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

CARINA FRIEDRICH DORNELES

Uma Estratégia Genérica para Casamento Aproximado de Instâncias

Tese apresentada como requisito parcial
para a obtenção do grau de
Doutor em Ciência da Computação

Prof. Dr. Carlos Alberto Heuser
Orientador

Prof. Dr. Altigran da Silva
Co-orientador

Porto Alegre, janeiro de 2006

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Dorneles, Carina Friedrich

Uma Estratégia Genérica para Casamento Aproximado de Instâncias / Carina Friedrich Dorneles. – Porto Alegre: PPGC da UFRGS, 2006.

107 f.: il.

Tese (doutorado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2006. Orientador: Carlos Alberto Heuser; Co-orientador: Altigran da Silva.

1. Similaridade. 2. Funções de similaridade. 3. Casamento de instâncias. 4. Revocação e precisão. I. Heuser, Carlos Alberto. II. da Silva, Altigran. III. Título.

"Aos meus pais,
com todo amor"

AGRADECIMENTOS

A seção de Agradecimentos é sempre a parte mais complicada para quem escreve porque sempre se corre o risco de esquecer alguém, mas eu aposto que é a parte que todo mundo mais gosta de ler, principalmente para ver se for citado. Então, para quem estiver lendo estes agradecimentos, obrigada, é sinal que vai ler a tese de alguma forma, e isso é bom.

De qualquer forma, existem pessoas muitíssimo importantes que devem ser citadas explicitamente. Em primeiríssimo lugar, meus amados pais Carlos André Beck Dorneles e Odete Ana Dorneles. Eles são a razão de tudo estar acontecendo, são os que mais torcem e os que mais incentivam. Muito obrigada, mesmo! Aos meus irmãozinhos queridos, Pablo Friedrich Dorneles e André Friedrich Dorneles que são torcida organizada sempre. Ao meu amado, Shálako Rodriguez Torrico pela "santa paciência", principalmente durante o período do sanduíche. Bom, um chato também, porque foi a pessoa que mais repetiu a pergunta proibida para todo doutorando "quando você vai acabar este doutorado?", principalmente nos últimos meses, "acabou?", "entregou?"...

Ao meu orientador, Carlos Alberto Heuser, um agradecimento muito especial, mesmo, pelo grande impulso ao meu crescimento e amadurecimento científico. Aliás, ele também deve estar agradecendo... já deve estar querendo me ver longe. Primeiro, com a bolsa DTI, depois o Mestrado e agora este Doutorado infindável; coitado.

Ao meu co-orientador, Altigran da Silva, obrigada pelas horas de discussão no MSN, Skype, SBBDs; foram ótimas e estão todas guardadas para os diversos artigos pensados. E claro, não poderia ficar de fora um agradecimento ao seu inseparável amigo Edleno S. de Moura pela idéia da precisão estimada.

Ao pessoal da sala 215, que mudou tanto desde que eu entrei na Pós, que se eu for citar todo mundo, vou acabar esquecendo alguém, e a lista vai ser bem grandinha.

Às minhas grandes amigas, Renata e Vanessa. Amigas que eu fiz neste período da Pós, e que vão ser eternas no meu coração. Meninas, obrigada, por tudo! Tanto profissional, quanto pessoal. Vou sentir muita saudades desta época, que vai ficar guardada com muito carinho nas minhas lembranças, vocês foram fantásticas sempre. Obrigada a Mirella pela leitura de artigos, pela compania nas conferências (essa menina é uma guia turístico de mãos cheia!) e pela hospedagem durante o SIGMOD. À amiga Daniela, pela parceria de todas as horas. Ah! E a todas vocês pela parceria nas organizações de eventos!

Ao pessoal do Instituto de Informática. Ao trio Luis Otávio, Lourdinha e Sil, que sempre fizeram este instituto andar. Ao pessoal da biblioteca, que sempre está pronto a nos atender e a ajudar.

A Capes, pelo financiamento do doutorado no Brasil e do sanduíche nos EUA.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

ASSC	Adjusted Similarity Score
DTD	Document Type Definition
HA	Hungarian Algorithm
FAT	Funções para valores ATômicos
FAG	Funções para valores AGregados
MHA	Modified Hungarian Algorithm
ODMG	Object Data Management Group
SSP	Successive Shortest Paths
SGBDR	Sistema Gerenciador de Banco de Dados Relacional
TaME	Tabela de Mapeamento de Escores
WHIRL	Word-based Heterogeneous Information Representation Language
XML	eXtensible Markup Language

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Casamento aproximado de instâncias é um problema central em muitos processos de gerenciamento de dados, tais como integração de dados, *data cleaning* e consulta aproximada. O principal objetivo de casamento aproximado é determinar se duas instâncias representam o mesmo objeto do mundo real. Para valores atômicos, diversas funções de similaridade têm sido definidas, que geralmente são dependentes do domínio de valores. Por outro lado, casamento de valores agregados, como tuplas ou árvores XML, ainda é um problema importante. Neste cenário, dois problemas podem ser identificados. O primeiro diz respeito a como os resultados gerados por diferentes funções de similaridade devem ser combinados em um escore único, ou para um escore normalizado. Funções individuais geralmente geram escores que não são comparáveis, pode-se obter diferentes distribuições a partir de cada função. Isto significa que não existe uma forma simples de combinar escores gerados por funções de similaridade distintas usando uma medida simples, em casamento de agregados. Nesta tese, a proposta é, ao invés de utilizar os escores originalmente gerados pelas funções de similaridade, aplicar um método para estimar a precisão dos resultados de cada função, e usar esta precisão estimada como um escore ajustado. Através deste método, a proposta apresentada nesta tese envolve duas contribuições a este problema. Primeiro, é possível permitir que o usuário especifique valores de ponto de corte (*thresholds*) que sejam significativos, usando para isso um valor de precisão ajustada como um escore de similaridade. Além disso, usando o escore ajustado, são obtidos resultados mais precisos em um processo de casamento aproximado de agregados. O segundo problema, surge quando os escores são combinados em casamento de agregados, e diz respeito à função de similaridade utilizada para combinar os valores. Particularmente, um agregado pode ser estruturado de diferentes maneiras, tais como tupla, conjunto e lista. O processo de combinação usado em cada caso deve ser distinto, a fim de se alcançar resultados mais exatos. Entretanto, não é claro como escores de similaridade individuais podem ser combinados para calcular, apropriadamente, escores para um agregado. O processo de combinação deveria ser distinto em cada caso. A contribuição apresentada para este problema é a definição de funções de similaridade específicas para cada tipo de agregado, dependendo da estruturação.

Palavras-chave: Similaridade, funções de similaridade, casamento de instâncias, revocação e precisão.

A Generic Strategy for Instance Approximate Matching

ABSTRACT

Approximate matching is a central part of several data management processes like data integration, data cleaning and approximate querying. The goal of approximate matching is to assess whether two different data instances represent the same real world object or not. For atomic values several similarity functions have been defined, which generally are dependent on the value domain. On the other hand, aggregate matching, i.e., matching of complex data instances, like tuples or XML trees, is still an important problem. In this scenario, there are two problems which can be identified. The first one is about how the results of several different functions must be combined into a single score. Individual functions usually generate scores that are not comparable. Different score distributions may be obtained even when a specific function is applied to different domains. For aggregate matching this means that there is no general straightforward way of combining scores resulting from independent distinct functions into a single measure. In our approach, instead of combining the scores computed by different functions (or some naive normalization there of) we propose a method to estimate the precision of results given by each function, and to use this estimated precision as an universal adjusted score. Using this method, our approach brings two main contributions to the problem of approximate matching. First, we allow the user to specify thresholds that are meaningful to him by using estimated precision as an adjusted similarity. Further, using the adjusted similarity score, we can better combine different functions when performing approximate matching of aggregate instances. The second problem, arising when several different functions must be combined into a single score, concerning to the similarity function used to combine the score values. In particular, an aggregate can have different structures, such as tuple, list and set. The combination process used in each case must be distinct, in order to have a more accurate result. However, it is not clear how individual similarity scores for single values occurring in an aggregate can be combined to properly compute similarity scores for the whole aggregate. Specifically, in our work we propose some similarity functions to be used in each case, which are dependent on the aggregate structure.

Keywords: approximate matching, tuple matching, similarity function, similarity score combination.

1 INTRODUÇÃO

Instâncias de um mesmo objeto do mundo real, provenientes de fontes distintas, geralmente apresentam diferenças na representação de seus valores. Exemplos simples e práticos deste cenário são as informações contidas em servidores bibliográficos, como o da DBLP (LEY, 1998), do CiteSeer (LAWRENCE; GILES; BOLLACKER, 1999a,b) e do BDBComp (LAENDER; GONÇALVES; ROBERTO, 2004). Nestas bases de dados, alguns dados, como nomes de autores por exemplo, podem aparecer com duplicações devido às diferentes formas nas quais aparecem representados em diferentes artigos. Exemplos destas duplicações são facilmente identificados nestes repositórios. Por exemplo, no servidor do BDBComp quatro diferentes representações do mesmo autor podem ser encontradas: i) Agma J. M. Traina, ii) Agma Juci Machado Train, iii) Agma Juci Machado Traina, e iv) Agma Traina. No servidor do DBLP pode-se encontrar: i) Marta L. Queiros Mattoso e ii) Marta Mattoso. E finalmente, no servidor do CiteSeer identifica-se: i) William W. Cohen e ii) William Cohen. Como se pode observar, este não é um problema raro, pois foi facilmente detectado nos três servidores em questão. Isso se torna uma questão problemática para aplicações onde existe a necessidade de se identificar que duas representações distintas se referem ao mesmo objeto do mundo real, como em integração de dados e sistemas de *data warehouse*, por exemplo. Como não é possível assegurar com exatidão que duas representações se referem ao mesmo objeto, a solução é prover um mecanismo que forneça uma noção de proximidade entre os valores. De forma geral, o termo usado para designar esta questão é casamento aproximado, ou *approximate matching* (NAVARRO, 2001).

O problema de casamento aproximado tem motivado pesquisa nas mais diversas áreas e contextos, desde questões relacionadas a consulta sobre bases de dados com mais de uma representação do mesmo objeto (como as citadas anteriormente) até a integração de fontes que possuem dados referentes ao mesmo domínio. Técnicas de processamento e execução de consulta por similaridade (ou consulta aproximada) (GRAVANO et al., 2003; JIN; LI; MEHROTRA, 2002; GRAVANO et al., 2001(b; MOTRO, 1988; AGRAWAL; FALOUTSOS; SWAMI, 1993; SHATKAY; ZDONIK, 1996; CHAUDHURI; GRAVANO, 1999; BRUNO; CHAUDHURI; GRAVANO, 2002; BUENO; TRAINA; TRAINA-JR., 2005; BARIONI et al., 2005), integração de dados (BILENKO et al., 2003; TEJADA; KNOBLOCK; MINTON, 2001; DOAN et al., 2003), *data cleaning* (GUHA et al., 2004; CHAUDHURI et al., 2003), deduplicação de dados em bibliotecas digitais (LAENDER; GONÇALVES; ROBERTO, 2004; OLIVEIRA; LAENDER; GONÇALVES, 2005) , entre outros (CARVALHO; SILVA, 2003), têm sido amplamente propostas na literatura. Em consulta aproximada (ou consulta por similaridade), o problema é identificar tuplas que re-

presentam o mesmo objeto do mundo real que aquele que o usuário especifica na consulta. Em integração de dados, o problema é reconhecer tuplas originadas de diferentes fontes que representam o mesmo objeto do mundo real. O problema em *data cleaning* é a detecção e correção de erros nos dados recebidos de fontes externas nos *data warehouses*. Estes dados podem conter inconsistências, tais como convenções diferentes, campos ausentes, erros tipográficos, ou de digitação, erros gramaticais, que devem ser solucionados antes de serem armazenados no *data warehouse*. Tais situações são muito comuns em bases de dados modernas, particularmente nos casos em que os dados armazenados são gerados e acessados por diferentes pessoas ou *softwares*.

Métodos de casamento aproximado geralmente contam com o uso de uma função de similaridade $f(a_1, a_2) \mapsto s$ que calcula um escore s para um par de valores a_1 e a_2. Quanto mais alto o valor do escore mais similares os dois valores a_1 e a_2 são entre si.

A literatura recente é abundante em funções de similaridade para comparação de valores textuais como nomes de pessoas, instituições, datas, etc (COHEN; RAVIKUMAR; FIENBERG, 2003a; CHAPMAN, 2004; COHEN; RAVIKUMAR; FIENBERG, 2003b; TEJADA; KNOBLOCK; MINTON, 2001; NAVARRO, 2001; LIMA, ????). Entretanto, na maioria das vezes, tais funções não apresentam um resultado perfeito na comparação de valores. Um par de valores que não deveria ser considerado similar pode ter um escore de similaridade mais alto do que o limiar e portanto aparecer no resultado final (falso positivo). Da mesma forma, um par de valores que deveria ser considerado similar pode não aparecer no resultado (falso negativo). Uma maneira de avaliar os resultados gerados por uma função de similaridade é utilizar medidas que quantificam a qualidade. Isso pode ser feito com o uso de medidas clássicas da comunidade de Recuperação de Informação, como revocação (*recall*) e precisão(*precision*) (BAEZA-YATES; RIBEIRO-NETO, 1999). As medidas de revocação e precisão indicam a proporção dos pares de valores corretamente identificados como similares que aparecem no resultado (precisão) e a proporção do total de pares similares existentes que aparecem no resultado final (revocação).

Em uma função de similaridade $f(a_1, a_2) \mapsto s$, o escore s possui um valor no intervalo $[0, 1]$, sendo 1 totalmente similar (igual) e 0 totalmente diferente (desigual). No entanto, por se tratar de um valor que fornece uma noção de proximidade entre os valores do par comparado, não é possível definir um limite único que indique onde acaba a desigualdade e começa a igualdade (ou vice-versa). No entanto, pode-se dizer que dois objetos são considerados similares, ou seja, são considerados o mesmo objeto do mundo real, se o escore gerado por uma função f for maior do que um determinado limiar, ou, ponto de corte (*threshold*) (BEYER et al., 1999). O valor de limiar deve ser cuidadosamente escolhido, dependendo da qualidade que se espera nos resultados. Por exemplo, para uma aplicação que requer poucos falsos negativos (alta revocação) um limiar mais baixo deve ser usado, por outro lado, quando uma aplicação requer poucos falsos positivos (alta precisão) um valor de limiar mais alto deve ser usado. A escolha de um valor de limiar não é uma tarefa trivial, tanto se escolhida automaticamente, quanto se deixada a cargo de um usuário. Vários trabalhos identificam esta escolha como um problema (BILENKO; MOONEY, 2003; SARAWAGI; BHAMIDIPATY, 2002), mas nenhuma estratégia é aceita universalmente, pois geralmente, o escore gerado por uma função de similaridade não tem um significado direto para o usuário. Provavelmente, o limiar deve ser um valor pre-

definido em um processo de estimativa ou de aprendizado, pois o escore retornado por uma função é um valor que depende de detalhes internos do algoritmo que a implementa, e até mesmo da semântica usada pela função de similaridade aplicada.

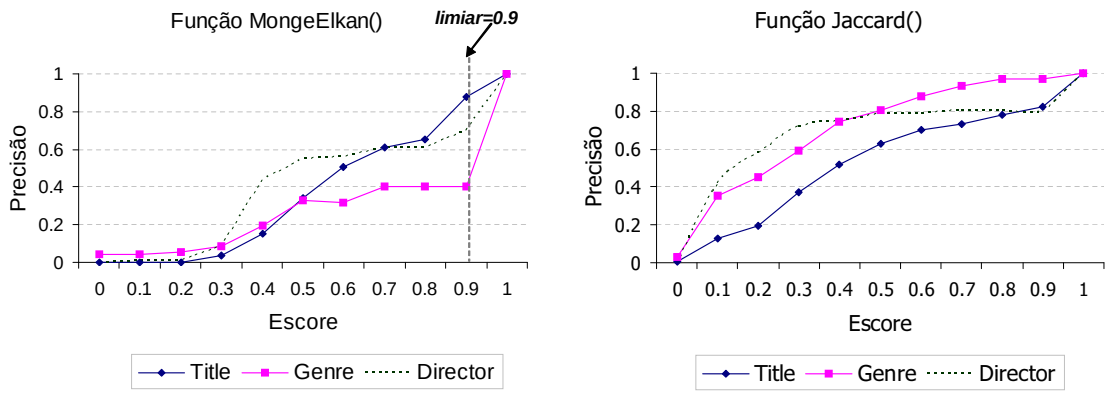
Diferentes funções geram escores em diferentes distribuições. Mais precisamente, diferentes funções são implementações de diferentes métricas de similaridade, o que significa que geram diferentes interpretações de escores. Neste caso, os escores gerados não são comparáveis entre si. Em casamento aproximado de objetos complexos, ou agregados[1] (objetos compostos por multi-atributos), geralmente diferentes funções são usadas para cada um dos atributos (TEJADA; KNOBLOCK; MINTON, 2001; COHEN; RAVIKUMAR; FIENBERG, 2003b; GUHA et al., 2004). Por exemplo, para o objeto instituição Inst = ⟨nome="UFRGS", cidade="PoA"⟩ composto pelos atributos nome e cidade, uma função de similaridade específica para cada um deles poderia ser usada. De forma geral, a estratégia aplicada para determinar a similaridade entre dois objetos complexos consiste em combinar os escores previamente calculados para seus atributos. O resultado desta combinação pode não estar completamente correto devido às diferenças nos escores usados.

Quando se avalia os resultados gerados por uma função de similaridade, pode-se observar que diferentes funções de similaridade geram escores com distribuições distintas sobre os conjuntos de dados. A execução de uma dada consulta sobre um conjunto específico pode gerar um resultado cujos pares retornados são considerados todos pares verdadeiramente similares, ou seja, precisão de 1.0. Neste caso, uma função f_1 pode gerar escores de no mínimo $0,8$ para pares verdadeiramente similares, enquanto que uma função f_2 pode gerar escores de $0,7$ para estes mesmos pares. Portanto, a qualidade dos resultados (medida em termos de revocação e precisão) pode variar de uma função para outra quando um determinado limiar é especificado. Utilizando o mesmo exemplo, caso fosse estipulado um limiar de $0,7$, a qualidade de f_1 cairia em termos de precisão, pois o escore $0,8$ é o mínimo retornado para pares relevantes, com a função f_1. Essa variação pode ocorrer, inclusive, quando a mesma função de similaridade é aplicada a diferentes conjuntos de dados. Isso quer dizer que um valor de limiar que foi predefinido para uma função de similaridade não poderá ser usado quando esta mesma função for aplicada a diferentes conjuntos de dados.

A Figura 1.1 apresenta alguns exemplos reais que mostram alguns casos desta distinção. As funções de similaridade apresentadas na figura foram avaliadas sobre uma base de dados contendo informações sobre filmes que possui os atributos Title, Genre e Director. A Figuras 1.1 A) e 1.1 B) apresentam, respectivamente, avaliações das funções de similaridade *MongeElkan()* e *Jaccard()* (CHAPMAN, 2004). Em ambos os casos pode-se observar que a distribuição é distinta nos diferentes conjuntos de dados. Por exemplo, considerando especificamente a Figura 1.1 A) e um valor de limiar de $0,9$, a qualidade dos resultados varia grandemente de uma função para outra para cada conjunto de dados dos atributos: em Title a precisão é de $0,87$, em Director a precisão cai para $0,70$ e em Genre a precisão cai ainda mais, ficando em $0,40$.

Quando os dados que estão sendo comparados são objetos complexos, tais como tuplas relacionais ou árvores XML, outro problema pode ser identificado: como combinar apropriadamente os escores gerados para os atributos de um objeto. Como um exemplo, supõem-se a tupla ⟨nome="Koffi Anan", organização="United Nations"⟩.

[1] O termo "agregado" é utilizado informalmente até sua definição formal no Capítulo 4



A) B)

Figura 1.1: Distribuições para três diferentes funções de similaridade em três diferentes atributos

Quando algum processo de casamento for aplicado a esta tupla, é desejado que altos escores de similaridade sejam dados a todas as tuplas que um usuário considera como representando a mesma entidade do mundo real, tais como ⟨nome="K. Anan",organização="United Nations"⟩ and ⟨nome="Kofi Anan",organização="UN"⟩, que representam um casamento aproximado, bem como a todas as tuplas cujos valores de atributos tenham correspondência exata. Neste caso, é importante saber como combinar apropriadamente os escores de similaridade de seus componentes (ou seja, de seus atributos) para gerar um grau de similaridade entre objetos de um base de dados (por exemplo, tuplas de uma relação). Trabalhos recentes reconhecem que esta questão é um problema e sugerem novas propostas e técnicas para resolvela (MOTRO, 1988; FERGUSON; BRIDGE, 1999; CHENG et al., 2004). Em muitos casos, estas técnicas são acompanhadas por algum processo de transformação dos valores (TEJADA; KNOBLOCK; MINTON, 2001). Entretanto, como funções individuais geram escore de similaridade que não são comparáveis (por não possuírem a mesma distribuição), não existe uma maneira fácil e evidente de como combinar escores de similaridade individuais em um valor único. Por exemplo, considerando o exemplo anterior, da tupla ⟨nome="Koffi Anan", organização="United Nations"⟩, uma função para comparação de nomes próprios (OLIVEIRA; LAENDER; GONÇALVES, 2005) poderia ser usada no atributo nome, enquanto que alguma função que considere siglas e abreviações (FRENCH; POWELL; SCHULMAN, 2000) deveria ser usada para organização. Para avaliação de um escore de similaridade entre as tuplas, os escores gerados por estas funções individuais deve ser apropriadamente combinados, levando em conta que os valores gerados estão distribuídos distintamente.

Nas abordagens propostas na literatura, a combinação dos escores calculados para os atributos de um objeto é feita da mesma forma para qualquer estrutura de objeto. No entanto, objetos complexos podem possuir diferentes estruturas e portanto, diferentes estratégias deveriam ser usadas para combinar os escores de seus atributos. De forma geral, um objeto complexo pode possuir uma estrutura de tupla ou de coleção. Um objeto complexo pode ser tratado como uma tupla, quando seus atributos possuem valores pertencentes a diferentes domínios de valores. Ao contrário, para um objeto ser tratado como uma coleção, os atributos devem possuir

valores pertencentes ao mesmo domínio de valores. No caso de uma coleção, ela pode ser tratada como uma lista ou como conjunto, dependendo se a intenção é especificar a ordem dos membros da coleção (lista) ou não (conjunto). Elementos de documentos XML são exemplos clássicos de objetos complexos do tipo coleção, pois podem possuir como filhos a repetição de outro elemento. Por exemplo, considerando a definição de um elemento em uma DTD `<!ELEMENT Autores(nome+)>`, o elemento `Autores` pode ser considerado como objeto do tipo coleção composto por n atributos `nome`.

Por exemplo, seja um objeto com a seguinte estrutura: O_1 = {país="Brasil", país="Alemanha", país= "Turquia"}. Neste caso, ao se executar algum processo de casamento aproximado com os seguintes objetos: O_2 = {país="Turquia", país="Brasil", país= "Turquia"} e O_3 = {país="Alemanha", país="Brasil", país= "Turquia"}, o resultado , indicando se um objeto casa ou não com o outro, depende da intenção do usuário ou do domínio de aplicação. No exemplo apresentado, se os objetos a serem casados correspondem aos países participantes da Copa do Mundo 2002, então é desejado que altos escores de similaridade sejam alcançados na comparação de (O_1, O_2) e de (O_1, O_3), já que os três países participaram da Copa em 2002. No entanto, caso os objetos a serem casados são países ganhadores da Copa do Mundo 2002 na ordem de classificação, então os escores de similaridade gerados na comparação de (O_1, O_2) e de (O_1, O_3) não devem ser altos, pois a ordem dos valores dos atributos em O_2 e O_3 não corresponde à classificação correta dos vencedores. Portanto, para avaliação de um escore de similaridade entre conjunto de valores, os escores gerados para cada um dos atributos devem ser apropriadamente combinados, dependendo da aplicação em que o processo de casamento é executado. Neste caso, torna-se importante que a função de similaridade considere a ordem na qual os valores aparecem na coleção, se assim for necessário. Acredita-se que esta distinção entre tuplas e coleções é uma característica importante a ser considerada durante o casamento de objetos complexos já que a composição de cada um é distinta. Neste contexto, uma questão importante que surge é como combinar os escores gerados pelas funções de similaridade dos atributos, dependendo da estrutura do objeto.

Dados os problemas apresentados anteriormente, duas grandes motivações conduzem o trabalho desenvolvido nesta tese:

1. ausência de tratamento adequado aos escores gerados pelas funções de similaridade;

2. falta de uma abordagem que manipule tuplas e coleções de dados de forma diferenciada.

1.1 Objetivos

Especificamente, esta tese possui três principais objetivos:

1. encontrar uma alternativa para os escores de similaridade originalmente gerados por funções de similaridade, já que estes não são intuitivos quanto ao significado do valor;

2. estabelecer uma estratégia de combinação de escores gerados com diferentes distribuições de escores, pois estes não são adequados para combinação em um processo de casamento de objetos complexos;

3. definir funções de similaridade adequadas às diferentes formas de composição de um objeto complexo, tratando adequadamente as coleções de valores.

Para o primeiro caso, como alternativa, propõe-se o uso de um novo escore de similaridade, o escore ajustado, ao invés dos escores originalmente gerados pelas funções. A idéia é uma abordagem baseada em um processo de estimativa durante o qual a precisão é estimada para cada função de similaridade e para diferentes conjuntos de dados. O resultado desta fase é uma tabela que mapeia escores de similaridade para valores de precisão, para cada função sobre um conjunto de dados. Para o processo de casamento, o usuário apenas deve especificar a precisão esperada nos resultados. O sistema então configura como limiar o valor que corresponde a precisão esperada, como definido na tabela que foi gerada durante o processo de estimativa. O raciocínio por trás desta idéia é que a precisão estimada expressa a convicção de um valor de um dado atributo ser uma representação diferente de outro valor do mesmo atributo. Portanto, a precisão estimada pode ser usada como uma função que não é apenas única para qualquer tipo de atributo comparado, mas que também fornece um resultado que é significativo ao usuário, permitindo que seja expressado em consultas como um parâmetro que define o grau de proximidade permitida em um processo de casamento aproximado.

Para o segundo caso, a proposta é aplicar diferentes estratégias de combinação de escores gerados por funções de similaridade, de acordo com a estrutura do objeto: tupla ou coleção. Desta forma são definidas funções de similaridade que são consideradas dependentes da estrutura. No caso do tratamento de coleções, pode-se considerar ainda listas e conjuntos. Neste caso, dependendo do domínio de aplicação, uma ou outra estrutura deve ser usada, e para isso duas outras funções de similaridade foram definidas, para listas e para conjuntos, que são consideradas dependentes da aplicação.

1.2 Contribuições

Com o tratamento adequado aos problemas apresentados anteriormente, as duas grandes contribuições da tese são:

1. a proposta de um novo método para mapeamento de escores de similaridade gerados originalmente por diferentes funções de similaridade para um escore normalizado que pode ser utilizado em diversas técnicas de combinação de escores;

2. a definição de diferentes formas de combinação dos escores, introduzidas através de funções de similaridades para agregados, também caracteriza uma contribuição da tese.

Através das propostas apresentadas, é possível permitir:

- a um usuário especificar a qualidade esperada no processo de casamento usando um valor significativo (a precisão estimada);

- a combinação apropriada de escores de funções de similaridade usadas em valores atômicos para avaliar a similaridade entre tuplas;

- o uso de diferentes funções de similaridade que são dependentes da estrutura do agregado, tupla ou coleção.

Os resultados dos experimentos demonstram que os escores ajustados são perfeitamente combináveis quando usados no processo de casamento de agregados, produzindo resultados mais precisos do que quando os escores originais são usados. Além disso, é possível mostrar também, através dos experimentos, que os escores ajustados podem ser usados como valor de ponto de corte, de forma bastante intuitiva ao usuário. Finalmente, através dos experimentos é possível demonstrar que o tratamento diferenciado entre tuplas e coleção é um ponto relevante a ser considerado.

1.3 Organização do Texto

O texto da tese é organizado da forma como segue.

- No Capítulo 2, são apresentados trabalhos da literatura que possuem propostas do uso de funções de similaridade para casamento de valores agregados. São apresentados trabalhos propostos em diferentes áreas de aplicação, desde integração de dados e *data cleaning* até processamento de consultas. É importante observar que os trabalhos existentes na literatura podem fazer uso da estratégia proposta nesta tese.

- No Capítulo 3, é apresentada a descrição detalhada da proposta de mapeamento de escores de similaridade gerados por diferentes funções para um único escore gerado por uma única medida. O Capítulo apresenta o mecanismo utilizado para alcançar o escore normalizado, que pode ser usado para todas as funções de similaridade.

- No Capítulo 4, a abordagem proposta para combinação de escores de similaridade de atributos em objetos complexos é detalhadamente descrita. Nesse Capítulo, são apresentadas propostas de diferentes tipos de combinação de escores, dependendo da estrutura na qual um objeto é apresentado, tupla ou coleção de dados.

- No Capítulo 5, são apresentados os experimentos que foram executados a fim de validar a abordagem apresentada. O objetivo dos experimentos é apresentar a validação das propostas apresentadas nos Capítulos 4 e 3.

- No Capítulo 6, são descritas conclusões, apresentando as principais contribuições da tese, lista de artigos publicados e produção gerada, comparação com os trabalhos relacionados e uma breve discussões de trabalhos futuros.

2 TRABALHOS RELACIONADOS

Neste capítulo, são descritos alguns trabalhos que possuem propostas do uso ou combinação de funções de similaridade para casamento de agregados. O problema de casamento aproximado tem incentivado a pesquisa há muito tempo e nas mais diversas áreas e contextos. A nomenclatura não é única quando o problema é "identificar várias representações do mesmo objeto". O problema é tratado como *record linkage* (WINKLER, 1999; FELLEGI; SUNTER, 1969), *merge/purge problem* (HERNANDEZ; STOLFO, 1995), *duplicate detection* (BILENKO; MOONEY, 2003; SARAWAGI; BHAMIDIPATY, 2002; MONGE; ELKAN, 1997), *vague queries* (MOTRO, 1988), *hardening soft databases* (COHEN; KAUTZ; MCALLESTER, 2000), *reference matching* (MCCALLUM; NIGAM; UNGAR, 2000), *entity name matching* (COHEN; RICHMAN, 2002), *object matching* (DOAN et al., 2003), *object identification* (TEJADA; KNOBLOCK; MINTON, 2001), *instance matching* (RAHM; DO, 2000), e assim por diante. Na descrição dos trabalhos apresentados neste capítulo, procurou-se manter a nomenclatura original.

De forma geral, os trabalhos podem ser identificados sob dois aspectos: (1) aqueles que apresentam propostas para combinar os valores dos escores obtidos nos atributos através de alguma métrica de similaridade (MOTRO, 1988; GUHA et al., 2004; SCHALLEHN; SATTLER; SAAKE, 2004; CHAUDHURI et al., 2003); e (2) aqueles que propõem métodos em que alguma técnica de aprendizado de máquina é aplicada (TEJADA; KNOBLOCK; MINTON, 2001; BILENKO et al., 2003; BILENKO; MOONEY, 2003; DOAN et al., 2003).

Outra importante observação em relação aos trabalhos que tratam de casamento aproximado diz respeito às estratégias utilizadas para restringir o conjunto resposta: (1) limitadas por um limiar (ponto de corte, *threshold*) de similaridade que define um valor de abrangência para os resultados de uma consulta, conhecidas como *range queries*; e (2) limitadas por uma quantidade k de objetos que devem fazer parte da resposta, conhecidas como *top-k queries*. A maior parte dos trabalhos descritos neste capítulo utilizam a primeira estratégia. Trabalhos que fazem uso da segunda estratégia serão devidamente mencionados quando descritos.

Os trabalhos descritos são propostas apresentadas em diferentes áreas de aplicação, como em integração de fontes de dados, processamento de consultas e *data cleaning*. A seguir, algumas propostas são descritas.

2.1 Integração de dados

De forma geral, em integração de dados, diversos problemas são tratados, tais como, descobrir o mesmo objeto representado de diferentes formas nas várias fontes

sendo integradas (TEJADA; KNOBLOCK; MINTON, 2001), construção de modelos para *wrappers* a fim integrar fontes automaticamente (CARMAN; KNOBLOCK, 2005), gerar planos para integrar automaticamente os dados vindos das fontes (MICHALOWSKI et al., 2004; KNOBLOCK; AMBITE; THAKKAR, 2003) entre outros. No entanto, o objetivo geral das propostas de integração de dados é gerar, como saída do processo, mapeamentos que indicam quais pares de objetos, cada um com suas respectivas fontes, correspondem ao mesmo objeto do mundo real.

2.1.1 Active Atlas

O Sistema Active Atlas (TEJADA; KNOBLOCK; MINTON, 2001) implementa um método de casamento aproximado para identificação de objetos usando técnicas de aprendizado de máquina.

O objetivo principal é o aprendizado de informações úteis para efetuar o mapeamento entre objetos a fim de integrar fontes de dados. O processo, basicamente, é executado em dois passos, como descrito a seguir.

1. O cálculo do escore de similaridade é efetuado para cada atributo em um objeto usando transformações nas cadeias de caracteres e funções de similaridade específicas para o domínio de valores dos atributos.

2. Regras de aprendizado são especificadas para determinar quais os atributos de um objeto são mais importantes para o mapeamento. Estas regras são chamadas de Regras de Mapeamento, e são aprendidas em um processo de treinamento.

O sistema seleciona preliminarmente uma fonte, que serve como um alvo, e então fornece um mapeamento a partir dela com cada uma das outras fontes de mesmo domínio. Este processo pode ser feito de duas formas, uma maneira é criar uma tabela de mapeamento que especifica em cada entrada de uma fonte de dados qual a entidade equivalente na outra fonte de dados. Alternativamente, ele pode ser representado por uma função de mapeamento, em que um componente do par de objetos dado é transformado no outro. O sistema Active Atlas compara os atributos compartilhados pelo par de objetos a fim identificar aqueles que representam o mesmo objeto do mundo real. Determinados atributos são considerados mais importantes para decidir se um mapeamento deve existir entre dois objetos. O objetivo é automatizar o processo de construção de regras de identificação de objeto, ou regras de mapeamento entre objetos. A proposta do sistema é aprender a desenvolver tais regras através de pouca intervenção do usuário.

Foi desenvolvido um método semi-automático para construir tabelas e funções de mapeamento analisando previamente os dados das fontes. O método tenta criar pares de objetos de uma fonte com um objeto correspondente (ou os objetos, em alguns casos) em uma outra fonte. A idéia básica é usar técnicas da recuperação de informação para fornecer um mapeamento inicial, e aplicar então técnicas da aprendizagem de máquina para melhorar o mapeamento. Os mapeamentos iniciais casam objetos de duas fontes com base em sua similaridade textual. Na fase subseqüente da aprendizagem, o sistema aprende dois tipos de regra para ajudar a melhorar/verificar os mapeamentos iniciais, que são descritas a seguir.

Regras de Transformação. Identificam transformações textuais, tais como siglas, abreviaturas, e ordenação de frase, que são comuns a um determinado do-

mínio. Por exemplo, o sistema pode aprender que "Rep." é uma abreviação comum de "República", ou que uma fonte geralmente emprega siglas, ou que uma fonte representa nomes de pessoa como "Sobrenome, Nome", enquanto outra usa "Nome, Sobrenome".

