemas seja aprimorado.

5.4 Considerações Gerais

Neste capítulo foram apresentadas quatro técnicas estruturais de casamento de esquema. Estas técnicas podem ser divididas em dois grupos. O primeiro grupo é composto pelo casador de nomenclatura e o casador de relação (chamados aqui de casadores de base), enquanto o segundo grupo é composto pelo casador de vizinhança e o casador de cardinalidade (chamados aqui de casadores de atração). As técnicas do primeiro grupo diferenciam-se das do segundo grupo por utilizarem os nomes dos elementos como parte do cálculo da similaridade. Já as técnicas do segundo grupo utilizam apenas informações estruturais, como a disposição dos elementos dentro dos esquemas. Outra diferença é o fato dos casadores de atração precisarem ser alimentados por casamentos pré-computados para poder realizar seu processamento. De modo geral, os casadores de base são adequados para computar similaridades entre esquemas que utilizem uma nomenclatura similar. Já os casadores de atração são adequados para refinar os casamentos computados pelos casadores de base. Ou seja, as informações de vizinhança e cardinalidade podem ser usadas para eliminar ambigüidades de casamento que não puderam ser resolvidas pelos casadores de base. No capítulo 5 são descritos experimentos que foram realizados com as técnicas propostas, onde um dos temas discutidos é a forma como os casadores são combinados para gerar o resultado final de casamento.

6 EXPERIMENTOS REALIZADOS

Neste capítulo são apresentados os experimentos realizados com os algoritmos propostos neste trabalho. O texto está organizado em duas seções. Na primeira seção, são apresentados os experimentos envolvendo o casador lingüístico *Carla*. Na segunda seção são apresentados os experimentos envolvendo os casadores estruturais propostos.

Em ambas seções são usadas duas formas de medir a qualidade dos casamentos: Precisão (*Precision*) e Revocação (*Recall*). Essas métricas, originárias da área de recuperação de informação, são comumente usadas para validar técnicas de casamento (DO; MELNIK; RAHM, 2002). O propósito destas técnicas pode ser explicado com base na figura 6.1. A figura mostra o conjunto de casamentos reais e o conjunto de casamentos inferidos pelos algoritmos. A intersecção dos dois conjuntos mostra os casamentos corretos que foram identificados pelos algoritmos (verdadeiro-positivos). A revocação, computada como $\frac{|B|}{|A|+|B|}$ indica a porcentagem do número total de casamentos corretos que os algoritmos de casamento conseguiram deduzir. Já a precisão, computada como $\frac{|B|}{|B|+|C|}$, indica a porcentagem dos casamentos retornados pelos algoritmos que realmente são casamentos corretos. Quanto mais alto forem os valores de revocação e precisão, maior é a eficácia da técnica avaliada.

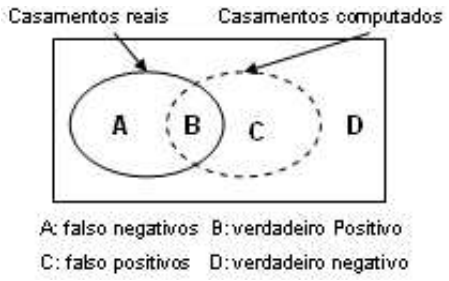


Figura 6.1: Conjunto de casamentos corretos e computados automaticamente

6.1 Experimentos com a técnica Carla

Os objetivos dos experimentos com a técnica Carla são: i) verificar sua eficácia e ii) verificar sua performance de execução. Para melhor avaliar a técnica proposta, os seus resultados são comparados com algumas técnicas existentes na literatura: *Jaro*, *Jaccard* e *Levenstein*.

Os experimentos consistem em consultas a uma fonte de dados XML composta por 2495 instâncias relativas a nomes de instituições. As instituições são descritas de

diversas formas, incluindo sua denominação por extenso (ex. Universidade Federal do Rio Grande do Sul) e o seu formato abreviado (UFRGS). Dada uma consulta por nome da instituição, a técnica avaliada retorna uma lista com as instituições ordenadas de acordo com sua similaridade com o objeto consulta. Com base nesta lista são construídos gráficos de precisão e revocação.

Ao total foram executadas 10 consultas, e a média dos resultados é mostrada no gráfico a) da figura 6.2. Nota-se que, de modo geral, as técnicas obtiveram resultados parecidos. A pior avaliação coube a *Jaro*, enquanto a técnica *Jaccard* teve uma eficácia levemente superior a *Levenstein* e *Carla*.

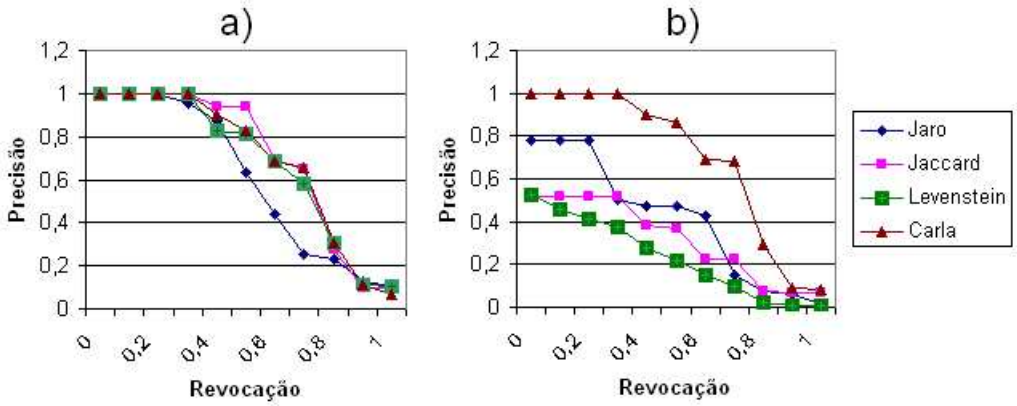


Figura 6.2: Gráficos de precisão e revocação dos casadores lingüísticos

Também foi realizada uma outra série de experimentos sobre a mesma base, porém com objetos de consulta alterados. As alterações incluem inversão de termos das cadeias e concatenação de termos. Por exemplo, a primeira série de experimentos possuía uma consulta por *"Universidade Federal de Santa Catarina"*. Já na segunda série de experimentos, a consulta foi alterada para *"Federal Universidade de SantaCatarina"*. A média de precisão e revocação das 10 consultas alteradas está representada no gráfico b) da figura 6.2. Desta vez a técnica *Carla* se sobressaiu em relação as demais. Na verdade, a precisão e revocação desta técnica não sofreu nenhuma mudança com o uso das consultas alteradas, o que mostra sua eficácia com dados não-comportados, caracterizados por inversão de termos e falta de espaços em brancos entre um termo e outro.

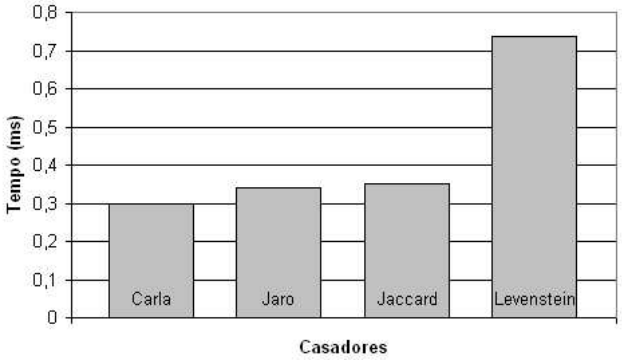


Figura 6.3: Tempo de execução dos casadores lingüísticos

O outro objetivo desta seção é verificar a performance de execução da técnica

proposta. A performance é medida em comparação com as outras 3 técnicas avaliadas. O gráfico da figura 6.3 mostra o tempo médio gasto no processamento de uma consulta sobre a fonte XML de instituições. O tempo está representado em milisegundos. Conforme o gráfico, as técnicas *Carla*, *Jaro* e *Jaccard* foram as mais rápidas para as consultas em questão. Já o desempenho da *Levenstein* foi bastante inferior às demais.

6.2 Experimentos com os casadores estruturais

O objetivo dos experimentos com os casadores estruturais é descobrir se os casamentos retornados são corretos, ou seja, se os casamentos retornados são realmente aqueles que um usuário especialista escolheria com base na sua própria experiência. Os resultados dos experimentos são traduzidos por gráficos de revocação e precisão. Valores de similaridade são abstraídos dos gráficos, já que a preocupação principal consiste em verificar quantos casamentos corretos estão dentro do grupo dos verdadeiro-positivos, e não em verificar os valores de similaridade propriamente ditos. Para os casos de baixa eficácia dos algoritmos, é feita uma análise dos casos não resolvidos de modo a compreender as limitações dos algoritmos propostos.

Para cada experimento são definidos os dois esquemas que se deseja casar e um conjunto de algoritmos de casamento que será aplicado. Os resultados de cada experimento são apresentados na forma de uma tabela que contém os pares de elementos casados e um valor de similaridade para o par. Cada experimento compreende a execução de uma série de etapas, que obedecem a seguinte ordem: Execução dos casadores, combinação das similaridades e filtro dos resultados. A seguir cada uma destas etapas será descrita. Também serão apresentados os domínios usados para os experimentos.

6.2.1 Execução dos casadores

Para cada experimento os quatro casadores propostos são executados, sendo que a sua execução obedece uma determinada ordem. A seqüência de execução é mostrada na figura abaixo. A seqüência não precisa ser necessariamente esta, desde que os dois primeiros algoritmos sejam executados antes dos dois últimos. Essa restrição deve ser obedecida porque os dois últimos algoritmos utilizam similaridades pré-computadas para inferir novos valores de similaridades. Para diferenciar os últimos casadores dos primeiros no tocante ao uso de similaridades pré-computadas, os dois casadores iniciais são referenciados por casadores de base, enquanto os últimos casadores são referenciados por casadores de atração.

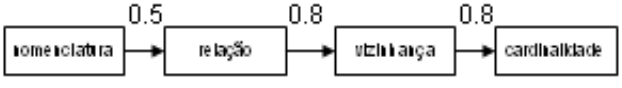


Figura 6.4: Seqüência de execução dos casadores

6.2.2 Combinação das similaridades

Outra questão importante é a combinação das similaridades de cada um dos casadores. Os valores de similaridade computados por um casador devem ser combinados com os valores de similaridade computados pelo casador anterior (se houver).

A combinação é dada pela média ponderada das similaridades computadas por dois algoritmos adjacentes. A figura 6.4 mostra como os pesos foram usados nos experimentos. Cada valor apresentado entre dois casadores indica o peso α usado para o casador da esquerda. O peso β do casador da direita é o complemento de 1 de α ($\alpha + \beta = 1.0$). Observa-se que os pesos dos dois primeiros algoritmos são maiores do que os pesos dos dois últimos. Essa configuração de pesos foi adotada porque os casadores de atração são usados apenas para complementar similaridades já computadas. Ou seja, sua principal função é valorizar mais os casamentos corretos que já possuíam valores altos de similaridade em detrimento dos casamentos incorretos. A descoberta de novos casamentos fica a cabo dos casadores de base. Vale lembrar que ambos casadores de base utilizam internamente a técnica *Carla* para verificar a similaridade lingüística entre os conceitos. A técnica foi configurada para usar um tamanho mínimo de termo igual a 3, e para valorizar as cadeias mais longas.

6.2.3 Filtro dos resultados

Somente são apresentados para o usuário os casamentos cuja similaridade tenha passado por uma etapa de filtro. Três formas de filtro são usadas:

Filtro de tipagem dos dados Considera a eliminação de casamentos cujos membros do par tenham tipos de dados distintos. Os tipos de dados de um esquema relacional podem ser automaticamente obtidos através da análise dos metadados do esquema. Já esquemas DTDs não definem tipos de dados, de modo que o trabalho de atribuição dos tipos de dados dos elementos teve que ser feita manualmente. Uma forma de obterem-se automaticamente os tipos de dados para elementos de uma DTD envolve análise das instâncias de documentos XML. Trabalhos futuros podem implementar essa solução para agilizar o processo de casamento. Outrossim, o uso alternativo de XML Schema resolve a questão, já que este tipo de esquema permite a definição dos tipos de dados dos elementos.

Filtro do melhor casamento Esta forma de filtro adotada já foi introduzida no capítulo 3. Basicamente trata-se de um processo que elimina casamentos caso um dos membros do par já esteja presente em outro casamento com uma similaridade mais elevada.

Filtro de limiar Elimina casamentos cujo valor de similaridade seja inferior a um limiar pré-estabelecido. Os experimentos usam um valor de limiar igual a 0.7. Este valor de limiar tende a maximizar os valores de precisão e revocação, pelo menos nos experimentos realizados. Não se pretende aqui estabelecer critérios e métodos para a melhor escolha do valor de limiar. Esta área por si só merece um estudo aprofundado a parte.

6.2.4 Domínios usados

Foram conduzidos experimentos com três domínios diferentes: Domínio de livros, locadora de imóveis e locadora de filmes. Para cada um dos domínios foram utilizados diversos esquemas. Cada esquema S_i é composto por um número de conceitos $|S_i|$. Neste contexto, o nome conceito é aplicado aos nós atômicos, ou seja, os nós que podem ser casados. Em esquemas XML os conceitos são os atributos e os elementos `#PCDATA`. Em esquemas relacionais, conceitos são atributos de tabela que

Tabela 6.1: Experimentos no domínio de livros

Experimento	DTD	RDB	num. corretos	num. possíveis
e1	d1 (16 conc.)	r1 (9 conc.)	3	144
e2	d2 (4 conc.)	r1 (9 conc.)	3	36
e3	d3 (8 conc.)	r1 (9 conc.)	5	72
e4	d4 (4 conc.)	r1 (9 conc.)	3	36
e5	d5 (4 conc.)	r1 (9 conc.)	3	36
e6	d6 (6 conc.)	r1 (9 conc.)	3	54

não sejam chave estrangeira. Dados dois esquemas S_1 e S_2, o número máximo de possibilidades de casamentos é dado por $|S_1| \times |S_2|$.

Observe que, como os esquemas foram construídos de forma independente, mesmo sendo do mesmo domínio, não há garantia de que eles modelem as mesmas informações. Em nenhum dos casos um esquema está completamente contido dentro de outro, ou seja, em nenhum dos casos todos os conceitos de um esquema tem correspondência com conceitos do outro esquema. Nas próximas seções são descritos os experimentos realizados com cada um dos domínios citados anteriormente.

6.3 Livros

Os experimentos neste domínio envolvem um esquema relacional (r1) composto por 6 tabelas e 9 conceitos. Este esquema já foi construído como parte deste trabalho especialmente para a validação dos algoritmos de casamento. Contra este esquema são casadas uma série de 6 DTDs (d1,d2,d3,d4,d5,d6), todas extraídas da *Web*, totalizando o número de 6 experimentos(e1,e2,e3,e4,e5,e6). Todas as DTDs (disponíveis no anexo I) foram extraídas de material instrucional que serve como exemplo de construção de DTDs. Em suma, tais DTDs possuem poucos elementos e são constituídas de estruturas simples. A tabela 6.2 mostra os dados relacionados aos experimentos realizados. De acordo com a tabela, na maioria dos experimentos existe uma grande sobreposição dos conceitos usados em ambos os esquemas. A maior exceção está em e1, onde apenas 3 dos 16 conceitos de d1 possui correspondência no esquema r1.

O gráfico da figura 6.5 mostra os valores de revocação e precisão obtidos para os 6 experimentos realizados. Os baixos valores de precisão, principalmente em e1, são resultados da baixa sobreposição dos conceitos dos dois esquemas. Isto é, apesar de serem esquemas do mesmo domínio, as informações modeladas são diferentes. No experimento em questão, existem apenas 3 casamentos corretos, em meio a 144 possibilidades de casamento. Ainda, casamentos errados obtém similaridade elevada devido a similaridade lingüística dos conceitos. Por exemplo, em e5, o casador de nomenclatura infere uma similaridade relativamente elevada entre *</catalogo/livro/titulo>* e *<capitulo.nome>*, uma vez que o elemento atômico *titulo* possui semelhança com a tabela *capitulo* (ambos compartilham a sub-cadeia "tulo").

Já no gráfico de revocação na maioria dos casos obteve-se o valor de revocação igual a 0.66. Esses casos equivalem a situações onde existiam 3 casamentos corretos

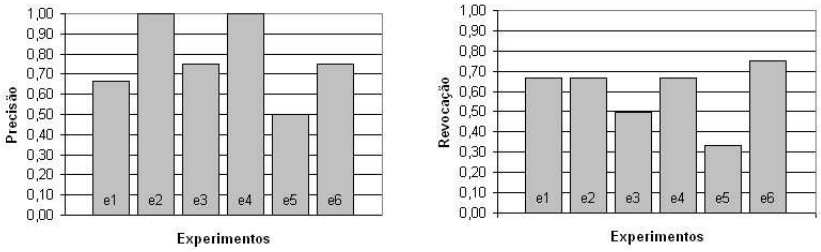


Figura 6.5: Gráficos de revocação e precisão para o domínio de livros

Tabela 6.2: Experimentos no domínio de locadora de filmes

Experimento	DTD	RDB	num. corretos	num. possíveis
e1	d1 (4 conc.)	r1 (23 conc.)	3	92
e2	d2 (10 conc.)	r1 (23 conc.)	5	230
e3	d3 (24 conc.)	r1 (23 conc.)	7	552
e4	d1 (4 conc.)	r2 (23 conc.)	3	92
e5	d2 (10 conc.)	r2 (23 conc.)	5	230
e6	d3 (24 conc.)	r2 (23 conc.)	7	552

e 2 foram encontrados pelos casadores. O casamento não encontrado corresponde ao caso envolvendo o atributo *livro.nome*. As DTDs modelam essa informação em um elemento chamado *titulo*, o que inviabiliza que uma similaridade lingüística seja encontrada.

Os casadores baseados em campos de atração não foram eficazes neste experimento, ou seja, as similaridades obtidas com ou sem o uso destes algoritmos foi a mesma. Isso deve-se ao fato de que os casadores de maior peso (casadores de base) não terem atribuído um valor alto de similaridade para os casamentos não-identificados. Com isso, os casadores de atração falharam em identificar os casamentos corretos que faltaram. Vale ressaltar que os casadores com peso menor são mais usados para resolver problemas de ambigüidade entre casamentos já computados do que identificar novos casamentos.

6.4 Locadora de filmes

Para este experimento foram utilizados esquemas relacionais construídos por alunos da disciplina de Modelagem de Bancos de Dados do curso de Ciências da Computação da UFRGS. Dos esquemas relacionais estudados, dois (r1 e r2) foram usados nos experimentos por serem os mais consistentes. Tais esquemas são compostos ambos por 23 conceitos. Três DTDs coletadas na *Web* foram utilizadas, sendo que elas possuem respectivamente 4, 10 e 24 conceitos. Ao total foram realizados 6 experimentos (e1,e2,e3,e4,e5,e6), conforme indicado na tabela 6.2.

Os esquemas DTD usados variam de tamanho, sendo que os conceitos do menor deles (d1) estão quase todos contidos dentro dos esquemas r1 e r2. Já nos esquemas maiores a sobreposição é menor. O esquema d3 possui poucas correspondências com conceitos dos esquemas relacionais. Os experimentos neste domínio se diferenciam dos demais pelo fato de tanto as DTDs como os esquemas relacionais terem sido

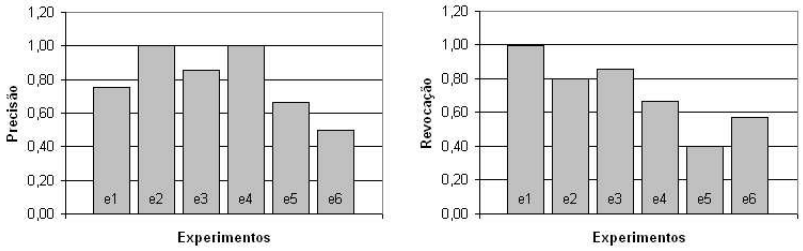


Figura 6.6: Gráficos de revocação e precisão para o domínio de locadora de filmes

modelados utilizando termos em inglês.

A figura 6.6 mostra os gráficos de revocação e precisão obtidos pelos casadores. O gráfico de precisão variou bastante de acordo com o experimento. Os falso-positivos mais uma vez são devido a similaridades lingüísticas, como em *</movie/cast/role/name>* e *<client.name>*. A menor revocação de todas ocorreu em e5. Isto é justificável pela baixa similaridade lingüística existente entre os casamentos corretos, visto que a notação usada para representar o gênero do filme difere em d2 (*</movies/movie/ @type>*) e em r2 (*<genre.name>*). Também não foi possível encontrar as correspondências envolvendo o nome do diretor e o nome dos atores. Por exemplo, em d2 essas informações estão modeladas respectivamente em *</movie/director>* e *</movie/actor>*. Já em r2, ambas informações são modeladas como *<artist.name>*. Este é um caso típico de casamento global 1:n. Nesses casos, a diferença reside na forma como a expressão de mapeamento[1] é construída, e não propriamente no casamento em si. No caso em questão, o mapeamento entre *</movie/actor>* e *<artist.name>* envolve a expressão (*actors.cod_tape = tape.code and actors.codeartist = artist.code*). Já o mapeamento entre *</movie/director>* e *<artist.name>* envolve a expressão (*tape.director = artist.code*).

Destacou-se neste experimento o casador de nomenclatura, que, através do uso dos nomes de tabelas, conseguiu identificar casamentos. Esse foi o caso, por exemplo, do casamento entre *</movie/language>* e *<language.name>*. Já o casador de relação permitiu que ambigüidades entre casamentos fossem resolvidas. Por exemplo, tanto *</movie/country>* como *</movie/certificates/cert/country>* são similares a *<movie.country>*. No entanto, apenas a primeira opção (*</movie/country>*) representa um casamento correto. Através do uso do casador de relação esta ambigüidade foi resolvida.