Regras de Mapeamento. São usadas quando se quer comparar objetos com múltiplos atributos. As regras de mapeamento contêm informação sobre quais as combinações de atributos são importantes para determinar o mapeamento entre dois objetos, bem como os pontos de corte (*thresholds*) para os escores de similaridade para cada atributo. Muitas regras de mapeamento podem ser necessárias para classificar apropriadamente os objetos de um domínio específico. Exemplos de regras de mapeamento, aplicadas a uma base de dados de Filmes, podem ser os seguintes:

Regra 1: $Titulo > 0,859 \ \bigwedge \ Diretor > 0,901 \Rightarrow mapeado$
Regra 2: $Titulo > 0,859 \ \bigwedge \ Genero > 0,954 \Rightarrow mapeado$

Estas regras são geradas através da aplicação de aprendizado com árvores de decisão (QUINLAN, 1996). Cada nodo da árvore contém um teste a ser executado sobre um atributo, e cada aresta é rotulada com um possível resultado do teste. A Figura 2.1 apresenta um exemplo de uma árvore de decisão referente às duas regras apresentadas acima. Primeiramente, o atributo Titulo é testado. Se o resultado for positivo (aresta "+"), então o atributo Diretor é testado a seguir. Caso o resultado também seja positivo (aresta "+"), o objeto é mapeado e o atributo Gênero não precisa ser testado. Este atributo só será testado no caso do teste executado sobre o atributo Diretor resultar em uma resposta negativa (aresta "-").

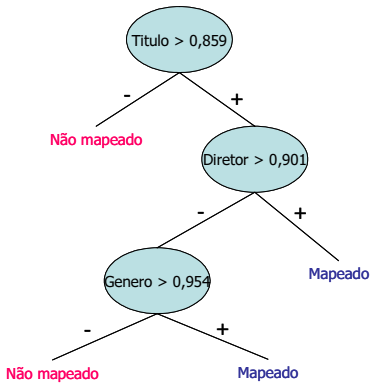


Figura 2.1: Árvore de decisão usada para determinar o casamento de objetos

A Figura 2.2 apresenta uma visão geral do funcionamento do Active Atlas. Para gerar os escores de similaridade, o gerador de candidatos compara todos os atributos existentes em ambos os objetos com um conjunto de funções de similaridade dependentes do domínio e/ou funções de transformações de strings. A partir daí, o sistema calcula o escore de similaridade para cada um dos atributos do par de objetos e finalmente cria um conjunto de mapeamentos candidatos, cada um com seu conjunto de escores de similaridade de atributos mais o escore total correspondente.

Nesta fase de aprendizado é que são definidas as regras de mapeamento e o peso de transformação das strings.

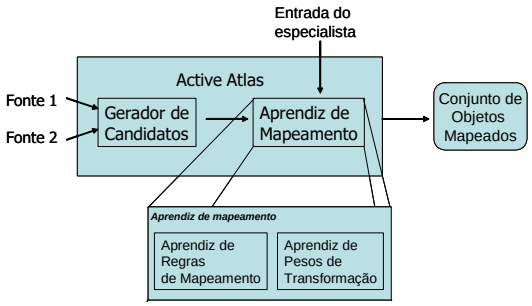


Figura 2.2: Visão geral do processo do Active Atlas

O aprendiz de regras de mapeamento é responsável por determinar quais atributos, ou conjunto de atributos, são mais importantes para o mapeamento de objetos. A proposta de aprendizado de regras de mapeamento é alcançar a exatidão mais alta possível de mapeamento de objetos sobre os vários domínios de aplicação. Na proposta do Active Atlas, o sistema escolhe ativamente os candidatos a mapeamentos mais informativos (exemplos de treinamento) para o especializa classificar como combinados ou não.

O protótipo do Active Atlas possui um método de aprendizado ativo para o aprendizado de "regras de transformação" e "regras de mapeamento". Na proposta, um especialista é requisitado em casos onde é preciso verificar o mapeamento inicial de alguns pares, ou seja, para indicar se os pares foram combinados corretamente ou não. A partir daí o sistema tenta aprender regras novas, e seleciona pares adicionais para que o especialista verifique novamente. O sistema seleciona os pares que ele considera como os "mais valiosos" para confirmar ou não as hipóteses exploradas pelo algoritmo da aprendizagem. O objetivo é obter mapeamentos para os quais o sistema é altamente confiável e minimizar o tempo que um especialista deveria gastar.

2.1.2 PROM

O sistema PROM (*Profile-Based Object Matching*) (DOAN et al., 2003) implementa uma técnica de casamento aproximado através da exploração de atributos disjuntos, ou seja, atributos que não são comuns a um par de objetos. A principal característica está na exploração destes atributos e na possibilidade de construir e transferir conhecimentos a respeito de casamentos já efetuados para novos casamentos.

O sistema fornece um *framework* estensível dentro do qual são definidos perfis, os quais podem ser acrescidos a qualquer momento do processo. Um perfil contém informações sobre cada classe de objeto em um domínio, e pode ser construído por usuários conhecedores dos dados ou aprendido a partir dos dados no domínio. Os perfis contêm dados sobre o que constitui uma instância típica de um objeto. O principal foco do trabalho é definir, construir e combinar os perfis. Existem dois tipos de perfil: os rígidos e os leves. O perfil rígido (impõe fortes restrições sobre os objetos) é uma restrição que toda e qualquer instância daquele objeto deve cumprir (por exemplo, o ano de revisão de um filme não deve ser menor do que o ano de produção). O perfil leve (impõe restrições mais flexíveis sobre um objeto) é uma restrição que

qualquer instância de um objeto provavelmente vai satisfazer (por exemplo, duas classificações de um mesmo filme na maioria das vezes são muito similares e diferem no máximo em 3 pontos). Os perfis pode ser construídos manualmente por usuários especialistas ou através de alguma técnica de Inteligência Artificial, como Redes Bayesianas (PEARL, 1988).

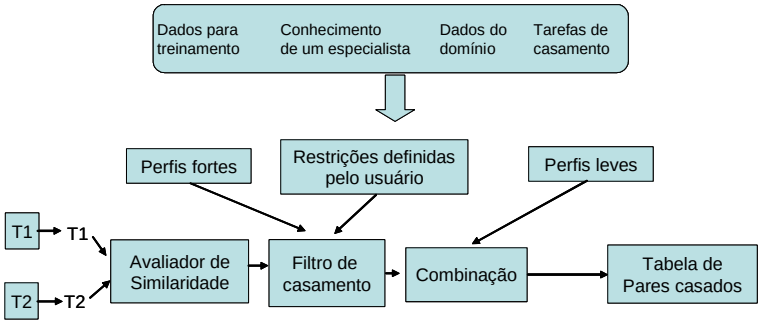


Figura 2.3: O Sistema PROM

O processo executado para determinar casamentos de agregados é apresentado na Figura 2.3. O sistema inicia o processo de combinação usando os atributos compartilhados. Nesta etapa qualquer abordagem de casamento de objetos existente na literatura pode ser aplicada, ou seja, os autores não propõem um novo método para atributos compartilhados. Se o escore de similaridade desta etapa é baixo, então o sistema descarta o par. Este passo é executado pelo módulo Avaliador de Similaridade. Por outro lado, se a similaridade for satisfatória[1], então o processo segue adiante para o módulo Filtro de casamento. Neste passo, para cada par de objetos, são realizadas verificações utilizando módulos que aplicam diferentes perfis. Um perfil pode ser definido como um conjunto de regras que formam conhecimentos específicos que são aplicados a algum objeto. Estes perfis servem para determinar se há relacionamento entre os atributos disjuntos de um par de tuplas.

Os resultados gerados pelos dois perfis são diferentes, o perfil rígido gera um prognóstico boleano, ou seja, "sim" ou "não", enquanto que o o perfil leve gera escores de similaridade. Desta forma, cada um deles é combinado por diferentes módulos. O filtro de casamento combina os resultados do perfil rígido, enquanto que o módulo de combinação utiliza o resultado dos perfis leves para calcular o valor final de escore. O filtro de casamento utiliza o operador AND para combinar os prognósticos gerados pelo perfis rígidos, ou seja, quando a resposta é negativa, o objeto é descartado. O módulo de combinação calcula uma média ponderada dos escores dos escores gerados pelos perfis leves. O peso dado a cada valor é configurado manualmente. A saída do processo é o par de tupla com o valor do escore de similaridade.

Com os escores de similaridade calculados para cada atributo, o componente realiza a junção dos valores para obter um escore total, ou seja, um valor único de similaridade. Com base no valor, é identificado se o par de tuplas é similar ou não. Caso o par de tuplas for equivalente, elas são armazenadas, juntamente com os valores de similaridade na Tabela de pares casados.

[1]O processo de escolha de um valor de limiar para esta etapa não foi descrito no trabalho

2.1.3 Outros trabalhos em integração de dados

A literatura é vasta em trabalhos que tratam a integração de dados. Em (HUFFMAN; STEIER, 1995a,b), os autores descrevem um *framework* para combinação de dados provenientes de diferentes fontes. Neste *framework* é definido um operador de junção heurística (*heuristic join*), o qual possui um conjunto de parâmetros que são usados durante o processamento de integração. O projeto do Citeseer (GILES; BOLLACKER; LAWRENCE, 1998; BOLLACKER; LAWRENCE; GILES, 1998) é um sistema de recuperação de informação que é capaz de localizar artigos automaticamente, extrair citações, identificar citações para o mesmo artigo que ocorrem em formatos diferentes e identificar o contexto das citações no corpo dos artigos.

Em (SCHALLEHN; SATTLER; SAAKE, 2004, 2001) operadores baseados em similaridade para junção e agrupamento de dados são propostos para sistemas de integração de dados. Os autores descrevem a semântica e implementação, bem como as técnicas de otimização utilizadas durante o processamento de integração. Em (HERNANDEZ; STOLFO, 1995) um método chamado *Multi Pass Neighborhood Method* é proposto para detectar o número máximo de registros duplicados exatos e aproximados no menor espaço de tempo. A detecção de registros duplicados é feita através de um conjunto de regras, escritas em linguagem de programação.

2.2 Consultas por similaridade

Na comunidade de banco de dados, algumas abordagens têm sido propostas para trabalhar com casamento aproximado no contexto de linguagens de consultas que suportam operadores que processam a comparação de valores por aproximação (MOTRO, 1988; AGRAWAL; FALOUTSOS; SWAMI, 1993; SHATKAY; ZDONIK, 1996; CHAUDHURI; GRAVANO, 1999; BRUNO; CHAUDHURI; GRAVANO, 2002; BUENO; TRAINA; TRAINA-JR., 2005; BARIONI et al., 2005). Neste cenário, é muito comum encontrar trabalhos que propõem a implementação destes operadores nos processadores de consulta, usando operadores de junção por similaridade (*similarity joins*) (GRAVANO et al., 2003; JIN; LI; MEHROTRA, 2002; GRAVANO et al., 2001(b).

2.2.1 WHIRL

WHIRL (*Word-based Heterogeneous Information Representation Language*) (COHEN, 1998a) é uma linguagem lógica, baseada em álgebra probabilística, proposta com a intenção de ser usada para a integração de bases de dados. O objetivo é usar medidas estatísticas de similaridade entre documentos utilizadas na comunidade de Recuperação de Informação. WHRIL foi construída para acessar relações STIR (*Storing Text In Relations*), em que os atributos de uma relação são fragmentos de texto. Para representação destes fragmentos é usado uma variante do modelo vetorial (SALTON; MCGILL, 1984), ou seja, cada fragmento é representado por um vetor onde cada índice representa os termos mais importantes presentes naquele fragmento, sendo fornecido um escore para cada uma das tuplas de uma relação. Este escore é fornecido pelo usuário e indica o grau de confiança nos dados que estão armazenados naquela tupla, cujo valor varia de 0 a 1. Da mesma forma que em sistemas de recuperação de informação tradicionais, a resposta fornecida ao usuário é um conjunto de tuplas ordenadas

pela "melhor" resposta.

Da mesma forma que em SQL, o usuário necessita conhecer o esquema da base de dados para efetuar uma consulta. Uma consulta WHIRL é construída através de literais, $L_1 \wedge ... \wedge L_n$, onde L é um literal. Existem dois tipos de literal: o literal EDB (*extensional database*) e o literal de similaridade. Um literal EDB é escrito da seguinte forma: $rel(Var_1, ..., Var_2)$, onde "rel" é o nome de uma relação e "Var" é uma variável. Um literal de similaridade é construído da seguinte maneira: $(Var1 \sim Var2)$. Para exemplificar uma consulta construída em WHIRL, supõe-se a seguinte requisição do usuário: "Encontrar sites Web de companhias de telecomunicações":

$\texttt{rel}_1\texttt{(CompCom, empresa)} \wedge \texttt{rel}_2\texttt{(CompCom Inc., site Web)}$
$\wedge \texttt{(CompCom} \sim \texttt{CompCom Inc.)}$

O literal definido em rel_1(CompCom, empresa) expressa que em `rel1` a variável `CompCom` é uma `empresa`, enquanto que o literal $\texttt{rel}_2$`(CompCom Inc., site Web)` expressa através da variável `CompCom Inc.` que esta é um `site Web`. O literal `(CompCom` $\sim$ `CompCom Inc.)`, por sua vez, expressa que `CompCom Inc.` e `CompCom` são similares.

Durante a execução da consulta WHRIL descrita acima, as variáveis `CompCom` e `CompCom Inc.` nos literais EDB (em $\texttt{rel}_1$`(CompCom, empresa)` e $\texttt{rel}_2$`(CompCom Inc., site Web)`) são mapeadas para os vetores dos atributos da relação $\texttt{rel}_1$ e da $\texttt{rel}_2$.

Durante o mapeamento, os escores definidos nas tuplas são usados para as variáveis (`CompCom Inc.` e `CompCom`) caso o valor exista no vetor dos atributos. Já às variáveis `CompCom` e `CompCom Inc.` no literal de similaridade (em `(CompCom` $\sim$ `CompCom Inc.)`) o escore é dado pela similaridade entre os termos de cada vetor das respectivas relações com cada uma das variáveis (`CompCom Inc.` com os vetores de $\texttt{rel}_2$ e `CompCom` com os vetores de $\texttt{rel}_1$). Neste caso, a medida de similaridade não é proposta pelos autores e pode ser definida pelo usuário.

Valores de limiar, que definem o ponto de corte no escore produzido como resultado, são definidos pelo usuário na especificação da consulta. Uma consulta especificada em WHIRL pode pressupor, ainda, que o usuário informe o grau de similaridade aceito.

A linguagem WHIRL é usada em diferentes aplicações que implementem o conceito de similaridade textual (COHEN, 1999, 1998b; COHEN; RICHMAN, 2002). Em (CHINENYANGA; KUSHMERICK, 2002), a linguagem XML-QL é utilizada como interface de consulta, e o sistema possui um processador que traduz as condições de similaridade em subconsultas, que são avaliadas usando WHIRL.

2.2.2 VAGUE

No sistema VAGUE (MOTRO, 1988), o modelo vetorial é refinado para incluir um operador de proximidade. Umas das situações em que o modelo é usado é na definição das respostas a uma consulta. Neste caso, a distância entre as tuplas da resposta e a tupla "ideal" é usada para gerar o *ranking*.

A similaridade entre dois registros é calculada usando funções especificas do domínio de valores de cada atributo, o qual pode possuir uma ou mais funções específicas associadas. De forma geral as consultas são redefinidas através de um operador de similaridade definido como *similar-to*, representado pelo símbolo $\sim$.

Valores de distância individuais, com uso de pesos ou não, de cada atributo são combinados em um único valor de relevância que representa a tupla. Este valor é usado para ordenar o resultado produzido. As distâncias individuais são combinadas através da raiz da soma de seus quadrados ($d_t = \sqrt{(d_{a1})^2 + , ... + (d_{an})^2}$, onde $d()$ é a função de distância e a é o atributo. Basicamente, o autor utiliza a distância Euclidiana (RAGNEMALM, ????) para combinar os valores individuais gerados para cada atributo).

Nesta abordagem é levado em conta que diferentes usuários possuem noções diferentes do que seja um valor de similaridade. Então, cada atributo é associado com um domínio, e para cada domínio uma métrica é associada, bem como para cada um deles, um peso pode ser adicionado, indicando a importância de um determinado atributo na geração do escore final. Além disso, dois parâmetros são associados: um valor de diâmetro e um valor de raio. O diâmetro é o limite máximo entre todas as distâncias entre os valores do domínio; ele é usado para estimar distâncias desconhecidas (se um limite máximo não pode ser fornecido, é usado 0,3). O raio estabelece padrões de vizinhança; dado um valor, ele determina quais os valores mais próximos dele. O raio é também usado para escalar distâncias; dividindo uma distância pelo raio consegue-se uma medida de proximidade que é independente de um domínio particular ou de uma função de similaridade.

O modelo de consulta é dependente da qualidade da função de similaridade usada para cada domínio. O sistema permite ao projetista da base de dados quatro escolhas: 1) usar funções de similaridade pré-definidas embutidas no próprio mecanismo de consulta; 2) fornecer um procedimento para calcular as distâncias entre cada par de elementos de um domínio; 3) fornecer uma relação que armazene a distância entre cada par de elementos; ou iv) usar uma relação de referência, ou seja, uma relação da base de dados que contém as chaves apontando para as distâncias entre os valores dos atributos. As distâncias entre os valores dos atributos são combinadas para gerar distâncias entre tuplas. Cabe ao projetista também definir a fórmula para a combinação das distâncias de cada atributo, utilizando pesos ou não, para definir a ordem em que as tuplas aparecerão no resultado. Portanto, sendo x e y valores de atributos de um mesmo domínio, M a função de similaridade para o domínio e r o raio, o operador de similaridade é definido como $x \sim y$ se $M(x, y) \leq r$. As funções de similaridade são implementadas em forma de procedimentos que envolvem o cálculo da distância e a recuperação dos dados. Tais procedimentos são classificados em funções computacionais, quando as distâncias são derivadas por cálculo de valores dos atributos, sem envolver a recuperação dos valores; tabulares, quando a distância entre pares de valores é pré-computada e armazenada em uma tabela, envolvendo somente a recuperação dos valores; e referenciais, quando a distância representa diferença entre os atributos chave.

Os resultados são ordenados por relevância, e na elaboração da consulta o usuário tem a alternativa de escolher um valor de limiar. Esse valor representa a combinação dos valores de cada domínio; portanto, cada relação possui uma classificação das tuplas baseada nesse valor. No sistema VAGUE o processo de modelagem da base de dados inclui a determinação de funções de similaridade apropriadas para cada domínio e seus respectivos parâmetros, como diâmetro e raio. A definição de funções de similaridade e parâmetros é crítica para processar adequadamente as consultas vagas, e o autor apresenta algumas sugestões de modelagem para o projetista da base de dados.

O sistema possui um interpretador de consultas que determina e resolve qualificações usadas com o operador de similaridade. A estratégia do interpretador é direcionar o usuário para selecionar a função de similaridade, ou conjunto de funções, mais adequada para processar a consulta. Por exemplo, uma consulta sobre filmes, onde o usuário especifica um determinado título de filme, o interpretador solicita intervenção para indicar se a similaridade deve ser: (1) por filmes com títulos parecidos; (2) por filmes com atributos parecidos; (3) por comparação exata. Assim, através de perguntas com intervenção do usuário, o interpretador procura obter dados para processar a consulta e produzir o resultado de forma mais satisfatória.

2.2.3 Outros trabalhos em consulta por similaridade

Um trabalho similar à idéia da linguagem WHIRL foi apresentado em (GRAVANO et al., 2003), onde a proposta é a execução de junções textuais em um SGBDR, sem qualquer modificação no processador de consultas. O operador de junção utiliza a função de similaridade do cosseno (SALTON; MCGILL, 1984). Para executar o operador de junção em dados originados de fontes diferentes, são utilizados Serviços Web (*Web Services*) para extrair e materializar os dados em uma base de dados relacional local. A partir do momento que está materialização foi realizada, os problemas e inconsistências são manipuladas via o operador de junção.

Os trabalhos envolvendo processamento de consultas usando funções de similaridade são vários. Dentre eles pode-se citar extensões para a linguagem SQL (GAO et al., 2004; ILYAS; AREF; ELMAGARMID, 2003; SCHALLEHN; SATTLER; SAAKE, 2004), álgebras que fornecem suporte a similaridade (DEY; SARKAR, 1996; ILYAS; AREF, 2005; AL-KHALIFA; YU; JAGADISH, 2003), operadores por similaridade (GRAVANO et al., 2003, 2001(b). Um estudo completo sobre consultas por similaridade e os problemas envolvidos pode ser encontrado em (STASIU, ????).

2.3 Data Cleaning

O problema foco nos sistemas de *data cleaning* é a detecção e correção de erros em dados vindos de fontes externas e que serão armazenados em um *data warehouse*. Tais dados podem conter inúmeras inconsistências como convenções diferentes, campos ausentes, erros tipográficos, ou de digitação, erros gramaticais, que devem ser resolvidos antes de serem armazenados no *data warehouse*

2.3.1 MARLIN

O sistema MARLIN (*Multiply Adaptative Record Linkage* with Induction) (BILENKO et al., 2003; BILENKO; MOONEY, 2003) utiliza técnicas de aprendizado para detectar duplicação de valores. A abordagem proposta descreve um *framework* para detecção de duplicatas usando funções treináveis para similaridade textual (similaridade de valores atômicos), as quais são aplicadas a cada atributo de uma tupla. Através de um processo de treinamento, estas medidas são capazes de se adaptar a cada domínio de valores de cada atributo. Como parte do *framework*, duas medidas de similaridade para cadeias de caracteres são propostas: 1) uma delas avalia um escore de similaridade entre um par (s_1, s_2) de strings comparando cada uma caracter a caracter; e 2) a outra calcula o escore de similaridade entre um par (s_1, s_2) de strings comparando palavra a palavra. A primeira utiliza o algoritmo EM (*Expectation* Maximization), que estima parâmetros para um modelo

de função baseado em uma distância de edição. A segunda medida utiliza um algoritmo SVM (*Support Vector Machine*) (BOSER; GUYON; VAPNIK, 1992), uma técnica de aprendizado de máquina, para obter uma similaridade estimada baseada no modelo vetorial.

O processo de cálculo de similaridade consiste de dois passos: i) as funções de similaridade usadas para cada atributo da base de dados passam por um processo de treinamento, usando os algoritmos mencionados acima; e ii) as funções de similaridade consideradas "treinadas" são usadas para calcular a distância de cada atributo em um par de tuplas (t_1, t_2), que são posteriormente usadas por um "classificador binário" que identifica cada par de tuplas como "duplicada" ou "não duplicada".

A similaridade entre tuplas compostas por múltiplos atributos é calculada através da combinação das similaridades estimadas nos atributos individuais. Durante o processo de combinação, são atribuídos pesos aos atributos de acordo com sua contribuição à real similaridade entre as tuplas. Dado uma base de dados que contenha tuplas compostas por k atributos e um conjunto $D = \{d_1, ..., d_m\}$ de funções de distância, um par de tuplas é representado por um vetor mk-dimensional. Cada componente do vetor representa a similaridade entre dois valores de atributos das tuplas que é calculada usando uma das m funções de distâncias.

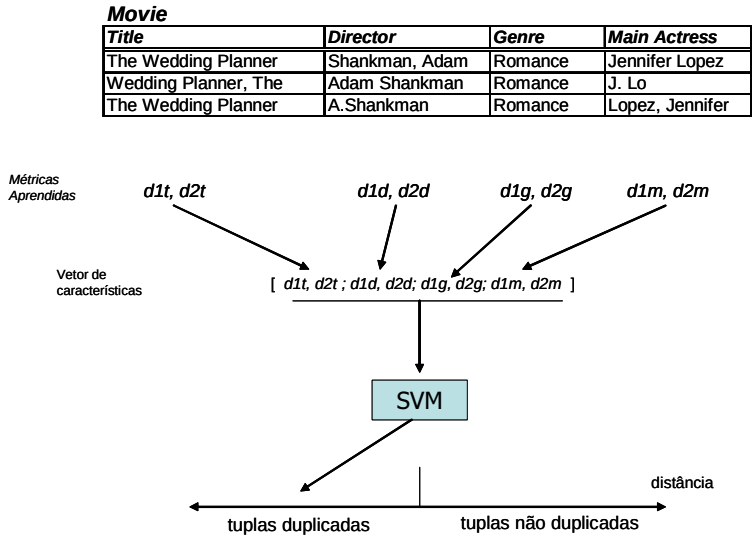


Figura 2.4: Passos para cálculo da similaridade entre tuplas no Sistema MARLIN

Pares casados de tuplas duplicadas podem ser usadaos para construir um conjunto de treinamento, e são rotulados como pares "positivos". Pares não rotulados como positivos de forma implícita formam o conjunto de exemplos "negativos". Depois, um classificador binário é treinado usando estes pares para discriminar entre pares de tuplas correspondentes a duplicatas e não-duplicatas. A Figura 2.4 ilustra o processo de cálculo de similaridade entre tuplas usando duas funções de similaridade para cada atributo e um classificador binário para categorizar o vetor de característica resultante como pertencente a classe de "duplicatas" ou de "não duplicatas", resultando uma distância estimada. Para cada atributo da base de dados, duas funções de similaridade d_1 e d_2 são treinadas e usadas para calcular a similaridade daquele atributo. Os valores calculados por estas funções formam o vetor de características que é então classificado por um algoritmo SVM, produzindo um valor de confiança que representa a similaridade entre as tuplas da base de dados.

A fase de detecção de duplicatas inicia com a geração de pares de duplicatas em potencial, ou seja, aquele pares que de fato são duplicatas. O sistema MARLIN utiliza um método de *clustering*, o método *canopies clustering* (MCCALLUM; NIGAM; UNGAR, 2000) usando a bem conhecida função de similaridade *Jaccard* para separar tuplas em grupos de duplicatas. Pares de tuplas que se enquadram nos *clusters* tornam-se candidatos para uma comparação de similaridade completa.

Diversas estratégias alternativas ao uso dos algoritmos SVM, para a geração de escores para tuplas, foram testadas em (BILENKO et al., 2003). O aprendizado para as funções de similaridade textuais é feito a partir de pequenas amostras de exemplos rotulados, cada amostra relacionada a um domínio diferente. Os autores avaliam algumas funções de similaridade usando um conjunto de problemas de casamento de registros (o chamado *recording-matching* (WINKLER, 1999)). Diferentes experimentos são testados a fim de se identificar a melhor solução. Se um registro possui múltiplos campos, ele é transformado em uma string simples. Entretanto, em alguns casos algumas funções retornam resultados não satisfatórios, pois os campos pequenos recebem o mesmo peso dos campos maiores (por exemplo, o campo `sexo` tem o mesmo peso do campo de nome de uma pessoa). Para solucionar este problema, os campos são novamente segmentados em suas estruturas originais. Em um primeiro experimento, apenas a média entre os valores de similaridade é executado. Em um outro experimento, cada campo do registro é representado através de um vetor de características e treinado com um classificador binário SVM usando este vetor. A distância métrica é medida pelo grau de confiança do classificador aprendido.

Um dos principais focos do trabalho é a avaliação de performance usando estas varias alternativas, sobre o conjunto de dados do Censo Norte-Americano. Os autores assumem que o método de aprendizagem é limitado pelo fato de que não se pode modificar as medidas de similaridade que cada campo usa, e que portanto funções estatísticas podem falhar na estimativa de um escore de similaridade satisfatório. Assim, eles propõem uma versão adaptativa da distância de *Levenshtein* (ou distância de edição como ela é popularmente conhecida), a qual se adapta a tipos particulares de dados do tipo string. Dado um conjunto de pares de strings treinados, a similaridade entre duas strings é a probabilidade de gerar o mais provável alinhamento entre eles (quanto mais alta a probabilidade maior é a similaridade). Esta versão adaptativa da distância de edição é similar à proposta em (RISTAD; YIANILOS, 1998), onde se propõe um modelo que é usado para aprender uma função de distância de edição a partir de um conjunto de dados exemplo.

2.3.2 Combinando *rankings* individuais

Em (GUHA et al., 2004), a proposta é gerar *rankings* individuais para cada atributo presente em um objeto dado como consulta e depois combiná-los a fim de obter o *ranking* global (FAGIN; KUMAR; SIVAKUMAR, 2003). A idéia é encontrar um *ranking* global na qual a distância entre os individuais possa ser minimizada. A noção de distância entre os *rankings* é baseada no *Spearman Coefficient* (SPEARMAN, 1906).

A Figura 2.5 apresenta um exemplo, em que *rankings* individuais são gerados para cada atributo de uma relação R, de acordo com uma consulta C. A tupla t_5 por exemplo, está em segundo lugar no *ranking* para o atributo local, mas está mal colocada nos outros *rankings*. A idéia neste caso, é fundir as posições e/ou escores

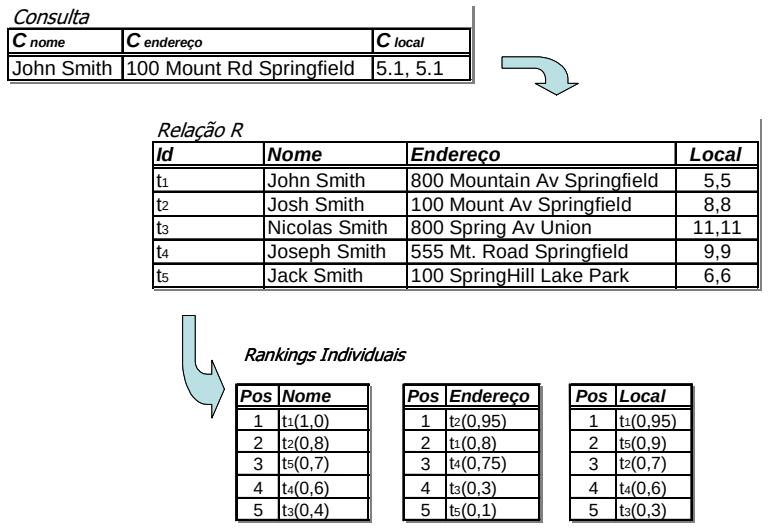


Figura 2.5: Exemplo de *rankings* individuais

de uma tupla em todos os *rankings*. Ou seja, fundir os escores de cada atributo da tupla t_5 considerando também suas posições em cada um dos *rankings*. Para isso, os autores propõem o uso de funções de fusão (*merging function*).

Para fundir os resultados de escores de similaridade e de posições no *rankings*, os autores consideraram o problema de seleção das *top-k* (CHAUDHURI; GRAVANO, 1999) respostas mais relevantes. O objetivo é derivar um *ranking* final "ótimo" de tamanho k, definido pelo usuário.

Os autores propõem a adaptação de três algoritmos para o problema de seleção de *top-k*: i) *HA - Hungarian Algorithm* (AHUJA; MAGNANTI; ORLIN, 1993); ii) *MHA - Modified Hungarian Algorithm*; e iii) *SSP - Successive Shortest Paths*. O primeiro deles, *Hungarian Algorithm* é um algoritmo existente na literatura originalmente usado para resolver o problema de casamento aproximado perfeito a custo mínimo, que é adaptado para obter uma solução ao problema de seleção de *top-k*. O segundo algoritmo, o *MHA - Modified Hungarian Algorithm*, é uma modificação do primeiro visando a melhoria de performance. O último algoritmo, o *SSP - Successive Shortest Paths* é uma nova solução especificamente projetada para o problema de seleção de *top-k*. A intuição por trás deste algoritmo é representar o problema de seleção de *top-k* como uma instância do problema de casamento aproximado perfeito a custo mínimo, construindo um grafo bipartido a partir das tuplas de uma base de dados. Usando os três algoritmos para para a criação de *rankings* individuais para cada atributo de uma tupla, um *ranking* global é sintetizado usando uma função de fusão, chamada de *footrule distance*.

Um exemplo da aplicação do algoritmo proposto pode ser ilustrado com a utilização da Figura 2.6. A Figura apresenta, hipoteticamente, dois *rankings* representados pelos pontos, cujas posições são apresentadas pelos números. Assim, considerando como exemplo uma base de dados consistindo de 5 tuplas, se constrói um grafo bipartido G completo. Um custo de *ranking* $\sigma^r(t_i, j)$ é calculado para a criação de cada arco entre um nodo t_1 e uma posição j. O objetivo é, tendo construído uma resposta $top - i$, deseja-se estendê-la e construir uma resposta $top - (i + 1)$. Como a construção de uma resposta $top - 1$ é fácil, o arco com mínimo custo de *ranking* pertencente a posição 1 é selecionado. A partir daí, através de algum mecanismo de

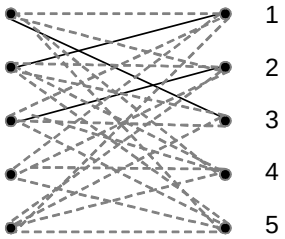


Figura 2.6: Exemplo de combinação de *rankings*

definição de *top-k*, a resposta *top-i* (por exemplo, i=3) é identificada.

Diferentemente de nossa proposta, este trabalho visa a performance do algoritmo de geração de *ranking*, que é implementado em um processador de consulta SQL. Nenhum experimento foi efetuado para avaliar a qualidade dos *rankings* gerados em relação a uma consulta.

2.3.3 Outros trabalhos em *data cleaning*

Em (CHAUDHURI et al., 2003) é proposto um algoritmo *fuzzy* para a tarefa de *data cleaning* em *data warehousing*. O foco do trabalho está no desenvolvimento de um algoritmo que estende a distância Levenshtein a fim de que possa ser aplicada a um registro, e em testes de performance. A proposta apresentada em (GALHARDAS et al., 2000, 2001) sugere o uso de um operador de casamento (*matching operator*) que basicamente calcula o valor de distância entre cada par de tuplas, de um produto cartesiano, usando uma função de distância arbitrária.

2.4 Outras propostas

Outros trabalhos focam a comparação de agregados que possuem atributos com conflitos de valores, criando neste caso, regras para resolução de conflitos (KIM et al., 1995; LIM; SRIVASTAVA; SHEKHAR, 1994; LIM et al., 1996; LIM; SRIVASTAVA; SHEKHAR, 1996; WANG; MADNICK, 1989; CALVANESE; GIACOMO; ROSATI, 1999). Estas regras não avaliam os valores de atributos por suas similaridades textuais, e sim pelo "conhecimento" que elas contêm sobre como comparar atributos quando existe um conflito. Por exemplo, um objeto de uma fonte pode possuir um atributo idade que possui como valor a idade de uma pessoa, e um objeto de outra fonte pode ter o atributo datNasc indicando a data de nascimento de alguém. Se existir uma regra de resolução de conflito que possa efetuar a conversão de datNasc para idade, então os valores podem ser comparados. Este tipo de resolução de conflitos é muito útil em métodos de casamento de objetos, como o proposto nesta tese, mas não é o foco do presente trabalho.

2.5 Comparação entre os trabalhos

As Tabelas 2.1 e 2.2 apresentam uma análise resumida das propostas apresentadas neste capítulo. As características foram divididas em duas tabelas distintas por questões de espaço e melhor visualização.

Na Tabela 2.1, as características consideradas são descritas a seguir.

Processo de combinação. Indica como a combinação dos valores de escores gerados para os atributos de um objeto é executada. Como discutido inicialmente, alguns trabalhos propõem o uso de alguma métrica de similaridade, enquanto que outros propõem métodos que utilizam alguma técnica de aprendizado de máquina.