6.5 Locadora de imóveis

Os experimentos neste domínio envolvem o casamento de uma DTD (d1) com uma série de esquemas relacionais, sendo todos esquemas construídos por alunos da disciplina de Modelagem de Bancos de Dados do curso de Ciências da Computação da UFRGS. A DTD possui 81 conceitos (elementos atômicos) que podem ser casados. Muitos destes conceitos são elementos de mesmo nome, mas que diferem na sua hierarquia, como por exemplo os conceitos *<imobiliaria/locacao/locatario/nome>* e *<imobiliaria/locacao/fiador/nome>*. Foram modelados 6 esquemas relacionais, no entanto apenas 3 serão apresentados (r1, r2, r3), por se tratarem de esquemas mais consistentes. Dois dos esquemas escolhidos utilizavam conceitos de herança

[1] o mapeamento entre esquemas não é abordado nesta dissertação

Tabela 6.3: Experimentos no domínio de locadora de imóveis

Experimento	DTD	RDB	num. corretos	num. possíveis
e1	d1 (81 conc.)	r1 (43 conc.)	33	3483
e2	d1 (81 conc.)	r2 (39 conc.)	22	3159
e3	d1 (81 conc.)	r3 (52 conc.)	34	4212

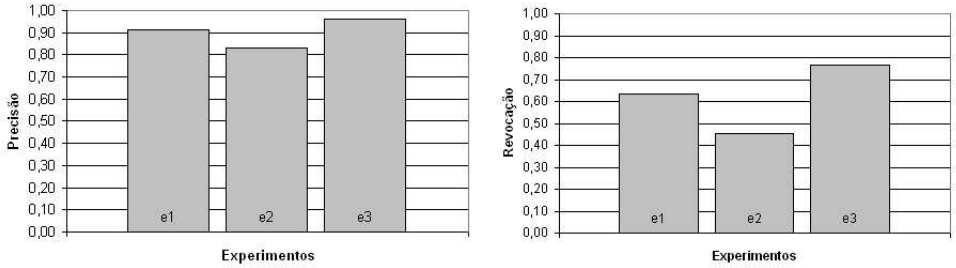


Figura 6.7: Gráficos de revocação e precisão para o domínio de locadora de imóveis

para modelar as entidades. Tais esquemas foram adaptados de forma a eliminar herança, visto que os algoritmos de casamento desenvolvidos não lidam com este tipo de estrutura. A tabela acima mostra os dados relacionados com os experimentos realizados.

Os esquemas deste domínio são bem maiores do que os esquemas do domínio de livros. Conseqüentemente, o número de casamentos corretos e o número de casamentos possíveis de serem computados também é maior. O gráfico da figura 6.7 mostra os valores de revocação e precisão obtidos para os 3 experimentos.

Os valores altos de precisão do gráfico mostram que a maioria dos casamentos indicados pelos algoritmos está correto. As exceções cabem a conceitos diferentes que compartilham sub-cadeias, como no casamento entre *</imobiliaria/locacao/fiador/imovelfiador/valormercado>* e *<imovel.valordemercado>*. Nota-se a semelhança tanto do elemento atômico *valormercado* com o atributo de tabela *valordemercado*, quanto do elemento imediatamente superior *imovelfiador* com a tabela *imovel*. Tanto o algoritmo de nomenclatura como o de relação usam essas informações para erroneamente atribuir um alto valor de similaridade para este par de conceitos.

Já os valores de revocação são baixos, o que mostra que muitas correspondências não foram encontradas pelos algoritmos. Um caso em particular ocorreu em todos os esquemas casados: as informações sobre endereço eram particionadas no esquema DTD (rua, cep, cidade, ...), enquanto nos esquemas relacionais ela era modelada em um único campo (ora como enderecoResidencial, ora como enderecoComercial). Esse tipo de casamento é classificado como casamento local 1:n, onde um conceito de um esquema equivale a concatenação de vários conceitos do outro esquema. O tratamento deste tipo de casamento não foi previsto para esta dissertação, que contempla apenas casamentos 1:1.

Neste experimento constatou-se a importância dos casadores de atração para encontrar determinadas correspondências. O casador de cardinalidade foi usado para encontrar os casamentos dos conceitos relacionados a entidade *FIADOR*. Partiu-se do fato de que, tanto no esquema XML como no esquema relacional, existem múltiplos fiadores para um único locatário. Utilizando as informações de casamento

pré-computadas da entidade *LOCATARIO*, o casador de cardinalidade conseguiu descobrir correspondências envolvendo a entidade *FIADOR*.

Já o casador de vizinhança foi útil em inúmeros casos, como por exemplo, para encontrar os casamentos envolvendo a entidade CONJUGE. Partiu-se do fato de que, tanto no esquema XML como no esquema relacional, um locatário possui apenas 1 cônjuge. Utilizando as informações de casamento pré-computadas da entidade *LOCATARIO*, o casador de vizinhança conseguiu descobrir correspondências envolvendo a entidade *CONJUGE*.

A revocação poderia ser aumentada se a técnica *Carla* fosse configurada de forma diferente. É o caso por exemplo do casamento entre (<*imobiliaria/locacao/imovel /zona_local*>, <*imovel.zonadelocalizacao*>), e o casamento entre (<*imobiliaria/ locacao/imovel/numerodepecas*>, <*imovel.nodepecas*>). Se a técnica valorizasse mais as cadeias curtas, os casamentos mostrados acima teriam valores de similaridade mais elevada. No entanto, se a técnica fosse configurada assim, muitos casamentos incorretos figurariam na lista de falso-positivos, tornando a precisão muito baixa. Em síntese, valorizar cadeias mais curtas significa maior revocação e menor precisão, enquanto valorizar cadeias mais longas significa maior precisão e menor revocação. Optou-se por valorizar mais a precisão, sob o ponto de vista de que é preferível o algoritmo retornar poucos pares de casamentos, sendo a maioria deles correto, do que retornar todos os casamento corretos em meio a muitos casamentos incorretos.

6.6 Considerações gerais

Os resultados obtidos em cada experimento foram bastante diversos, variando valores de precisão e revocação desde taxas relativamente baixas até taxas relativamente altas. De modo geral, as taxas baixas estão associadas a experimentos cuja terminologia difere bastante entre os esquemas comparados. Já as taxas mais elevadas estão associadas a experimentos cujas terminologias dos esquemas são mais similares.

Os casadores baseados em informações puramente estruturais(vizinhança e cardinalidade) foram úteis para refinar alguns casamentos lingüísticos onde existia ambigüidade, mas não foram capazes de descobrir novos casamentos. Também foi concluído que esses algoritmos podem ser usados para descobrir novos casamentos, desde que os esquemas que se deseja casar sejam compostos por poucos elementos. Já no caso de esquemas maiores, como os dos experimentos, os casadores puramente estruturais falham nesta tarefa.

7 CONCLUSÃO

Nesta dissertação foram propostas técnicas para o casamento semi-automático entre esquemas XML e esquemas relacionais. O casamento entre esses esquemas pode ser usado em aplicações como integração de informação e intercâmbio de dados). As técnicas existentes são limitadas, uma vez que elas ou são projetadas para casar esquemas desenvolvidos no mesmo modelo, como por exemplo o casamento de dois esquemas relacionais, ou são técnicas genéricas que se aplicam a diversos modelos. Neste último caso, as técnicas não são adequadas porque elas não exploram as particularidades existentes entre os esquemas para calcular a similaridade.

Para o casamento entre esquemas XML e esquemas relacionais são necessárias técnicas que possuam formas inovadoras de analisar as estruturas dos esquemas para inferir a similaridade. Nesta dissertação foram desenvolvidas 5 técnicas de casamento, sendo uma técnica lingüística e 4 técnicas estruturais:

Carla Técnica de casamento lingüístico projetada para encontrar similaridades entre cadeias que compartilham termos em comum. O diferencial desta técnica com relação as demais é o fato dela encontrar similaridades mesmo que os termos estejam invertidos e não existam espaços em branco entre um termo e outro. Uma vez que esta técnica é puramente baseada em nomes, ela é de propósito geral, não ficando restrita ao problema de casamento entre esquemas XML e relacional.

Nomenclatura Técnica que explora padrões de nomenclatura que podem ser usados quando um esquema XML é construído como uma visão de um esquema relacional. Inicialmente foram identificados 3 padrões, que são os mais comumente usados. Futuramente este casador pode ser estendido com o acréscimo de novos padrões.

Relação Técnica que explora as similaridades lingüísticas dos nós relacionados com os nós que estão sendo casados. As relações ocorrem entre nós que possuem arcos entre si, e o grau de relacionamento entre dois nós diminui conforme o número de arcos entre os nós aumenta.

Vizinhança Técnica que parte da suposição que nós vizinhos em um esquema também sejam vizinhos no outro esquema. Assim, um casamento entre dois nós pode ser usado para inferir os casamento entre dois outros nós que sejam vizinhos aos nós que foram previamente casados.

Cardinalidade Técnica que parte da suposição que nós com dependência multivalorada em um esquema também possuam as mesmas dependências multivalo-

radas no outro esquema. Assim, um casamento entre dois nós pode ser usado para inferir os casamento entre dois outros nós que possuam dependências multivaloradas com os nós que foram previamente casados.

Os dois primeiros casadores mostrados acima foram tema do artigo "*Matching of XML Schemas and Relational Schemas*", publicado no SBBD 2004. Os dois últimos casadores mostrados acima utilizam a noção de campos de atração para computar similaridades. Em síntese, um casamento entre dois nós gera campos de atração nos dois esquemas, e considera-se que os nós internos aos campos possuam uma similaridade equivalente a similaridade dos nós que geraram os campos. A noção de campos de atração foi criada neste trabalho e é usada para ilustrar o comportamento das técnicas. Uma característica dos casadores de atração é que eles necessitam que outros casadores já tenham sido executados, de modo que valores de similaridade pré-computadas possam ser usados para gerar os campos. Uma diferença importante entre os dois primeiros casadores descritos (casadores de base) e os casadores de atração é que os primeiros são usados para descobrir novos valores de similaridade enquanto os últimos são adequados para resolver situações de ambigüidade, onde um nó de um esquema apresenta alta similaridade com mais de um nó do outro esquema.

Também foram conduzidos diversos experimentos com as técnicas propostas. Através da análise geral de todos os experimentos realizados, chegou-se a uma conclusão sobre os tipos de situações onde os casadores apresentados são eficazes. O casador lingüístico demonstrou ser bastante adequado em todas as consultas onde foi usado, aliando qualidade dos casamentos com alto desempenho. Já a qualidade dos casamentos computados pelos casadores estruturais depende da nomenclatura usada nos esquemas. Se os esquemas são compostos por nomes muito distintos, as técnicas falham em encontrar os casamentos corretos. Caso os nomes sejam mais parecidos, as técnicas demonstram melhores resultados, inclusive resolvendo situações de ambigüidade.