Função para os atributos. Mostra qual a função (ou conjunto de funções) de similaridade é usada para comparar os atributos de um objeto. Pode-se observar, na relação dos trabalhos comparados, casos em que várias funções são usadas para um mesmo atributo, casos em que as funções são adaptáveis aos valores de domínios, entre outras características.

Área de Aplicação. Indica em que contexto o trabalho foi desenvolvido. Este ponto trata as três grandes áreas descritas anteriormente: integração de dados, *data cleaning* e consultas por similaridade.

Atributos usados. Este ponto indica quais os atributos utilizados durante o casamento de objetos, ou seja, quais os atributos são comparados durante o processo. Nos trabalhos relacionados, podem haver duas alternativas: atributos comuns aos dois objetos comparados ou atributos comuns e disjuntos.

Na Tabela 2.2 as características descritas são descritas a seguir.

Termo usado para "processo de casamento". Indica qual a terminologia para indicar a estratégia usada para solucionar o problema de "identificar várias representações do mesmo objeto". Como descrito inicialmente neste capítulo, a nomenclatura é variada e apresenta várias alternativas: *record linkage*, *merge/purge problem*, *duplicate detection*, *vague queries*, *hardening soft databases*, *reference matching*, *entity name matching*, *object matching*, *object identification*, *instance matching*, entre outras.

Termo usado para designar um objeto. Descreve o termo utilizado para indicar um objeto complexo. Da mesma forma que o item anterior, a nomenclatura para designar um objeto do mundo real é variada, tupla, registro, objeto, agregado, etc.

Importância dos atributos. Apresenta as alternativas utilizadas pelos trabalhos para indicar a importância de cada atributo no processo de casamento de objetos. Alguns trabalhos apresentam a proposta de indicação de pesos, outros definem valores de limiar que indicam que, quando um escore não atinge o valor de limiar definido, o par de objetos não pode ser casado.

Objetivo. Mostra qual o objetivo final de cada trabalho. A descrição desta característica foi baseada na descrição dos experimentos executados em cada um dos trabalhos, usados para demonstrar a efetividade das propostas criadas.

Tabela 2.1: Tabela comparativa

Trabalho	Processo de combinação	Função para os atributos	Área de Aplicação	Atributos usados
(BILENKO et al., 2003)	SVM	versão adaptável da distância de edição	integração de dados	comuns
MARLIN	SVM	funções aprendidas específicas do domínio de valores	*data cleaning*	comuns
WHIRL	variante do modelo vetorial	definida pelo usuário	consulta por similaridade	comuns
Active Atlas	árvores de decisão	transformações e funções específicas do domínio de valores	integração de dados	comuns
VAGUE	média ponderada	distância Euclidiana	consulta por similaridade	comuns
PROM	média ponderada dos escores gerados pelos perfis	NA	integração de dados	comuns e disjuntos
(GUHA et al., 2004)	combinação de *rankings*	NA	*data cleaning*	comuns

Tabela 2.2: Tabela comparativa (*continuação*)

Trabalho	Termo usado para "processo de casamento"	Termo usado para designar um objeto	Importância dos atributos	Objetivo
(BILENKO et al., 2003)	*name matching*	tupla	NA	qualidade do casamento
MARLIN	detecção de duplicatas	tupla	uso de um valor de limiar	qualidade no casamento
WHIRL	consulta vaga	tupla	uso de pesos definidos pelo usuário	performance no processamento de consulta
Active Atlas	*object identification*	objeto	uso de um valor de limiar	qualidade no casamento
VAGUE	*data matching*	registro	uso de pesos	performance no processamento de consulta
PROM	*object matching*	objeto	uso de um valor de limiar	qualidade no casamento
(GUHA et al., 2004)	Combinação de *rankings*	tuplas	k valores mais importantes	performance

2.6 Conclusões

Três problemas em aberto podem ser identificados nos trabalhos descritos neste Capítulo: (1) as propostas não levam em consideração coleção de valores; (2) os valores gerados por uma função de similaridade não possuem um significado claro ao usuário; e (3) estes valores não são comparáveis entre si.

O tratamento de coleções em casamento aproximado, como já discutido no Capítulo 1, é essencial quando existe a manipulação de listas ou conjuntos de dados. Cada um dos casos possui características importantes, e distintas, que devem ser levadas em conta no processo de combinação dos escores de seus componentes.

Entender o significado dos escores gerados pelas funções de similaridade é crucial para permitir que os usuários possam definir pontos de corte para os resultados esperados nos processos de casamento ou consulta aproximados. Em geral, apenas aqueles usuários que têm conhecimento prévio sobre a função de similaridade adotada pelo sistema estariam aptos a entender o significado dos escores resultantes e tirar vantagens deste conhecimento na formulação de consultas e execução de casamento em processos de integração.

Além disso, os escores gerados por cada uma das funções em um casamento de agregados não são comparáveis. A Figura 1.1, apresentada no Capítulo 1, descreve três gráficos contendo a relação de escore versus a precisão obtida pela execução de muitos casamentos usando três diferentes funções de similaridade sobre atributos de uma base de dados contendo informações sobre músicas. Pode-se observar que a precisão em um dado escore (por exemplo, $0,5$) para o atributo Artist é completamente

diferente da precisão no mesmo escore para o atributo Song.

O fato que diferentes funções de similaridade têm diferentes distribuições de valores de escores leva a problemas durante o casamento de agregados, quando os resultados de diferentes funções de similaridade aplicadas aos atributos devem ser combinadas, de alguma forma. Este problema tem sido investigado em alguns trabalhos (GUHA et al., 2004; TEJADA; KNOBLOCK; MINTON, 2001). Soluções tais como normalização dos escores de similaridade em um intervalo predefinido de valores têm sido propostos. No entanto, apesar de tais normalizações ajudarem de alguma forma, os valores de escores resultantes das funções de similaridade continuam tendo significados distintos e não intuitivos aos usuários.

3 ESCORE AJUSTADO

Neste capítulo, é apresentada a descrição detalhada da proposta de mapeamento de escores de similaridade gerados por diferentes funções para um único escore gerado por uma única medida. A idéia consiste em obter, para os valores de similaridade resultantes de cada função de similaridade específica, novos valores de escores que sejam significativos para usuários e que sejam comparáveis entre si. O objetivo é mostrar que existe uma correspondência entre os escores originais gerados pelas funções de similaridade e o novo escore. Na proposta apresentada, esta correspondência é dada pela estimativa da qualidade dos escores fornecidos por cada função de similaridade. A estimativa de qualidade é feita medindo-se a precisão dos resultados fornecidos por uma função. O uso da precisão, como uma medida para a avaliação da qualidade dos resultados de recuperação de dados das funções de similaridade e métodos de busca inexatos, é amplamente aceito e utilizado na comunidade de Recuperação de Informação (RI) (BAEZA-YATES; RIBEIRO-NETO, 1999).

O raciocínio por trás da idéia é que, com o resultado conseguido no processo de estimativa de qualidade é possível quantificar a probabilidade de dois valores de atributo serem representações diferentes do mesmo objeto do mundo real. Além disso, o novo escore pode ser usado como uma medida que não é apenas única para qualquer tipo de atributo comparado, mas que também possibilita que o escore de similaridade seja significativo ao usuário. Com a proposta do novo escore, é possível que ele seja especificado em uma consulta como um parâmetro que define o grau de proximidade permitido em um processo de casamento aproximado. Além disso, usando o novo escore, tal parâmetro terá sempre o mesmo significado, independentemente de qual função de similaridade está sendo usada.

Nas seções abaixo, são apresentados os detalhes de como calcular o novo escore, chamado aqui de ASSC (*Adjusted Similarity Score* - Escore de Similaridade Ajustado). Como mencionado, o cálculo é baseado na estimativa de qualidade dos resultados, medido aqui através da precisão estimada, obtidos com o escore gerado pelas funções de similaridade.

3.1 Processo de Estimativa de Qualidade

De forma geral, o processo de estimativa de qualidade de uma função de similaridade, ou treinamento[1], é efetuado a fim de se identificar a relação entre os escores originais gerados pelas funções de similaridade e o novo escore, o ASSC. Conside-

[1]Os termos "estimativa" e "treinamento" serão usados no decorrer do texto como termos sinônimos.

rando uma função f que avalia a similaridade entre dois valores a_1 e a_2 de um dado atributo A, tal que $f(a_1, a_2) \mapsto g$ e $0 \leq g \leq 1$, qualquer valor retornado por f é chamado aqui de *escore original*. Este raciocínio é formalizado na Definição 1.

Citation

Title	***Journal***	***Publisher***
The state of the art in distributed query processing	ACM Comput. Surv. (CSUR)	ACM Press
The state of the art in distributed query processing	ACM Computing Surveys	ACM
...	...	...
Relational expressive power of constraint query languages	Journal of the ACM (JACM)	ACM
Relational expressive power of constraint query languages	J. ACM	ACM Press

Movie

Title	***Direction***	***Genre***
Charlie's Angels: Full Throttle	Joseph McGinty Nichol	Action
Charlie's Angels II	McGinty J.	Action, comedy
Charlie's Angels II: Full Throttle		Action, comedy

Music

Song	***Artist***	***Album***
Young Hearts Run Free	Candi Staton	Young Hearts Run Free
Young Hearts Run Free	Candi Staton	Young Hearts Run Free: The Best Of Candi Staton
Young Hearts Run Free	Staton	Young Hearts Run Free

PosComp

Instituição	***Cidade***	***Data***
Universidade Federal do Paraná (UFPA)	Belém	20/nov/2003
UFPA - Universidade Federal do Paraná	Belem	20/11/2003
CGEC	Belmé	

Figura 3.1: Domínios, atributos e valores usados nos exemplo

***Definição* 1** *(Escore Original) Seja D_A um domínio, uma função de similaridade $f : D_A \times D_A \longrightarrow \mathcal{R}_S \subseteq [0, 1]$ avalia a similaridade entre um par de valores de D_A e retorna um valor entre $[0, 1]$.*

A função f é chamada de função de similaridade original *ou simplesmente* função de similaridade *e qualquer valor $g \in \mathcal{R}_S$ é chamado de* escore original.

Intuitivamente, pode-se dizer que a função f estima a probabilidade de dois valores comparados serem semanticamente iguais, ou em outras palavras, a convicção de que os dois valores, a_1 e a_2, são representações distintas do mesmo objeto do mundo real. Por exemplo, tendo $a_1 = \{Universidade\ Federal\ do\ Rio\ Grande\ do\ Sul\}$ e $a_2 = \{UFRGS\}$ como dois valores distintos que representam o mesmo objeto do mundo real, a Universidade Federal do estado do Rio Grande do Sul, a função $f(a_1, a_2)$ deve retornar a probabilidade dos dois valores , a_1 e a_2, representarem o mesmo objeto.

Assumindo que f seja consistente (ou seja, gera escores no intervalo de $[0, ..., 1]$), pode-se afirmar que quanto mais próximo de 1 for o escore original mais similares são os dois valores comparados. Por outro lado, quanto mais próximo de 0 menos similares eles são entre si. Idealmente, escores originais maiores do que 0 deveriam ser associados apenas a valores que representem o mesmo objeto do mundo real. Entretanto, funções de similaridade são imprecisas, não havendo forma de afirmar o quão perto de 1 o escore original deve ser para que dois valores sejam considerados similares, ou, representação do mesmo objeto do mundo real. Esta questão está relacionada com a *qualidade* que uma função de similaridade identifica diferentes representações de um mesmo objeto em um domínio de valores.

Durante o processo de estimativa de qualidade, cada função de similaridade é avaliada sobre um determinado conjunto de dados. É importante esclarecer que a

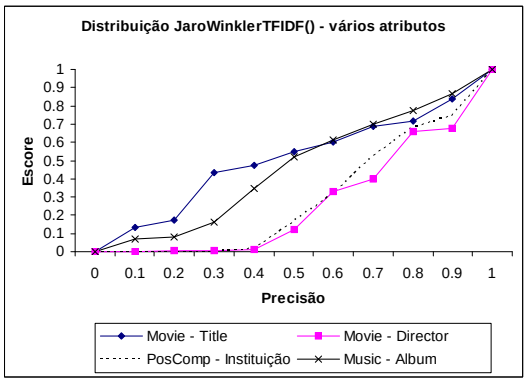


Figura 3.2: Distribuição da função *JaroWinklerTFIDF()* em diferentes atributos

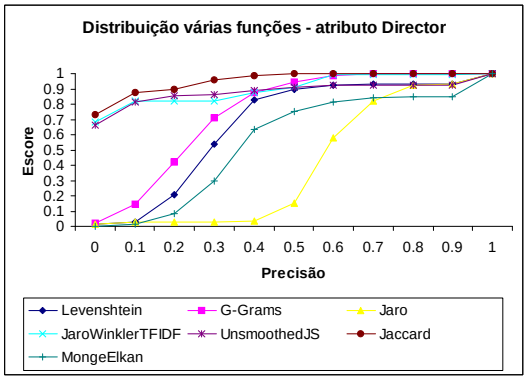


Figura 3.3: Distribuições de diferentes funções de similaridade sobre o mesmo conjunto de valores do atributo *Director*

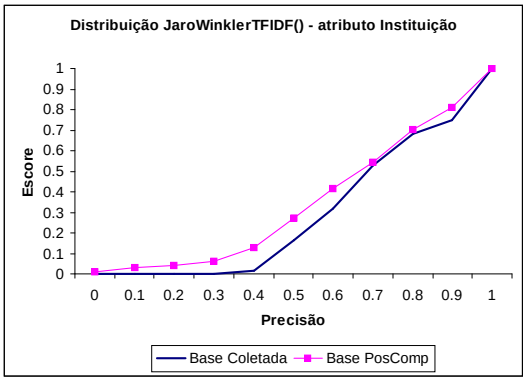


Figura 3.4: Distribuição da função *JaroWinklerTFIDF()* no atributo Instituição sobre dois diferentes conjuntos de dados

qualidade de uma função de similaridade é estimada sobre um conjunto específico de valores de um atributo e que esta estimativa é invalidada caso a função seja usada sobre outro conjunto de valores. Isto é uma importante questão a ser considerada pois a distribuição dos escores de similaridade de uma função em um atributo é diretamente afetada pelos valores deste atributo. Isso porque uma função de similaridade possui diferentes comportamentos dependendo do conjunto de dados em que é avaliada, mesmo se tratando de um conjunto diferente de valores que se refira ao mesmo atributo. Para ilustrar estes comportamentos distintos, alguns exemplos[2] são introduzidos. A Figura 3.1 apresenta domínios[3] de aplicação, os respectivos atributos de cada um, bem como exemplos de valores para cada um dos atributos. Estes valores serão utilizados em exemplos no decorrer do trabalho.

As Figuras 3.2, 3.4 e 3.3 apresentam alguns casos onde se pode visualizar diferentes comportamentos. Por exemplo, a Figura 3.2 mostra os resultados de vários

[2]Todos os exemplos ilustrados neste trabalho são extraídos de fontes reais. Por este motivo, o idioma original de cada um deles será preservado

[3]Os domínios descritos são usados na fase experimental do trabalho e são detalhadamente explicados no Capítulo 5

processos de estimativa de qualidade da função de similaridade *JaroWinklerTFIDF()* sobre diferentes conjuntos de valores em diferentes atributos. O gráfico mostra o caso em que a mesma função apresenta distribuições de escores de similaridade diferentes dependendo do domínio de valores onde é avaliada. Os resultados apresentados nesta figura se referem aos atributos Title e Director do domínio Movie, Album do domínio Music e Instituição do domínio PosComp. A Figura 3.3 apresenta distribuições de diferentes funções de similaridade sobre o mesmo conjunto de valores do atributo *Director*, mostrando que diferentes funções de similaridade possuem distribuições diferentes mesmo quando aplicadas ao mesmo atributo no mesmo conjunto de valores. Já a Figura 3.4 descreve a distribuição da mesma função de similaridade, *JaroWinklerTFIDF()*, em um mesmo atributo, Instituição, mas sobre diferentes conjuntos de dados. A linha contínua, sem marcadores apresenta dados coletados em páginas Web, enquanto que a linha com marcadores representa dados da base do domínio PosComp. Neste caso, pode-se observar que em um mesmo domínio, mas com uma pequena diferença no conjunto de valores, a distribuição dos escores de similaridade também é diferente.

De forma geral, as curvas apresentadas nas figuras representam a relação do escore original gerado pelas funções de similaridade com a qualidade dos resultados, representada pela precisão. Para que esta relação possa ser especificada, é necessário que se crie um *ranking* dos resultados gerados por uma função de similaridade. Assim, considerando um valor q de um dado atributo, é possível ordenar um conjunto de valores deste mesmo atributo de acordo com o escore de similaridade entre q e cada valor do conjunto de valores usando uma função de similaridade.

Definição 2 *(Ordenação Simples) Seja A um atributo cujo domínio é D_A e seja f uma função de similaridade (como apresentado na Definição 1).*

Uma permutação $R(q) = \{a_1, a_2, \ldots, a_n\}$ de A é chamada de ordenação simples *de acordo com $q \in A$ iff $f(q, a_k) \geq f(q, a_{k+1})$ para $k = 1, \ldots, n-1$.*

Com base na ordenação $R(q) = \{a_1, a_2, \ldots, a_n\}$ é possível avaliar a qualidade dos resultados. Para isso, cada par $< q, a_1 >, ..., < q, a_n >$ deve ser identificado previamente como sendo similar/não-similar, a fim de que a estimativa da qualidade seja executada. Isso permite que seja definido o número de valores similares a q no conjunto de valores de A. Este procedimento é amplamente utilizado pela comunidade de Recuperação de Informação para avaliação dos resultados de sistemas de busca, e consiste em usar o conhecimento prévio dos pares identificados como similar/não similar para estabelecer posteriormente no *ranking* $R(q) = \{a_1, a_2, \ldots, a_n\}$ cada valor a_k como "relevante" para aqueles valores considerados previamente como similares (ou seja, "representa o mesmo objeto do mundo real que q representa") ou como "irrelevante" para os identificados previamente como não similares (ou seja, "não representa o mesmo objeto do mundo real que q representa"). Após este processo, a avaliação da qualidade dos resultados da função de similaridade pode ser especificada. Esta avaliação é feita através do cálculo da precisão (BAEZA-YATES; RIBEIRO-NETO, 1999; SALTON; MCGILL, 1984) e pode ser descrita como segue.

Seja $R(q)_i$ o número de ítens no *ranking* $R(q)$ já identificados como "relevantes" até a posição i. A precisão é calculada para cada posição i em $R(q)$ pela equação:

$$p_i = R(q)_i / i \tag{3.1}$$

Por exemplo, considerando um atributo Publisher que contém nomes de Editoras, a Figura 3.5 apresenta uma lista de valores de Publisher ordenados de acordo com a comparação com o valor q ="Morgan Kaufman". Os escores são obtidos através da função de similaridade *MongeElkan()*, implementação da métrica proposta por Monge&Elkan (MONGE; ELKAN, 1996). No *ranking* apresentado na Figura 3.5, a primeira coluna contém os valores do atributo a_i. A segunda coluna contém o escore original $f(q, a_i)$ gerado pela função *MongeElkan()*. A terceira coluna apresenta a avaliação feita por um especialista que classificou cada um dos valores como "relevante" caso o valor tenha sido definido previamente com similar a q e "irrelevante" caso contrário. A quarta coluna contém a posição (i) de cada valor a_i no *ranking*. A quinta coluna apresenta o valor de precisão p_i calculado em cada posição i (Equação 3.1). A última coluna apresenta o número de objetos marcados como "relevante" até a posição i.

Valor	*Escore Original*	*Avaliação*	*Posição*	*Precisão*	*R(q) k*
Morgan Kaufman.	1	relevante	1	1	1
Morgan Kaufman	1	relevante	2	1	2
Morgan Kaufmann Publishers Inc.	0,93333333	relevante	3	1	3
Morgan Kaufmann Publishers	0,93333333	relevante	4	1	4
Morgan Kaufmann	0,93333333	relevante	5	1	5
	...	...	...	...	...
Morgan-Kaufmanm	0,826666666666	relevante	8	1	8
Morgan Kalfmann	0,826666666666	relevante	9	1	9
Pergamon.	0,57777777777	irrelevante	10	0,8	9
	...	...	...	...	9
McMillan .	0,459	irrelevante	14	0,639	9
Kogan Page	0,44	irrelevante	15	0,6	9
Waxmann	0,428571429	irrelevante	16	0,5625	9

Figura 3.5: *Ranking* para comparação de valores do atributo Publisher

Ao se observar a Figura 3.5 pode-se notar que todos os valores acima da posição $i = 9$ são considerados como relevantes, ou seja, considera-se que os valores representam o mesmo objeto do mundo real que q e que foram identificados previamente como similares a q. Desta forma, tem-se precisão $p_i = 1$ para estas nove posições. A partir da posição $i = 10$ em diante, a precisão cai, indicando uma queda na qualidade dos resultados obtidos com a função de similaridade em questão para escores menores ou iguais a $0,577777778$.

Os valores de precisão da Figura 3.5 podem também ser interpretados como uma forma de quantificar a convicção nos escores gerados pela função de similaridade *MongeElkan()* em avaliar a similaridade entre os elementos neste conjunto de dados, dado um valor q. Mais precisamente, se um valor p_i aparece associado a um escore original g_i em uma posição i nesta figura, p_i pode ser interpretado como sendo a probabilidade de um valor ser similar a q se o escore original for considerado maior ou igual a g_i. Considerando por exemplo a posição 9 na Figura 3.5, o escore original é $g_9 = 0,826666667$ e a precisão é $p_9 = 1$. Uma possível interpretação disso é que 100% dos resultados obtidos até a posição $i = 9$ serão relevantes para um escore original $g \geq 0,826666667$. Da mesma forma, considerando a posição 14, com o escore original $g_{14} = 0,459$ e precisão $p_{14} = 0,639$, a possível interpretação seria 63.9% dos resultados obtidos até a posição $i = 14$ serão relevantes para um escore original $g \geq 0,459$. Ainda, se o valor na coluna "Avaliação" for considerado com uma

variável randômica, os valores na coluna de precisão também podem ser considerados como probabilidades associadas a ela.

Pode-se dizer que a precisão é usada como uma indicação da adequabilidade de uma função de similaridade em determinar se dois valores são similares ou não, em um determinado domínio D de valores. Por exemplo, a Figura 3.6, da mesma forma que a Figura 3.5, apresenta uma lista de valores de Publisher ordenados de acordo com a comparação com o valor $q =$"Morgan Kaufman". A diferença entre os dois *rankings* é que a função de similaridade usada para obtenção dos escore na Figura 3.6 é a *NeedlemanWunsch()*, uma implementação da métrica de alinhamento de seqüencia proposta por Needleman&Wunsch (NEEDLEMAN; WUNSCH, 1970). Este *ranking* mostra um exemplo de uma função de similaridade não adequada a este conjunto de valores, e isso pode ser observado através do valor de precisão $p = 0,011568123$ na posição $i = 778$, considerado um valor relevante a q.

Valor	*Escore original (g_i)*	*Avaliação*	*Posição (i)*	*Precisão (p_i)*	*R(q) i*
Morgan Kaufman	1	relevante	1	1	1
Morgan Kaufman.	0,933333337306976	relevante	2	1	2
Morgan Kaufmann	0,933333337306976	relevante	3	1	3
Morgan Kalfmann	0,899999976158142	relevante	4	1	4
Morgan Kaufman.	0,899999976158142	relevante	5	1	5
Morgan-Kaufmanm	0,899999976158142	relevante	6	1	6
Morgan Kaufmann	0,782608687877655	relevante	7	1	7
W.H. Freeman	0,642857134342193	irrelevante	8	0,875	7
...	...	...	...	...	...
Morgan Kaufmann Publishers	0,538461565971374	relevante	67	0,119402985074627	8
The Q.E.D. Monograph Series	0,537037014961242	irrelevante	68	0,117647058823529	8
...	...	...	...	...	...
Morgan Kaufmann Publishers Inc.	0,5	relevante	778	0,0115681233933162	9
Mosby.	0,5	irrelevante	779	0,0115089514066496	9

Figura 3.6: *Ranking* de valores do atributo Publisher, com a função *NeedlemanWunsch()*

Estas considerações podem ser generalizadas a quaisquer outras funções de similaridade que geram valores de similaridade consistentes. Ou seja, dada uma função de similaridade f, um valor q, um atributo A e um *ranking* $R(q)$ sobre A de acordo com q, pode-se construir uma tabela similar àquela apresentada Figura 3.5 que possua os mapeamentos dos escores originais para os valores de precisão.

A tabela da Figura 3.5 foi construída usando um único valor para q. Entretanto, para se ter um melhor entendimento do comportamento de uma função de similaridade aplicada a um dado conjunto de valores de um atributo, um conjunto de valores representativos $Q = \{q_1, q_2, \ldots, q_m\}$ deve ser usado para gerar um *ranking* $R(q_i)$ para cada valor $q_i \in Q$. Neste caso, é apropriado ter uma única tabela que resuma o comportamento da função para todas as consultas. Assim, um simples mapeamento pode ser gerado para expressar a qualidade da função de similaridade considerando todas as consultas em Q.

Esta tabela única, que resume o comportamento do conjunto de consultas $Q = \{q_1, q_2, \ldots, q_m\}$, pode ser conseguido através do cálculo da média dos valores de precisão, em cada cada valor $q_i \in Q$, conseguidos para o mesmo valor de escore original. Por exemplo, a Figura 3.7 apresenta, de forma resumida, três rankings $R(q_1)$, $R(q_2)$ e $R(q_n)$. A qualidade da função de similaridade usada no caso da figura poderia ser alcançada através do cálculo da média dos valores de precisão para cada ponto de escore. Por exemplo, para $g_i = 1$ (como apresentado na Definição 1, $g \in \mathcal{R}_S$ e é chamado de escore original, sendo o valor de escore origi-

$R(q_1)$

Valor	Escore original (g_i)	Posição (i)	Precisão (p_i)
ACM Press (Association of Computing Machinery)	1	1	1
ACM Press	1	2	1
ACM	1	3	1
Apress	0,833333333	4	0,75
UC Press	0,825	5	0,6
CS Press	0,8	6	0,5
ACM Pres.	0,75	7	0,571429
ACC Press	0,7	8	0,5
...	...	...	...
IEEE Press	0,6	17	0,307629

$R(q_2)$

Valor	Escore original (g_i)	Posição (i)	Precisão (p_i)
Addison-Wesley Publishing Company	1	1	1
Addison-Wesley Pub Co	1	2	1
Addison-Wesley	1	3	1
Addison-Wesley Publishing Company, Inc., USA	0,9	4	0,75
...	...	5	0,6
Addison Wesley.	0,8	6	0,5
...	...	7	0,57
West	0,8	8	0,5
...	...	...	...
Addison Wesley Longaman Limited	0,7	17	0,3

...

$R(q_n)$

Valor	Escore original (g_i)	Posição (i)	Precisão (p_i)
Makron Books	1	1	1
TAB books	0,7	2	0,7
SRA	0,7	3	0,5
Hayden Books	0,6	4	0,4

Figura 3.7: Parte de alguns *rankings* usados para conseguir a precisão média

nalmente gerado por uma função de similaridade) teria-se uma precisão média de 1, para $g_i = 0,7$ teria-se uma precisão média aproximada de 0,63333. Entretanto, como pode-se observar na figura, os escores originais podem ser distintos em cada *ranking*. Por exemplo, nos *rankings* $R(q_1)$ e $R(q_2)$ pode-se identificar a existência de $g_i = 0,8$, mas no *ranking* $R(q_n)$ não existe $g_i = 0,8$. Para que a média das precisões sejam calculadas de forma forma correta, todos os valores de escores originais devem existir em todos os ranking de $R = (q_1, ..., q_n)$. Para solucionar este problema, outro procedimento comum da comunidade de Recuperação de Informação é usado (BAEZA-YATES; RIBEIRO-NETO, 1999), a média nos 11 pontos de precisão (*the eleven point precision average*). Primeiro, é definido um conjunto de valores arbitrários de escores originais de similaridade, os quais são dispostos como pontos em uma escala dentro de um intervalo, os 11 pontos: $0,0; 0,1; \ldots, 0,9; 1,0$. Para cada $R(q_i)$, um valor estimado de precisão, a precisão interpolada é calculada em cada um dos 11 pontos, com base nos valores de precisão real obtidos no *ranking*. Este processo pode ser formalizado da seguinte forma:

***Definição* 3** (Precisão interpolada em um ponto de escore original) *Seja $R(q)$ um ranking de acordo com um valor q (como apresentado na Definição 2) e considerando que para cada posição i em $R(q)$ existe um par $\langle g_i, p_i \rangle$, onde g_i representa o escore original em i e p_i representa a precisão nesta mesma posição. Também, considerando que g_{MIN} e g_{MAX} são respectivamente o mínimo e o máximo valores de escores originais $R(q)$.*

Domínio Movie

Valor	Escore Original (g_i)	Posição (i)	Precisão (p_i)
The Green Mile	1	1	1
Green Mile, The	1	2	1
Green mile (The)	1	3	1
The GreenMile	0,9	4	1
The Green Man	0,8	5	0,8
The Man with the Golden Gun	0,6	6	0,8
Man on the Moon	0,6	7	0,7
The Greatest	0,5	8	0,6
Men, The	0,5	9	0,5
The Man in the Iron Mask	0,5	10	0,4
Man in the Iron Mask, The	0,5	11	0,4
The Thin Red Line	0,5	12	0,4
Jewel of the Nile (The)	0,5	13	0,3
...	...		...
Name of the Rose, The	0,5	22	0,3
The Thin Blue Line	0,5	23	0,2
...	...		...
Are We There Yet?	0,5	43	0,2
Wrong man (The)	0,5	44	0,2
Three Amigos	0,5	45	0,2
Third Man, The	0,5	46	0,1
...	...		...

Figura 3.8: *Ranking* para ilustrar o cálculo da precisão interpolada.

Seja $0 \leq \hat{g} \leq 1$ *um valor arbitrariamente predefinido. A* precisão interpolada *para* $\hat{g}$ *é dada pela fórmula:*

$$\dot{p}(\hat{g}, R(q)) = \frac{p_{low} + p_{high}}{2} \tag{3.2}$$

onde:

- p_{low} *é o menor valor de precisão que ocorre com* g_{low} *no ranking* $R(q)$ *e* g_{low} *é o mais alto valor de escore original no ranking* $R(q)$ *tal que* $\hat{g} \geq g_{low} \geq g_{MIN}$*;*

- p_{high} *é o menor valor de precisão que ocorre com* g_{high} *no ranking* $R(q)$ *e* g_{high} *é o mais baixo valor de escore original no ranking* $R(q)$ *tal que* $\hat{g} \leq g_{high} \leq g_{MAX}$*.*

A Definição 3 mostra como calcular a precisão interpolada para um valor arbitrário $0 \leq \hat{g} \leq 1$. A intuição por trás desta idéia é que se $\hat{g}$ é um escore original que já existe no *ranking*, a precisão interpolada possui o mesmo valor da precisão existente no *ranking*. Caso contrário, o valor da precisão interpolada será a média entre os valores de precisão existentes que estejam próximos a $\hat{g}$. É importante observar que podem haver dois ou mais valores de precisão para um mesmo valor de escore original no *ranking*. Neste caso, o valor mais baixo de precisão é escolhido como precisão interpolada.

Para ilustrar como a precisão interpolada deve ser calculada, a Figura 3.8 apresenta um ranking para o atributo Title do domínio Movie. A precisão interpolada para o escore $\hat{g} = 0,6$ é $\dot{p} = 0,7$ pelo fato de que $0,7$ é o mais baixo valor de precisão no *ranking* que está associado ao escore original $0,6$. Já o valor de precisão interpolada para $\hat{g} = 0,7$, que não ocorre no *ranking*, deve ser calculado através da média entre o valor de precisão mais baixo no escore original $0,8$ e $0,6$, que é, $0,8$

e $0,7$. Agora, supondo que seja necessário calcular a precisão interpolada em um escore original $0,4$. Neste caso, como $0,4$ é mais baixo do que o mais baixo escore do *ranking*, que é $0,5$, de acordo com a Definição 3, a precisão no ponto mais baixo de escore é usada, ou seja, $0,1$. As linhas em destaque representam os valores de precisão escolhidos para os valores de escore original existentes no *ranking*.

Finalmente, a média de todos valores de precisão interpolada é calculada para cada um dos 11 pontos predefinidos. No presente trabalho, o termo escore ASSC é usado para referenciar estas médias. Em avaliação de Sistemas de Recuperação de Informação, um procedimento similar é usado para determinar a chamada precisão nos onze pontos (*eleven point average precision* (BAEZA-YATES; RIBEIRO-NETO, 1999)). Mais precisamente, estas médias são calculadas como descrito a seguir.

Com a Equação 3 tem-se todos os valores de precisão para cada um dos 11 pontos de escore original. Depois disso, a média de todos os valores de *precisão interpolada* em todos os *rankings* é calculada para cada um dos 11 pontos predefinidos. A média final em cada um dos 11 pontos de escore original é chamada de ASSC (*Adjusted Similarity Score*). Em sistemas de Recuperação de Informação, um procedimento similar é usado para determinar o chamado *eleven point average precision*, ou precisão aos 11 pontos[4]. A média final em cada um dos 11 pontos de escore original é calculada e especificada em uma tabela, a TaME - (Tabela de Mapeamento de Escores), como descrito abaixo.

***Definição* 4** *(TaME - Tabela de Mapeamento de Escores) Seja $Q = \{q_1, \ldots, q_m\}$, onde Q é um conjunto de valores de um dado atributo A. Seja $R(q_k)$ um ranking original gerado de acordo com $q_k \in Q$ (Definição 2). Considera-se que, para cada posição i de $R(q_k)$, existe um par $\langle g_{i,k}, p_{i,k} \rangle$ que representa o escore original e a precisão naquela posição.*

Uma TaME - Tabela de Mapeamento de Escores *é uma lista de pares $\langle \hat{g}_i, \hat{p}_i \rangle$ onde $\hat{g}_i \in \{0,0; 0,1, \ldots, 0,9; 1,0\}$ e é um valor de escore representativo predefinido arbitrariamente e $\hat{p}_i$ é a média dos valores de precisão interpolada em $\hat{g}_i$ para cada ranking $R(q_k)$, que é,*

$$\hat{p}_i = \frac{\sum_Q \dot{p}(\hat{g}_i, R(q_k))}{|Q|} \tag{3.3}$$

onde $\dot{p}(\hat{g}_i, R(q_k)$ é a precisão estimada para $\hat{g}$ como especificado na Definição 3.

Cada $\hat{p}_i$ é chamado de valor de Adjusted Similarity Score*, ou ASSC.*

A Figura 3.9 apresenta um exemplo de uma TaME construída após a geração de diversos *rankings* sobre a mesma base de dados usada para construir o *ranking* apresentado na Figura 3.5.

A Figura 3.10 apresenta uma visão geral dos passos executados no processo de estimativa do escore ASSC.

Através da tabela TaME é possível definir duas funções, uma que cria mapeamentos entre o escore original e o escore ASSC, e outra que cria mapeamentos entre o escore ASSC e o escore original. As funções são definidas a seguir e serão usadas nas próximas seções.

[4]Em Sistemas de Recuperação este procedimento é usado para a geração de curvas Revocação/Precisão, e a precisão é calculada aos 11 pontos de revocação

TaME - Tabela de Mapeamento de Escores

Escore Original	ASSC
1,0	1,0
0,9	0,758333333
0,8	0,625
0,7	0,455
0,6	0,328333333
0,5	0,21
0,4	0,083333333
0,3	0,023333333
0,2	0,003333333
0,1	0
0,0	0

Figura 3.9: Um exemplo de uma tabela de mapeamento de escores

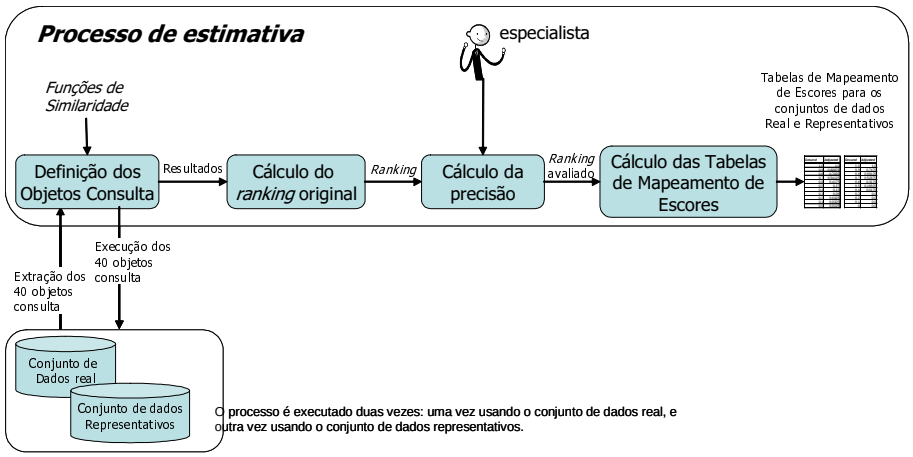


Figura 3.10: Visão geral do processo de estimativa do escore ASSC

***Definição* 5** *(Função de mapeamento do escore original para o escore ASSC) Seja* $M = \{\langle 0, 0, p_0\rangle, \ldots, \langle 1, 0, p_{10}\rangle\}$ *uma TaME para uma função de similaridade* f*, como apresentado na Definição 4, e um valor de* escore original $0 \leq g \leq 1$*.*

A função $\alpha_M(g) \mapsto [0, 1]$ *que mapeia escores originais para escores ASSC é definida como:*

$$\alpha_M(g) = \begin{cases} p_i & \text{if } \langle g, p_i\rangle \text{ ocorre em } M \\ \frac{p_{i-1}+p_{i+1}}{2}, \text{ onde } g_i < g < g_{i+1} & \text{caso contrário} \end{cases} \tag{3.4}$$

De acordo com a Equação 3.4, descrita na Definição 5, um escore original g é diretamente mapeado para um escore ASSC caso g esteja na TaME, por exemplo, o escore original $g = 0,6$ é diretamente mapeado para o escore ASSC correspondente, ou seja, para $0,328333333$. Por outro lado, caso o escore original gerado por uma função de similaridade não esteja representado pelos 11 pontos, como por exemplo $g = 0,65$, o valor do escore ASSC correspondente é $0,3916666665$, ou seja, a média entre os escores ASSC de 0,6 e 0,7.

***Definição* 6** *(Função de mapeamento do escore ASSC para o escore original) Seja* $M = \{\langle 0, 0, p_0\rangle, \ldots, \langle 1, 0, p_{10}\rangle\}$ *uma TaME para uma função* f *como mostrado na Definição 4. Considera-se* $0 < p \leq 1$ *um valor de escore ASSC.*

A função $\gamma_M(p) \mapsto [0, 1]$ *que mapeia escores ajustados para escores originais é definida como:*

$$\langle \gamma_M(p), p_i\rangle \in M, p_{i-1} < p \leq p_i \tag{3.5}$$

Pela Definição 6, o escore ASSC de valor 0,4, da Figura 3.9, seria mapeado para o escore original 0,7, assim como todos os escores ASSC maiores do que 0,328333333 e menores ou igual a 0,455. Observa-se que os valores são mapeados de diferentes formas de acordo com a direção de mapeamento. Quando o mapeamento é feito dos escores originais para os escores ASSC , o mapeamento é dado pela média entre os escores ASSC superior e inferior. Por outro lado, quando o mapeamento é feito dos escores ASSC para os escores originais, o mais alto escore correspondente é usado. As razões para esta assimetria de mapeamentos entre os dois tipos de escores são apresentadas a seguir, nas próximas seções.

3.2 Operadores de Casamento

Diversos trabalhos (GRAVANO et al., 2001; GUHA et al., 2002; MELNIK; GARCIA-MOLINA; RAHM, 2002; GUHA et al., 2003; GRAVANO et al., 2001(b) que envolvem abordagens de casamento aproximado, especificam o uso de um limiar (*threshold*), ou ponto de corte, que serve como um fator descriminante para definir se dois valores são representações do mesmo objeto do mundo real ou não. O método usado em sistemas de consulta por similaridade (CHENG et al., 2004; PAPADOPOULOS; MANOLOPOULOS, 1998; AGRAWAL; FALOUTSOS; SWAMI, 1993; CIACCIA; PATELLA; ZEZULA, 1997; PETRAKIS; FALOUTSOS, 1997), que fazem uso do limiar, é conhecido como *range query*. O limiar, ou ponto de corte, é geralmente interpretado como uma representação da estimativa de qualidade dos resultados esperados, ou seja, ele representa o ponto mínimo de qualidade exigida no resultado. Esta abordagem tem dois principais problemas. Primeiro, na maioria das vezes os valores de limiar não têm significado para o usuário, ou seja, não há uma maneira simples de saber que, para uma determinada função de similaridade, o valor ideal de limiar é x. Isso se deve principalmente ao fato de que estes valores dependem do modelo específico aplicado pela função de similaridade e/ou pela maneira como a função é implementada. Segundo, para cada diferente função, utilizada sobre diferentes conjuntos de dados, deve-se especificar um valor diferente para o limiar, pois cada função possui uma distribuição específica de escores de similaridade sobre diferentes bases, como já apresentado e discutido na Seção 3.1.

Por outro lado, o uso da precisão estimada como um novo escore, apresentado na Seção 3.1, torna fácil pensar no escore como um valor que indica a convicção que a resposta que está sendo retornada tenha no mínimo $x\%$ de valores relevantes. Desta forma, pode-se dizer que o novo valor é usado para quantificar a qualidade dos resultados. Com isso, fornecer o limiar como um valor que estabelece a qualidade desejada quando se está especificando casamento de valores é muito intuitivo para os usuários por dois motivos. Primeiro, os preserva do conhecimento de como cada função de similaridade se comporta em relação a cada conjunto de dados. Segundo, cria uma oportunidade de uma simples medida, a precisão média estimada, ser utilizada para todos os processos de comparação de valores, indiferentemente da função

de similaridade usada. Por exemplo, considerando o *ranking* $R(q_1)$ apresentado na Figura 3.7, e supondo que o usuário especifique, em um sistema de consulta por similaridade, um limiar de $0,57$, o resultado final seria um *ranking* composto pelos valores até a posição $k = 7$. Ao especificar o limiar, o usuário apenas deve ter em mente a porcentagem de valores relevantes que deve aparecer no resultado. No exemplo, indicando o limiar de $0,57$ o usuário estará indicando que deseja obter no resultado 57% de valores relevantes.

Nesta seção, é apresentada a proposta do uso do escore ASSC, definido na Seção 3.1, como um quantificador da qualidade dos resultados requeridos em um casamento aproximado de valores, ou seja, como um limiar. Inicialmente, um operador de casamento aproximado é formalizado, o qual é baseado no escore original, o *GS_ Matcher* (*Ground Score Matcher*). A seguir, a noção deste operador é estendida para um operador de casamento que é baseado no escore ASSC, o *ASSC_ Matcher*. Os operadores são definidos com o propósito de indicar quando dois valores, comparados por uma função de similaridade, são considerados similares, dado um limiar.

Definição 7 *(GS_ Matcher) Seja A um atributo, e a_1 e a_2 valores de A. Considera-se uma função de similaridade f (como especificado na Definição 1) e um valor de limiar baseado no escore original da função, o limiar original gt, onde $0 \leq gt \leq 1$.*

O operador baseado no escore original $GS\_Matcher(a_1, a_2, f, gt) \mapsto \{SIM,NÃO\}$ irá retornar sim significando que dois valores casam se $f(a_1, a_2) \geq gt$ e irá retornar não caso contrário.

Tendo como exemplo o atributo Publisher representado na Figura[5]3.5 e considerando um limiar baseado no valor original da função de similaridade, $gt = 0,9$, tendo a_1 ="Morgan Kaufman", o resultado da aplicação do operador $GS\_Matcher$, $GS_Matcher(a_1, a_2, f, gt)$ será SIM, significando que os values a_1 e a_2 casam. Por outro lado, se for considerado o valor a_2 ="Morgan Kalfmann" (valor da posição 9), o resultado será não, significando que os a_1 e a_2 não casam. O operador $GS\_Matcher$ simplesmente retorna o conjunto de valores considerados como sendo representações do mesmo objeto do mundo real que a_1, desde que o valor de similaridade resultante da função seja maior ou igual ao limiar definido. O problema deste operador é que, como o valor do limiar gt é um valor arbitrário entre 0 e 1 que é dependente da função e dos valores de a_1 e a_2, fica difícil definir o que é considerado um valor aceitável.

No entanto, seguindo as idéias apresentadas previamente, pode-se substituir o limiar original (gt) no operador *GS_ Matcher* por outro parâmetro, pt, chamado de *limiar ajustado*. Desta forma, um novo operador deve ser definido, chamado de *ASSC_ Matcher*.

Definição 8 *(ASSC_ Matcher) Seja A um atributo, e a_1 e a_2 valores de A. Seja f uma função de similaridade, como especificada na Definição 1 e seja $M = \{\langle 0, 0, p_0\rangle, \ldots, \langle 1, 0, p_{10}\rangle\}$ uma TaME para uma função de similaridade f, como apresentada na Definição 4. Seja γ_M uma função que mapeia escores originais para escores ASSC em uma tabela M, como apresentado na Definição 6. Seja $0 < pt \leq 1$ um valor de limiar baseado no escore ASSC, chamado de limiar ajustado.*

[5]Relembrando, a figura representa um *ranking* gerado para um valor a_1 ="Morgan Kaufman" e que os escores são dados pela função de similaridade *MongeElkan()*.

O operador baseado no escore ASSC, $ASSC\_Matcher(a_1, a_2, f, pt) \mapsto \{SIM,NÃO\}$ *é definido por:*
$ASSC\_Matcher(a_1, a_2, f, pt) = GS\_Matcher(a_1, a_2, f, \gamma_M(pt))$.

Informalmente, a Definição 8 especifica que o parâmetro pt, o limiar ajustado, deve ser primeiramente mapeado para o escore original correspondente através da função $\gamma_M(pt)$. Como definido anteriormente na Definição 6, a função $\gamma_M()$ mapeia o escore ASSC para o escore original correspondente de mais alto valor na TaME, que seja correspondente ao escore ASSC em questão. Pela Definição 8 a função $\gamma_M(pt)$ recebe como parâmetro o limiar ajustado pt, que por sua vez deve ser mapeado para o escore original de mais alto valor. Desta forma, o escore original é usado como parâmetro no operador *GS_Matcher*, para de fato funcionar como limiar no processo de comparação dos valores a_1 e a_2.

Para os valores de Publisher, por exemplo, representados na Figura 3.5, tendo $f = MongeElkan()$, se o limiar ajustado for definido como $pt = 0,75$, significa que o usuário espera que no mínimo 75% dos valores retornados no resultado sejam considerados similares a a_1. Na tabela M, apresentada na Figura 3.9, o limiar original correspondente ao limiar ajustado $pt = 0,75$ é $gt = 0,9$ (escore original de mais alto valor). Isso quer dizer que, para obter os 75% de resultados similares a a_1, o limiar original em um processo de casamento por similaridade deveria ser $gt = 0,9$.

Agora torna-se mais fácil a compreensão de porque os mapeamentos entre os escore são assimétricos dependendo da direção, ou seja, quando se mapeiam escores ASSC para escores originais, através da função γ_M, o mais alto escore correspondente é considerado, ao contrário de como é feito no mapeamento do escore original para o escore ASSC, em que se considera a média de dois valores. A idéia básica é que, ao mapear um limiar ajustado pt para um limiar original gt, se quer assegurar que o processo de casamento de valores resulta em, no mínimo, pt% resultados corretos. Para assegurar esta porcentagem de casamentos corretos, deve-se usar o próximo escore original mais algo que corresponde a pt na TaME M. Se for considerado qualquer valor menor do que $\gamma_M(pt)$, corre-se o risco de violar o requisito de fornecer no mínimo pt% resultados corretos para o casamento com esta função.

No Capítulo 5, são apresentados os resultados de experimentos executados com o propósito de confirmar a exatidão dos mapeamentos entre limiares originais e limiares ajustados.

3.3 Conclusões

Neste capítulo, foi apresentado como um escore originalmente gerado por uma função de similaridade pode ser mapeado para um novo escore, baseado na estimativa de precisão da função, chamado de escore ASSC. Usando este novo escore, é possível quantificar a probabilidade de dois valores de dados serem representações diferentes do mesmo objeto do mundo real. Como o escore ASSC é gerado a partir de uma mesma medida, a precisão estimada, pode ser usado como valor em um processo de combinação de escores. Além disso, com o uso do escore ASSC é possível fornecer um valor ao usuário que seja significativo e intuitivo, podendo ser usado com uma valor para ponto de corte.

Pode-se considerar a proposta do uso do escore ASSC como a contribuição mais importante desta tese. O uso do escore ASSC pode ser considerado uma estratégia

genérica para casamento aproximado de valores.

O processo de estimativa tem um custo relativamente aceitável, pois o comportamento das funções de similaridade é avaliada com base em um conjunto bastante limitado de objetos-consulta (40). Como será demonstrado nos experimentos, este pequeno número de consultas de treinamento é suficiente para alcançar bons resultados de estimativa em todos os domínios e conjuntos de dados experimentados, indicando que a abordagem proposta pode ser facilmente aplicada na prática.

Como apresentado neste capítulo, o escore ASSC pode ser utilizado como valor de limiar em qualquer aplicação que utilize casamento aproximado de valores, tendo em mente o mesmo significado: porcentagem de resultados retornados como relevantes. Esta abordagem pode ser usada em sistemas de processamento de consultas por similaridade, em sistemas de integração de dados, *data cleaning*, entre outros.

4 COMBINAÇÃO DE ESCORES

Este capítulo apresenta a abordagem proposta para combinação de escores de similaridade de atributos em objetos complexos. Na abordagem apresentada aqui, utiliza-se o conceito de agregado (formalmente definido a seguir) para designar tais objetos complexos[1]. A idéia se baseia na intuição de que agregados estruturados de diferentes formas devem possuir diferentes maneiras de combinar escores de seus atributos. Basicamente, dados os escores de similaridade dos atributos de um agregado, pretende-se especificar a estratégia mais adequada de combinação, dependendo da estrutura do agregado. Na proposta apresentada, os tipos de estrutura que um agregado pode possuir são classificados, basicamente, como tupla e coleção. Além disso, cada um dos casos possui características importantes, e distintas, que devem ser levadas em conta no processo de combinação dos escores de seus atributos, que são descritas como segue.

Coleção. Diz respeito aos valores que compõem o agregado. Neste caso, é importante distinguir que tipo de coleção está sendo manipulada, lista, considerando que os valores são ordenados, ou conjunto, levando em conta que os valores não são ordenados.

Tupla. Diz respeito às funções de similaridade utilizadas para cada atributo de um agregado. Como uma tupla é geralmente composta por valores que pertencem a diferentes domínios, é natural que se utilize diferentes funções de similaridade aplicadas a cada um deles. Neste caso, como visto no Capítulo 3, cada atributo pode ser comparado por funções de similaridade que possuem diferentes distribuições de escore, sendo portanto importante o tratamento destes escores.

Este capítulo apresenta, de forma geral, três principais contribuições no que diz respeito à combinação de escores de similaridade de atributos agregados:

- uso de diferentes abordagens de combinação de valores de escores de similaridades, que são dependentes da estrutura do agregados (Seção 4.3);
- definição de funções de similaridade específicas para coleção de valores (Seção 4.3);

[1]Optou-se por não utilizar o conceito de objeto pois as estruturas utilizas na presente proposta são mais simples do que aquela utilizada para o conceito de objeto no modelo ODMG (*Object Data Management Group*) (COOPER, 1997)

- uso do escore ASSC como uma estratégia genérica para solução do problema de combinação de escores de similaridade, em casamento de tuplas, gerados por funções que possuem diferentes distribuições (Seção 4.4).

Como visto no Capítulo 2, diferentes metodologias podem ser encontradas na literatura para combinar escores de funções de similaridade quando diferentes atributos devem ser combinados (TEJADA; KNOBLOCK; MINTON, 2001; BILENKO et al., 2003; MOTRO, 1988). No presente trabalho, uma metodologia simples é testada para combinar escores de diferentes atributos de um objeto, a qual é descrita nas próximas seções.

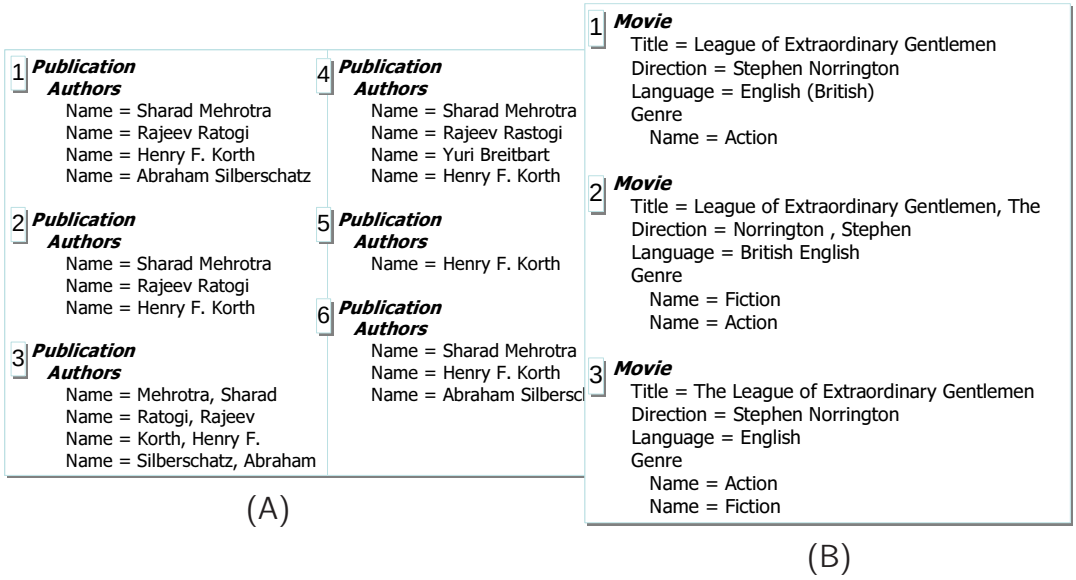


Figura 4.1: Exemplos de agregados

4.1 Estrutura de um agregado

Para explicar os distintos processos de combinação de escore propostos neste trabalho, é importante definir a noção do que é um agregado. Informalmente, um agregado é um objeto composto por um nome e um conjunto de atributos.

***Definição* 9** *(Agregado) Um* **Agregado** $O = (n, A)$ *é uma tupla onde* n *representa o nome do agregado e* $A = \{a_1, ..., a_m\}$ *é o conjunto de atributos daquele agregado, tal que* a_i *é um par* $\langle \eta_i, v_i \rangle$ *onde* η_i *representa o nome do atributo e* v_i *o valor, tal* v_i *pode ser um valor atômico ou, recursivamente, outro* Agregado O*, tal que* $m \geq i \geq 1$.

De acordo com a Definição 9, os atributos de um agregado possuem valores que podem ser do tipo atômico ou do tipo agregado[2]. Um tipo atômico possui um valor primitivo, ou seja, pode ser uma cadeia de caracteres, uma data, um número, etc. Por exemplo, na Figura 4.1 (A) pode-se identificar uma instância do agregado Authors (agregado de número 5) composto por um atributo Name, cujo valor é atômico, ou seja:

[2]É importante observar que, na Definição 9, n representa o nome de um agregado enquanto que η representa o nome de um atributo

```
(Authors, {<Name, "Henry F Korth">})
```

Um atributo do tipo agregado pode possuir uma estrutura do tipo aninhada, como por exemplo, o atributo Genre do agregado Movie, na Figura 4.1 (B). Um agregado O pode ser ainda, uma *tupla* ou uma *coleção*. A definição destas estruturas é formalizada a seguir.

Definição* 10** *(Tupla) Seja $A = \{a_1, ..., a_m\}$ o conjunto de atributos de um agregado O, tal que $1 \leq i \leq m$. Se cada atributo a_i em O possuir um* ***nome *η* ***diferente*** *de todos os demais, então O é chamado de* **Tupla**

Definição* 11** *(Coleção) Seja $A = \{a_1, ..., a_m\}$ o conjunto de atributos de um agregado O, tal que $1 \leq i \leq m$. Se cada atributo a_i em O possuir o* ***mesmo nome *η, então O é chamado de* **Coleção**

Informalmente, um agregado é considerado uma tupla caso todos os atributos a_i tenham nomes η diferentes. Por exemplo, considerando a Figura 4.1 (B), uma das instâncias do agregado Movie pode ser representada da seguinte forma:

```
(Movie,
   {<Title, "League of Extraordinary Gentlemen">
    <Director, "Stephen Norrington">
    <Language, "English (British)">
    <Genre,
        {<Name, "Action">}>
    }
)
```

Por outro lado, se todos os atributos a_i de um agregado possuírem o mesmo nome η, ele é considerado uma coleção. Por exemplo, o agregado Authors, representado na Figura 4.1 (A) possui atributos com o mesmo nome, ou seja, Name, e uma das instâncias de Authors (agregado de número 2) pode ser descrita da seguinte forma:

```
(Authors,
   {<Name, "Sharad Mehrotra">
    <Name, "Rajeev Ratogi">
    <Name, "Henry F Korth">})
```

Uma coleção de dados pode ser considerada uma lista ou um conjunto. A distinção entre lista e conjunto não é uma restrição de integridade, ou seja, não é uma característica do agregado e sim dependente do domínio da realidade modelada ou da interpretação do usuário. Por exemplo, considerando hipoteticamente os dados apresentados na Figura 4.2, pode-se observar dois casos distintos em que listas e conjuntos são usados para representar o domínio da realidade modelada. Os dados em (A) não possuem uma ordem a ser obedecida por se tratar de nomes de participantes de uma atividade, já em (B) a ordem é importante pois trata-se da colocação dos mesmos participantes da atividade. É evidente que tal distinção é dependente do domínio de problema que se está tratando (ou do usuário que está efetuando o casamento). Portanto, a escolha de qual tipo de coleção é aplicável deve ser definida em tempo de execução de comparação dos agregados, ou seja, no momento da escolha da função de similaridade [3].

[3]Esta afirmativa pode ser interpreta como equivocada caso se considere que o tipo de coleção é uma questão de modelagem, sendo portanto definida *a priori*.

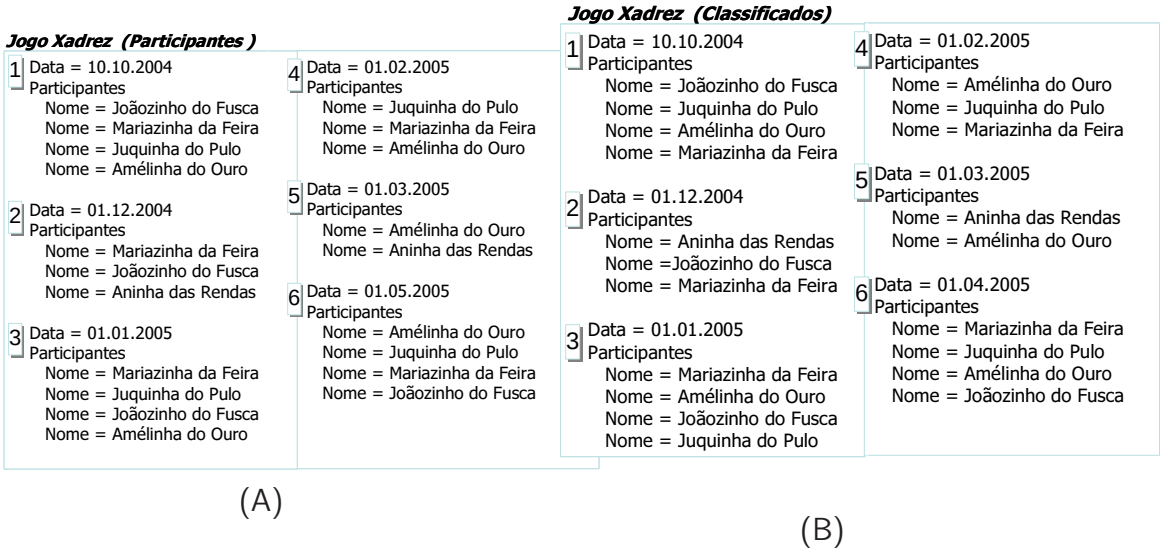


Figura 4.2: Exemplos do uso de listas e conjuntos

4.2 Taxonomia das funções de similaridade

Levando em conta que valores podem ser atômicos ou complexos, dois tipos de funções de similaridade são propostas: FAT (Funções para valores ATtômicos) e FAG (Funções para valores AGregados).

As funções do tipo FATs são dependentes do domínio de valores e podem ser quaisquer funções de similaridade aplicáveis a valores primitivos. Por exemplo, para um atributo Director de um agregado Movie (Figura 4.1 (B)), uma FAT específica para nomes próprios poderia ser usada.

As funções do tipo FAGs são dependentes da estrutura do agregado. Por exemplo, na Figura 4.1 (B), para o agregado Movie, que possui os atributos Title, Director e Genre, uma FAG específica para tuplas deve ser usada. Outro exemplo é representado na Figura 4.1 (A), o atributo Authors, de Publication, é um agregado do tipo coleção que possui vários atributos Name, que corresponde a cada um dos autores de uma publicação. Neste caso, uma FAG para coleções deve ser usada. As FAGs podem ainda ser classificadas em dois tipos, aquelas aplicadas a tuplas de dados e as aplicadas a coleções de dados, as quais são dependentes do tipo de estrutura de um agregado, e que são identificadas como funções dependentes da estrutura. Além disso, para agregados com estruturas de coleções, pode ser aceita a restrição de ordenação, que é dependente da aplicação ou da intenção do usuário. Uma coleção pode ser tratada como uma lista ou como um conjunto de valores, dependendo se a intenção é efetuar comparação de agregados considerando a ordem dos atributos de uma coleção (lista) ou não (conjunto). A Figura 4.3 apresenta a taxonomia utilizada para a classificação das funções de similaridade utilizadas na presente proposta.

4.3 Funções de similaridade para agregados

Com base nas questões discutidas previamente, algumas funções de similaridade são apresentadas a fim de calcular um escore de similaridade entre um par de valores agregados. As funções são definidas e explicadas em detalhes através de exemplos a seguir. Como será visto mais adiante, a maneira como as funções de similari-

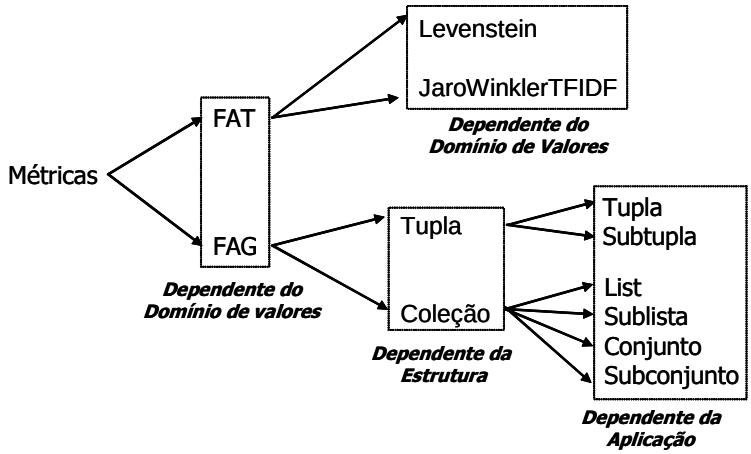


Figura 4.3: Taxonomia utilizada para a classificação das funções de similaridade

dade combinam os escores é diferente em cada caso: tupla, lista ou conjunto. As funções efetuam uma normalização através de um valor que varia, dependendo da interpretação que dá ao processo de casamento de agregados. As razões para isso, e os detalhes, são explicadas na Seção 4.3.3.

4.3.1 Tuplas

O objetivo da função de similaridade definida para tuplas[4] de dados é, dados dois agregados O_1 e O_2, comparar apenas os atributos comuns aos dois. Os atributos devem ser comparados usando uma função de similaridade específica para cada um. O resultado final é dado pela combinação dos escores gerados para os atributos, calculado através da média entre eles. Esta idéia é formalmente apresentada abaixo.

***Definição* 12** *(Similaridade entre Tuplas) Sejam O_1 e O_2 dois agregados do tipo tupla compostos por um conjunto A de atributos, conforme descrito na Definição 9, n e m o número de atributos em O_1 e O_2 respectivamente e f_η a função de similaridade associada e aplicada ao atributo de mesmo nome η, que é comum aos dois agregados. Seja d um valor, tal que d pode ser n, $max(n,m)$ ou $min(n,m)$. A* similaridade entre o par de agregados (O_1, O_2) do tipo tupla *é calculada através da seguinte equação:*

$$tupleSim(O_1, O_2) = \frac{\sum_{O_1.a.\eta = O_2.a.\eta} (f_\eta(O_1.a.v, O_2.a.v))}{d} \quad (4.1)$$

Neste caso, ao utilizar-se a função de similaridade *tupleSim()* cada atributo a do agregado O_1 é comparado com o atributo a correspondente (ou seja, de mesmo nome) em O_2, usando uma função f_η. A função f_η é uma função associada ao atributo de nome η e que recebe como entrada os valores $O_1.a.v$ e $O_2.a.v$, que correspondem respectivamente aos valores dos atributos de nome η dos agregados O_1 e O_2. Considerando $d = max(n,m)$, o somatório dos escores é normalizado pelo maior entre o número de atributos dos dois agregados, penalizando o resultado da comparação caso os agregados possuam um número diferente de atributos. Esta

[4] A função apresentada pode ser generalizada usando os resultados obtidos por Carvalho& Silva (CARVALHO; SILVA, 2003), ou por Tejada et al. (TEJADA; KNOBLOCK; MINTON, 2002), trabalhos que em que é levado em consideração o uso de pesos específicos para cada atributo

O	Director	Language	Title
O_1	Barry Sonnenfeld	English	Men in Black

1	2	3	4	5	6	7	8	9
O	Director	$f_\eta(...)$	Language	$f_\eta(...)$	Title	$f_\eta(...)$	$\Sigma(f_\eta(...))$	tupleSim(O_1, O_n)
O_2	Barry Sonnenfeld	1	English	1	Men in Black	1	3	1
O_3	Sonnenfeld, Barry	1	English	1	Men In Black	1	3	1
O_4	Barry Sonenfeld	1			Men in Black	1	2	0,66667
O_5	Barry Sonnenfeld	1	English	1	Men in Black II	0,875	2,875	0,958333333
O_6	Barry Sonnenfeld	1	English	1	Men in Black VHS	0,7222222	2,722222222	0,907407407
O_7	Barry Sonnenfeld	1	US EN	0,333	Men In Black 2	0,8666667	2,2	0,733333333
...	...	...	...	...	...	...	...	...
O_n	Barry Sonenfeld	1	English	1	Wild Wild West	0	2	0,5

Figura 4.4: Resultado da aplicação da função *tupleSim()*

penalidade é utilizada, por exemplo, no caso em que $O_1 = \langle a, b, c\rangle$ e $O_2 = \langle b, c\rangle$ e que seja desejável que o escore de similaridade resultante da comparação destes dois agregados não seja 1, pois não são totalmente similares. Basicamente, a função de similaridade para tuplas é dada pela média entre os escores de similaridade dos atributos.

***Exemplo* 1** *Considerando o agregado O_1 como:*

```
(Movie,
   {<Director, "Barry Sonnenfeld">
    <Language, "English">
    <Title, "Men in Black">
    }
)
```

e um conjunto de agregados O_n apresentados na Figura 4.4, cujos valores dos atributos Director*,* Language *e* Title *estão representados nas colunas 2, 4 e 6 respectivamente.*

O resultado do casamento de agregados do tipo tupla, usando a Equação 4.1, sobre um conjunto de agregados O_n é apresentado na Figura 4.4, através de uma tabela com os resultados. A terceira, quinta e sétima colunas apresentam o resultado da comparação dos atributos Director, Language e Title respectivamente. No caso do exemplo, as funções de similaridade[5] usadas para cada um deles são $f_{Director} = JaroWincklerTFIDF()$, $f_{Title} = QGrams()$ e $f_{Language} = Levenshtein()$. A oitava coluna apresenta o somatório dos escores calculados para cada atributo comum, e finalmente a última coluna apresenta o resultado gerado pela função de similaridade *tupleSim()*. O agregado O_3 da tabela de resultados apresenta um caso em que o escore final obteve um decréscimo pelo fato dos agregados possuírem um número diferente de atributos.

A Equação 4.1 é, essencialmente, similar à proposta feita por Fellegi&Sunter (FELLEGI; SUNTER, 1969) para comparação de registros com múltiplos atributos. O objetivo é somar os escores dos atributos e dividir pelo número de atributos. Como um simples escore é usado para expressar o casamento aproximado dos atributos, usar a média para combiná-los é bastante intuitivo.

[5]Nesta tese, a definição de qual a melhor função de similaridade utilizada em cada caso de valor atômico não foi explorada, pois não se caracteriza um foco da tese

4.3.2 Coleções

Como já mencionado, o processo de combinação de escores de similaridade de atributos componentes de uma coleção de dados pode ser executado considerando manipulação de listas e conjuntos de dados. No caso das listas, assume-se que os valores dos atributos devam obedecer uma ordem específica. No caso de conjuntos, a ordem dos valores não é considerada. É importante deixar claro, novamente, que esta não é uma restrição da estrutura do agregado e sim da intenção do usuário ou aplicação, especificada em tempo de execução do casamento de agregados. A principal meta que se quer atingir ao tratar coleções de dados de forma diferenciada é alcançar um resultado mais preciso no valor do escore gerado para a comparação de agregados verdadeiramente similares.

Na função de similaridade definida para listas de dados, o objetivo é comparar os valores de atributos que se encontram exatamente na mesma ordem. A combinação é feita através da média entre os escores calculados. A definição é apresentada a seguir.

***Definição* 13** *(Similaridade entre Listas) Sejam O_1 e O_2 dois agregados do tipo lista compostos por um conjunto A de atributos, conforme descrito na Definição 9, n e m o número de atributos em O_1 e O_2 respectivamente, f a função de similaridade utilizada para comparar os atributos dos agregados, e i a posição de cada atributo. Seja d um valor, tal que d pode ser n, $max(n, m)$ ou $min(n, m)$. A* similaridade entre o par de agregados (O_1, O_2) do tipo lista *é calculada através da seguinte equação:*

$$listSim(O_1, O_2) = \frac{\sum_{i=1}^{min(n,m)} (f(O_1.a_i.v, O_2.a_i.v))}{d} \tag{4.2}$$

O valor de escore de similaridade entre listas de dados é calculado pela comparação dos atributos de mesma posição nos dois agregados O_1 e O_2. Da mesma forma que em tuplas, considerando $d = max(n, m)$, o somatório dos escores é normalizado pelo maior entre o número de atributos dos dois agregados, penalizando a comparação de listas com número diferente de atributos.