O casamento de esquemas XML e esquemas relacionais engloba muitas questões, sendo que nem todas foram abordadas neste trabalho. Os tópicos listados a seguir referem-se a essas questões em aberto, cujo estudo pode ser tema de trabalhos futuros:

Novos Casadores Algoritmos de casamento podem explorar muitas outras características dos esquemas além das enfocadas neste trabalho. Isto inclui tratamento de elementos opcionais, tratamento dos identificadores ID/IDREF (esquema DTD) e melhor tratamento das chaves estrangeiras (esquema relacional). Além disso, casadores baseados em instância e o uso de aprendizado de máquina também podem ajudar na tarefa de casamento.

Novos modelos XML Além do modelo DTD, outros modelos são usados como gramática para documentos XML. O modelo mais difundido, que inclusive se tornou padrão W3C, é chamado XML Schema. Ele se distingue do modelo DTD por possuir construtores mais complexos que permitem definir melhor as estruturas de dados. Esquemas deste porte são mais ricos em informação semântica e abrem novas perspectivas para técnicas de casamento.

Filtro melhor elaborado Um dos filtros usados neste trabalho elimina casamentos caso algum dos nós do casamento em questão já fizer parte de um casamento

com um valor maior de similaridade. No entanto, não é feita uma verificação quanto ao melhor casamento global. O melhor casamento global, conhecido na literatura como *perfect matching* (MELNIK; GARCIA-MOLINA; RAHM, 2002), é a situação onde a média da similaridade de todos os casamentos resultantes após o filtro seja a maior possível. Para garantir tal disposição, o filtro deve ser melhor elaborado, inclusive podendo eliminar um casamento com um valor alto de similaridade para que a média global seja superior.

Casamentos 1:n As técnicas aqui propostas visam obter casamentos 1:1, onde um nó de um esquema é casado com apenas um nó de outro esquema. No entanto, podem existir situações onde um nó de um esquema seja uma combinação de mais de um nó de outro esquema (casamento local 1:n). Um exemplo de casamento local 1:n é encontrado nos experimentos realizados, onde um endereço aparece fragmentado na DTD enquanto no esquema relacional ele é modelado em um atributo atômico. Outra forma de casamento possível ocorre quando um nó de um esquema participa de mais de um casamento no outro esquema (casamento global 1:n). Nos experimentos com o domínio de filmes surgiu um exemplo de casamento global 1:n. Trata-se das correspondências envolvendo um atributo artista, que é casado tanto com um elemento destinado ao diretor de um determinado filme como a um elemento destinado a atores que atuaram neste filme.

Mapeamento entre esquemas Uma etapa posterior ao casamento de esquemas envolve o mapeamento (MILLER; HASS; HERNáNDEZ, 2000; XU; EMBLEY, 2003) entre as instâncias dos esquemas. O mapeamento é geralmente unidirecional, e indica como as instâncias armazenadas em um esquema podem ser convertidas em instâncias armazenadas em outro esquema. Situações onde é clara a necessidade de mapeamento são casos de casamentos com cardinalidade 1:n. O casamento local 1:n indica que um valor armazenado em um esquema é uma combinação de valores de outro esquema. Porém, o casamento por si só não indica como os valores devem ser combinados. A combinação pode envolver desde uma simples concatenação de dados (*nome = primeiroNome + sobreNome*) até a execução de operações aritméticas (*total = quant * preço*). Um exemplo de mapeamento para casamento global 1:n foi visto no tópico anterior. Neste caso os casamentos são diferenciados através de expressões de mapeamento, onde diferentes *joins* entre tabelas indicam como o atributo artista se relaciona com dois elementos distintos.

REFERÊNCIAS

ARENAS, M.; LIBKIN, L. A Normal Form for XML Documents. In: PODS, 2002, Madison, Wisconsin. **Proceedings...** [S.l.: s.n.], 2002.

BAEZZA-YATES, R.; RIBEIRO-NETO, B. **Modern Information Retrieval**. Menlo Park, California: Addison Wesley, 1999.

BATINI, C.; LENZERINI, M.; NAVATHE, S. B. A comparative analysis of methodologies for database schema integration. **ACM Comput. Surv.**, [S.l.], v.18, n.4, p.323–364, 1986.

BERGHOLZ, A.; FREYTAG, J. C. Querying Semistructured Data Based on Schema Matching. In: INTERNATIONAL WORKSHOP ON DATABASE PROGRAMMING LANGUAGES, 1999, Kinloch Rannoch, Scotland. **Proceedings...** Berlin: Springer, 1999. p.168–183. (Lecture Notes in Computer Science, v.1949).

BERLIN, J.; MOTRO, A. Autoplex: automated discovery of content for virtual databases. In: INTERNATIONAL CONFERENCE ON COOPERATIVE INFORMATION SYSTEMS, COOPIS, 9., 2001, Trento, Italy. **Proceedings...** Berlin: Springer, 2001. p.109–122. (Lecture Notes in Computer Science, v.2172).

BERLIN, J.; MOTRO, A. Database Schema Matching Using Machine Learning with Feature Selection. In: INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION SYSTEMS ENGINEERING, CAiSE, 14., 2002, Toronto, Canadá. **Proceedings...** Berlin: Springer, 2002. p.452–466. (Lecture Notes in Computer Science, v.2348).

BERNSTEIN, P. A.; RAHM, E. Data Warehouse Scenarios for Model Management. In: INTERNATIONAL CONFERENCE ON CONCEPTUAL MODELING, ER, 2000. **Proceedings...** Berlin: Springer - Verlag, 2000. p.1–15.

CASTANO, S. et al. A Disciplined Approach for the Integration of Heterogeneous XML Datasources. In: INTERNATIONAL CONFERENCE ON DATABASE AND EXPERT SYSTEMS APPLICATIONS, DEXA, 13., 2002, Aix-in-Provence, França. **Proceedings...** Berlin: Springer, 2002. p.103–110. (Lecture Notes in Computer Science, v.2453).

CHAWATHE, S.; GARCIA-MOLINA, H.; HAMMER, J.; IRELAND, K.; PAPAKONSTANTINOU, Y.; ULLMAN, J. D.; WIDOM, J. The TSIMMIS Project:

integration of heterogeneous information sources. In: MEETING OF THE INFORMATION PROCESSING SOCIETY OF JAPAN, 16., 1994, Tokyo, Japan. **Proceedings...** [S.l.: s.n.], 1994. p.7–18.

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

DO, H.; MELNIK, S.; RAHM, E. Comparison of schema matching evaluations. In: WEB, WEBSERVICES, AND DATABASE SYSTEMS: NODe 2002, WEB- AND DATABASE-RELATED WORKSHOPS, 2002. **Proceedings...** Berlin: Springer, 2002. p.221–237. (Lecture Notes in Computer Science, v.2593).

DO, H.; RAHM, E. COMA – A System for Flexible Combination of Schema Matching Approaches. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 28., 2002, Hong Kong, China. **Proceedings...** Saint Louis: Morgan Kaufmann, 2002. p.610–621.

DOAN, A. **Learning to Map between Structured Representations of Data**. 2002. Doctor of Philosophy Thesis — University of Washington.

DOAN, A.; DOMINGOS, P.; HALEVY, A. Reconciling Schemas of Disparate Data Sources: a machine-learning approach. In: ACM SIGMOD, 2001. **Proceedings...** [S.l.: s.n.], 2001.

DOAN, A.; DOMINGOS, P.; LEVY, A. Learning Mappings between Data Schemas. In: WORKSHOP ON LEARNING STATISTICAL MODELS FROM RELATIONAL DATA, AAAI, 2000, Austin, Texas. **Proceedings...** [S.l.: s.n.], 2000.

DOAN, A.; DOMINGOS, P.; LEVY, A. Y. Learning Source Description for Data Integration. In: WebDB, 2000. **Proceedings...** [S.l.: s.n.], 2000. p.81–86.

DOAN, A.; LU, Y.; LEE, Y.; HAN, J. Profile-Based Object Matching for Information Integration. **IEEE Intelligent Systems**, [S.l.], v.18, n.5, p.54–59, Sept./Oct. 2003.

DOAN, A.; MADHAVAN, J.; DOMINGOS, P.; HALEVY, A. Learning to map between ontologies on the semantic web. In: INTERNATIONAL WWW CONFERENCE, WWW, 11., 2002, Honolulu, Hawaii, USA. **Proceedings...** [S.l.: s.n.], 2002.

EMBLEY, D. W.; JACKMAN, D.; XU, L. Multifaceted Exploitation of Metadata for Attribute Match Discovery in Information Integration. In: WIIW, 2001. **Proceedings...** [S.l.: s.n.], 2001.

FAGIN, R. et al. **Data Exchange**: semantics and query answering. 2002.

GRAVANO, L. et al. Using q-grams in a DBMS for Approximate String Processing. **IEEE Data Engineering Bulletin**, [S.l.], v.24, n.4, p.28–34, 2001.

HERNANDEZ, M. A. et al. Clio: a semi-automatic tool for schema mapping. **ACM SIGMOD Record**, New York, v.30, n.2, p.607, June 2001.

HERRING, C.; MILOSEVIC, Z. Implementing B2B Contracts using BizTalk. In: HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES, HICSS, 34., 2001, Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2001. p.9081.

KANG, J.; NAUGHTON, J. On Schema Matching with Opaque Column Names and Data Values. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 2003, San Diego, CA. **Proceedings...** [S.l.: s.n.], 2003.

KURGAN, L.; SWIERCZ, W.; CIOS, K. Semantic Mapping of XML Tags using Inductive Machine Learning. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING AND APPLICATIONS, ICMLA, 2002, Las Vegas. **Proceedings...** [S.l.: s.n.], 2002. p.99–109.

LARSON, J. A.; NAVATHE, S. B.; ELMASRI, R. A Theory of Attribute Equivalence in Databases with Application to Schema Integration. **IEEE Transactions on Software Engineering**, New York, v.15, n.4, p.449–469, April 1989.

LEVY, A. Y. et al. Answering Queries Using Views. In: ACM SIGACT-SIGMOD-SIGART SYMPOSIUM ON PRINCIPLES OF DATABASE SYSTEMS, 14., 1995, San Jose, CA. **Proceedings...** [S.l.: s.n.], 1995. p.95–104.

LI, W.-S.; CLIFTON, C. Semantic Integration in Heterogeneous Databases Using Neural Networks. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 20., 1994, Santiago, Chile. **Proceedings...** Hove: Morgan Kaufmann, 1994. p.1–12.

LI, W.-S.; CLIFTON, C. SemInt: a tool for identifying attribute correspondences in heterogeneous databases using neural networks. **Data Knowledge Engineering**, Amsterdam, v.33, n.1, p.49–84, 2000.

LI, W.-S.; CLIFTON, C. Database integration using neural networks: implementation and experiences. **Knowledge Information Systems**, [S.l.], v.1, n.2, p.73–96, 2000.