***Exemplo* 2** *Considerando o agregado O_1 como:*

```
(Authors,
   {<Name, "Sharad Mehrotra">
    <Name, "Rajeev Ratogi">
    <Name, "Henry F. Korth">
    <Name, "Abraham Silberschatz">
    }
)
```

O resultado da comparação entre O_1 e alguns agregados de um conjunto de dados é apresentado na Figura 4.5. A terceira, quinta, sétima e nona colunas apresentam os resultados da comparação dos atributos nome que estão na primeira, segunda, terceira e quarta posições respectivamente. A décima coluna apresenta o somatório e a última o resultado final da comparação. No resultado, pode-se observar que os dois primeiros agregados, O_2 e O_3 representam exatamente o mesmo objeto do

O	Name	Name	Name	Name
O_1	Sharad Mehrotra	Rajeev Ratogi	Henry F. Korth	Abraham Silberschatz

1	2	3	4	5	6	7	8	9	10	11
O	Name	$f_q(...)$	Name	$f_q(...)$	Name	$f_q(...)$	Name	$f_q(...)$	$\Sigma(f_q(...))$	listSim(O_1, O_n)
O_2	Sharad Mehrotra	1	Rajeev Ratogi	1	Henry F. Korth	1	Abraham Silberschatz	1	4	1
O_3	Mehrotra, Sharad	1	Ratogi, Rajeev	1	Korth, Henry F.	1	Silberschatz, Abraham	1	4	1
O_4	Sharad Mehrotra	1	Rajeev Rastogi	1	Yuri Breitbart	0.5	Henry F. Korth	0,4933	3	0,75
O_5	Sharad Mehrotra	1	Rajeev Ratogi	1	Henry F. Korth	1		0	3	0,75
O_6	Sharad Mehrotra	1	Henry F. Korth	0,233	Silberschatz, A.	0,140766		0	1,374066	0,3435165
...	...	...	...	...	...	...	...	...	...	...
O_n	Henry F. Korth	0,478		0		0		0	0,4777	0,119425

Figura 4.5: Resultado da aplicação da função *listSim()*

mundo real que O_1. Os restantes possuem alguns valores equivalentes, entretanto em ordens diferentes.

Casos em que não exista a necessidade de considerar a ordenação dos valores em uma coleção de dados, é possível utilizar outra função de similaridade, chamada de *setSim()*. A função *setSim()* é utilizada para comparação de conjuntos de dados. Como os valores em um conjunto não possuem uma ordem, a identificação dos valores comuns aos atributos dos agregados torna-se mais exaustiva. A estratégia usada é comparar todos os atributos e combinar apenas os escores de similaridade resultantes de mais alto valor. O escore de similaridade resultante, da mesma forma que em tuplas e listas, é dado através da média entre os escores calculados. A definição formal desta função é apresentada a seguir.

***Definição* 14** *(Similaridade entre Conjuntos) Sejam O_1 e O_2 dois agregados do tipo conjunto compostos por um conjunto A de atributos, conforme descrito na Definição 9, n e m o número de atributos em O_1 e O_2 respectivamente, f a função utilizada para comparar os atributos dos agregados, e i a posição dos atributos em O_1 e j a posição em O_2. Seja d um valor, tal que d pode ser n, $max(n, m)$ ou $min(n, m)$. A* similaridade entre o par de agregados (O_1, O_2) do tipo conjunto *é calculada através da seguinte equação:*

$$setSim(O_1, O_2) = \frac{\sum_{i=1,j=1}^{n,m} (\max_{j=1 \ to \ m} (f(O_1.a_i.v, O_2.a_j.v)))}{d} \tag{4.3}$$

Na função de similaridade para conjuntos cada atributo do agregado O_1 é comparado com todo o conjunto de atributos do agregado O_2 e o maior escore de similaridade é retornado através da função max(). Da mesma forma que em tuplas e listas, sendo $d = max(n, m)$, o somatório dos valores de escore é normalizado pelo maior número entre a quantidade de atributos de O_1 e O_2.

***Exemplo* 3** *Considerando o agregado O_1 como:*

```
(Genre,
   {<Name, "Action">
    <Name, "Fantasy">
    }
)
```

e O_2 como:

```
(Genre,
   {<Name, "Fantasy">
    <Name, "Action">
    <Name, "Comedy">
    }
)
```

O casamento entre os agregados O_1 e O_2 apresentados no Exemplo 3 é apresentado na Figura 4.6. A figura apresenta a comparação de cada atributo do agregado O_1, com todos os atributos de O_2, que compõem Genre.

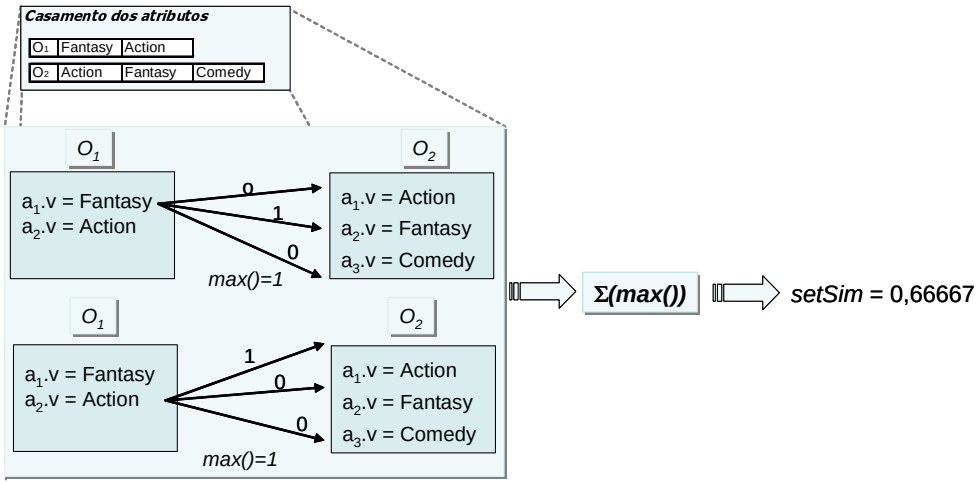


Figura 4.6: Exemplo de comparação de componentes de um conjunto

4.3.3 Sub-estruturas de agregados

Os exemplos apresentados para as funções de similaridade definidas anteriormente têm $d = max(n, m)$. Nestes casos, as funções são usadas em aplicações de casamento de agregados, onde o número de atributos deve ser o mesmo para os dois objetos. Neste caso, se o número de atributos não for igual, os escores resultantes da comparação refletem a diferença através de valores mais baixos. Entretanto, em certos casos, pode ser desejada a comparação de estruturas que estão contidas em outras. No contexto da tese, durante o processo de casamento é possível casar dois agregados, em que um é uma sub-estrutura do outro, independentemente dos demais componentes. Por este motivo, o divisor d é descrito nas Definições 12, 13 e 14 como um valor que pode ser uma escolha entre várias alternativas: n, $max(n, m)$ ou $min(n, m)$. Relembrando que n representa o número de atributos do agregado O_1, e m o número de atributos do agregado O_2. O caso $d = max(n, m)$ foi detalhadamente explicado e exemplificado anteriormente.

As duas outras alternativas, $d = n$ e $d = min(n, m)$ são especialmente úteis em aplicações que tratam com agregados que nem sempre são representados com o mesmo número de atributos. Em sistemas de consultas por similaridade, por exemplo, o usuário especifica certas condições de busca, que não são o agregado como um todo. Neste cenário é interessante utilizar $d = n$ indicando o número

de atributos constantes na consulta (a consulta seria representada como um "objeto consulta" O_1), pois reflete a intenção do usuário.

Outro exemplo é o caso de integração de fontes, como em citações bibliográficas, em que algumas fontes podem representar apenas o primeiro autor de uma lista de autores, enquanto outras fontes representam a lista completa. Neste caso, $d = min(n, m)$ pois não é possível identificar qual das duas fontes de dados possui um agregado com menor número de atributos.

A seguir, alguns exemplos são apresentados a fim de mostrar a idéia do uso de sub-estruturas. Ao todo, são apresentados três exemplos, um para cada estrutura: sub-tupla, sub-lista e sub-conjunto.

O	Director	Language
O_1	Barry Sonnenfeld	English

1	2	3	4	5	6	7	8
O	Direction	$f_\eta(...)$	Language	$f_\eta(...)$	Title	$\Sigma(f_\eta(...))$	$tupleSim(O_1, O_n)$
O_2	Barry Sonnenfeld	1	English	1	Men in Black	2	1
O_3	Sonnenfeld, Barry	1	English	1	Men In Black	2	1
O_4	Barry Sonenfeld	1			Men in Black	1	1
O_5	Barry Sonnenfeld	1	English	1	Men in Black II	2	1
O_6	Barry Sonnenfeld	1	English	1	Men in Black VHS	2	1
O_7	Barry Sonenfeld	1	English	1	Wild Wild West	2	1
O_8	Sonnenfeld, Barry	1	English	1	Get Shorty	2	1
...	...	...	...	...	...	...	...
O_n	Barry Sonnenfeld	1	US EN	0,333	Men In Black 2	1	0,5

Figura 4.7: Resultado da aplicação da função *tupleSim()* com $d = n$

***Exemplo* 4** *Considerando uma aplicação que trate consultas por similaridade, um conjunto de agregados, apresentado na Figura 4.7, cujos valores dos atributos são apresentados nas colunas 3, 5 e 7, e o agregado O_1 como:*

```
(Movie,
   {<Director, "Barry Sonnenfeld">
    <Language, "English">
    }
)
```

deseja-se obter filmes dirigidos por "Barry Sonnenfeld" cujo idioma é "Inglês", não importando o título.

O resultado da aplicação da Equação 4.1, tendo $d = n$, sobre um conjunto de agregados é apresentado na Figura 4.7. Neste exemplo, as colunas 3 e 5 apresentam valores que referem aos escores calculados para os atributos Director e Language respectivamente. O atributo Title não é usado na comparação. Pode-se observar que aparecem no resultado agregados com diferentes valores para o atributo Title e que possuem alto valor de similaridade final.

O exemplo a seguir apresenta um caso em que seria útil utilizar uma função que compara sub-estruturas de listas.

Figura 4.8: Exemplo de dados com diferentes valores em listas

***Exemplo* 5** *Supõe-se uma aplicação de integração de fontes de citações bibliográficas em que se deseja integrar publicações. A Figura 4.8 apresenta alguns dados que podem ser considerados representações de agregados constantes em uma das fontes a serem integradas, e o agregado O_1 como um dos agregados provenientes de outra fonte, e representado como:*

```
(Publication,
   {<Title, "Ensuring Consistency in Multidatabase by Preserving Two-Level">
    <Journal, "ACM Trans. Database Syst.">
    <Authors,
       {<Name, "Sharad Mehrotra">}>
   }
)
```

Neste caso, o objetivo é integrar Publicações (agregado Publication) relaxando o casamento entre a lista de autores (atributo Authors), pois em certas fontes ela pode estar representada apenas com o primeiro autor. Considerando, para fins de simplificação do exemplo, apenas o casamento entre a lista de autores, o resultado da comparação entre os agregados do tipo lista, usando a função de similaridade da Equação 4.2, com $d = min(n, m)$, é apresentado na Figura 4.9 (a figura apresenta apenas o resultado relativo à comparação das coleções de dados, Authors).

Similarmente ao Exemplo 5, existem casos em que os valores de um conjunto de dados não são os mesmos em todas as fontes.

***Exemplo* 6** *Presume-se uma aplicação de integração de fontes, cujo objetivo é integrar dados sobre filmes. Para este exemplo, supõe-se que alguns dados provenientes de uma das fontes estão representados na Figura 4.10, e que o agregado O_1, proveniente de outra fonte, seja representado por:*

```
(Publication,
```

O	Name
O_1	Sharad Mehrotra et.al

O	Name	$f_\eta(...)$	Name	Name	Name	$\Sigma(f_\eta(...))$	listSim(O_1, O_n)
O_2	Sharad Mehrotra	1	Rajeev Ratogi	Henry F. Korth	Abraham Silberschatz	1	1
O_3	Mehrotra, Sharad	1	Ratogi, Rajeev et.al.			1	1
O_4	Sharad Mehrotra et.al	0,9555				0,9555	0,9555

Figura 4.9: Resultado da aplicação da função *listSim()* com $d = min(n, m)$

```
   {<Title, "League of Extraordinary Gentlemen, The">
    <Journal, "Norrington,  Stephen>
    <Genre,
       {<Name, "Action">}>
   }
)
```

No Exemplo 3, o objetivo é integrar dados sobre filmes (Movies, relaxando a comparação entre os gêneros dos filmes). Considerando, para fins de simplificação do exemplo, apenas o casamento entre o conjunto de gêneros, usando a função de similaridade da Equação 4.3, com $d = min(n, m)$, o resultado do casamento de O_1 com os agregados das fontes é apresentado na Figura 4.11 (a figura apresenta apenas o resultado relativo à comparação das coleções de dados, Genre), e o que se pode observar é que o gênero de um filme nem sempre é um conjunto de valores iguais em todas as fontes.

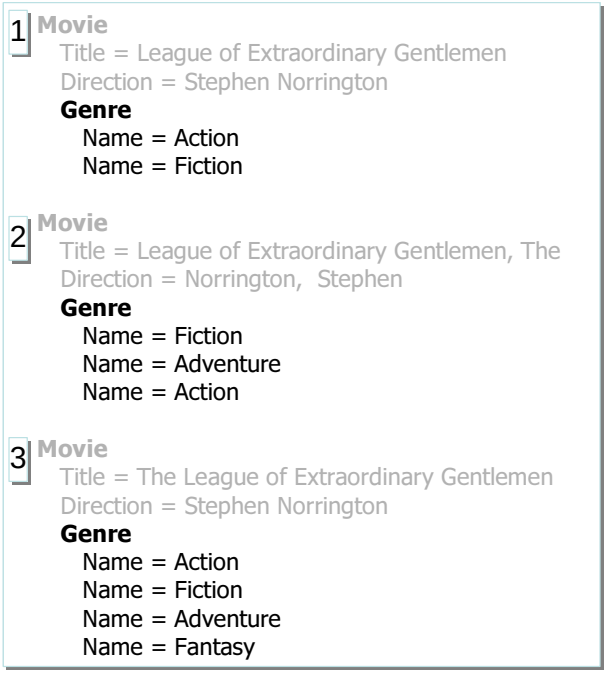
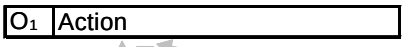


Figura 4.10: Exemplo de dados com diferentes valores em conjuntos

Em geral, todas as funções de similaridade propostas calculam um escore de similaridade entre agregados pela soma dos escores de similaridade de seus atributos dividida pelo maior número de atributos entre os agregados que estão sendo



1	2	3	4	5	6	7	8	9
O	Name	max(f η(...))	Name	max(f η(...))	Name	max(f η(...))	Σ(max(f η(...))	setSim()
O2	Action	1	Fantasy	1	Comedy	0	2	1
O3	Action	1	Comedy	0	Fantasy	1	2	1
O4	Fantasy	1	Comedy	0	Action	1	2	1
O5	Fantasy	1	Action	1			2	1
O6	Action	1	Fantasy	1			2	1
...	...	...	...	...	...	...	...	...
On	Action	1	Drama	0	Sci Fi	0	1	0,5

Figura 4.11: Resultado da aplicação da função *setSim()* com $d = min(n, m)$

comparados. A distinção entre as funções está basicamente no processo de como os escores dos atributos são combinados, ou seja, como o somatório é feito. Esta combinação depende do tipo de estrutura do agregado, bem como de certas necessidades específicas de comparação (quando se usa as sub-estruturas). Como esta combinação, algumas vezes, depende de certas necessidades, decidiu-se por permitir um relaxamento das funções no caso de se necessitar apenas parte do agregado.

4.4 Combinação de escores ASSC

Um dos problemas discutidos no Capítulo 3 é que os escores gerados por diferentes funções de similaridade possuem diferentes distribuições sobre as bases de dados. Por exemplo, um escore de aproximação entre duas datas pode ter um significado completamente diferente de um escore resultante da comparação entre dois nomes próprios. Assim, os escores não são comparáveis.

Como discutido neste capítulo, pode-se observar que é comum haver casamento de tuplas compostas por vários atributos com diferentes domínios de valores, e portanto possuindo diferentes distribuições (uma para cada atributo) esta incompatibilidade das funções de similaridade deve ser analisada. As funções de similaridade propostas para agregados, como estão definidas, não resolvem o problema das diferentes distribuições. Trabalhos prévios propõem soluções tais como transformações do escore de similaridade em valores para um intervalo unificado, por exemplo valores entre [0..1] (TEJADA; KNOBLOCK; MINTON, 2001). Estas transformações podem ajudar, mas os resultados das diferentes funções ainda continuam tendo diferentes significados. Além disso, como apresentado no Capítulo 3, mesmo os resultados de uma mesma função de similaridade sobre dois diferentes atributos pode resultar valores incompatíveis devido a diferenças no domínio e na qualidade dos dados armazenados em cada um dos atributos.

Com este problema, a segunda grande motivação no uso de escores ajustados é permitir a combinação correta de escores de similaridade que são gerados por diferentes funções de similaridade, usando para isso um simples escore. Neste caso, a proposta é usar a Tabela de Mapeamento de Escores (TaME) para efetuar a substituição dos escores originais gerados pelas funções pelo escore ASSC correspondente.

Exemplificando novamente o problema, considerando que dois agregados Film, compostos pelos seguintes atributos: Director, title e genre, necessitem ser casados.

Supondo ainda que as funções de similaridade, utilizadas para cada um dos atributos, sejam f_1 para Director, por ser mais adequada a nomes próprios, f_2 para Title, por ser aplicável a cadeias de caracter de forma geral, e f_3 para Genre, por ser uma função recomendada para dados compostos por uma única palavra (um *token*). A fim de avaliar um escore de similaridade entre dois objetos Film, primeiro a similaridade entre seus atributos deve ser calculada, e posteriormente combinada usando *tupleSim()*. Para tornar tal combinação viável, os escores gerados pelas funções f_1, f_2 e f_3 devem ser compatíveis, e isso provavelmente não acontece em funções de similaridade distintas. A proposta então é que a combinação dos valores de escores seja efetuada usando os escores ajustados (proposto no Capítulo 3) ao invés dos escores originais gerados pelas funções de similaridade f_1, f_2 e f_3. Esta é uma solução aceitável pois os escores ASSC são sempre (independente da função de similaridade) gerados a partir da mesma medida, a precisão, e portanto, são considerados combináveis.

Desta forma, as funções de similaridade para agregados devem utilizar uma nova função no lugar de cada f_η usada para seus atributos. Esta nova função é definida a seguir.

***Definição* 15** *(Função Ajustada de Similaridade)*
Seja $M_f = \{\langle 0, 0, p_0\rangle, \ldots, \langle 1, 0, p_{10}\rangle\}$ *uma tabela de mapeamento de escores para uma função de similaridade* f*, como apresentada na Definição 4. Seja* α_M *uma função que mapeia os escores originais gerados por* f *para escores ASSC em uma tabela* M_f*, como apresentado na Definição 6. Seja* (v_1, v_2) *o par de valores a ser comparado.*

A função ajustada de similaridade $\hat{f}()$ *que utiliza os escores ASSC de uma TaME é definida como:*

$\hat{f}(v_1, v_2) = \alpha_{M_f}(f(v_1, v_2))$.

A função especificada na Definição 15 apenas formaliza a substituição dos escores originais pelos escores ASSC. A função $\alpha_{M_f}()$, introduzida na Definição 6, é a função que de fato executa o mapeamento do escore orginal para o escore ASSC.

Agora, com a função ajustada de similaridade, é possível utilizar as funções de similaridade definidas para agregados combinando os escores dos atributos de forma adequada. A seguir, alguns exemplos demonstram esta idéia.

***Exemplo* 7** *Considerando um sistema de integração e sendo:*
O_1*:*

```
(Movie,
   {<Director, "Stephen Norrington">
    <Language, "Britis English">,
    <Title, "The League of Extraordinary Gentlemen">}
)
```

e

O_2*:*

```
(Movie,
   {<Director, "Norrington, S.">
    <Language, "British English">,
    <Title, "League of Extraordinary Gentlemen">}
)
```

JaroWincklerTFIDF()

Escore	ASSC
1	1
0.9	1
..	...

QGrams()

Escore	ASSC
1	1
0.9	0,91001
..	...

Levenshtein()

Escore	ASSC
1	1
0.9	0,8324
..	...

Figura 4.12: TaMEs para as funções $JaroWincklerTFIDF()$, $QGrams()$ e $Levenshtein()$

obter o casamento entre os agregados.

Para obter o casamento entre os agregados apresentados no Exemplo 7 são necessárias funções de similaridade para os atributos Title, Director e Language, que são $f_{Director} = JaroWincklerTFIDF()$, $f_{Title} = QGrams()$ e $f_{Language} = Levenshtein()$. Para que a combinação dos escores de seus atributos seja adequada, a Definição 15 deve ser aplicada. Desta forma, é necessário que se conheça as seguintes TaMEs: $M_{JaroWincklerTFIDF()}$, $M_{QGrams()}$ e $M_{Levenshtein()}$, as quais são apresentadas na Figura 4.12.

Casamento dos atributos

O	Director	Language	Title
O_1	Stephen Norrington	Britis English	The League of Extraordinary Gentlemen

O	Director	Language	Title
O_2	Norrington, S.	British English	League of Extraordinary Gentlemen

Resultado

Director	$f_\eta(...)$	Language	$f_\eta(...)$	Title	$f_\eta(...)$
	0,9		0,93		1

$\alpha_{M_f}()$ ⇐ **TaMEs**

Resultado Final

Director	$\alpha_{M_f}()$	Language	$\alpha_{M_f}()$	Title	$\alpha_{M_f}()$	$\Sigma(f_\eta(...))$	tupleSim(O_1, O_n)
	1		0,90501		1	2,905005	0,968335

Figura 4.13: Resultado da combinação de escores ASSC em casamento de agregados

Além disso, é necessário definir a função utilizada para a combinação dos escores gerados para os atributos, que, no caso, deve ser a função *tupleSim()*, com a função ajustada de similaridade:

$$tupleSim(O_1, O_2) = \frac{\sum_{O_1.a.\eta = O_2.a.\eta} (\hat{f}_\eta(O_1.a.v, O_2.a.v))}{max(n, m)} \quad (4.4)$$

O resultado do casamento dos agregados do Exemplo 7 é apresentado na Figura 4.13. Na figura são apresentados os passos executados desde o casamento dos

atributos, passando pelo uso da função $\alpha_{M_f}(f())$, que utiliza os valores das tabelas TaME, até o resultado final.

No Capítulo 5 são apresentados em detalhes os resultados de experimentos realizados com a proposta apresentada aqui.

4.5 Conclusões

Este capítulo apresentou duas principais contribuições: o uso de diferentes funções de similaridade para combinação de escore em agregados, que são dependentes da estrutura, e o uso do escore ASSC como uma estratégia genérica para combinação de escores de similaridade, em tuplas, gerados por funções que possuem diferentes distribuições.

As funções definidas para tupla, lista e conjunto, são o primeiro passo para a pesquisa no tratamento diferenciado de agregados. Como demonstrado no Capítulo 5, os experimentos realizados comprovam que a combinação de escores, usando diferentes estratégias, gera resultados mais precisos.

As funções de similaridade para agregados, apresentadas neste Capítulo, foram implementadas em um protótipo que processa consultas em uma linguagem que é extensão da XPath, a SimXPath (suporte à Similaridade em XPath) que permite o uso de funções de similaridade em expressões de caminho (PADILHA; DORNELES; HEUSER, 2005). A intenção da linguagem SimXpath não é modificar a sintaxe da XPath, pois deseja-se utilizar o processador já existente. Desta forma, a proposta consiste da extensão do conjunto de funções já disponíveis na linguagem. O trabalho apresenta, de forma geral, três contribuições: (1) suporte a consultas com condições de busca aproximada através da adição de novas funções para uma linguagem; (2) definição de uma extensão, sem alteração da sintaxe, para uma linguagem padrão já adotada pela comunidade; (3) conjunto de experimentos preliminares que demonstram a utilidade de funções de similaridade em consultas a base de dados geradas pela integração de diferentes fontes.

5 EXPERIMENTOS

Neste Capítulo, são descritos os experimentos realizados a fim de validar a proposta apresentada. Os principais objetivos destes experimentos estão listados a seguir.

- Demonstrar a efetividade das idéias apresentadas no Capítulo 4. Através destes experimentos, pretende-se apresentar que a manipulação apropriada de coleções leva a melhores resultados, principalmente quando existe a diferenciação entre o tratamento de uma coleção ordenada e uma não ordenada (DORNELES et al., 2004). A descrição detalhada dos experimentos efetuados nesta fase encontram-se em (DORNELES; LIMA; HEUSER, 2003a).

- Demonstrar que o escore ASSC, isto é, a precisão estimada através do processo de treinamento, pode ser usada ao invés do escore original de similaridade calculado por uma função de similaridade, permitindo assim, que o usuário especifique a qualidade do processo de casamento em termos significativos. Estes experimentos comprovam as idéias apresentadas no Capítulo 3.

- Demonstrar que pela combinação de escores ASSC, ao invés dos escores originais, resultados mais precisos são alcançados em casamento de tuplas, comprovando as idéias apresentadas no na Seção 4.4 do Capítulo 4.

A apresentação destes experimentos está dividida em duas partes: a Seção 5.1 apresenta a validação das idéias introduzidas no Capítulo 4, e a Seção 5.2 descreve os resultados da proposta apresentada no Capítulo 3.

5.1 Avaliação das funções para agregados

Nesta seção, são apresentados os resultados dos experimentos executados para demonstrar a efetividade das idéias apresentadas no Capítulo 4. A seção mostra a descrição da elaboração dos dados utilizados nos experimentos, a execução do processo de casamento de agregados e os resultados obtidos.

5.1.1 Preparação dos dados

A seguir, é apresentada uma descrição de como as bases de dados foram preparadas para a execução dos experimentos, incluindo as fontes a partir das quais os dados foram originados, os atributos envolvidos e funções de similaridade utilizadas para cada um deles, bem como o número de agregados contidos em cada base.

Bases de dados e atributos envolvidos

Duas bases de dados, contendo dados reais, foram utilizadas para os experimentos efetuados nesta etapa, as quais são descritas a seguir.

Citações Bibliográficas. (*Citation*) Os dados desta base são provenientes de arquivos BibTEX. A base de citações bibliográficas é composta pelos atributos Authors, que corresponde a uma lista de autores, contendo o atributo Name, composta pelo atributo Nome, e os atributos Date e Title, representando respectivamente a data de ocorrência do evento, ou publicação da revista, em que o artigo foi publicado, e o título da publicação.

Os dados foram extraídos do servidor de citações *The Collection of Computer Science Bibliographies*[1] e dos arquivos BibTEX dos membros do Grupo de Banco de Dados do Instituto de Informática da UFRGS[2], totalizando 16.519 entradas de citações bibliográficas. No primeiro caso, os arquivos usados foram os seguintes: *ACM/SIGMOD*, *Information Systems*, *Bibliography on database systems*, *Bibliography on Semistructured Data*, *VLDB journal*, *VLDB Conference and ACM Transactions on Information Systems*.

Filmes. (*Movie*) Os dados desta base são provenientes de video locadoras que possuem sites na Web. Os atributos dos agregados desta base de dados são Genre, que corresponde a um conjunto de dados que representa os gêneros de um filme, Director que corresponde ao Diretor de um filme, e Title que representa o título do filme.

Os dados são originados de múltiplas fontes, tais como *Blockbuster US*, *Blockbuster UK* e *Blockbuster CA*. Os dados foram extraídos manualmente, e os rótulos usados nas páginas HTML para descrever cada dado foram usados como nomes dos atributos dos agregados.

Instâncias

A Figura 5.1 apresenta o tamanho de cada uma das bases de dados. A segunda coluna, Total, apresenta o total de tuplas da base de dados, com valores redundantes. A terceira coluna, Total sem redundância, representa a quantidade total de instâncias, sem valor redundante da mesma cadeia de caracteres (neste caso, as cadeias de caracter "ONU" e "Organização das Nações Unidas" não são consideradas redundantes). E finalmente, a última coluna, Número de objetos do mundo real, mostra a total de objetos distintos presentes em cada uma das bases de dados (os valores "ONU" e "Organização das Nações Unidas" representam um único objeto).

Base de Dados	*Total*	*Total sem redundância*	*Quantidade de objetos do mundo real*
Movies	287	172	102
Citation	16.519	14.536	9.136

Figura 5.1: Tamanho total das bases de dados

[1]http://liinwww.ira.uka.de/ bibliography/ Database/index.html
[2]http://www.inf.ufrgs.br/ DBGroup/

Funções utilizadas para os atributos

Como já mencionado, foram usadas funções de similaridade específicas para cada atributo, dependendo do domínio de valores de cada um deles.

Domínio	*Atributo/Função*
Movie	Title - QGrams() Director - QGrams() Genre - Levenstein()
Citation	Title - QGrams() Date - dateSim() Author Name - QGrams()

Figura 5.2: Funções de similaridade utilizadas nos experimentos das funções para agregados

A Figura 5.2 apresenta as funções de similaridade que foram aplicadas a cada um dos atributos de cada uma das bases. As funções foram implementadas e testadas em um trabalho prévio (LIMA, ????). Avaliações e discussões sobre funções de similaridade podem ser encontradas em trabalhos na literatura (COHEN; RAVIKUMAR; FIENBERG, 2003b; LEE, 2001), e esse ponto não caracteriza um foco da tese.

5.1.2 Execução do processo de casamento

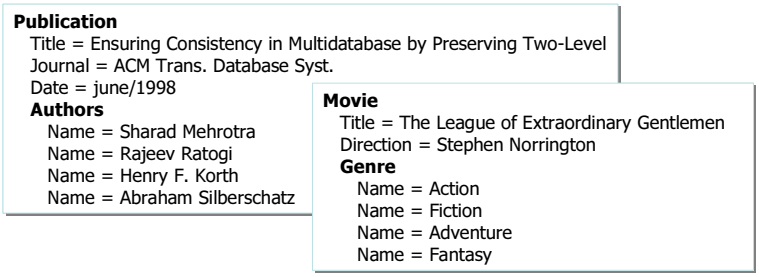


Figura 5.3: Exemplos da estrutura dos agregados usados nos testes

Especificamente, os experimentos realizados nesta fase possuem três objetivos: (1) mostrar que o tratamento diferenciado entre tuplas e coleções é um ponto importante a ser tratado em um processo de casamento de agregados; (2) apresentar resultados que demonstram o ganho quando existe a distinção entre lista e conjunto; (3) confirmar resultados apresentados na literatura de que o uso de funções específicas para o domínio de valores de dados atômicos aumenta a precisão dos resultados.

Para a execução destes experimentos, foram definidos agregados, chamados de objetos-consulta, que representam o agregado O_1, introduzido na Definição 12. Cada objeto-consulta é casado com cada instância da base de dados, e os resultados são apresentados como *rankings* em relação a O_1.

Esta estratégia, de apresentação dos resultados, foi utilizada para que através do *ranking* fosse possível a posterior avaliação através de curvas de revocação/precisão. Os experimentos consistem em executar diversos casamentos nas bases de dados reais dos domínios Citation e Movie e avaliá-los usando a curva de revocação/precisão, uma

medida de avaliação amplamente utilizada e aceita na comunidade de Recuperação de Informação (BAEZA-YATES; RIBEIRO-NETO, 1999).

Definição dos objetos-consulta

São definidos quatro conjuntos diferentes de consultas $Q_i = \{q_1, q_2, \ldots, q_{30}\}$, para cada base de dados. Cada conjunto de objetos-consulta possui diferentes combinações de funções de similaridade utilizadas. As variações construídas são descritas a seguir.

- **GnE**
  Gn - Função para agregado: somente para tupla
  E - Função para atômicos: específicas do domínio
  A configuração utilizada neste conjunto de consultas, Q_{GnE}, foi a utilização da função para tuplas para agregados, Definição 12, e funções de similaridade específicas do domínio de valores dos atômicos.

GnE - *Publication*

	Função de similaridade
Publication	*tupleSim()*
Title	*QGrams()*
Date	*dateSim()*
Authors	*QGrams()*

GnE - *Movie*

	Função de similaridade
Movie	*tupleSim()*
Title	*QGrams()*
Director	*QGrams()*
Genre	*QGrams()*

Figura 5.4: Funções do conjunto de objetos-consulta Q_GnE

- **StE**
  St - Função para agregado: tuplas e conjuntos
  E - Função para atômicos: específicas do domínio
  No segundo conjunto de consultas, Q_{StE}, a função utilizada para um agregado foi dependente da estrutura, e para valores atômicos as funções de similaridade específicas do domínio de valores.

StE - Publication

	Função de similaridade
Publication	*tupleSim()*
Title	*QGrams()*
Date	*dateSim()*
Authors	*setSim() (d = n)*
Name	*QGrams()*
Name	
Name	
Name	

StE - Movie

	Função de similaridade
Movie	*tupleSim()*
Title	*QGrams()*
Director	*QGrams()*
Genre	*setSim() (d = n)*
Name	*QGrams()*
Name	
Name	
Name	

Figura 5.5: Funções do conjunto de objetos-consulta Q_StE

- **ApG**
  Ap - Função para agregado: dependente da aplicação
  G - Função para atômicos: genérica
  As consultas deste conjunto, Q_{ApG}, foram construídas usando as três funções de similaridade para agregados, *tupleSim()* (Definição 12), *tupleList()* (Definição 13) e *tupleSet* (Definição 14), e uma função de similaridade geral para os atributos de valores atômicos.

ApG - Publication

	Função de similaridade
Publication	*tupleSim()*
Title	*Levenshtein()*
Date	*Levenshtein()*
Authors	*listSim() (d = n)*
Name	*Levenshtein()*
Name	
Name	
Name	

ApG - Movie

	Função de similaridade
Movie	*tupleSim()*
Title	*Levenshtein()*
Director	*Levenshtein()*
Genre	*listSim() (d = n)*
Name	*Levenshtein()*
Name	
Name	
Name	

Figura 5.6: Funções do conjunto de objetos-consulta Q_{ApG}

- **ApE**
  Ap - Função para agregado: dependente da aplicação
  E - Função para atômicos: específicas do domínio
  Para o último conjunto de consultas, Q_{ApE}, foram construídas usando as três funções de similaridade para agregados, Definição 12, Definição 13 e Definição 14 e uma função específica do domínio de valores para os atômicos.

ApE - Publication

	Função de similaridade
Publication	*tupleSim()*
Title	*QGrams()*
Date	*dateSim()*
Authors	*listSim()*
Name	*QGrams()*
Name	
Name	
Name	

ApE - Movie

	Função de similaridade
Movie	*tupleSim()*
Title	*QGrams()*
Director	*QGrams()*
Genre	*setSim()*
Name	*QGrams()*
Name	
Name	
Name	

Figura 5.7: Funções do conjunto de objetos-consulta Q_{ApE}

Para cada base de dados, os seguintes passos foram executados:

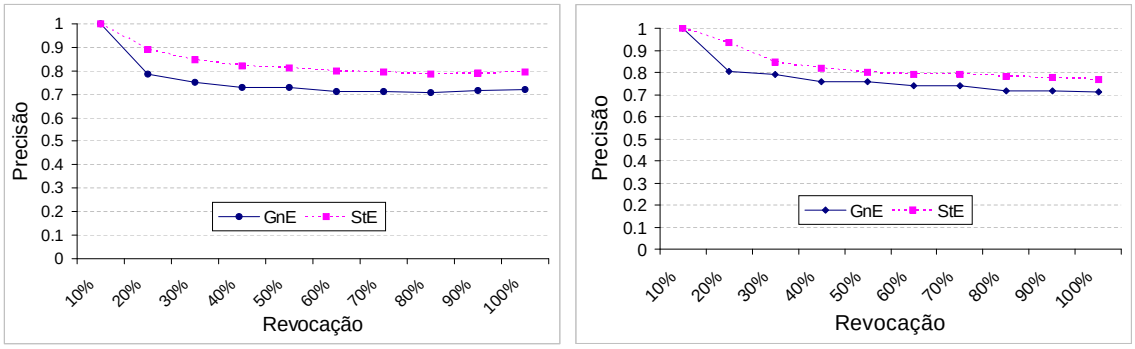
1. quatro conjuntos de 30 objetos-consulta, foram randomicamente escolhidos na base de dados real;

2. cada conjunto tem suas consultas configuradas de acordo com o apresentado anteriormente;

3. para cada objeto conjunto, um *ranking* é gerado sobre a base de dados. Os dados foram ordenados pela média dos escores originais obtidos para cada atributo;

4. revocação e precisão são calculadas através do procedimento padrão usado pela comunidade de Recuperação de Informação (BAEZA-YATES; RIBEIRONETO, 1999) e os resultados são apresentados em curvas de Revocação/ Precisão.

A Figura 5.8 apresenta a curva de Revocação/Precisão para comparação de objetos-consulta GnE vs. StE. Observando os resultados, é possível notar que a distinção entre tupla e coleção é uma importante característica a ser considerada quando se comparam agregados, pois o ganho de precisão neste caso corresponde a um valor em torno de 11%, nas duas bases de dados. Os resultados não apresentaram precisão de 100% na base de citações, pois o atributo (Authors), neste caso, foi

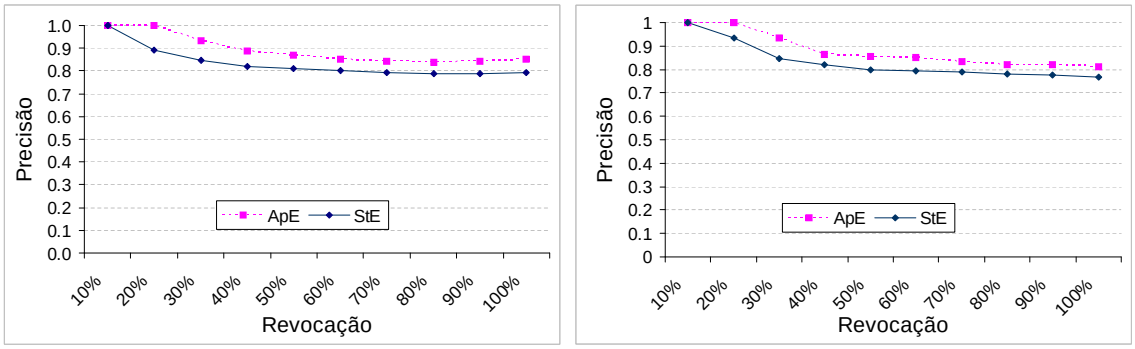


A) Citations B) Movies

Figura 5.8: Revocação/Precisão na comparação GnE vs. StE

considerado uma cadeia de caracteres. O mesmo ocorre com a base de filmes, onde o atributo (Genre) também é uma string de caracteres.

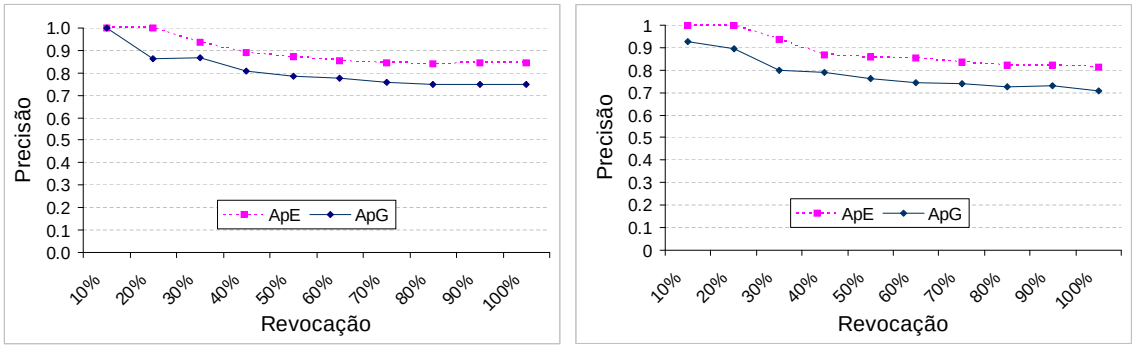
As curvas apresentadas na Figura 5.9 mostram a comparação de objetos-consulta StE vs. ApE. Nesta comparação, o segundo objetivo proposto foi testado, o qual consiste apresentar resultados que demonstram o ganho no tratamento de ordenação em componentes de coleções. Foram consideradas funções dependentes da estrutura StE, na qual a ordem não é considerada, contra funções ApE, nas quais a ordem dos componentes de uma coleção são considerados. Usando funções que tratam ordem, obteve-se resultados melhores. Os resultados de ApE melhoram em relação a StE pois em ambos os casos as coleções Authors e Genre foram consideradas coleções de dados, mas sem restrição de ordem nos valores.



A) Citations B) Movies

Figura 5.9: Revocação/Precisão na comparação StA vs. ApA

A Figura 5.10 apresenta a comparação ApE vs. ApG. Nesta comparação, o terceiro objetivo foi testado, ou seja, confirmar resultados apresentados na literatura de que o uso de funções específicas para o domínio de valores de dados atômicos aumenta a precisão dos resultados. Com os resultados apresentados, é possível afirmar que, usando funções específicas do domínio de aplicação de valores atômicos, a precisão dos resultados melhora.



A) Citations B) Movies

Figura 5.10: Revocação/Precisão na comparação ApE vs. StE

5.2 Avaliação do uso do escore ASSC

Esta seção apresenta os resultados dos experimentos executados para demonstrar a efetividades das idéias apresentadas no Capítulo 3.

5.2.1 Preparação dos dados

A seguir, é mostrado como as bases de dados foram preparadas para a execução dos experimentos efetuados, quais as fontes originárias dos dados, atributos envolvidos e quais as funções de similaridade para atômicos foram utilizadas.

Bases de dados

Os experimentos foram conduzidos em quatro diferentes domínios do mundo real, conforme apresentados a seguir.

Citações Bibliográficas. (*Citation*) Cada artigo tem três atributos: o título do artigo, representado por Title, o nome do periódico na qual o artigo foi publicado, representado pelo atributo Journal, e a editora do periódico, indicado pelo atributo Publisher. Os valores neste domínio estão todos em Inglês;

Filmes. (*Movie*) Este domínio compreende dados sobre filmes. Cada filme possui três atributos: o título do filme, representado pelo atributo Title, o diretor, representado por Director e o gênero do filme, indicado pelo atributo Genre. Os valores neste domínio também estão descritos em Inglês;

PosComp. Este domínio diz respeito aos dados sobre as pessoas que se inscreveram para o exame do PosComp. Para cada pessoa, foram escolhidos os seguintes atributos: o nome da instituição onde a pessoa trabalha, ou estuda, Instituição, a cidade origem da pessoa Cidade e a data na qual a pessoa se inscreveu na prova, representado pelo atributo Data. Os valores neste domínio estão em Português;

Música. (*Music*). Este domínio possui dados sobre músicas. Cada música possui três atributos: o nome do artista que interpreta a musica, representado pelo atributo Artist, o título da música, indicado pelo atributo Song, e o álbum na qual a música foi gravada, representado por Album. Os valores armazenados neste domínio estão em Inglês.

Nos experimentos, primeiramente é necessário executar o processo de treinamento a fim de gerar a TaME (Tabela de Mapeamento de Escores). Duas formas alternativas de uso dos conjuntos de dados foram avaliadas para o processo de treinamento:

- execução do processo de treinamento usando a mesma base de dados que contém as instâncias usadas no processo de casamento de agregados (chamado de base de dados real);
- execução do processo de treinamento usando um conjunto de dados maior de valores que representam o domínio de valores de um dado valor atômico (chamado de conjunto de dados representativo).

Usando o conjunto de dados real para treinamento, os resultado obtidos serão mais apurados pois o mesmo conjunto de valores é usado durante o processo de treinamento e o casamento de agregados. Por outro lado, o processo de treinamento deve ser repetido a cada nova base de dados usada. Além disso, se a base de dados sofreu alguma alteração significativa nos dados, um novo treinamento se faz necessário. Para a segunda alternativa, o uso de um conjunto de dados representativo, um único treinamento deveria ser adotado para qualquer conjunto de instâncias de um atributo, e usado em qualquer aplicação que possua o atributo com o mesmo domínio. No entanto, os escores ASSC podem não ser tão apurados quanto aqueles conseguidos sobre a base de dados real.

Em ambos os casos, o conjunto de consultas Q usado no processo de treinamento consiste de 40 valores distintos correspondendo aos objetos do mundo real que ocorrem na base de dados usada. Este conjunto de consultas foi usado para calcular a TaME para cada função de similaridade em cada atributo na base de dados. Isso significa que o comportamento da função de similaridade é estimado com base em um conjunto bastante limitado de consultas, o que torna o custo de treinamento aceitável. Como mostrado mais adiante nos experimentos, este pequeno número de consultas de treinamento é suficiente para alcançar bons resultados de estimativa em todos os domínios e conjuntos de dados experimentados, indicando que o método pode ser facilmente aplicado na prática.

Os dados dos experimentos foram organizados de três formas:

1. Para cada domínio (*Citation*, *Movie*, *PosComp* e *Music*), uma relação foi construída, chamada de ("base de dados real"), com os atributos descritos acima. Os valores desta base foram coletados de muitas fontes de dados (detalhes serão descritos abaixo) e são usados nos experimentos para casamento de agregados;
2. Para cada atributo, em cada domínio, dois diferentes conjuntos de dados foram construídos, como apresentado a seguir.
   (a) Os valores dos atributos da base de dados real foram usados para construir uma nova base de dados, chamada de "conjunto de dados real". Conjuntos de dados reais são usados no processo de estimativa do escores ASSC, bem como no processo de verificação de sua exatidão.
   (b) Para cada atributo, foram coletados valores de muitas fontes de dados, a fim de construir um conjunto de que representasse o domínio do atributo,

chamado de conjunto de dados representativo. Os conjuntos de dados representativos são utilizados apenas durante o processo de estimativa do escore ASSC.

As bases e conjuntos de dados foram construídas como descrito a seguir.

Instâncias

Como os experimentos tratam da existência de diferentes representações do mesmo objeto do mundo real, é importante identificar nas bases usadas a quantidade de instâncias existentes.

As instâncias na base de dados Citation são originadas do servidor *The Collection of Computer Science Bibliographies*[3] e de arquivos individuais coletados entre os membros do Grupo de Banco de Dados da UFRGS[4], totalizando 16.519 entradas de citações bibliográficas. Da primeira fonte de dados, os seguintes arquivos foram extraídos: *ACM/SIGMOD*, *Information Systems*, *Bibliography on database systems*, *Bibliography on Semistructured Data*, *VLDB journal*, *VLDB Conference* e *ACM Transactions on Information Systems*.

As instâncias da base de dados Movie são originadas de múltiplas fontes de dados, especificamente dados de fontes de videolocadoras, como *Blockbuster US*, *Blockbuster UK* e *Blockbuster CA*. As instâncias foram extraídas manualmente da Web e totalizam 353 entradas de filmes.

A base de dados do PosComp contém dados sobre as pessoas que efetuaram a prova em 2003 e 2004. Esta base de dados compreende 8.991 instâncias de dados.

Os dados coletados para o domínio Music são originados de três fontes de dados: a máquina de busca de músicas do Yahoo[5], as bases de dados WWW Music Database[6] e Web Music DB[7], totalizando 1,692 instâncias de dados.

A Figura 3.1 apresenta o esquema alguns exemplos de instâncias em cada uma das bases de dados. Pode-se observar que cada um dos bancos de dados contém muitas cópias da mesma tupla, bem como muitas tuplas diferentes que representam o mesmo objeto do mundo real. Similarmente, cada atributo pode conter valores duplicados, além disso, muitos valores diferentes podem representar o mesmo objeto. A Figura 5.11 mostra, na coluna Total o número total de instâncias, com repetições, a coluna Total sem redundância representa a quantidade total de instâncias, sem valor redundante da mesma cadeia de caracteres (como já mencionado, neste caso, as cadeias de caracter "ONU" e "Organização das Nações Unidas" não são consideradas redundantes); e por fim, a última coluna, Número de objetos do mundo real, mostra a total de objetos distintos presentes em cada uma das bases de dados (da mesma forma como já descrito, os valores "ONU" e "Organização das Nações Unidas" representam um único objeto). O número na última coluna foi obtido por inspeção manual em cada base de dados.

Bases de dados real e representativo

Os valores dos atributos da base de dados real foram usados para construir o conjunto de dados real. A Figura 5.12 apresenta o número de valores para cada atributo.

[3]http://liinwww.ira.uka.de/bibliography/Database/index.html
[4]http://metropole.inf.ufrgs.br/DBGroup/
[5]http://music.yahoo.com/musicengine/
[6]http://www.roadkill.com/MDB/
[7]http://www.webmusicdb.com/

Base de Dados	Total	Total sem redundância	Quantidade de objetos do mundo real
Movies	353	191	141
Music	1.692	1.503	1.135
PosComp	8.991	8.272	3.545
Citation	16.519	14.536	9.136

Figura 5.11: Tamanho total das bases de dados reais

Base de dados	Atributo	Total	Número de Agregados	Número de objetos do mundo real
Movies				
	Title	358	136	69
	Director	328	91	79
	Genre	310	32	26
Music				
	Artist	1.709	1.507	919
	Song	1.732	779	587
	Album	1.679	1.044	887
PosComp				
	Instituição	2.484	1.523	821
	Cidade	4.287	619	265
	Data	4.286	291	122
Citation				
	Title	18.403	10.91	8.416
	Journal	5.39	519	254
	Publisher	4.689	470	147

Figura 5.12: Tamanho, por atributo, das bases de dados reais

Além do conjunto de dados real, um conjunto de dados representativo também foi construído em cada domínio de atributo. É importante enfatizar que o conjunto de dados representativo foi usado apenas para o processo de estimativa. Os valores deste conjunto de dados foram coletados manualmente de diversos sites da Web; através disso, pretende-se remover qualquer tendência relacionada a alguma fonte que os conjuntos de dados possam ter.

No domínio Citation, os valores para os atributos title, journal e publisher foram extraídos da DB&LP, Citeseer, site de Periódicos e Conferências, além de Web sites de editoras, como Elsevier, ACM Press, IEEE pub., e assim por diante. Para o domínio Movie, os valores de title, director e genre foram extraídos de várias listas de filmes, com seus títulos originais, na Web. Para o domínio PosComp os valores dos atributos instituição e cidade foram extraídos de sites como CNPq, Capes, FAPERGS, além de diversas Universidades Brasileiras. Os valores para o atributo data foram extraídos de diferentes sites na Web (por exemplo, listas de lançamento de livros, lançamentos de livros, data de aniversários de artistas, entre outras) e de diferentes bases de dados (por exemplo, datas de inscrição em vestibulares de algumas Universidades). Para o domínio Music, os valores dos atributos Artist, Song e Album foram extraídos de diversos sites na Web[8], incluindo MTV e *Billboard*[9]. O número de valores para cada atributo no conjunto de dados representativo é mostrado na Figura 5.13.

Funções de similaridade para valores atômicos

[8]http://www.lyrics.com/, http://www.azlyrics.com/
[9]http://www.mtv.com/, http://www.billboard.com

Base de dados	*Atributo*	*Total*	*Número de Agregados*	*Número de objetos do mundo real*
Movies				
	Title	3 709	2 260	1 890
	Director	11 359	10 445	6 758
	Genre	778	410	230
Music				
	Artist	1 551	1 123	987
	Song	1 554	590	453
	Album	1 549	1 370	1 102
PosComp				
	Instituição	5 571	2 253	1 014
	Cidade	6 809	3 440	2 428
	Data	5 215	1 720	778
Citation				
	Title	22 710	13 204	5 281
	Journal	9 979	1 197	536
	Publisher	5 790	586	161

Figura 5.13: Tamanho, por atributo, das bases de dados representativas

Domínio	*Atributo/Função*
Movie	Title - Ggrams() Director - JaroWinklerTFIDF() Genre - Levenstein()
Music	Artist - Soundex() Song - JaroWinkler() Album - Qgrams()
PosComp	Instituição - JaroWinklerTFIDF() Cidade - Levenstein() Data - DateSim()
Citation	Title - Grams() Journal - JaroWinkler() Publisher - MongeElkan()

Figura 5.14: Funções de similaridade aplicadas a cada um dos atributos

Como mencionado previamente, foram usadas funções de similaridade específicas para cada atributo, dependendo do domínio de aplicação do atributo. A Figura 5.14 mostra as funções de similaridade que foram aplicadas a cada um dos atributos de cada um dos domínios. Algumas funções usadas estão disponíveis nos pacotes Java *SecondString*[10] e *SimMetrics*[11]. Baseado em experimentos previamente executados (STASIU; HEUSER; SILVA, 2005; DORNELES et al., 2004), foi tentado utilizar funções específicas para cada atributo, dependendo de seu domínio de aplicação. Avaliações e discussões sobre muitas funções de similaridade podem ser encontradas na literatura (COHEN; RAVIKUMAR; FIENBERG, 2003b; LEE, 2001), e esse ponto não caracteriza um foco da tese.

[10] secondstring.sourceforge.net/

[11] sourceforge.net/projects/simmetrics/

Domínio	*Atributo*	*Exemplos de consultas*
Movies		
	Title	"Charlie's Angels II", ..., "Prince of Thieves"
	Director	"Joseph McGinty Nichol", ..., "S. Spielberg"
	Gente	"Animated", ..., "Classical"
Music		
	Artist	"Tora Tora Torrance", ..., "J. Vix"
	Song	"Bury The Hatchet In Yr Chest", ..., "I Love You (We're Evil)"
	Album	"A Cynics Nightmare", ..., "Hope Springs Noturnal"
PosComp		
	Instituição	"II - UFRGS", ..., "Unicamp"
	Cidade	"Porto Alegre", ..., "Sao Pualo"
	Data	"23/11/2003", ..., "nov/11/2003"
Citation		
	Title	"An Introduction to DB Systems", ..., "GARLIC Project"
	Journal	"ACM Computing Surveys", ..., "Information System"
	Publisher	"Prentice-Hall publishers", ..., "Comm. of the ACM."

Figura 5.15: Exemplos de consultas usadas no processo de estimativa

5.2.2 Processo de treinamento

O processo de treinamento efetuado para a geração das Tabelas de Mapeamento de Escores (Definição 4) é descrito abaixo. Este processo foi executado para cada atributo, usando as bases de dados real e representativa. Como existem, ao todo, 12 atributos, os passos abaixo foram executados 24 vezes.

1. **Definição do objeto-consulta**
   Como descrito na Seção 3.1, o processo de estimativa é baseado na evolução dos resultados da execução de uma série de casamentos contra os valores de uma atributo a. Assim, o primeiro passo para a estimativa é a definição de um conjunto de objetos-consulta $Q = \{q_1, q_2, \ldots, q_{40}\}$. Para os experimentos, 40 diferentes valores para cada atributo foram escolhidos, randomicamente, a partir dos valores das bases de dados. Valores duplicados foram eliminados, pois o objetivo é avaliar a habilidade da proposta em encontrar diferentes representações de um objeto do mundo real. Este processo foi executado para a base de dados real e para seu correspondente conjunto de dados representativos. A Figura 5.15 apresenta alguns exemplos de consultas.

2. **Geração do *ranking***
   A seguir, para cada objeto-consulta $q \in Q$ um *ranking* $R(q) = \{a_1, a_2, \ldots, a_n\}$, como apresentado na Definição 2, foi gerado.

3. **Cálculo da precisão**
   Depois da geração do *ranking*, um especialista deve marcar cada valor a_i no ranking como "similar" (ou seja, "representa o mesmo objeto do mundo real que q") ou como "não similar" (ou seja, "não representa o mesmo objeto do mundo real que q"). Finalmente, para cada posição i no *ranking* a precisão nesta posição é calculada usando a Equação 3.1.

   É importante observar que o processo de identificar objetos similares e não similares pode ser executado com intervenção humana reduzida através do processo descrito em (STASIU; HEUSER; SILVA, 2005).

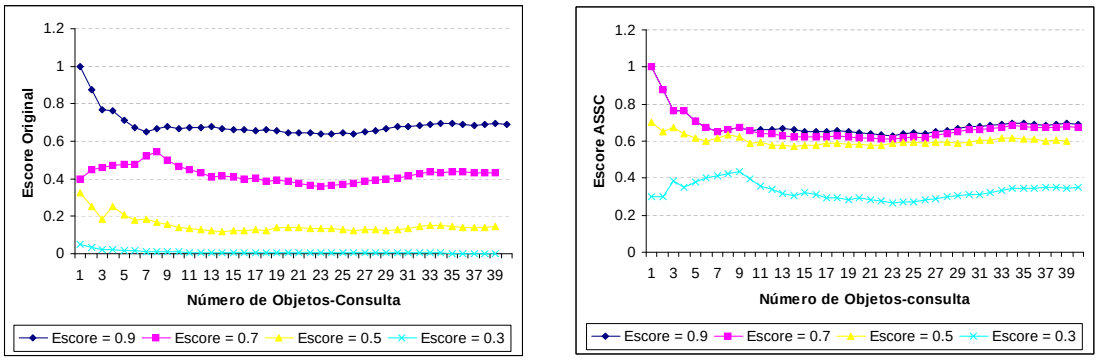
a) Base de dados real

– Escores –	
Original	**ASSC**
1,0	1,0
0,9	0,80875
0,8	0,7025
0,7	0,54125
0,6	0,4175
0,5	0,27
0,4	0,13
0,3	0,06
0,2	0,0425
0,1	0,0325
0,0	0,0125

a) Base de dados representativa

– Escores –	
Original	**ASSC**
1,0	1,0
0,9	0,7475
0,8	0,68125
0,7	0,52875
0,6	0,3175
0,5	0,16625
0,4	0,0175
0,3	0,0
0,2	0,0
0,1	0,0
0,0	0,0

Figura 5.16: TaME para o atributo Instituição do domínio PosComp



A) B)

Figura 5.17: Valores de escore com número crescente de objetos-consulta

A Tabela 3.5 apresenta um exemplo de uma ranking para o atributo Instituição na base de dados real do domínio do PosComp. Um *ranking* similar foi gerado para cada um dos atributos de todos os domínios. O resultado deste passo é um conjunto de 40 *rankings* para cada atributo.

4. **Cálculo da TaME**
   Tendo como entrada os 40 *rankings* gerados no passo anterior, o próximo passo é calcular a *precisão interpolada* (Definição 3) em cada um dos 11 pontos $[0, 0, 0, 1, \ldots, 0, 9, 1, 0]$ de escore. O resultado deste processo é a TaME (Definição 4) para cada atributo.

A Figura 5.16 mostra a TaME para o atributo Instituição, do PosComp, tanto da base de dados real quanto da base de dados representativa.

O número de objetos-consulta usado no processo de estimativa foi estabelecido observando o comportamento dos escores em relação ao aumento do número de objetos-consulta. Em todos os casos, foi observada uma estabilidade nos escores com o número de consultas maior do que 40. Por exemplo, a Figura 5.17 mostra o comportamento para o atributo Director no domínio Citation, na base de dados representativa. A Figura 5.17 A) apresenta a relação do escore original com o número

de objetos-consulta, e a Figura 5.17 B) apresenta a relação do escore ASSC com o número de objetos-consulta. Como pode ser observado na Figura 5.17, os valores do escore, em ambos os casos, tendem a se estabilizar ao redor de 30 objetos-consulta. O mesmo comportamento foi confirmado com experimentos para o restante dos atributos.

5.2.3 Verificação da exatidão dos escores ASSC

Os passos anteriores resultam em duas TaME (Definição 4) para cada atributo: uma tabela (M_a) corresponde à base de dados real e a outra tabela (M_r) corresponde à base de dados representativa. Como já mencionado na Seção 3.1, uma TaME mapeia um escore original para um escore ASSC (função α_M na Definição 5).

Os experimentos apresentados nesta seção visam a comparação dos resultados de precisão obtidos ao se usar o escore original e o escore ASSC. Para isso, usa-se os operadores *GS_Matcher* (Definição 7) e o *ASSC_Matcher* (Definição 8) para consultar a base de dados com diferentes pontos de corte e avaliar a precisão obtida em cada caso.

Os passos executados neste experimento são descritos abaixo. Estes passos foram executados duas vezes para cada atributo, uma vez usando a TaME obtida a partir da base de dados real, e outra vez usando a TaME obtida a partir da base de dados representativa.

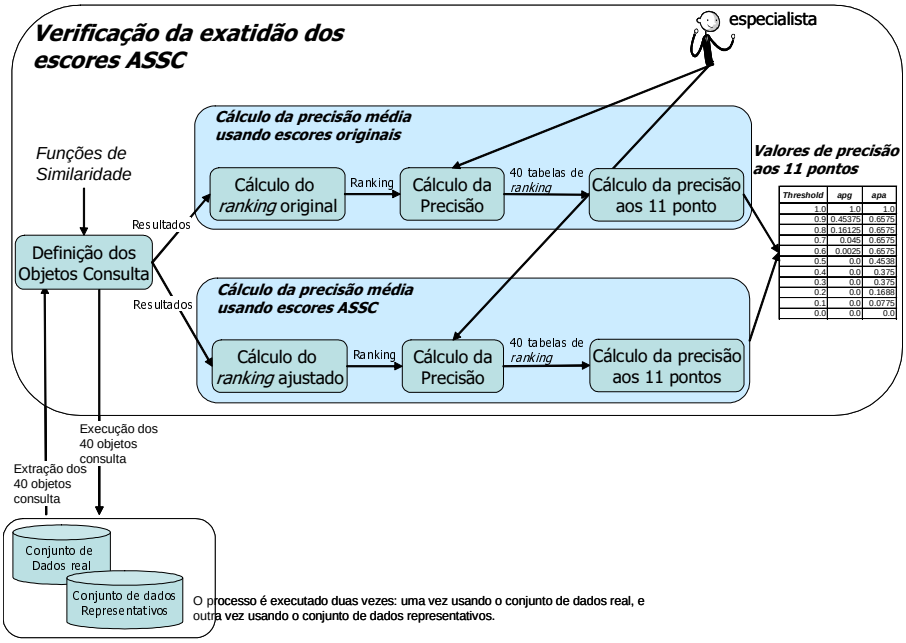
1. **Definição do objeto consulta**
   Um conjunto de 40 objetos-consulta $Q = \{q_1, q_2, \ldots, q_{40}\}$ foi randomicamente escolhido entre os valores de atributo. Da mesma forma que no processo de treinamento, valores duplicados foram eliminados.

2. **Cálculo da precisão média usando escores originais**
   Os passos abaixo descrevem como a precisão média foi calculada para muitos pontos de corte, ou diferentes valores de limiar, quando o escore original obtido diretamente da função de similaridade é aplicado. Isso corresponde a aplicar o operador *GS_Matcher* (Definição 7). Estes passos são similares àqueles executados no processo de treinamento, quando a TaME foi gerada.

   (a) **Cálculo do *ranking***
   Para cada objeto-consulta $q \in Q$ um *ranking* $R(q) = \{a_1, a_2, \ldots, a_n\}$ (Definição 2) foi construído.

   (b) **Cálculo da precisão**
   Após um especialista avaliar cada valor a_i no *ranking* como "similar" (ou seja, "representando o mesmo objeto do mundo real que q") ou como "não similar" (i.e., "não representando o mesmo objeto do mundo real que q"). Finalmente, para cada posição i no *ranking* a precisão neste ponto é calculada através da Equação 3.1.

   (c) **Cálculo da precisão nos 11 pontos**
   Tendo como entrada os 40 *rankings* gerados no passo anterior, a precisão média gap_i em cada um dos 11 pontos de escore original $g_i \in \{0, 0, 0, 1, \ldots, 0, 9, 1, 0\}$ foi calculada. O processo pela qual os valores de precisão são obtidos nos 11 pontos é aquele usado para calcular a precisão interpolada (passo 4 do processo de estimativa, o qual foi formalizado na Definição 3.



Threshold	apg	apa
1.0	1.0	1.0
0.9	0.45375	0.6575
0.8	0.16125	0.6575
0.7	0.045	0.6575
0.6	0.0025	0.6575
0.5	0.0	0.4538
0.4	0.0	0.375
0.3	0.0	0.375
0.2	0.0	0.1688
0.1	0.0	0.0775
0.0	0.0	0.0

Figura 5.18: Visão geral do processo de verificação da exatidão do escore ASSC

3. **Cálculo da precisão média usando escores ASSC**
   Os passos abaixo descrevem como a precisão média foi calculada para muitos pontos de corte, ou diferentes valores de limiar, quando o escore ASSC na TaME é usado. Este passo corresponde a aplicação do operador *ASSC_Matcher* (Definição 8).

   (a) **Geração do *ranking* ajustado**
   Para cada objeto-consulta $q \in Q$ um *ranking* ajustado $R(q) = \{a_1, a_2, \ldots, a_n\}$ é gerado. Um *ranking* é similar ao *ranking original* (Definition 2) com a diferença que, ao invés de ordenar os valores pelo escore original (g), o escore ASSC ($\alpha_M(g)$) correspondente é usada para ordenação dos valores do *ranking*.

   (b) **Cálculo da Precisão**
   Após a geração do *ranking*, para cada posição i no *ranking* a precisão em i é calculada pela Equação 3.1. É importante recordar que no passo 2b um especialista já definiu cada valor em a_i como "similar" ou " não similar".

   (c) **Cálculo da precisão nos 11 pontos**
   Tendo como entrada os 40 *rankings* gerados previamente, a precisão média apa_i é calculada em cada um dos 11 pontos de escore ASSC $g_i \in \{0,0,0,1,\ldots,0,9,1,0\}$. O processo pela qual os valores de precisão são obtidos nos 11 pontos é o mesmo para calcular a precisão interpolada (passo 4 do processo de estimativa) é foi formalizado pela Definição 3.

A Figura 5.18 apresenta uma visão geral dos passos executados no processo de verificação da exatidão dos escores ASSC gerados.

Um exemplo do resultado deste experimento para o atributo Journal na base de dados Citation é mostrada na Figura 5.19. A primeira coluna contém os 11 valores

de ponto de corte para os quais a precisão foi calculada. A segunda coluna (*apg*) contém os valores de precisão que foram calculadas usando o operador *GS_Matcher*, ou seja, valores de precisão alcançados usando o escore original. A terceira coluna (*apa*) contém os valores de precisão gerados pelo operador *ASSC_Matcher*, ou seja, valores de precisão alcançados usando o escore ASSC.

Limiar	apg	apa
1,0	1,0	1,0
0,9	0,45375	0,6575
0,8	0,16125	0,6575
0,7	0,045	0,6575
0,6	0,0025	0,6575
0,5	0,0	0,45375
0,4	0,0	0,375
0,3	0,0	0,375
0,2	0,0	0,16875
0,1	0,0	0,0775
0,0	0,0	0,0

Figura 5.19: Valores de precisão nos 11 pontos para o atributo Journal na base de dados Citation

Um aspecto central na proposta descrita aqui é permitir ao usuário especificar a qualidade do processo de casamento através de um valor de limiar que expresse a precisão esperada nos resultados. Se a proposta descrita aqui trabalha corretamente, a precisão média obtida com os escores ASSC (*apa*) devem ser iguais ao limiar fornecido. Portanto, como o processo de estimativa provavelmente não é perfeito, haverá normalmente diferenças nestes dois valores (escores ASSC e limiar fornecido). Por exemplo, na Tabela 5.19 a precisão média calculada para o valor de limiar $0,5$ é $0,45375$.

A fim de avaliar a proposta descrita, a diferença entre a precisão média, obtida nos experimentos, e a precisão esperada, fornecida por um usuário como um valor de limiar, devem ser avaliadas

A Figura 5.20 mostra os valores de precisão média para os atributos no domínio Citation. As curvas para o atributo Journal aparecem nos dois gráficos no topo, para o atributo Title nos dois gráficos no meio e para o atributo Publisher, os dois últimos gráficos. Os gráficos à esquerda contêm os resultados usando o escore ASSC obtidos com os conjuntos de dados reais, enquanto que os gráficos à direita apresentam os resultados usando os escores ASSC obtidos com os conjuntos de dados representativos. O eixo x representa os valores de limiar e o eixo y apresenta a precisão média para ambos os escores, original e ASSC.

Cada gráfico contém três curvas. A linha contínua, $y = x$ representa o caso ideal em que a precisão média é exatamente a precisão esperada fornecida pelo usuário através do valor de limiar. A linha pontilhada corresponde aos valores apa_i da precisão média alcançada quando se usa os escores ASSC, enquanto que a linha tracejada mostra os valores de precisão média alcançados quando se usa o escore original. Em outras palavras, as linhas pontilhadas refletem a aplicação da proposta apresentada nesta tese, enquanto que a linha tracejada[12] reflete os resultados de

[12]A linha tracejada é usada com um *baseline* para avaliação da proposta apresentada nesta tese.