MADHAVAN, J.; BERNSTEIN, P.; CHEN, K.; HALEVY, A.; SHENOY, P. Corpus-based Schema Matching. In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, IJCAI, 18., 2003. **Proceedings...** San Franciso: CA:Morgan Kaufmann, 2003.

MADHAVAN, J.; BERNSTEIN, P.; RAHM, E. Generic Schema Matching with Cupid. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 27., 2001. **Proceedings...** [S.l.: s.n.], 2001.

MELNIK, S.; GARCIA-MOLINA, H.; RAHM, E. Similarity Flooding: a versatile graph matching algorithm and its application to schema matching. In: ICDE CONFERENCE, 18., 2002. **Proceedings...** [S.l.: s.n.], 2002.

MILLER, R.; HASS, L.; HERNáNDEZ, M. Schema Mapping as Query Discovery. In: VLDB, 26., 2000, Cairo, Egito. **Proceedings...** [S.l.: s.n.], 2000.

MILLER, R. J. et al. The Clio project: managing heterogeneity. **SIGMOD Record**, New York, v.30, n.1, p.78–83, 2001.

MILO, T.; ZOHAR, S. Using Schema Matching to Simplify Heterogeneous Data Translation. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 24., 1998. **Proceedings...** [S.l.: s.n.], 1998. p.122–133.

POPA, L.; HERNáNDEZ, M.; VELEGRAKIS, Y.; MILLER, R. Mapping XML and Relational Schemas with Clio. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, ICDE, 2002. **Proceedings...** [S.l.: s.n.], 2002.

RAHM, E.; BERNSTEIN, P. A. A survey of approaches to automatic schema matching. **The VLDB Journal: The International Journal on Very Large Data Bases**, Heidelberg, v.10, n.4, p.334–350, 2001.

REYNAUD, C.; SIROT, J.-P.; VODISLAV, D. Semantic Integration of XML Heterogeneous Data Sources. In: INTERNATIONAL DATABASE ENGINEERING AND APPLICATIONS SYMPOSIUM, IDEAS, 2001. **Proceedings...** Los Alamitos: IEEE Computer Society, 2001.

RISHE, N. et al. Semantic Access: semantic interface for querying databases. **The VLDB Journal**, [S.l.], p.591–594, 2000.

VAN RIJSBERGEN, C. J. **Information Retrieval**. [S.l.]: Dept. of Computer Science, University of Glasgow, 1979.

WANG, G. et al. Interactive Schema Matching With Semantic Functions. In: SEMANTIC INTEGRATION WORKSHOP, 2003, Sanibel Island FL. **Proceedings...** [S.l.: s.n.], 2003.

XU, L.; EMBLEY, D. Discovering Direct and Indirect Matches for Schema Elements. In: INTERNATIONAL CONFERENCE ON DATABASE SYSTEMS FOR ADVANCED APPLICATIONS, DASFAA, 8., 2003, Kyoto. **Proceedings...** [S.l.: s.n.], 2003.

ZOBEL, J.; DART, P. W. Phonetic String Matching: lessons from information retrieval. In: INTERNATIONAL CONFERENCE ON RESEARCH AND DEVELOPMENT IN INFORMATION RETRIEVAL, 19., 1996, Zurich, Switzerland. **Proceedings...** New York: ACM Press, 1996. p.166–172.

APÊNDICE A EXPERIMENTOS REALIZADOS

Este anexo contém a descrição de todos os esquemas que foram utilizados nos experimentos apresentados no capítulo 5. Os esquemas DTD são apresentados em seu formato textual, enquanto os esquemas relacionais são apresentados em um modelo lógico. Este modelo é composto por tabelas, atributos e relacionamentos entre tabelas.

A.1 Esquemas do domínio de livros

Nestes experimentos foram utilizados 6 esquemas DTD (todos coletados da *Web*) e um esquema relacional. O esquema relacional foi criado neste trabalho especialmente para a realização de testes com os algoritmos de casamento de esquemas. A seguir são descritos todos estes esquemas:

Esquema DTD 1

```
<!-- ======= A DTD CatalogoLivro (Versão Básica) ======= -->

<!ELEMENT CatalogoLivro (Catalogo, Editora+, Livro*) >

<!-- seção <Catalogo> -->

<!ELEMENT Catalogo (TituloCatalogo, DataCatalogo) >

<!ELEMENT TituloCatalogo (#PCDATA) >
<!ELEMENT DataCatalogo (#PCDATA) >

<!-- seção <Editora> -->

<!ELEMENT Editora (NomeCorp, WebSite*, Endereco) >

<!ELEMENT NomeCorp (#PCDATA) >
<!ELEMENT WebSite (#PCDATA) >

<!ELEMENT Endereco (Rua*, Cidade, Regiao?, CodigoPostal?, Pais) >

<!ELEMENT Rua (#PCDATA) >
<!ELEMENT Cidade (#PCDATA) >
<!ELEMENT Regiao (#PCDATA) >
```

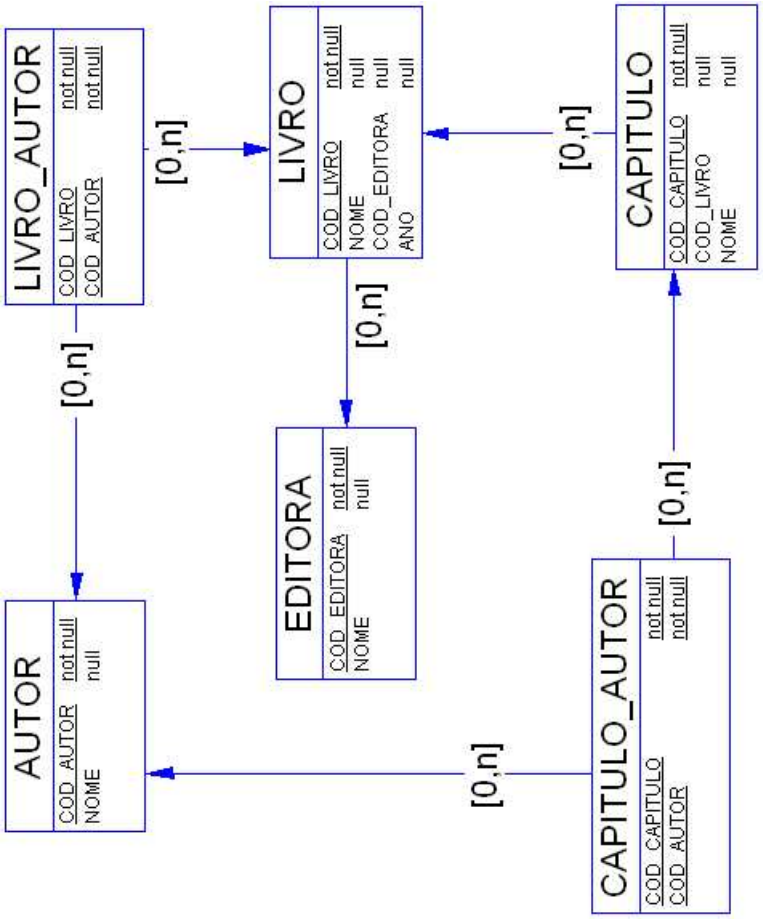


Figura A.1: Modelo lógico para domínio de livros

```
<!ELEMENT CodigoPostal (#PCDATA) >
<!ELEMENT Pais (#PCDATA) >

<!-- seção <Livro> e <Autor> -->

<!ELEMENT Livro (ISBN, Titulo, Autor+, Editor*, Resumo?, Paginas, Preco,
Assuntos?) >

<!ELEMENT ISBN (#PCDATA) >
<!ELEMENT Titulo (#PCDATA) >
<!ELEMENT Autor (#PCDATA) >
<!ELEMENT Editor (#PCDATA) >
<!ELEMENT Resumo (#PCDATA) >
<!ELEMENT Paginas (#PCDATA) >
<!ELEMENT Preco (#PCDATA) >

<!ELEMENT Assuntos (Assunto, Assunto?, Assunto?) >
<!ELEMENT Assunto (#PCDATA) >

<!-- ======= Fim da DTD  CatalogoLivro ======= -->
```

Esquema DTD 2

```
<!ELEMENT livro (titulo, autor, data, (capitulo)*) >
<!ATTLIST livro lang (ingles | portugues) #REQUIRED>
<!ELEMENT titulo (#PCDATA) >
<!ELEMENT autor (#PCDATA) >
<!ELEMENT data (#PCDATA) >
<!ELEMENT capitulo (#PCDATA) >
```

Esquema DTD 3

```
<!ELEMENT livro (infolivro, prefacio, parte+)>
<!ELEMENT infolivro (titulo, autor, direitos)>
<!ELEMENT autor (primeironome, ultimonome)>
<!ELEMENT direitos (ano, detentor)>
<!ELEMENT prefacio (titulo)>
<!ELEMENT parte (titulo, capitulo*, apendice*)>
<!ELEMENT capitulo (titulo, para)>
<!ELEMENT apendice (titulo, para)>
<!ELEMENT ano (#PCDATA)>
<!ELEMENT detentor (#PCDATA)>
<!ELEMENT primeironome (#PCDATA)>
<!ELEMENT ultimonome (#PCDATA)>
<!ELEMENT titulo (#PCDATA)>
<!ELEMENT para (#PCDATA)>
```

Esquema DTD 4

```
<!ELEMENT livro (Titulo, Capitulo+)>
<!ATTLIST Livro Autor CDATA #REQUIRED>
<!ELEMENT Titulo (#PCDATA)>
<!ELEMENT Capitulo (#PCDATA)>
<!ATTLIST Capitulo id ID #REQUIRED>
```

Esquema DTD 5

```
<!ELEMENT bd (livro*)>
<!ELEMENT livro (titulo, autor*, capitulo*, ref*)>
<!ELEMENT capitulo (texto | secao)*>
<!ELEMENT ref (#PCDATA)>
<!ELEMENT titulo (#PCDATA)>
<!ELEMENT autor (#PCDATA)>
<!ELEMENT secao (#PCDATA)>
<!ELEMENT texto (#PCDATA)>
```

Esquema DTD 6

```
<!ELEMENT CATALOGO (LIVRO)*>
<!ELEMENT LIVRO (TITULO, DESCRICAO)>
<!ELEMENT TITULO (#PCDATA)>
<!ELEMENT DESCRICAO (AUTOR,EDITORA, ANO,ISBN)>
<!ELEMENT AUTOR (#PCDATA)>
<!ELEMENT EDITORA (#PCDATA)>
<!ELEMENT ANO (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ATTLIST TITULO EDICAO CDATA #IMPLIED>
```

A.2 Esquemas do domínio de locadora de filmes

Nestes experimentos foram utilizados 3 esquemas DTD (todos coletados da *Web*) e 2 esquemas relacionais. Os esquemas relacionais foram criados por alunos da disciplina de Modelagem de Bancos de Dados do curso de Ciências da Computação da UFRGS. A seguir é descrito o enunciado usado pelos alunos juntamente com os modelos lógicos criados por eles:

Enunciado usado para a criação dos esquemas relacionais

Uma vídeo-locadora quer implantar um sistema automatizado com dados sobre seus clientes e seu acervo de fitas. Cada cliente tem nome, código identificador, endereço e telefone, bem como uma lista de dependentes que pode retirar fitas em nome dele. Cada cliente tem também um saldo em reais, que pode ser positivo (se o cliente tiver crédito junto a vídeo-locadora) ou negativo (se o cliente tiver débito com a vídeo-locadora).