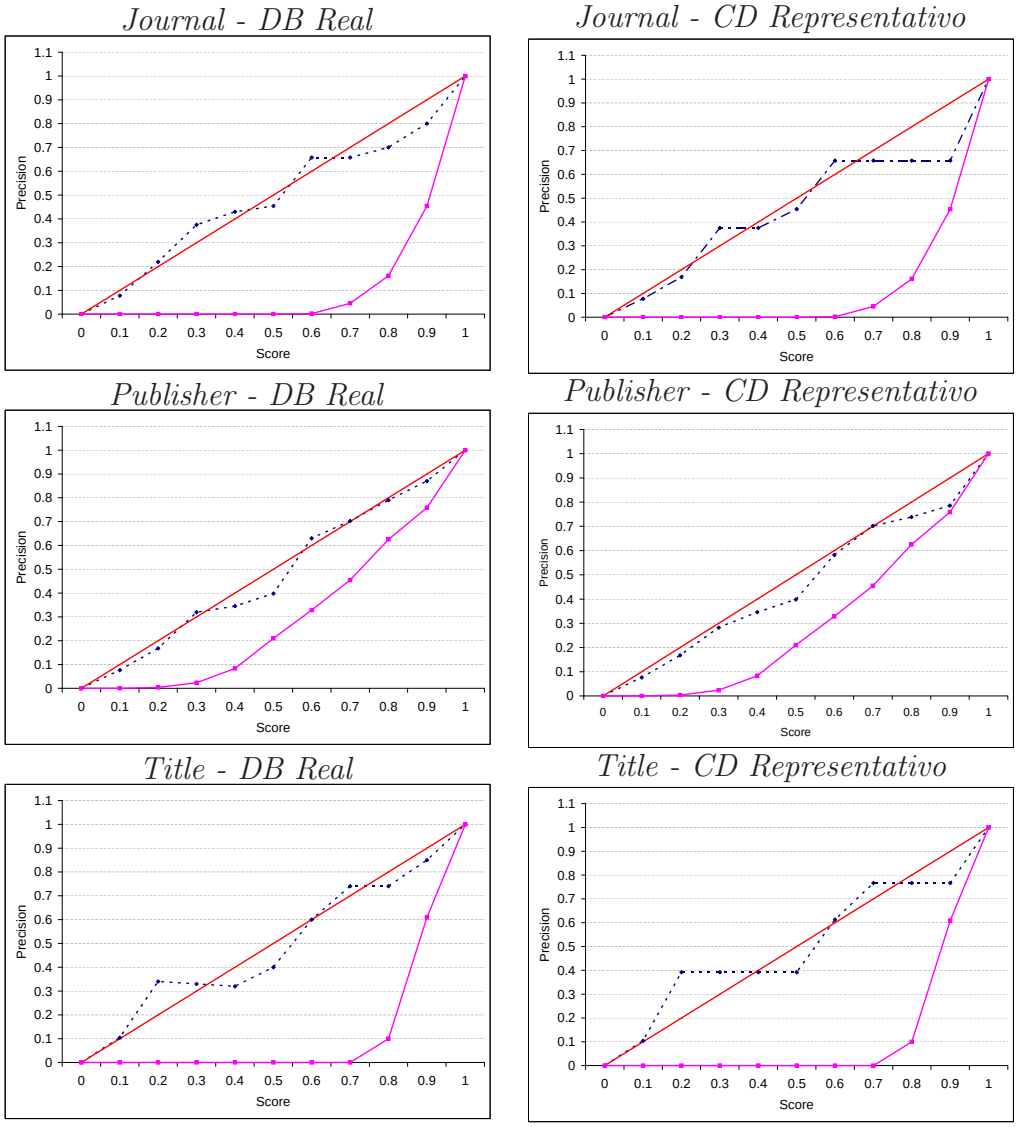


Figura 5.20: Precisão média calculada usando os escores originais e os escores ASSC para o domínio Citation

quando a proposta não é aplicada, ou seja, a abordagem atualmente usada nos trabalhos da área. Como os gráficos claramente mostram, a proposta de uso dos escores ASSC leva a valores que são relacionados a precisão esperada por um usuário (linha contínua), os quais são valores muito próximos do comportamento ideal. É importante enfatizar que estes resultados foram obtidos com um esforço mínimo de treinamento, envolvendo apenas 40 consultas por atributo.

A fim de quantificar as diferenças entre a precisão esperada fornecida pelo usuário através de um valor de limiar e a precisão média obtida representada pelo escore ASSC, foi calculada o desvio dos quadrados das diferenças entre as linhas (como efetuado no método dos mínimos quadrados (RODRIGUES, 2003)). O desvio é dado pela seguinte equação:

$$d_a = \sum_{i=1}^{11} [apa_i - t_i]^2 \tag{5.1}$$

onde ep_i é um dos onze valores de limiar $[0,0,0,1,\ldots,0,9,1,0]$, significando a preci-

são esperada pelo usuário, e apa_i é a precisão média alcançada pelos escores ASSC (passo 3b).

Base de dados	Atributo	d_g	d_a (Conjunto de Dados Representativos)	d_a (Base de Dados Real)
Movies				
	Title	0,2202	0,0088	0,0081
	Director	0,6646	0,0782	0,0214
	Genre	0,1967	0,0374	0,0151
Music				
	Artist	2,1936	0,7701	0,6511
	Song	0,9372	0,0753	0,0665
	Album	0,0812	0,0444	0,0321
PosComp				
	Instituição	0,4262	0,2115	0,1152
	Cidade	0,3499	0,04576	0,02757
	Data	2,8144	0,85	0,667
Citation				
	Title	1,9748	0,0805	0,0446
	Journal	1,9431	0,0941	0,0347
	Publisher	0,4941	0,0325	0,0173

d_g Desvio entre o caso ideal e os resultados obtidos quando a função de similaridade é aplicada diretamente
d_a Desvio entre o caso ideal e os resultados quando o escore ASSC é aplicada nas bases de dados real e representativa

Figura 5.21: Valores de desvio

O desvio entre os resultados obtidos com a proposta descrita nesta tese não é aplicado (linha tracejada), ou seja, quando os resultados da função de similaridade são diretamente aplicadas, com o caso ideal (linha contínua). Este desvio foi calculado como segue. Seja $M = \langle g_0, p_0\rangle, \ldots, \langle g_1 0, p_1 0\rangle\}$ uma TaME. A diferença é dada pela fórmula:

$$d_g = \sum_{i=1}^{11} [apg_i - t_i]^2 \tag{5.2}$$

onde ep_i é um dos onze valores de limiar $[0, 0, 0, 1, \ldots, 0, 9, 1, 0]$ e apg_i é a precisão média calculada usando os escores originais (passo 2c).

A Tabela 5.21 mostra os valores de desvio para cada um dos atributos. A coluna d_g contém dos valores de desvio entre o caso ideal e os resultados obtidos com o escore original. Coluna d_a contém os valores de desvio obtidos com o uso da TaME, os quais resultam tanto da base de dados real quanto da base de dados representativa.

Os resultados obtidos nestes experimentos mostram que, usando a proposta apresentada nesta tese, com o operador *ASSC_Matcher* (Definição 8) é obtido um comportamento muito mais próximo do ideal do que usando os valores originais das funções de similaridade. Este fato pode ser observado nos gráficos da Figura 5.20 e é confirmado pelos baixos valores de desvios mostrados na Tabela 5.21. É importante observar que os resultados apresentados na Tabela 5.21 são, de fato, uma representação compacta dos resultados apresentados na Figura 5.20.

Além disso, os resultados mostram que o processo de estimativa executado para o conjunto de dados representativo leva a resultados muito próximos àqueles obtidos com a base de dados real. Este é um resultado promissor, porque ele sugere que com o conjunto de dados representativo, um processo de estimativa simples para um atributo é suficiente e que o processo de estimativa não precisa ser repetido para cada nova base de dados.

5.3 Combinando escores ASSC em casamento de tuplas

Nesta seção, são relatados experimentos executados para demonstrar que, através da combinação de escores ASSC em casamento de tuplas, são alcançados melhores resultados do que combinar diretamente os escores originais gerados pelos funções de similaridade.

Os experimentos consistem em executar diversos casamentos nas bases de dados reais dos domínios Citation, Music, Movie e PosComp, e avaliá-los usando a clássica curva de Revocação/Precisão (BAEZA-YATES; RIBEIRO-NETO, 1999).

Como um *baseline* para avaliação da proposta, é utilizada a função de similaridade para tuplas (Definição 12) proposta no Capitulo 4, a qual efetua a média entre os valores de escore originais gerados para cada atributo em uma tupla.

Para cada base de dados, os seguintes passos foram executados:

1. 20 objetos-consulta, representados por tuplas, $\{t_1, t_2, \ldots, t_n\}$ os quais foram randomicamente escolhidos na base de dados real.

2. Para cada objeto-conjunto, um *ranking* foi gerado sobre a base de dados. Os dados foram ordenados pela média dos escores originais obtidos para cada atributo, ou seja, pela função apresentada na Definição 12.

3. Para cada objeto-consulta, um *ranking* foi gerado sobre a base de dados. Neste caso, os dados foram ordenados pela média dos escores ASSC obtidos para cada atributo, ou seja, os valores de escores originais são mapeados para os respectivos escores ASSC e média entre estes valores é usada. Nesta etapa, é utilizada a função de similaridade para tuplas, (*tupleSim()* - Definição 12) com a função ajustada de similaridade para os valores dos atributos (Definição 15)

4. Revocação e precisão são calculadas através do procedimento padrão usado pela comunidade de Recuperação de Informação (BAEZA-YATES; RIBEIRO-NETO, 1999) e os resultados são apresentados em curvas de Revocação/Precisão.

A Figura 5.22 apresenta uma visão geral do processo de combinação de escores.

Os resultados destes experimentos são apresentados nos gráficos na Figura 5.23 para as quatro bases de dados reais. As linhas contínuas representam resultados obtidos quando os escores originais são combinados e a linha tracejada representa os resultados obtidos pela combinação dos escores ASSC. Como pode ser observado nos gráficos, em todas as bases de dados, a proposta apresentada levou a melhores resultados.

Para ilustrar os efeitos da proposta apresentada, a Figura 5.24 apresenta dois *rankings* para o objeto-consulta:

```
(PosComp,
   {<Cidade, "Curitiba">},
   {<Instituição, "Pontifícia Universidade Católica do Paraná">},
   {<Data, "21.10.1974"})
```

O *ranking* no topo da figura foi gerado usando a função da Definição 12 e outro *ranking* foi gerado usando valores de escore ASSC. Como pode ser observado, quando se compara as funções usadas para Cidade e Instituição, a função usada para Data leva

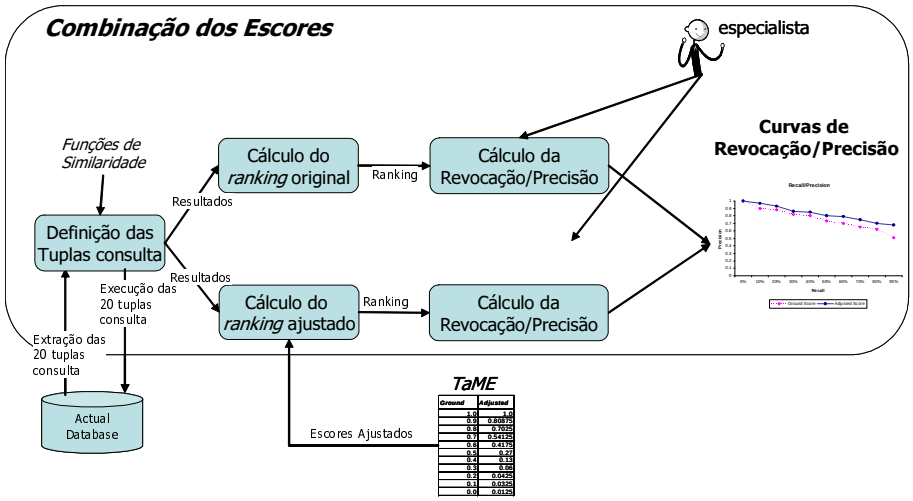


Figura 5.22: Visão geral do processo de combinação de escores

a escores relativamente mais altos. O resultado é que os agregados nas posições 9 e 17, que casariam com o resultado, aparecem depois de agregados que não casam com o objeto-consulta.

No *ranking* gerado com os escores ASSC, os valores gerados pela função de similaridade para datas foram ajustados e se tornaram comparáveis aos escores ASSC gerados pelas duas outras funções de similaridade. O resultado é que os agregados mencionados anteriormente, agora caem com o restante dos agregados relevantes, levando a um melhor resultado final.

5.4 Conclusões

Os experimentos demonstram a exatidão do mapeamento entre escores obtidos pela função de similaridade e a precisão esperada correspondente. Além disso, os experimentos mostram que o processo de estimativa executado em uma base de dados representativa (conjunto de valores de um atributo coletado a partir de diversas fontes simulando o domínio de valores de um atributo) leva a valores que são próximos àqueles obtidos quando o processo de estimativa é executado sobre a base de dados real. Este resultado é promissor, significando que o processo de treinamento pode ser executado uma vez para muitos bancos de dados que incluem dados do mesmo domínio.

A precisão estimada, ou o escore ASSC, pode ser vista como um tipo de escore de similaridade normalizado que é independente dos detalhes de uma função de similaridade específica. Nos experimentos, para cada base de dados, a abordagem proposta levou a melhores resultados do que aqueles quando os escores de funções de similaridade são combinados diretamente. Os experimentos executados para estimar o escore ASSC de uma função de similaridade não são úteis apenas para estimar a precisão propriamente dita, mas também para determinar se uma função de similaridade é adequada ou não para um conjunto de dados. Portanto, acredita-se que o esforço gasto no processo de estimativa é aceitável e é compensado pelo ganho que pode ser conseguido na qualidade das respostas. É importante notar que o mapeamento dos escores originais para os escore ASSC é baseado em estimativa, o que significa que o processo também pode falhar, caso a função de similaridade original

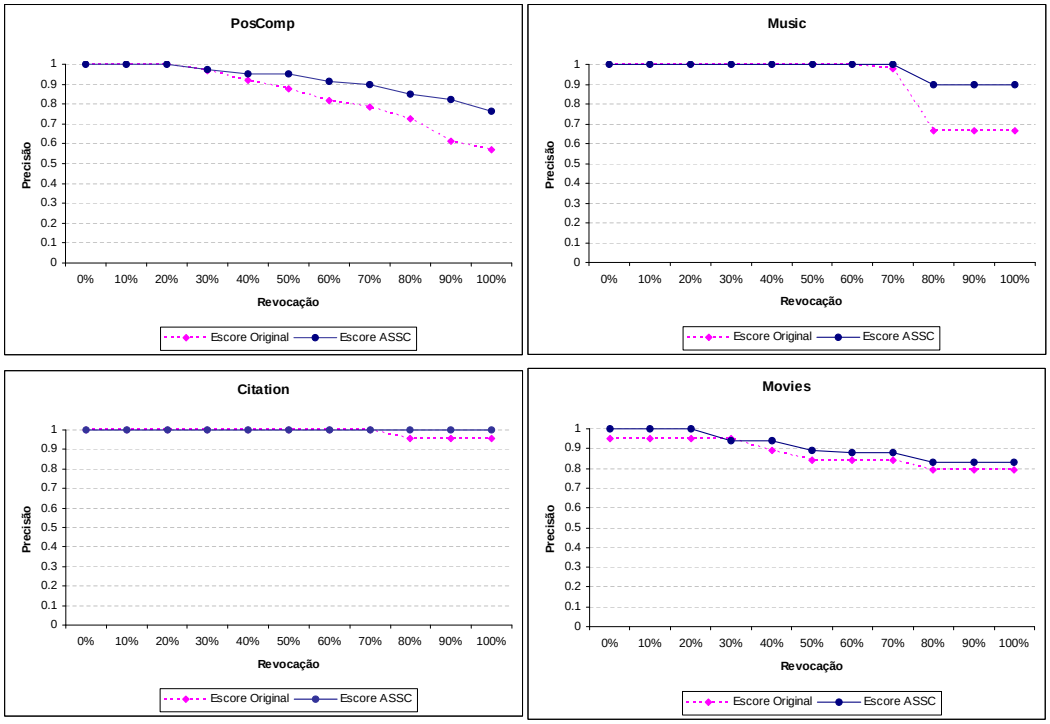


Figura 5.23: Curvas de Revocação/Precisão para casamento de agregados

Ranking usando escores originais

	Cidade		*Data*		*Instituição*			
Pos	**Valor**	**EO**	**Valor**	**EO**	**Valor**	**EO**	**Escore Final**	***Aval.***
1	Curitiba	1	21/10/1974	1	Pontifícia Universidade Católica do Paraná-PUCPR	0.8825	0.9608	**SIM**
2	Curitiba	1	1974.outubro.21	1	Pontificia Universidade Catolica do Parana	0.8452	0.9484	**SIM**
3	Curitiba	1	21/outubro/1974	1	Pontifícia Universidade Católica do Paraná	0.8135	0.9378	**SIM**
4	Curitiba	1	1974.outubro.21	1	Pontifícia Universidade Católica do Paraná	0.8135	0.9378	**SIM**
5	Curitiba	1	21/10/1974	1	Pontificia Univerisdade Catoloica do Prana	0.6695	0.8898	**SIM**
6	Curitiba	1	21/10/1974	1	PUC-PR	0.5345	0.8448	**SIM**
7	Curitiba	1	1976-out-05	0.9514	Universidade Federal do Paraná	0.5543	0.8352	NÃO
8	Curitiba	1	29 de setembro (	0.8508	Universidade Federal do Paraná	0.5543	0.8017	NÃO
9	CURITIBA	1	21/10/1974	1	PUC	0.3780	0.7927	**SIM**
10	CURITIBA	1	29/12/1981	0.8214	UNIVERSIDADE FEDERAL DO PARANÁ	0.5543	0.7919	NÃO
11	Curitiba	1	29/12/1981	0.8214	Universidade Federal do Paraná	0.5543	0.7919	NÃO
12	Curitiba	1	29/12/1981	0.8214	Universidade Federal do Paraná	0.5543	0.7919	NÃO
13	Curitiba	1	29/12/1981	0.8214	Universidade Federal do Paraná	0.5543	0.7919	NÃO
14	Curitiba	1	29/12/1981	0.8214	Universidade Federal do Paraná	0.5543	0.7919	NÃO
15	Curitiba	1	29/12/1981	0.8214	Universidade Tuiuti do Paraná	0.5543	0.7919	NÃO
16	Curitiba	1	16.05.1982	0.8120	Universidade Federal do Paraná	0.5543	0.7888	NÃO
17	Curitiba	1	21/10/1974	1	PUCPR	0.3427	0.7809	**SIM**
18	Curitiba	1	29/12/1981	0.8214	UFPR - Universidade Federal do Paraná	0.4958	0.7724	NÃO

Ranking usando escores ASSC

	Cidade		*Data*		*Instituição*			
Pos	**Valor**	**EA**	**Valor**	**EA**	**Valor**	**EA**	**Escore Final**	***Aval.***
1	Curitiba	1	21/10/1974	1	Pontifícia Universidade Católica do Paraná-PUCPR	0.6813	0.8938	**SIM**
2	Curitiba	1	1974.outubro.21	1	Pontificia Universidade Catolica do Parana	0.6813	0.8938	**SIM**
3	Curitiba	1	21/outubro/1974	1	Pontifícia Universidade Católica do Paraná	0.6813	0.8938	**SIM**
4	Curitiba	1	1974.outubro.21	1	Pontifícia Universidade Católica do Paraná	0.6813	0.8938	**SIM**
5	Curitiba	1	21/10/1974	1	Pontificia Univerisdade Catoloica do Prana	0.3175	0.7725	**SIM**
6	Curitiba	1	21/10/1974	1	PUC-PR	0.1663	0.7221	**SIM**
7	CURITIBA	1	21/10/1974	1	PUC	0.0000	0.6667	**SIM**
8	Curitiba	1	21/10/1974	1	PUCPR	0.0000	0.6667	**SIM**
9	Curitiba	1	1976-out-05	0.02	Universidade Federal do Paraná	0.1663	0.3954	NÃO

EO - Escore Original
EA - Escore ASSC
Aval. - Avaliação de similar ou não similar

Figura 5.24: *Rankings* usando escores original e ASSC

também falhar.

Em relação às funções de similaridade para agregados, pode-se afirmar que são funções bastante intuitivas que podem ser o primeiro passo em direção ao tratamento diferenciado de tuplas e coleções de valores. Experimentos iniciais foram realizados para demonstrar a efetividade das idéias apresentadas, concluindo-se que a diferenciação entre tuplas, listas e conjuntos é um ponto muito importante a ser considerado durante o processo de casamento de agregados.

6 CONCLUSÃO

A proposta apresentada nesta tese envolve duas grandes contribuições: (1) uso da precisão estimada como uma métrica útil para determinar valores de pontos de corte e combinar escores gerados por funções distintas; e (2) definição de funções de similaridade específicas para diferentes tipos de agregado.

No primeiro caso, a precisão é alcançada através de um processo de estimativa. Durante o processo, é definido um mapeamento entre escores originais gerados por uma função de similaridade e a precisão alcançada em cada ponto destes escores. Este mapeamento permite a definição de um operador de casamento que recebe como parâmetro a precisão que se espera como resultado de um processo de casamento. Esta precisão esperada é um valor que é significativo ao usuário, ao contrário do valor de limiar especificado em termos de escores das funções de similaridade.

A proposta foi avaliada através de extensivos experimentos, que confirmaram a hipótese da existência de uma relação entre escores obtidos pela função de similaridade e a precisão esperada correspondente. Como já mencionado, a precisão estimada, ou o escore ASSC, pode ser vista como um tipo de escore de similaridade normalizado que é independente dos detalhes de uma função de similaridade específica. Nos experimentos, para cada base de dados, a abordagem proposta levou a melhores resultados do que aqueles quando os escores de funções de similaridade são combinados diretamente. O esforço gasto no processo de estimativa é aceitável pelo número de objetos-consulta utilizados durante o processo, e é compensado pelo ganho que pode ser conseguido na qualidade das respostas.

Em relação às funções de similaridade para agregados, acredita-se que sejam o primeiro passo em direção ao tratamento diferenciado para diferentes estruturas. Foram definidas funções de similaridade que são dependentes da estrutura, tupla e coleção, além de funções que consideram o domínio de aplicação dos agregados, listas e conjuntos. Neste caso, dependendo do domínio de aplicação, uma ou outra estrutura deve ser usada, e para isso duas outras funções de similaridade foram definidas, para listas e para conjuntos, que são consideradas dependentes da aplicação. Os experimentos demonstram que a idéia apresentada vai em direção a um caminho a ser pesquisado em mais detalhes.

6.1 Contribuições

Um estudo sobre casamento de agregados. Um estudo completo sobre casamento de agregados e os problemas envolvidos e tratados nas diversas áreas de aplicação, como integração de dados, consultas por similaridade e *data cleaning* foi descrito e documentado. Neste estudo, três grandes problemas foram

detectados: (1) não há tratamento de estruturas de coleção de valores; (2) os valores gerados por uma função de similaridade não possuem um significado claro ao usuário; e (3) os valores gerados por uma função de similaridade não são comparáveis entre si, pois não possuem a mesma distribuição sobre uma base de dados. Para cada um dos problemas, uma alternativa de solução foi proposta.

Proposta de um escore significativo para definição de limiar. Proposta de criação de um processo de estimativa de um valor que pode ser substituto do escore original gerado pelas funções de similaridade, a precisão estimada, chamada aqui de escore ASSC. Através do escore ASCC é possível permitir que um usuário especifique a qualidade esperada no processo de casamento usando um valor significativo (a precisão estimada).

Custo de treinamento aceitável. O processo proposto para estimar o valor de escore ASSC tem um custo relativamente aceitável, pois o comportamento das funções de similaridade é avaliada com base em um conjunto bastante limitado de objetos-consulta (40). Como foi demonstrado nos experimentos, este pequeno número de consultas de treinamento é suficiente para alcançar bons resultados de estimativa em todos os domínios e conjuntos de dados experimentados, indicando que a abordagem proposta pode ser facilmente aplicada na prática.

Definição de funções de similaridade específicas para agregados. As funções de similaridade definidas para agregados combinam, de forma distinta, os valores de escores gerados para seus atributos, dependendo da estrutura. Esta estratégia garante resultados mais precisos em processos de casamento de agregados.

Proposta de um escore normalizado para uso em casamento de tuplas. Usando os escores ASSC ao invés dos escores originais gerados pelas funções de similaridade, pode-se garantir que os valores combinados são todos referentes a um mesmo escore, que é a precisão estimada.

Demonstração da eficácia do uso de bases de dados representivas. Através dos experimentos, foram demonstradas duas alternativas de conjuntos de dados que podem ser usados durante o processo de treinamento: os conjuntos de dados real e representativo. Usando o conjunto de dados real para treinamento, os resultado obtidos são mais apurados pois o mesmo conjunto de valores é usado durante o processo de treinamento e o casamento de agregados. Por outro lado, o processo de treinamento deve ser repetido a cada nova base de dados usada. Além disso, se a base de dados sofreu alguma alteração significativa nos dados, um novo treinamento se faz necessário. Para a segunda alternativa, o uso de um conjunto de dados representativo, um único treinamento deveria ser adotado para qualquer conjunto de instâncias de um atributo, e usado em qualquer aplicação que possua o atributo com o mesmo domínio. No entanto, os escores ASSC podem não ser tão apurados quanto aqueles conseguidos sobre a base de dados real.

6.2 Relação com estado da arte

Nesta seção, algumas características dos trabalhos propostos por Tejada (TEJADA; KNOBLOCK; MINTON, 2001), Guha (GUHA et al., 2004) e Bilenko (BILENKO; MOONEY, 2003) são resumidas com o objetivo de fazer uma análise da sua relação com a abordagem proposta nesta tese. A escolha destes trabalhos para a análise se justifica por serem aqueles que apresentam propostas de combinação de escores de atributos em processos de casamentos de tuplas, e fazer uso dos atributos comuns para a combinação de escores. As demais propostas apresentadas no Capítulo 2 são mais dirigidas a processos que não tratam necessariamente de tuplas.

As características escolhidas para esta comparação são apresentadas na Tabela 2.1 e podem ser descritas da forma como segue. A coluna Processo indica qual a estratégia utilizada para gerar o escore de similaridade a partir da comparação de tuplas. A coluna Escore indica que valor de escore é usado para cálculo do valor final na comparação de dois objetos. Os valores apresentados nesta coluna indicam se o escore é o originalmente gerado por uma função, ou se algum processo de normalização, ou ajuste, foi efetuado antes do processo de combinação. Na coluna Estruturas consideradas são descritas quais os tipos de estrutura de objetos complexos são tratados em cada um dos trabalhos. Por fim, a última coluna Estratégia de combinação indica qual a abordagem usada para combinar os escores gerados para os atributos dos objetos. Uma característica comum a todos os trabalhos é a proposta do uso de diferentes funções de similaridade na comparação dos atributos de tuplas.

Tabela 6.1: Características dos trabalhos

Trabalho	Processo	Escores Usados	Estruturas consideradas	Estratégia de combinação
ASSC	Qualquer um	Valores de precisão estimada	Tuplas com atributos compartilhados e coleções	Qualquer um
Active Atlas	Árvore de decisão	Valores originais gerados pelas funções de similaridade*	Tuplas com atributos compartilhados	Árvore de decisão com valor de limiar para atributos
MARLIN	SVM	Valores originais gerados pelas funções de similaridade	Tuplas com atributos compartilhados	Valor de limiar e combinação por função
Fusão de *rankings*	SSP em um grafo bipartido	Valores originais gerados pelas funções de similaridade	Tuplas com atributos compartilhados	Combinação de resultados de *rankings* gerados com estratégias de *top-k*

* Em trabalhos posteriores (TEJADA; KNOBLOCK; MINTON, 2002), os autores propões o uso de escores ajustados através de pesos.

De forma geral, pode-se dizer que a estratégia apresentada nesta tese pode ser usada juntamente com qualquer uma das propostas apresentadas no Capítulo 2.

Assim pode-se afirmar que o trabalho desenvolvido é um complemento ao trabalho existente na literatura.

Dificilmente pode-se dizer que a estratégia apresentada nesta tese é melhor ou pior do que os trabalhos citados, pois ela é uma ferramenta muito útil nos processos de escolha de limiar, como a apresentada no sistema Active Atlas, proposto por Tejada et al. (TEJADA; KNOBLOCK; MINTON, 2001). No Active Atlas, o uso de escores ASSC poderia ser facilmente utilizado como valores de limiar na árvore de decisão. Da mesma forma, o uso de escores ASSC pode ser de grande utilidade no sistema MARLIN, como entrada para o classificador binário. Dos trabalhos citados, aquele onde o uso escores ASSC deveria ser analisado com mais atenção, para identificar a real utilidade, seria na proposta de fusão de *rankings*, pois naquele trabalho não se requer valores de limiar para limitar uma resposta e sim a indicação das k respostas mais similares. No entanto, poderia-se supor que só entrariam nas k respostas aqueles objetos recuperados com $x\%$ de precisão estimada. Apesar de não ter sido incluído nesta seção, o trabalho desenvolvido por Doan et al. (DOAN et al., 2003) pode fazer uso dos escores ASSC no módulo avaliador de similaridade, onde um valor de limiar é usado para descartar pares de agregados que não terão seus atributos disjuntos testados.

6.3 Sub-produtos da tese

Alguns sub-produtos foram criados durante o desenvolvimento da tese, e são apresentados a seguir. Estes sub-produtos são baseados em idéias apresentadas nesta tese, as quais foram implementadas no sistema Eris, no FAPA, e na linguagem SimXPath. O primeiro permite a avaliação das consultas apresentadas no Capítulo 5, na fase de combinação de escores em agregados. O protótipo FAPA é um auxiliar no processo de estimativa dos escores ASSC. A linguagem SimXPath define o uso das funções de similaridade para agregados.

Eris. Os primeiros estudos desenvolvidos durante a tese geraram um trabalho de conclusão de curso (LIMA, ????), cujo objetivo foi o estudo, criação e prototipação de função de similaridade para valores atômicos. Eris é uma ferramenta destinada a realizar consultas por similaridade em bases de dados XML. A ferramenta foi implementada em Java e possibilita a escolha de algumas funções de similaridade, bem como do valor de limiar a ser aplicado para a criação do *ranking* dos resultados. A partir de uma consulta, todos os arquivos constantes em um diretório são processados e comparados com a mesma, na tentativa de encontrar aqueles que mais se aproximam da requisição do usuário.

FAPA - Ferramenta de Apoio ao Processo de Avaliação. A estratégia proposta para o processo de estimativa dos valores de precisão foi implementada em um trabalho de conclusão de curso. FAPA é uma ferramenta para automatização do processo de avaliação de funções de similaridade. A ferramenta provê o usuário (ou avaliador de funções de similaridades) a facilidade da inclusão de novas funções de similaridade, bem como a geração automática de gráficos. Ao usuário cabe a escolha de quais funções avaliar, qual base de dados deve ser utilizada para a avaliação ser efetuada, a escolha dos objeto consulta e a identificação dos objetos relevantes para o objeto consulta. A ferramenta foi desenvolvida em trabalho de conclusão de curso (GROSSI, ????).

Linguagem de consulta SimXPath. O trabalho desenvolvido sobre funções de similaridade criadas para agregados, apresentadas no Capítulo 4, originou um trabalho de Dissertação de Mestrado (PADILHA, 2005. 61 p.; PADILHA; DORNELES; HEUSER, 2005). O trabalho desenvolvido foi a implementação de um protótipo que processa consultas em uma linguagem que é extensão da XPath, a SimXPath (suporte à Similaridade em XPath), que permite o uso de funções de similaridade em expressões de caminho. O trabalho foi de autoria de Alvaristo Padilha, orientado pelo Prof. Carlos Alberto Heuser, no Instituto de Informática, UFRGS.

6.4 Artigos

6.4.1 Submetidos

1. DORNELES, Carina F.; HEUSER, Carlos A.; SILVA, Altigran S.; MOURA, Edleno S. A Generic Strategy for Combining Similarity Metrics in Approximate Matching. Information Systems, Elsevier B.V. Submetido em 20 de Dezembro de 2005 (DORNELES et al., ????);

6.4.2 Publicados

Os seguintes são os artigos diretamente resultantes do trabalho desenvolvido durante a tese:

1. DORNELES, Carina F.; HEUSER, Carlos A.; SILVA, Altigran S.; MOURA, Edleno S. Measuring Similarity Between Collection of Values. WIDM, 2004. Washington DC, USA (DORNELES et al., 2004).

2. DORNELES, Carina F.; HEUSER, Carlos A. Similarity Queries over Semistructured data. In WTDBD - I Workshop de Teses e Dissertações em Banco de Dados, in conjunction with SBBD'2002, Gramado, Rio Grande do Sul, Brasil, 17 Outubro, 2002 (DORNELES; HEUSER, 2002a)

3. DORNELES, Carina F.; LIMA, Andrei; HEUSER, Carlos A. Eris: Uma ferramenta para integração de dados XML. I Sessão de Demos, SBBD, 2004, Brasília.

4. PADILHA, Alvaristo; DORNELES, Carina F.; HEUSER, Carlos A. Suporte a argumentos de consulta vagos através da XPath. I Escola Regional de Banco de Dados, I ERBD, 2005, Porto Alegre (PADILHA; DORNELES; HEUSER, 2005).

Alguns trabalhos não estão diretamente relacionados com a tese, mas foram importantes no desenvolvimento dos primeiros estudos.

1. VITORI, Cesar; DORNELES, Carina F.; HEUSER, Carlos A. Creating XML Documents from Relational Data Sources. Second International Second International on Electronic Commerce and Web Technologies - EC-Web, 2001. páginas 60-70 (VITTORI; DORNELES; HEUSER, 2001);

2. WENZEL, Maíra Colling. Uma interface visual para a linguagem XML/SQL. Trabalho de Conclusão de Curso, Instituto de Informática, UFRGS. Dezembro de 2001. Carlos A. Heuser (orientador); Carina F. Dorneles (co-orientador);

Alguns relatórios técnicos foram também desenvolvidos. Eles apresentam resultados de experimentos realizados inicialmente até se chegar ao resultado final da tese.

1. DORNELES, Carina F.; LIMA, Andrei Eneas Nunes; HEUSER, Carlos A. Experiments on similarity query over XML data sources. Technical Report, RP-337, Universidade Federal do Rio Grande do Sul, 2003 (DORNELES; LIMA; HEUSER, 2003a).

2. DORNELES, Carina F.; HEUSER, Carlos A. Consulta por similaridade a dados semiestruturados. Relatorio Tecnico, RP-326, Universidade Federal do Rio Grande do Sul, 2003 (DORNELES; LIMA; HEUSER, 2003b).

3. DORNELES, Carina F.; NADVORNY, Cesar Feijo; KROTH, Eduardo; LIMA, Andrei Eneas Nunes; HEUSER, Carlos A. Pesquisa por similaridade em dados XML. Relatorio Tecnico, RP-327, Universidade Federal do Rio Grande do Sul, 2003 (DORNELES et al., 2003).

4. DORNELES, Carina F.; HEUSER, Carlos A. Applying visual information retrieval techniques to querying semistructured databases. Submetido ao 9th String Processing and Information Retrieval, SPIRE'02, Lisbon, Portugal, 2002 (DORNELES; HEUSER, 2002b)

5. DORNELES, Carina F.; PADILHA, Alvaristo; HEUSER, Carlos A. Suporte a argumentos de consultas vagos através da XPath. Technical Report, RP-347, Universidade Federal do Rio Grande do Sul, 2004 (DORNELES; PADILHA; HEUSER, 2004a);

6. DORNELES, Carina F.; LIMA, Andrei Eneas Nunes; HEUSER, Carlos A.; DA SILVA, Altigran; DE MOURA, Edleno S. Accessing XML data by allowing vague query arguments. Technical Report, RP-342, Universidade Federal do Rio Grande do Sul, 2004 (DORNELES; PADILHA; HEUSER, 2004b).

Finalmente, o trabalho desenvolvido nesta tese originou Projetos de Trabalho de Conclusão de Curso.

1. GROSSI, Marcelo Aguiar. Avaliando a performance das funções de similaridade através da revocação e precisão. Trabalho de Conclusão de Curso, Instituto de Informática, UFRGS. Dezembro de 2005. Carlos A. Heuser (orientador); Carina F. Dorneles (co-orientador);

2. CONY, Carlos; BORGES, Eduardo Nunes. Implementação de funções de similaridade no PostgreSQL. Trabalho de Conclusão de Curso, Engenharia da Computação. FURG - Fundação Universidade Federal de Rio Grande. Dezembro de 2005. André Freitas (orientador); Carina F. Dorneles (co-orientador);

3. SUDER, Rodrigo. Integração de fontes XML usando algoritmos genéticos. Trabalho de Conclusão de Curso, Curso de Ciência da Computação, UPF. TC-I Finalizado em Dezembro de 2005. TC-II em andamento. Carina F. Dorneles (orientador);

4. SANTOS, Rodrigo. Clustering de documentos XML. Trabalho de Conclusão de Curso, Instituto de Informática, UFRGS. Dezembro de 2003. Carlos A. Heuser (orientador); Carina F. Dorneles (co-orientador);

5. LIMA, Andrei Enéas Nunes. Pesquisa de Similaridade em XML. Trabalho de Conclusão de Curso, Instituto de Informática, UFRGS. Dezembro de 2002. Carlos A. Heuser (orientador); Carina F. Dorneles (co-orientador);

6.5 Trabalhos Futuros

Esta seção apresenta duas direções para trabalhos a serem desenvolvidos: (1) as limitações que a proposta descrita apresenta e que podem ser corrigidas; e (2) algumas propostas de trabalhos adjacentes que podem se tratados.