Cada fita tem título, gênero (aventura, comédia, drama, etc), diretor, lista de atores / atrizes principais, ano de produção e país onde foi produzida. Cada fita tem também uma língua original, que pode ser português ou não. Se a língua original não for português, a fita deve ser dublada ou legendada. A vídeo-locadora pode ter varias cópias (exemplares) de uma mesma fita. Cada cópia tem um código que a identifica. Uma cópia de uma fita pode estar na prateleira ou alugada a um cliente. Caso esteja alugada, a data prevista de devolução é uma informação importante.

Além de guardar informações sobre os exemplares das fitas que no momento estão alugadas aos clientes, deseja-se também manter dados históricos que informem que fitas cada cliente já alugou (a cópia particular da fita é irrelevante aqui). Desta forma os atendentes poderão confirmar se um cliente desejas mesmo alugar uma fita já vista.

Esquema DTD 1

```
<!ELEMENT MovieDatabase (Movie+)>
<!ELEMENT Movie (Title, Year, Cast, Rating)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Year (#PCDATA)>
<!ELEMENT Cast (Actor+)>
<!ELEMENT Actor (#PCDATA)>
<!ELEMENT Rating (#PCDATA)>
```

Esquema DTD 2

```
<!ELEMENT movies (movie)+>
<!ELEMENT movie (title, writer+, producer+, director+, actor*, comments?)>
<!ATTLIST movie
  type (drama | comedy | adventure | sci-fi | mystery | horror | romance |
    documentary) "drama"
  rating (G | PG | PG-13 | R | X) "PG"
  review (1 | 2 | 3 | 4 | 5) "3"
  year CDATA #IMPLIED>
```

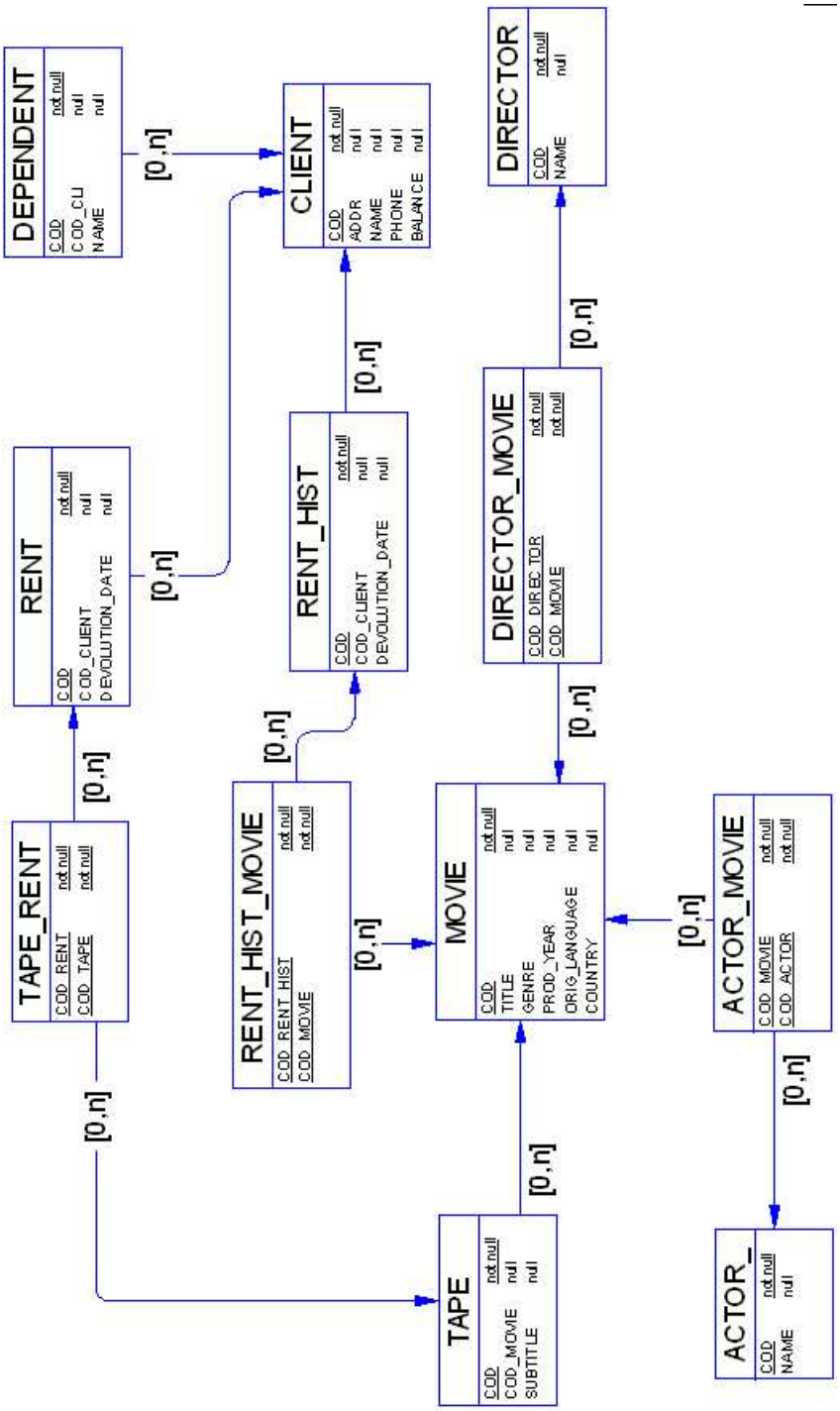


Figura A.2: Modelo lógico 1 para domínio de locadora de filmes

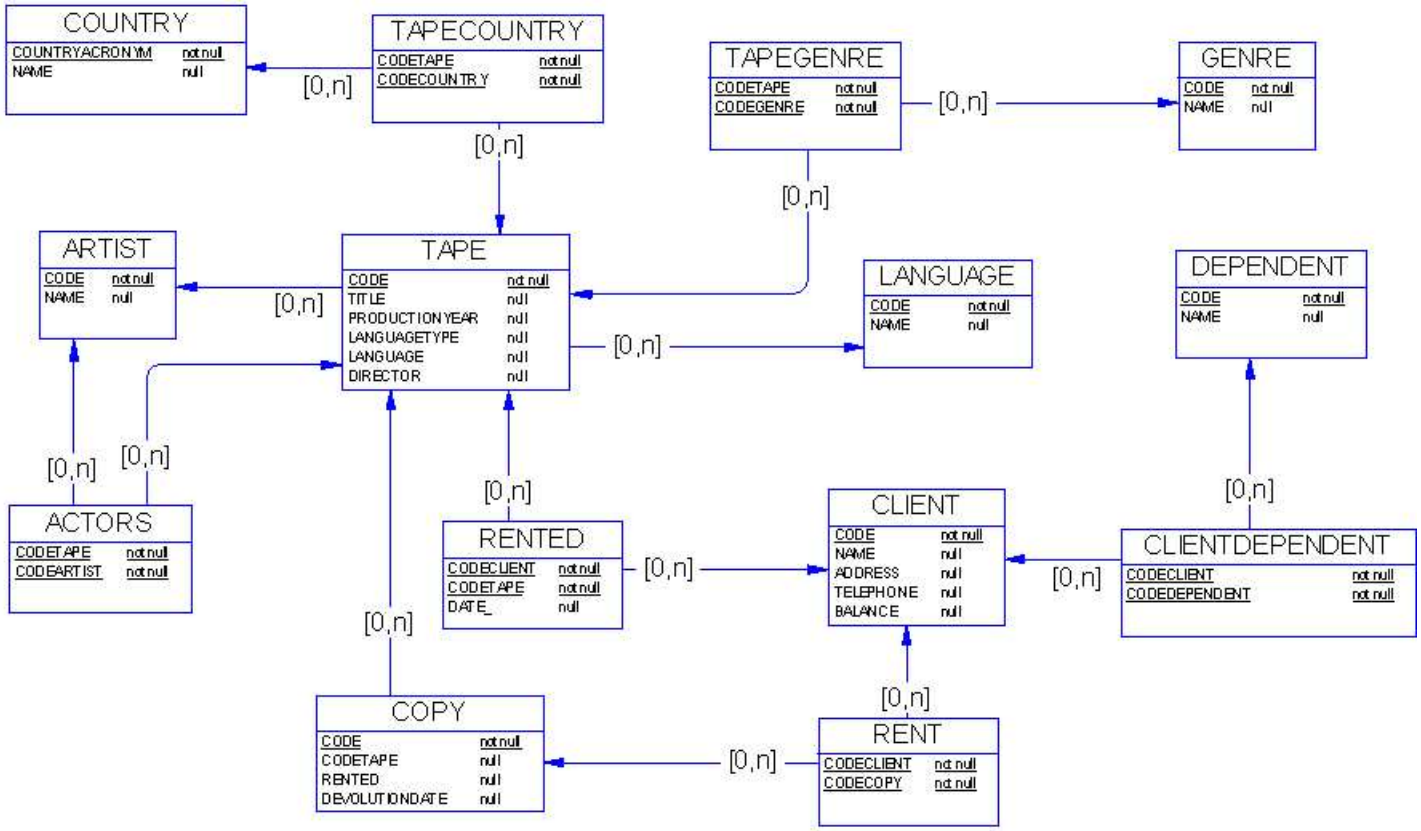


Figura A.3: Modelo lógico 2 para domínio de locadora de filmes

```
<!ELEMENT title (#PCDATA)>
<!ELEMENT writer (#PCDATA)>
<!ELEMENT producer (#PCDATA)>
<!ELEMENT director (#PCDATA)>
<!ELEMENT actor (#PCDATA)>
<!ELEMENT comments (#PCDATA)>
```

Esquema DTD 3

```
<!ENTITY % CREW "(PRODUCER | DIRECTOR | WRITER | COMPOSER | CINEMATOGRAPHER |
                  EDITOR | PRODUCTIONDESIGNER | COSTUMEDESIGNER)*">
<!ENTITY % COMP "(PRODCO | DIST | SFXCO)*">

<!-- Basiselement -->
<!ELEMENT MOVIE (TITLE, YEAR, TAGLINE*, %COMP;, COUNTRY, CERTIFICATES+, LOCATION*,
                  LANGUAGE+, PLOT*, KEYWORDS+, %CREW;, CAST, MISCCREW+)>
<!ELEMENT CERTIFICATES  (CERT+)>
<!ELEMENT CERT  (COUNTRY, RATING)>
<!ELEMENT KEYWORDS  (KEYWORD+)>
<!ELEMENT CAST  (ROLE+)>
<!ELEMENT ROLE  (ACTOR,NAME+)>
<!ELEMENT MISCCREW  (MISC+)>

<!-- PCDATA Elemente -->
<!ELEMENT PRODCO  (#PCDATA)>
<!ELEMENT DIST  (#PCDATA)>
<!ELEMENT SFXCO  (#PCDATA)>
<!ELEMENT PRODUCER  (#PCDATA)>
<!ELEMENT DIRECTOR  (#PCDATA)>
<!ELEMENT WRITER  (#PCDATA)>
<!ELEMENT COMPOSER  (#PCDATA)>
<!ELEMENT CINEMATOGRAPHER  (#PCDATA)>
<!ELEMENT EDITOR  (#PCDATA)>
<!ELEMENT PRODUCTIONDESIGNER  (#PCDATA)>
<!ELEMENT COSTUMEDESIGNER  (#PCDATA)>
<!ELEMENT ACTOR (#PCDATA)>
<!ELEMENT MISC (#PCDATA)>
<!ELEMENT TITLE (#PCDATA)>
<!ELEMENT YEAR (#PCDATA)>
<!ELEMENT TAGLINE (#PCDATA)>
<!ELEMENT COUNTRY (#PCDATA)>
<!ELEMENT RATING (#PCDATA)>
<!ELEMENT LOCATION (#PCDATA)>
<!ELEMENT LANGUAGE (#PCDATA)>
<!ELEMENT PLOT (#PCDATA)>
<!ELEMENT KEYWORD (#PCDATA)>
<!ELEMENT NAME (#PCDATA)>
```

A.3 Esquemas do domínio de locação de imóveis

Nestes experimentos foram utilizados 1 esquema DTD (criado pelo grupo de pesquisa de banco de dados da UFRGS) e 3 esquemas relacionais. Os esquemas relacionais foram criados por alunos da disciplina de Modelagem de Bancos de Dados do curso de Ciências da Computação da UFRGS. A seguir é descrito o enunciado usado pelos alunos juntamente com os modelos lógicos criados por eles:

Enunciado usado para a criação dos esquemas relacionais

Os proprietários de imobiliárias de uma cidade resolveram montar uma cooperativa para entre outras coisas, trocar informações e dados entre si sobre seus imóveis. Cada imobiliária continua autônoma para definir e alterar valores, administrar ou não condomínios, entre outros serviços. O objetivo da cooperativa é apenas centralizar dados de imóveis, locatários e proprietários a fim de executar consultas através de um único site http://imobiliariasDaCidades.cidade. Apesar de serem autônomas e independentes, elas possuem as mesmas informações:

Possuem dois tipos de clientes: os locatários, que locam os imóveis e os proprietários, donos destes. Existem vários imóveis para venda e aluguel, cada um dos quais pertencente a um proprietário. Sobre estes imóveis, cada imobiliária tem disponível informação sobre: a zona de localização, o número de peças, o preço e o ano de construção. Sobre os proprietários dos imóveis, deve ser armazenado o nome, endereço residencial (rua, numero, complemento, bairro, cep, cidade), endereço comercial (nome da empresa, rua, numero, complemento, bairro, cep, cidade), RG, CPF, estado civil. Um proprietário pode possuir vários imóveis em seu nome, tanto para venda como para aluguel, ou até mesmo em ambas as situações.

O processo de aluguel de um imóvel inicia com o proponente a locatário manifestando seu interesse em locar certo imóvel que ele acaba de conhecer. O funcionário da administradora fornece, a ele, a documentação a ser preenchida: uma ficha-proposta, na qual deverão constar os dados do proponente (nome, endereço residencial - rua, numero, complemento, bairro, cep, cidade -, endereço comercial - nome da empresa, rua, numero, complemento, bairro, cep, cidade -, RG, CPF, estado civil e renda mensal) e de seu cônjuge (nome, RG, CPF, renda mensal), se houver, e uma ficha de fiador contendo nome, endereço residencial (rua, numero, complemento, bairro, cep, cidade), endereço comercial (nome da empresa, rua, numero, complemento, bairro, cep, cidade), RG, CPF, renda mensal, além dos dados dos imóveis que ele possui (endereço, área, valor de mercado e número de matrícula no Registro de Imóveis)

Posteriormente, o proponente retorna com as fichas preenchidas, anexando à ficha-proposta seus comprovantes de rendimentos, e à ficha do fiador, os comprovantes de propriedade de imóveis, que é um requisito exigido para o fiador. As fichas são conferidas separadamente e logo os comprovantes são devolvidos. A ficha-proposta pode não ser aceita se, por exemplo, o proponente não comprovar uma renda adequada para sustentar o aluguel do imóvel. Nesse caso, o proponente não terá outra chance naquele imóvel. O fiador também pode ser rejeitado, mas neste caso o proponente tem chance de buscar outro fiador e reapresentar. Se ambas as fichas são aprovadas, são remetidas para o setor jurídico, que elabora o contrato de locação em duas vias, que terá dados do proponente, do cônjuge, do fiador, do proprietário do imóvel, do imóvel (endereço - rua, numero, complemento, bairro, cep,

cidade - área, valor de mercado,), além da data de assinatura e prazo de validade do contrato. As duas vias são entregues ao proponente para assinar, obter a assinatura do fiador e reconhecer firmas das mesmas.

Quando devolvido, a imobiliária verifica se o contrato foi devidamente assinado e se as firmas foram reconhecidas; se houver qualquer irregularidade, as duas vias do contrato voltam para o proponente para a devida regularização. Se tudo estiver correto, a imobiliária assina também o contrato, obtém a assinatura de duas testemunhas, debita na conta-corrente do imóvel o valor da taxa de locação e entrega as chaves ao proponente, juntamente com uma das vias do contrato assinado. A outra via do contrato é arquivada na pasta de contratos em vigor, juntamente com a ficha proposta e a ficha de fiador. A conta-corrente de cada imóvel é composta por lançamentos, cada um com data, histórico, valor e saldo.