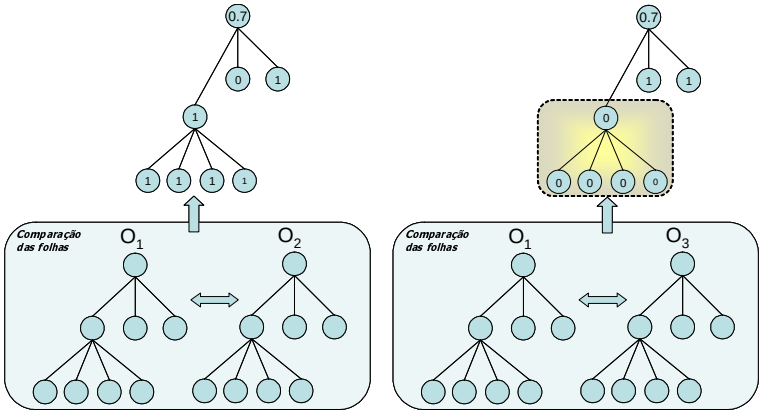


Figura 6.1: Exemplificação do problema de pesos em estruturas aninhadas

Utilização de pesos em estruturas aninhadas. Durante a execução de alguns experimentos sobre bases de dados XML, em que foram utilizadas estruturas aninhadas, detectou-se o problema de que elementos que estão mais acima na hierarquia possuem maior peso quando os valores são combinados. Por exemplo, considerando-se a estrutura apresentada na Figura 6.1, com os valores de escore gerados para cada um dos elementos e o valor final. Para este problema, deve-se levar em conta "similaridade horizontal" medida nos termos mostrados no exemplo, e "similaridade vertical", medida em termos da profundidade de um caminho em uma árvore para chegar a um mesmo elemento em dois objetos.

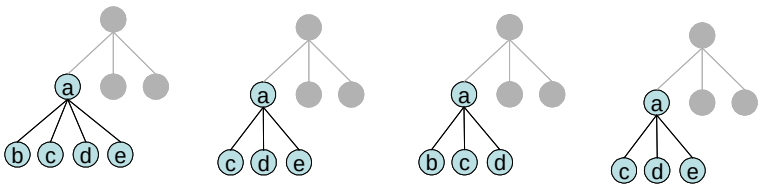


Figura 6.2: Exemplificação do problema de avaliação das funções de similaridade para agregados.

Avaliação das funções de similaridade para agregados. Outra questão importante que necessita de análise, também considerando objetos aninhados,

é a que diz respeito aos escores gerados pelas funções de similaridade para agregados. Neste caso, existe a necessidade de uma análise sobre a estimativa de precisão das funções de similaridade para agregados, além das estimativas efetuadas para funções aplicadas a valores atômicos. O maior problema que surge neste caso, está exemplificado na Figura 6.2, onde para um mesmo agregado a, diversas variações de componentes podem ser identificadas. Desta forma, haveria a necessidade de se estimar a precisão para todas as possíveis estruturas.

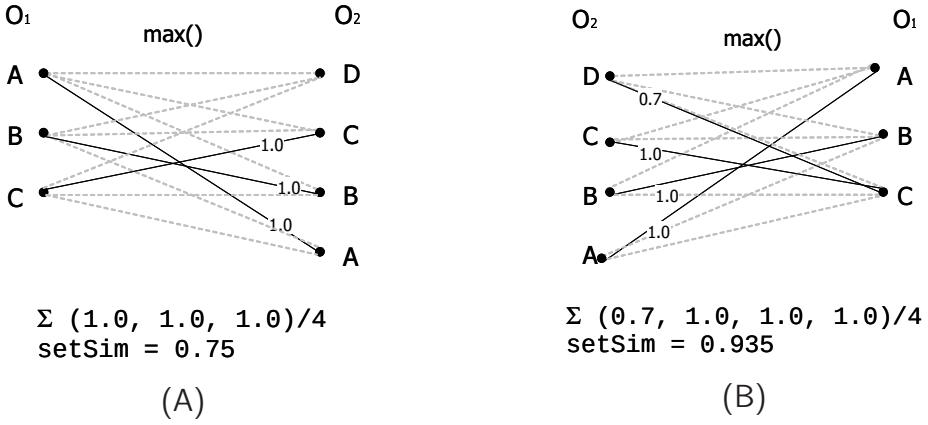


Figura 6.3: Exemplo da assimetria da função de similaridade *setSim()*

Resolução da assimetria da função para conjunto de dados. A função de similaridade definida para conjuntos não é uma função simétrica quando os agregados que estão sendo comparados possuem número diferente de atributos ($m \neq n$). A Figura 6.3 apresenta um exemplo do problema. Pode-se observar que tendo número diferente de atributos ($m \neq n$), o número de comparações também será diferente. A solução apresentada nesta tese é uma proposta simplista para casamento de conjuntos. Este problema tem sido estudado na literatura sob o nome de *The Stable Marriage Problem* (GUSFIELD; IRVING, 1999; MAIRSON, 1992) e usado em muitos trabalho de casamento de dados.

Uso de escores ajustados gerados através da revocação estimada. Se ao invés de utilizar a precisão estimada, a revocação estimada fosse usada, seria possível que os usuários especificassem os resultados esperados através da indicação da porcentagem do total de pares similares existentes na base de dados que se espera que sejam recuperados. Assim, seria possível permitir a indicação dos resultados relevantes que se deseja recuperar. Fundamentalmente, a idéia apresentada nesta tese é permitir ao usuário especificar os resultados esperados a partir do casamento através da indicação da porcentagem de resultados corretos requeridos. Entretanto, o uso da medida de revocação poderia ser perfeitamente utilizada, pois trata-se de outra medida muito intuitiva quanto ao significado de sua avaliação. Alguns resultados preliminares foram obtidos mostrando que esta idéia é aplicável.

Redução da intervenção humana. Na abordagem proposta, intervenção humana é requerida durante o processo de treinamento a fim de rotular cada valor como "similar" ou "não similar". Uma alternativa para reduzir está intervenção humana pode ser a abordagem apresentada em (STASIU; HEUSER;

SILVA, 2005). Na proposta, um usuário especialista indica o número de objetos distintos existentes no banco, e posteriormente o sistema aplica um processo de agrupamento (*clustering*) em que cada cada grupo deve conter todas as representações de cada objeto contido na base de dados. O número de grupos criados corresponde ao número indicado inicialmente pelo usuário.

Escolha da melhor função de similaridade para valores atômicos. Neste caso, a proposta seria a utilização dos resultados obtidos no processo de treinamento para escolher a melhor função de similaridade a ser utilizada em cada um dos atributos atômicos. O processo poderia funcionar, basicamente, da seguinte forma: dadas várias funções de similaridade e diferentes conjuntos de dados de atributos, encontrar a mais adequada a um domínio de valores, de acordo com a tabela de mapeamentos de escore (TaME) gerada. Quanto mais precisos os resultados apresentados na TaME, mais adequada seria a função.

Utilização de mais de uma função de similaridade para o mesmo atributo. Em alguns casos, uma única função de similaridade pode não ser suficiente para detectar variações em um par de valores. Por exemplo, considerando os valores s_1 = "Universidade Federal do Rio Grande do Sul - UFRGS" s_2 = "UFRGS" pode-se observar a representação de um mesmo objeto duas vezes em s_1. A comparação de s_1 com s_2 provalvemente resulta em um valor não muito alto de similaridade, mesmo sendo representações do mesmo objeto do mundo real. Esta é uma representação muito comum para identificar nomes de instituições, mas uma simples função de similaridade, como as existentes na literatura, não resolve este problema. Uma solução alternativa, seria a utilização de mais de uma função, onde uma trataria a comparação entre as siglas, e outra entre a sigla e o nome por extenso, e por fim efetuaria-se a combinação dos resultados.

Uso de técnicas de aprendizado de máquina no processo de treinamento. Utilizar técnicas de aprendizado de máquina para minimizar o trabalho do usuário durante o processo de estimativa dos escores ASSC. O processo utilizado atualmente para a geração dos escores ASSC é completamente manual. As 40 consultas submetidas aos conjuntos de dados são extraídas por um usuário e avaliadas por ele. Este processo de treinamento poderia ser efetuado através de uma técnica de aprendizado de máquina, supervisionado ou não, como uma rede neural que pudesse encontrar os padrões de valores de um determinado atributo, e assim definir exatamente o conjunto de valores completo que pode ser possível para um determinado atributo, com todas as possíveis representações de um mesmo objeto do mundo real. Através deste treinamento, os resultados de precisão estimada seriam muito mais exatos.

Combinação de escores ASSC com estratégias de *rankings*. Combinar valores de escores ASSC gerados para cada atributo de um agregado com estratégias de *ranking*. No trabalho proposto em (GUHA et al., 2004), os autores sugerem o uso de técnicas aplicadas ao problema de *top-k* para gerar *rankings* individuais de cada cada atributo de um agregado, e posteriormente fundem estes *rankings* utilizando um novo algoritmo , o *SSP - Successive Shortest Paths*. Como trabalho futuro, usando a proposta descrita neste tese, a mesma

fusão de *rankings* poderia ser executada, entretanto, como estratégia aplicada ao problema de restrição no número de respostas, utilizaria-se um limiar de similaridade. Desta forma, cada *ranking* individual seria gerado com resultados limitados por um limiar, definido através de um escore ASSC, indicando assim que o usuário apenas deve especificar a precisão esperada nos resultados.

Comparação experimental. Realizar comparação com trabalhos relacionados, testando a qualidade dos resultados. Esta proposta pode ser executada de duas formas, conforme apresentado a seguir.

Comparação com proposta de fusão de escores. No trabalho apresentado em (GUHA et al., 2004), os autores apresentam os resultados de um estudo de caso, com variação de parâmetros, a fim de efetuar uma comparação de performance dos algoritmos envolvidos na proposta. Nenhum resultado relativo à qualidade dos resultados é descrita. Desta forma, poderia ser interessante a comparação da qualidade dos resultados conseguidos com a proposta apresentada pelos autores com a proposta apresentada aqui.

Comparação com proposta de uso das árvores de decisão. Na proposta apresenta em (TEJADA; KNOBLOCK; MINTON, 2001), o valor de escore resultante da comparação entre dois objetos é gerado com uma árvore de decisão. Como já discutido, estas árvores possuem regras de mapeamento que contêm informação sobre quais as combinações de atributos são importantes para determinar o mapeamento entre dois objetos, bem como os valores de limiar para os escores de similaridade para cada atributo. Seria interessante utilizar os escores ASSC como valores destes limiar.

Resolução de conflitos. Resolver conflitos de valores encontrados entre os atributos de um par de agregados. Como exemplo, já mencionado no Capítulo 2, uma fonte de dados possui informações sobre produtos e preços, cuja região que abrange é a Comunidade Européia, enquanto que outra fonte possui armazena informações sobre a América do Norte. As fontes podem tratar das mesmas informações referentes aos mesmos produtos, no entanto, uma trata com valores em Euros enquanto que a outra armazena seus dados em Dólares Americanos.

Uso de atributos disjuntos. Utilizar os atributos não compartilhados por um par de agregados como auxiliares no processo de casamento. Este problema pode ser visto como casamento de agregados com estruturas diferentes. Para este problema, o uso de um modelo conceitual, gerado a partir da integração dos esquemas das fontes, pode ser útil no processo, pois através dele é possível identificar mapeamentos existentes entre os conceitos. Por exemplo, dadas duas fontes de dados, um agregado de uma fonte pode possuir um atributo tempoTrab que possui como valor o tempo que um funcionário trabalha em uma empresa, e um agregado de outra fonte pode ter o atributo datAdmis indicando a data de admissão de funcionário em uma empresa. Pode-se utilizar os valores destes atributos disjuntos para encontrar alguma relação entre um par de agregados.

REFERÊNCIAS

AGRAWAL, R.; FALOUTSOS, C.; SWAMI, A. N. Efficient Similarity Search In Sequence Databases. In: INTERNATIONAL CONFERENCE ON FOUNDATIONS OF DATA ORGANIZATION AND ALGORITHMS, 4., 1993, Chicago, Illinois, USA. **Proceedings...** Berling: Springer-Verlang, 1993. p.69–84.

AHUJA, R.; MAGNANTI, T.; ORLIN, J. **Network Flows**. [S.l.]: Prentice Hall, 1993.

AL-KHALIFA, S.; YU, C.; JAGADISH, H. V. Querying structured text in an XML database. In: INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 2003. **Proceedings...** New York:ACM Press, 2003. p.4–15.

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

BARIONI, M. C. N.; RAZENTE, H. L.; TRAINA-JR., C.; TRAINA, A. J. M. Querying complex objects by similarity in SQL. In: SIMPÓSIO BRASILEIRO DE BANCOS DE DADOS, SBBD, 20., 2005, Uberlândia, MG, Brazil. **Anais...** [S.l.: s.n.], 2005. p.130–144.

BEYER, K.; GOLDSTEIN, J.; RAMAKRISHNAN, R.; SHAFT, U. When Is "Nearest Neighbor" Meaningful? In: INTERNATIONAL CONFERENCE ON DATABASE THEORY, ICDT, 7., 1999. **Proceedings...** [S.l.: s.n.], 1999.

BILENKO, M.; MOONEY, R.; COHEN, W.; RAVIKUMAR, P.; FIENBERG, S. Adaptive Name Matching in Information Integration. **IEEE Intelligent Systems**, [S.l.], v.18, n.5, p.16–23, September/October 2003.

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

BOLLACKER, K.; LAWRENCE, S.; GILES, C. L. CiteSeer: an autonomous web agent for automatic retrieval and identification of interesting publications. In: INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS, 2., 1998, New York. **Proceedings...** New York:ACM Press, 1998. p.116–123.

BOSER, B. E.; GUYON, I.; VAPNIK, V. A training algorithm for optimal margin-classifiers. **Computational Learing Theory**, [S.l.], p.144–152, 1992.

BRUNO, N.; CHAUDHURI, S.; GRAVANO, L. Top-k selection queries over relational databases: mapping strategies and performance evaluation. **ACM Trans. Database Syst.**, New York, NY, USA, v.27, n.2, p.153–187, 2002.

BUENO, R.; TRAINA, A. J. M.; TRAINA-JR., C. Accelerating approximate similarity queries using genetic algorithms. In: SYMPOSIUM ON APPLIED COMPUTING, SAC, 2005, Santa Fe, New Mexico, USA. **Proceedings...** [S.l.: s.n.], 2005. p.617–622.

CALVANESE, D.; GIACOMO, G. D.; ROSATI, R. Data Integration and Reconciliation in data Warehousing: conceptual modeling and reasoning support. **Networking and Information Systems**, [S.l.], v.2, n.4, p.413–432, 1999.

CARMAN, M. J.; KNOBLOCK, C. A. Inducing source descriptions for automated web service composition. In: AAAI WORKSHOP ON EXPLORING PLANNING AND SCHEDULING FOR WEB SERVICES, GRID, AND AUTONOMIC COMPUTING, 2005. **Proceedings...** Menlo Park:AAAI Press, 2005.

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

CHAPMAN, S. **SimMetrics**: a Java & C# .NET Library of Similarity Metrics. Available at: http://sourceforge.net/projects/simmetrics/. Visited on 05.01.2006.

CHAUDHURI, S.; GANJAM, K.; GANTI, V.; MOTWANI, R. Robust and efficient fuzzy match for online data cleaning. In: INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, ACM SIGMOD, 2003, New York, USA. **Proceedings...** New York:ACM Press, 2003. p.313–324.

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

CHENG, R.; XIA, Y.; PRABHAKAR, S.; SHAH, R.; VITTER, J. S. Efficient Indexing Methods for Probabilistic Threshold Queries over Uncertain Data. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 30., 2004. **Proceedings...** San Francisco: CA:Morgan Kaufmann, 2004. p.876–887.

CHINENYANGA, T. T.; KUSHMERICK, N. An expressive and efficient language for XML information retrieval. **JASIST**, [S.l.], v.53, n.6, p.438–453, 2002.

CIACCIA, P.; PATELLA, M.; ZEZULA, P. M-tree: an efficient access method for similarity search in metric spaces. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 23., 1997, Athens, Greece. **Proceedings...** Morgan Kaufmann, 1997. p.426–435.

COHEN, W. W. Integration of Heterogeneous Databases Without Common Domains Using Queries Based on Textual Similarity. **SIGMOD Record (ACM Special Interest Group on Management of Data)**, [S.l.], v.27, n.2, p.201–212, June 1998.

COHEN, W. W. Providing Database-like Access to the Web Using Queries Based on Textual Similarity. **SIGMOD Record (ACM Special Interest Group on Management of Data)**, [S.l.], v.27, n.2, p.558–560, June 1998.

COHEN, W. W. Recognizing Structure in Web Pages using Similarity Queries. In: NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, AAAI, 16., 1999, Orlando, Florida, USA. **Proceedings...** Menlo Park:AAAI Press, 1999. p.59–66.

COHEN, W. W.; KAUTZ, H. A.; MCALLESTER, D. A. Hardening soft information sources. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, ACM SIGKDD, 6., 2000, Boston, MA, USA. **Proceedings...** [S.l.: s.n.], 2000. p.255–259.

COHEN, W. W.; RAVIKUMAR, P.; FIENBERG, S. **SecondString**: open source java-based package of approximate string-matching. Available at: http://secondstring.sourceforge.net/. Visited on 20.12.2005.

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, IIWEB, 2003. **Proceedings...** [S.l.: s.n.], 2003. p.73–78.

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

COOPER, R. **Object Database**: an ODMG approach. (part two). [S.l.]: International Thomson Computer Press, 1997. p.97 – 220.

DEY, D.; SARKAR, S. A Probabilistic Relational Model and Algebra. **ACM Transactions on Database Systems**, [S.l.], v.21, n.3, p.339–369, Sept. 1996.

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, 2003.

DORNELES, C. F.; HEUSER, C. A. Similarity Queries over Semistructured data. In: WORKSHOP DE TESES E DISSERTAÇÕES EM BANCO DE DADOS, WTDBD, 1.; SBBD, 2002, Gramado, Rio Grande do Sul, Brasil. **Anais...** [S.l.: s.n.], 2002.

DORNELES, C. F.; HEUSER, C. A. **Applying visual information retrieval techniques to querying semistructured databases.** Submetido ao 9th String Processing and Information Retrieval, SPIRE, Lisboa, Portugal, 2002.

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

DORNELES, C. F.; HEUSER, C. A.; SILVA, A. S. da; MOURA, E. S. de. A Generic Strategy for Combining Similarity Metrics in Approximate Matching. **Information Systems**, [S.l.]. Submetido em 20 de dezembro de 2005.

DORNELES, C. F.; LIMA, A. E. N.; HEUSER, C. A. **Experiments on similarity query over XML data sources.** [S.l.]: Universidade Federal do Rio Grande do Sul, 2003. (RP-337).

DORNELES, C. F.; LIMA, A. E. N.; HEUSER, C. A. **Consulta por similaridade a dados semiestruturados**. [S.l.]: Universidade Federal do Rio Grande do Sul, 2003. (RP-326).

DORNELES, C. F.; NADVORNY, C. F.; KROTH, E.; LIMA, A. E. N.; HEUSER, C. A. **Pesquisa por similaridade em dados XML.** [S.l.]: Universidade Federal do Rio Grande do Sul, 2003. (RP-327).

DORNELES, C. F.; PADILHA, A.; HEUSER, C. A. **Suporte a argumentos de consultas vagos através da XPath.** [S.l.]: Universidade Federal do Rio Grande do Sul, 2004. (RP-347).

DORNELES, C. F.; PADILHA, A.; HEUSER, C. A. **Accessing XML data by allowing vague query arguments.** [S.l.]: Universidade Federal do Rio Grande do Sul, 2004. (RP-342).

FAGIN, R.; KUMAR, R.; SIVAKUMAR, D. Comparing top k lists. In: SYMPOSIUM ON DISCRETE ALGORITHMS, SODA, 14., 2003, Philadelphia, PA, USA. **Proceedings...** [S.l.]:Society for Industrial and Applied Mathematics, 2003. p.28–36.

FELLEGI, I. P.; SUNTER, A. B. A theory for record linkage. **Journal of the American Statistical Society**, [S.l.], v.64, p.1183–1210, 1969.

FERGUSON, A.; BRIDGE, D. Generalised Prioritisation: a new way of combining similarity metrics. In: CONFERENCE ON ARTIFICIAL INTELLIGENCE & COGNITIVE SCIENCE, AICS, 10., 1999. **Proceedings...** [S.l.: s.n.], 1999. p.137–142.

FRENCH, J. C.; POWELL, A. L.; SCHULMAN, E. Using clustering strategies for creating authority files. **Journal of the American Society for Information Science, JASIS**, [S.l.], v.51, n.8, p.774–786, 2000.

GALHARDAS, H.; FLORESCU, D.; SHASHA, D.; SIMON, E. An Extensible Framework for Data Cleaning. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, ICDE, 16., 2000, San Diego, California, USA. **Proceedings...** [S.l.]:IEEE Computer Society, 2000. p.312.

GALHARDAS, H.; FLORESCU, D.; SHASHA, D.; SIMON, E.; SAITA, C.-A. Declarative Data Cleaning: language, model, and algorithms. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 27., 2001, Roma, Italy. **Proceedings...** San Francisco: CA:Morgan Kaufmann, 2001. p.371–380.

GAO, L.; WANG, M.; WANG, X. S.; PADMANABHAN, S. Expressing and Optimizing Similarity-Based Queries in SQL. In: INTERNATIONAL CONFERENCE ON CONCEPTUAL MODELING, ER, 23., 2004, Shanghai, China. **Proceedings...** Berlin:Springer-Verlang, 2004. p.464–478.

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

GRAVANO, L.; IPEIROTIS, P. G.; JAGADISH, H. V.; KOUDAS, N.; MUTHUKRISHNAN, S.; PIETARINEN, L.; SRIVASTAVA, D. Using q-grams in a DBMS for Approximate String Processing. **IEEE Data Eng. Bull.**, [S.l.], v.24, n.4, p.28–34, 2001.

GRAVANO, L.; IPEIROTIS, P. G.; JAGADISH, H. V.; KOUDAS, N.; MUTHUKRISHNAN, S.; SRIVASTAVA, D. Approximate String Joins in a Database (Almost) for Free. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 27., 2001(b), San Francisco, CA, USA. **Proceedings...** San Francisco: CA:Morgan Kaufmann, 2001(b). p.491–500.

GRAVANO, L.; IPEIROTIS, P. G.; KOUDAS, N.; SRIVASTAVA, D. Text joins in an RDBMS for web data integration. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE, WWW, 2003. **Proceedings...** New York:ACM Press, 2003.

GROSSI, M. A. **Avaliando a performance das funções de similaridade através da revocação e precisão**. 2005. 60 p. Trabalho de Conclusão (Ciência da Computação), Instituto de Informática, UFRGS, Porto Alegre.

GUHA, S.; JAGADISH, H. V.; KOUDAS, N.; SRIVASTAVA, D.; YU, T. Approximate XML joins. In: INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, ACM SIGMOD, 21., 2002, New York, NY, USA. **Proceedings...** New York:ACM Press, 2002. p.287–298.

GUHA, S.; KOUDAS, N.; MARATHE, A.; SRIVASTAVA, D. Merging the Results of Approximate Match Operations. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 2004. **Proceedings...** [S.l.: s.n.], 2004. p.636–647.

GUHA, S.; KOUDAS, N.; SRIVASTAVA, D.; YU, T. Index-Based Approximate XML Joins. In: ICDE - INTERNATIONAL CONFERENCE ON DATA ENGINEERING, 2003. **Proceedings...** [S.l.: s.n.], 2003. p.708–710.

GUSFIELD, D.; IRVING, R. W. **The Stable Marriage Problem Structure and Algorithms**. [S.l.]: MIT Press, 1999.

HERNANDEZ, M. A.; STOLFO, S. J. The merge/purge problem for large databases. In: INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, ACM SIGMOD, 14., 1995, New York, NY, USA. **Proceedings...** New York:ACM Press, 1995. p.127–138.

HUFFMAN, S. B.; STEIER, D. Heuristic Joins to Integrate Structured Heterogeneous Data. In: SPRING SYMPOSIUM ON INFORMATION GATHERING, AAAI, 1995. **Proceedings...** [S.l.: s.n.], 1995.

HUFFMAN, S. B.; STEIER, D. A Navigation Assistant for Data Source Selection and Integration. In: FALL SYMPOSIUM SERIES ON AI APPLICATIONS IN KNOWLEDGE NAVIGATION AND RETRIEVAL, 1995, Cambridge, MA. **Proceedings...** [S.l.: s.n.], 1995. p.72–77.

ILYAS, I. F.; AREF, W. G. Rank-Aware Query Processing and Optimization. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, ICDE, 21., 2005, Tokyo, Japan. **Proceedings...** [S.l]:IEEE Computer Society, 2005. p.1144.

ILYAS, I. F.; AREF, W. G.; ELMAGARMID, A. K. Supporting Top-k Join Queries in Relational Databases. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 27., 2003, Berlin, Germany. **Proceedings...** Berlin:Morgan Kaufmann, 2003. p.754–765.

JIN, L.; LI, C.; MEHROTRA, S. Efficient similarity string joins in large data sets. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 28., 2002, Hong Kong, China. **Proceedings...** New York:ACM Press, 2002.

KIM, W.; CHOI, I.; GALA, S. K.; SCHEEVEL, M. On Resolving Schematic Heterogeneity in Multidatabase Systems. In: KIM, W. (Ed.). **Modern Database Systems**: the object model, interoperability, and beyond. [S.l.]: ACM Press:Addison-Wesley, 1995. p.521–550.

KNOBLOCK, C. A.; AMBITE, J. L.; THAKKAR, S. A View Integration Approach to Dynamic Composition of Web Services. In: ICAPS WORKSHOP ON PLANNING FOR WEB SERVICES, 2003, Trento, Italy. **Proceedings...** [S.l.: s.n.], 2003.

LAENDER, A. H. F.; GONÇALVES, M. A.; ROBERTO, P. A. BDBComp: building a digital library for the brazilian computer science community. In: JOINT CONFERENCE ON DIGITAL LIBRARIES, JCDL, 2004, Tucson, Arizona, USA. **Proceedings...** New York: ACM Press, 2004.

LAWRENCE, S.; GILES, L.; BOLLACKER, K. Digital Libraries and Autonomous Citation Indexing. **IEEE Computer**, [S.l.], v.32, n.6, p.67–71, 1999.

LAWRENCE, S.; GILES, L.; BOLLACKER, K. **CiteSeer.IST Scientific Literatura Digital Library**. Available at: http://citeseer.ist.psu.edu/. Visited on 05/01/2006.

LEE, L. On the Effectiveness of the Skew Divergence of Statistical Language Analysis. **Artificial Intelligence and Statistical**, [S.l.], p.65–72, 2001.

LEY, M. **DB&LP Computer Science Bibliography**. Available at: http://dblp.uni-trier.de/. Visited on 10/12/2005.

LIM, E.-P.; SRIVASTAVA, J.; PRABHAKAR, S.; RICHARDSON, J. Entity Identification in Database Integration. **Information Science**, [S.l.], v.89, n.1, p.1–38, 1996.

LIM, E.-P.; SRIVASTAVA, J.; SHEKHAR, S. Resolving Attribute Incompatibility in Database Integration: an evidential reasoning approach. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, 10., 1994, Houston, TX. **Proceedings...** [S.l.]:IEEE Computer Society Press, 1994. p.154–165.

LIM, E.-P.; SRIVASTAVA, J.; SHEKHAR, S. An Evidential Reasoning Approach to Attribute Value Conflict Resolution in Database Integration. **IEEE Transactions on Knowledge and Data Engineering**, [S.l.], v.8, n.5, p.707–723, Oct. 1996.

LIMA, A. E. N. de. **Pesquisa de Similaridade em XML.** 2002. 81 p. Projeto de Diplomação (Bacharelado em Ciência da Computação), Instituto de Informática, UFRGS, Porto Alegre.

MAIRSON, H. The Stable Marriage Problem. **The Brandeis Review**, [S.l.], v.12, n.1, 1992.

MCCALLUM, A.; NIGAM, K.; UNGAR, L. H. Efficient clustering of high-dimensional data sets with application to reference matching. In: KNOWLEDGE DISCOVERY AND DATA MINING, ACM SIGKDD, 6., 2000, Boston, MA, USA. **Proceedings...** New York:ACM Press, 2000. p.169–178.

MELNIK, S.; GARCIA-MOLINA, H.; RAHM, E. Similarity Flooding: a versatile graph matching algorithm and its application to schema matching. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, ICDE, 18., 2002, Washington, DC, USA. **Proceedings...** [S.l.]:IEEE Computer Society, 2002. p.117.

MICHALOWSKI, M.; AMBITE, J. L.; KNOBLOCK, C. A.; MINTON, S.; THAKKAR, S.; TUCHINDA, R. Retrieving and Semantically Integrating Heterogeneous Data from the Web. **IEEE Intelligent Systems**, [S.l.], v.19, n.3, 2004.

MONGE, A. E.; ELKAN, C. The Field Matching Problem: algorithms and applications. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, ACM SIGKDD, 2., 1996, Portland, Oregon, USA. **Proceedings...** New York:ACM Press, 1996. p.267–270.

MONGE, A. E.; ELKAN, C. An Efficient Domain-Independent Algorithm for Detecting Approximately Duplicate Database Records. In: WORKSHOP ON RESEARCH ISSUES ON DATA MINING AND KNOWLEDGE DISCOVERY, DMKD, 1997. **Proceedings...** New York: ACM Press, 1997.

MOTRO, A. VAGUE: a user interface to relational databases that permits vague queries. **ACM Transactions on Office Information Systems**, [S.l.], v.6, n.3, p.187–214, July 1988.

NAVARRO, G. A Guided Tour of Approximate String Matching. **ACM Computing Surveys**, [S.l.], n.1, p.31–88, March 2001.

NEEDLEMAN, S.; WUNSCH, C. A general method applicable to the search for similarities in the amino acid sequence of two proteins. **J Mol Biol**, [S.l.], v.48, n.3, p.443–53, 1970.

OLIVEIRA, J. W. A.; LAENDER, A. H. F.; GONÇALVES, M. A. Remoção de Ambiguidades na Identificação de Autoria de Objetos Bibliográficos. In: SIMPÓSIO BRASILEIRO DE BANCOS DE DADOS, SBBD, 20., 2005, Uberlândia, MG, Brazil. **Anais...** [S.l.: s.n.], 2005. p.205–219.

PADILHA, A. **Suporte a argumentos de consulta vagos através da XPath**. 2005. 61 p. Dissertação (Mestrado em Ciência da Computação) — Instituto de Informática, UFRGS, Porto Alegre, RS.

PADILHA, A.; DORNELES, C. F.; HEUSER, C. A. Suporte a argumentos de consulta vagos através da XPath. In: ESCOLA REGIONAL DE BANCO DE DADOS (ERBD), 1., 2005. **Anais...** [S.l.: s.n.], 2005. p.5–11.

PAPADOPOULOS, A. N.; MANOLOPOULOS, Y. Similarity query processing using disk arrays. In: INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, ACM SIGMOD, 1998, New York, NY, USA. **Proceedings...** New York:ACM Press, 1998. p.225–236.

PEARL, J. **Probabilisic reasoning in intelligent systems.** San Mateo, CA.: Morgan Kaufmann, 1988.

PETRAKIS, E. G. M.; FALOUTSOS, C. Similarity Searching in Medical Image Databases. **IEEE Trans. Knowl. Data Eng.**, [S.l.], v.9, n.3, p.435–447, 1997.

QUINLAN, J. Improved use of continuous attributes in C4.5. **Journal of Artificial Intelligence Research**, [S.l.], v.4, p.77–90, 1996.

RAGNEMALM, I. **The Euclidean Distance Transform**. 1993. Thesis, Linköping University, Dept. of Electrical Engineering, Linköping, Sweden.

RAHM, E.; DO, H.-H. Data Cleaning: problems and current approaches. **IEEE Bulletin of the Technical Committee on Data Engineering**, [S.l.], v.23, n.4, December 2000.

RISTAD, E. S.; YIANILOS, P. N. Learning String Edit Distance. **IEEE Transactions on Pattern Recognition and Machine Intelligence**, [S.l.], v.20, n.5, p.522–532, May 1998.

RODRIGUES, J. A. **Métodos Numéricos**: introdução, aplicação e programação. [S.l.]: Ed. Sílabo, 2003.

SALTON, G.; MCGILL, M. **Introduction to Modern Information Retrieval.** [S.l.]: McGraw-Hill Book Company, 1984.

SARAWAGI, S.; BHAMIDIPATY, A. Interactive deduplication using active learning. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, ACM SIGKDD, 8., 2002, New York, NY, USA. **Proceedings...** New York:ACM Press, 2002. p.269–278.

SCHALLEHN, E.; SATTLER, K.-U.; SAAKE, G. Advanced grouping and aggregation for data integration. In: INTERNATIONAL CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT, CIKM, 10., 2001, New York, NY, USA. **Proceedings...** New York:ACM Press, 2001. p.547–549.

SCHALLEHN, E.; SATTLER, K.-U.; SAAKE, G. Efficient similarity-based operations for data integration. **Data Knowl. Eng.**, Amsterdam, v.48, n.3, p.361–387, 2004.

SHATKAY, H.; ZDONIK, S. B. Approximate Queries and Representations for Large Data Sequences. In: ICDE - INTERNATIONAL CONFERENCE ON DATA ENGINEERING, 1996. **Proceedings...** [S.l.: s.n.], 1996. p.536–545.

SPEARMAN, C. Footrule' for measuring correlations. **British Journal of Psychology**, [S.l.], n.2, p.89–108, 1906.

STASIU, R. K. **Suporte a processamento de consultas vagas em banco de dados**. 2005. 57 p. Proposta de Tese (Doutorado em Ciência da Computação), Instituto de Informática, UFRGS, Porto Alegre.

STASIU, R. K.; HEUSER, C. A.; SILVA, R. Estimating recall and precision for vague queries in databases. In: CONFERENCE ON ADVANCED INFORMATION SYSTEMS ENGINEERING, CAISE, 17., 2005, Porto, Portugal. **Proceedings...** Berlin:Springer-Verlag, 2005. (Lecture Notes in Computer Science).

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

TEJADA, S.; KNOBLOCK, C. A.; MINTON, S. Learning domain-independent string transformation weights for high accuracy object identification. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, ACM SIGKDD, 8., 2002, New York, NY, USA. **Proceedings...** New York:ACM Press, 2002. p.350–359.

VITTORI, C. M.; DORNELES, C. F.; HEUSER, C. A. Creating XML Documents from Relational Data Sources. In: INTERNATIONAL CONFERENCE ON ELECTRONIC COMMERCE AND WEB TECHNOLOGIES, EC-WEB, 2., 2001, Munich, Germany. **Proceedings...** Berlin:Spring-Verlang, 2001. p.60–70.

WANG, Y. R.; MADNICK, S. E. The Inter-Database Instance Identification Problem in Integrating Autonomous Systems. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, 5., 1989. **Proceedings...** [S.l.: s.n.], 1989.

WINKLER, W. E. **The state of record linkage and current research problems.** [S.l.]: US Bureau of the Census, 1999. (R99/04).