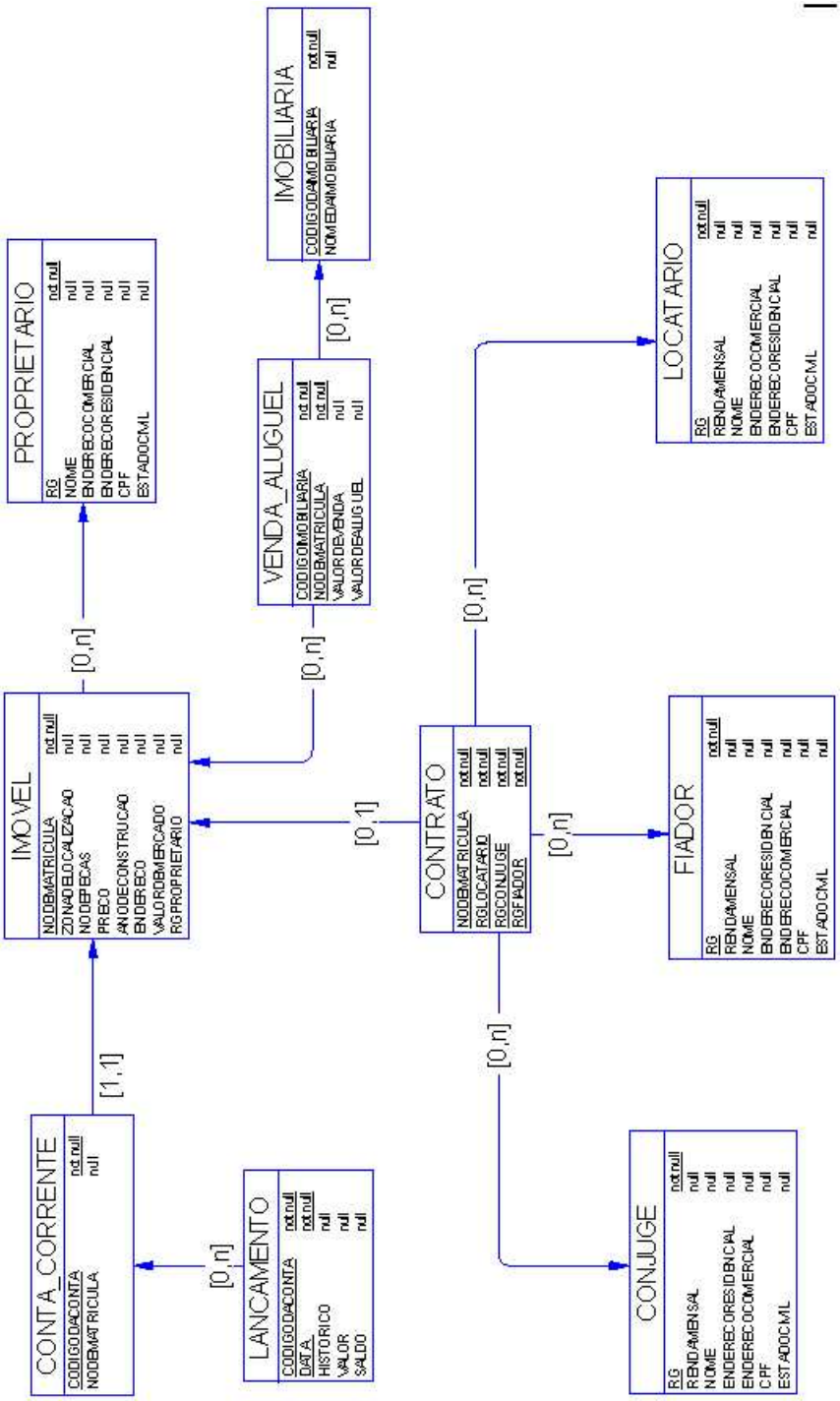


Figura A.4: Modelo lógico 1 para domínio de locadora de imóveis

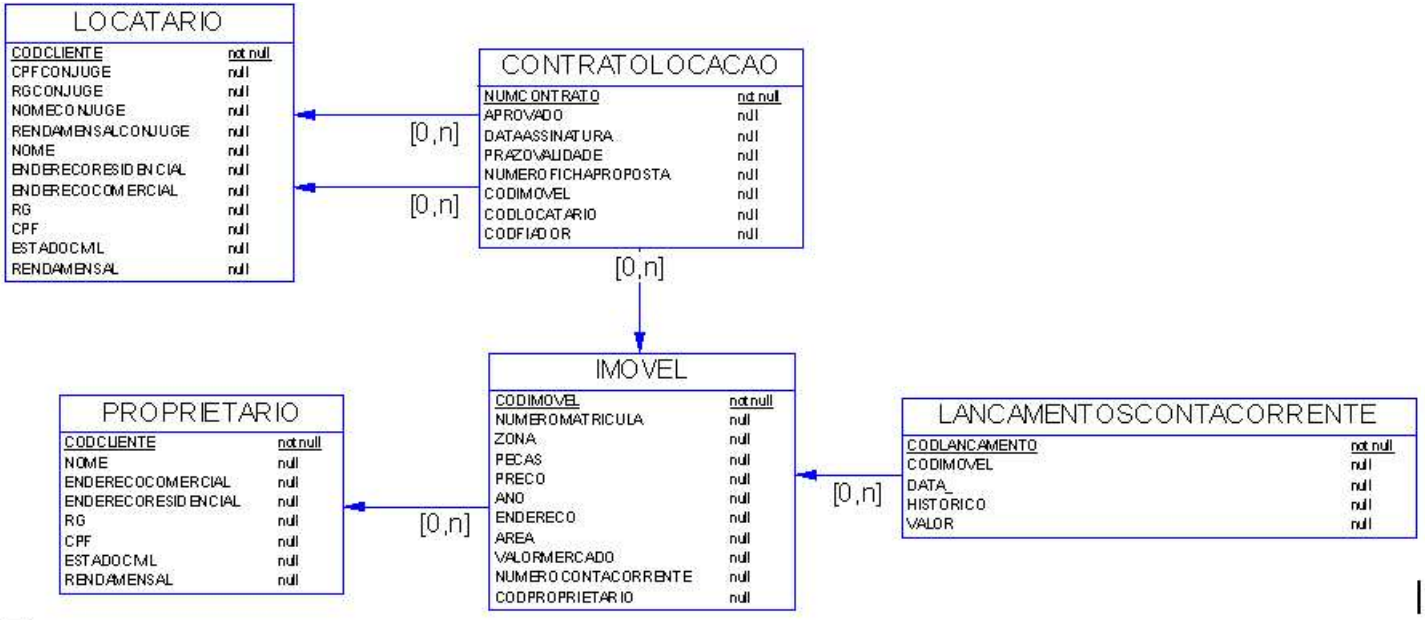


Figura A.5: Modelo lógico 2 para domínio de locadora de imóveis

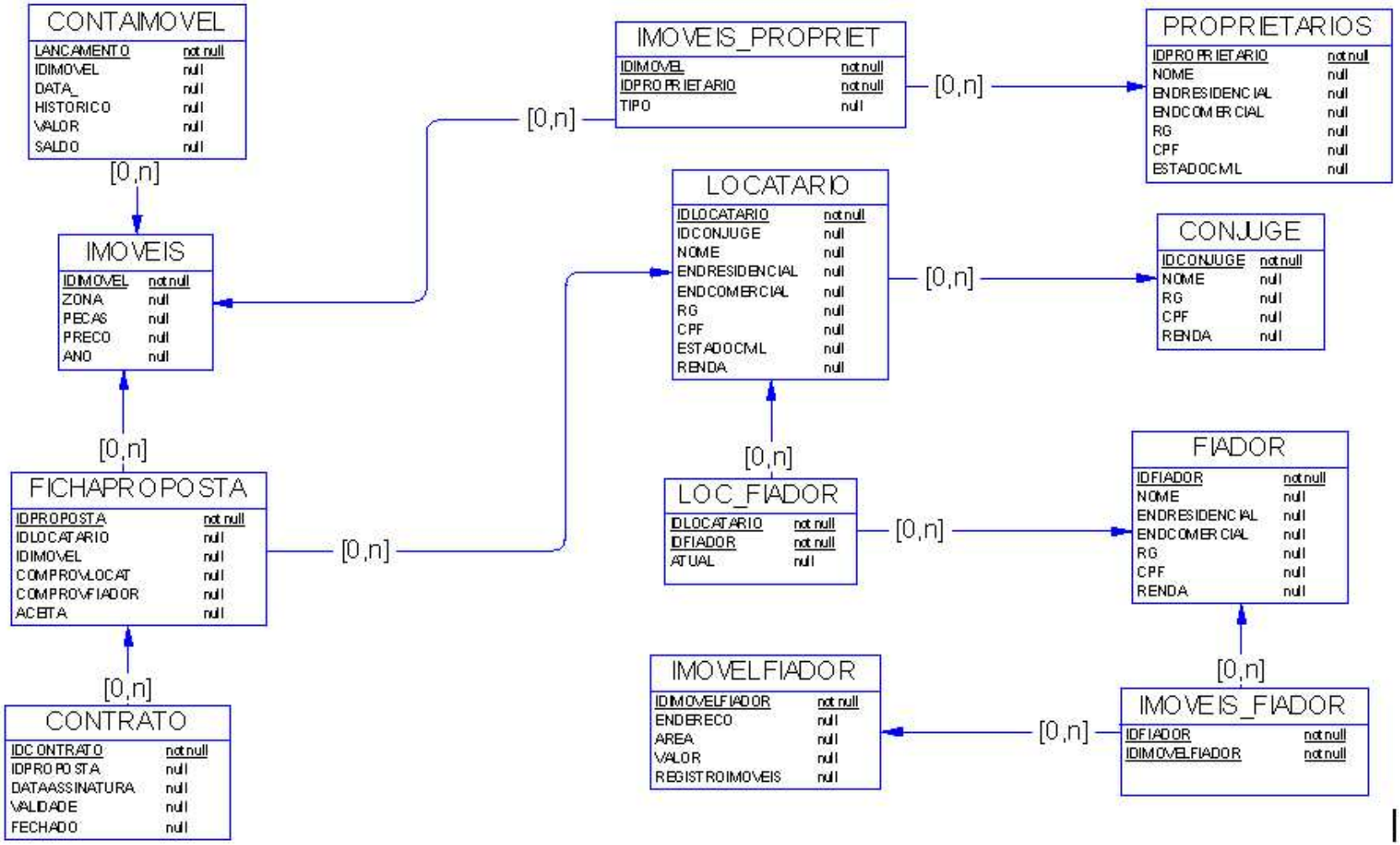


Figura A.6: Modelo lógico 3 para domínio de locadora de imóveis

Esquema DTD 1

```
<?xml version="1.0" encoding="UTF-8"?>

<!ENTITY % dados_pessoais "(nome, enderecores, enderecoprof, rg, cpf,
                            estadocivil)">
<!ENTITY % endereco "(rua, numero, complemento, bairro, cep, cidade)">

<!-- NÃO HA NECESSIDADE DE COLOCAR A ENTIDADE AQUI, ISSO EH DADO,
     VAI NO DOCUMENTO XML. COLOQUEM APENAS A NOTAÇÃO
        <!ENTITY foto_imovel SYSTEM "c:/temp/">              -->

<!NOTATION JPG SYSTEM "c:/temp/">

<!ELEMENT imobiliaria (locacao*)>

<!-- A imobiliária poderia ter ou não os elementos contrato de locação
     ou contrato de venda, e os dados sobre os terrenos a disposição-->

<!ELEMENT locacao ( locatario, fiador+, imovel, datadaassinatura,
                    valordoaluguel, tempodecontrato, condicoes)>

<!-- O contrato de locação conteria os dados do proprietário, do locatário
     na ficha de proposta, dos fiadores, sobre o imóvel, e mais informações
     adicionais sobre o mesmo                                          -->

<!ATTLIST locacao numero CDATA #REQUIRED
                                contrato_assinado (sim|nao) #REQUIRED
                                assinatura_test (sim|nao) #REQUIRED
                                assinaturas_rec (sim|nao) #REQUIRED>

<!-- Os atributos confirmam se a imobiliária assinou o contrato e se teve
     assinatura de testemunhas. Serve para ver se o contrato esta com as
     assinaturas reconhecidas                                        -->

<!ELEMENT nome (#PCDATA)>
<!ELEMENT rua (#PCDATA)>
<!ELEMENT numero (#PCDATA)>
<!ELEMENT complemento (#PCDATA)>
<!ELEMENT bairro (#PCDATA)>
<!ELEMENT cep (#PCDATA)>
<!ELEMENT cidade (#PCDATA)>

<!ELEMENT enderecores (%endereco;)>
<!ELEMENT enderecoprof ((%endereco;), nomeempresa)>
<!ELEMENT nomeempresa (#PCDATA)>

<!ELEMENT rg (#PCDATA)>
<!ELEMENT cpf (#PCDATA)>
<!ELEMENT estadocivil (#PCDATA)>
```

```
<!ELEMENT imovel (proprietario, (%endereco;), zona_local, numerodepecas,
                  preco, anodeconstrucao, situacao+, foto_imovel*)>

<!-- Informações sobre o terreno, localização do mesmo, preço de aluguel
     ou de venda, ano em que foi construído, a situação se é para venda
     ou para aluguel, e foto do terreno se tivesse não obrigatório.  -->

<!ATTLIST imovel tipo (casa|apartamento) #REQUIRED>
<!ELEMENT zona_local (#PCDATA)>
<!ELEMENT proprietario (%dados_pessoais;)>
<!ELEMENT numerodepecas (#PCDATA)>
<!ELEMENT preco (#PCDATA)>
<!ELEMENT anodeconstrucao (#PCDATA)>


<!ELEMENT situacao (lancpagamentos)>
<!ATTLIST situacao tipo (alugar|vender|ambos) #REQUIRED>

<!-- Se é para alugar ou vender, e mais lançamentos dos pagamentos -->

<!ELEMENT lancpagamentos (contacorrente, data+, descricao, valorpago+)>

<!-- Aqui seriam lançados os pagamentos de aluguel ou de parcela de
     pagamento do terreno.                                        -->

<!ELEMENT contacorrente (#PCDATA)>
<!ELEMENT data (#PCDATA)>
<!ELEMENT descricao (#PCDATA)>
<!ELEMENT valorpago (#PCDATA)>
<!ELEMENT foto_imovel EMPTY>
<!ATTLIST foto_imovel nome CDATA #IMPLIED>

<!ELEMENT locatario ((%dados_pessoais;), rendamensal, conjuge?)>

<!-- Definição dos dados do locatário e de sua renda mensal. -->

<!ELEMENT rendamensal (#PCDATA)>
<!ELEMENT conjuge (nome, rg, cpf, rendamensal)>

<!-- Definição dos dados do cônjuge se houvesse e sua renda mensal. -->

<!ELEMENT fiador ((%dados_pessoais;), rendamensal, imovelfiador+)>

<!-- A ficha do fiador conteria seus dados pessoais, sua renda mensal
     e mais os dados sobre seus imóveis.                          -->

<!ELEMENT imovelfiador ((%endereco;), area, valormercado, registrodeimovel)>

<!-- Imóvel do fiador vai definir a localização, mais valor de mercado,
     mais o registro de imóvel.                                    -->
```

```
<!ELEMENT area (#PCDATA)>
<!ELEMENT valormercado (#PCDATA)>
<!ELEMENT registrodeimovel (#PCDATA)>
<!ELEMENT datadaassinatura (#PCDATA)>
<!ELEMENT valordoaluguel (#PCDATA)>
<!ELEMENT tempodecontrato (#PCDATA)>
<!ELEMENT condicoes (#PCDATA)>

<!-- condicoes: ver se o locatário tem as condições para alugar o imóvel,
     e ver se pode ou não alugar                                        -->
```