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

FELIPE HOPPE LEVIN

# Desambiguação de Autores em Bibliotecas Digitais utilizando Redes Sociais e Programação Genética

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

Prof. Dr. Carlos Alberto Heuser
Orientador

Porto Alegre, setembro de 2010.

**CIP – CATALOGAÇÃO NA PUBLICAÇÃO**

Levin, Felipe Hoppe

Desambiguação de Autores em Bibliotecas Digitais utilizando Redes Sociais e Programação Genética / Felipe Hoppe Levin – Porto Alegre: Programa de Pós-Graduação em Computação, 2010.

62 f.:il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação. Porto Alegre, BR – RS, 2010. Orientador: Carlos Alberto Heuser.

1. desambiguação de autores 2. redes sociais 3. casamento de registros 4. bibliotecas digitais. I. Heuser, Carlos A. II. Desambiguação de Autores em Bibliotecas Digitais utilizando Redes Sociais e Programação Genética.

# SUMÁRIO

# LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| BDBComp | Biblioteca Digital Brasileira de Computação |
| CAP | *Contextual Atraction Principle* |
| DBLP | *Digital Bibliography and Library Project* |
| FC | Função de Casamento |
| HHC | *Hiererchical Heuristic-Clustering* |
| PG | Programação Genética |
| PMA | Pureza Média por Autor |
| PMG | Pureza Média por Grupo |
| RD | *Relationship Distance* |
| RE | *Relationship Existence* |
| RQ | *Relationship Quantity* |
| RS | *Relationship Strength* |
| SGB | *Semantic Graph Blocking* |

# LISTA DE FIGURAS

# LISTA DE TABELAS

# RESUMO

Bibliotecas digitais tornaram-se uma importante fonte de informação para comunidades científicas. Entretanto, por coletar dados de diferentes fontes, surge o problema de informações ambíguas ou duplicadas de nomes de autores. Métodos tradicionais de desambiguação de nomes utilizam informação sintática de atributos. Todavia, recentemente o uso de redes de relacionamentos, que traz informação semântica, tem sido estudado em desambiguação de dados. Em desambiguação de nomes de autores, relações de co-autoria podem ser usadas para criar uma rede social, que pode ser utilizada para melhorar métodos de desambiguação de nomes de autores.

Esta dissertação apresenta um estudo do impacto de adicionar análise de redes sociais a métodos de desambiguação de nomes de autores baseados em informação sintática de atributos. Nós apresentamos uma abordagem de aprendizagem de máquina baseada em Programação Genética e a utilizamos para avaliar o impacto de adicionar análise de redes sociais a desambiguação de nomes de autores. Através de experimentos usando subconjuntos de bibliotecas digitais reais, nós demonstramos que o uso de análise de redes sociais melhora de forma significativa a qualidade dos resultados. Adicionalmente, nós demonstramos que as funções de casamento criadas por nossa abordagem baseada em Programação Genética são capazes de competir com métodos do estado da arte.

**Palavras-Chave:** desambiguação de nomes, análise de relacionamentos, redes sociais, programação genética, funções de casamento, bibliotecas digitais.

**Author name disambiguation in digital libraries using social networks and genetic programming**

# ABSTRACT

Digital libraries have become an important source of information for scientific communities. However, by gathering data from different sources, the problem of duplicate and ambiguous information about author names arises. Traditional methods of name disambiguation use syntactic attribute information. However, recently the use of relationship networks, which provides semantic information, has been studied in data disambiguation. In author name disambiguation, the co-authorship relations can be used to create a social network, which can be used to improve author name disambiguation methods.

This dissertation presents a study of the impact of adding social network analysis to author name disambiguation methods based on syntactic attribute information. We present a machine learning approach based on Genetic Programming and use it to evaluate the impact of social network analysis in author name disambiguation. Through experiments using subsets of real digital libraries, we show that the use of social network analysis significantly improves the quality of results. Also, we demonstrate that match functions created by our Genetic Programming approach are able to compete with state-of-the-art methods.

**Keywords:** name disambiguation, relationship analysis, social networks, genetic programming, match functions, digital libraries.

# 1 INTRODUÇÃO

Bibliotecas digitais são complexos sistemas de informação dedicados ao armazenamento e à apresentação de coleções *online* de informações. Elas proporcionam serviços para busca e visualização dessas informações e armazenam metadados que descrevem seu conteúdo (e.g., autor, editora) e os relacionamentos entre os dados. São construídas, coletadas e organizadas com o objetivo de atender as necessidades de informação de uma comunidade específica (BORGMAN, 1999).

Na comunidade científica, bibliotecas digitais vêm se tornando uma fonte de informação cada vez mais importante, apresentando uma interface centralizada para o acesso a publicações científicas. Ao agrupar publicações através de metadados como autor, assunto e local de publicação, usuários podem utilizar o conteúdo destas bibliotecas para análises diversas. Uma instituição pode, por exemplo, utilizar a informação contida em uma biblioteca digital para validar a produção de um pesquisador e realizar uma contratação.

Entretanto, ao avaliar autores usando bibliotecas digitais, os usuários geralmente assumem que o conteúdo é livre de erros e ambigüidades, o que raramente é o caso. Uma das características de bibliotecas digitais é que elas obtêm seus dados de diferentes fontes. Estas fontes utilizam, muitas vezes, diferentes padrões e abreviaturas, o que acaba por gerar ambigüidades quando estes dados são unificados. Uma das mais comuns é a ambigüidade de nomes, pois existe um relacionamento do tipo muitos-para-muitos entre pessoas e suas denominações. Uma pessoa pode ter diferentes denominações: o nome de solteiro, o nome de casado, o nome completo, o nome abreviado. Com isso, ao fazer uma busca pelo trabalho de um determinado autor em uma biblioteca digital, precisamos saber todas as denominações que representam este autor, do contrário não encontraremos todas as suas publicações. Além disso, pessoas diferentes podem utilizar uma mesma denominação: pessoas com o mesmo nome, pessoas cujo nome abreviado é idêntico. Assim, uma busca por determinado nome pode trazer publicações de autores diferentes que possuem o mesmo nome e temos dificuldade em saber quais publicações pertencem ao autor que buscamos.

O problema de autores ambíguos em bibliotecas digitais pode ser dividido em dois subproblemas: citações separadas (*split citations*) e citações misturadas (*mixed citations*) (LEE; ON; KANG, 2005). O problema de citações separadas ocorre quando um autor possui diferentes denominações em uma biblioteca digital. Desta forma, suas publicações estão divididas entre nomes distintos. Por exemplo, na biblioteca digital DBLP, o autor Abel Guilhermino Silva-Filho possui diferentes denominações. Na Figura 2.1, temos uma tela da DBLP que mostra algumas das publicações deste autor onde ele aparece sob o nome de Abel Guilhermino S. Filho. Já na Figura 2.2 temos outra tela da DBLP que mostra outras publicações deste mesmo autor, porém agora sob o nome Abel G. Silva-Filho. Assim, uma busca por um destes nomes nunca traz a obra

completa do autor, o que dificulta buscas e análises do seu trabalho e do trabalho de outros autores que possuem diferentes denominações.

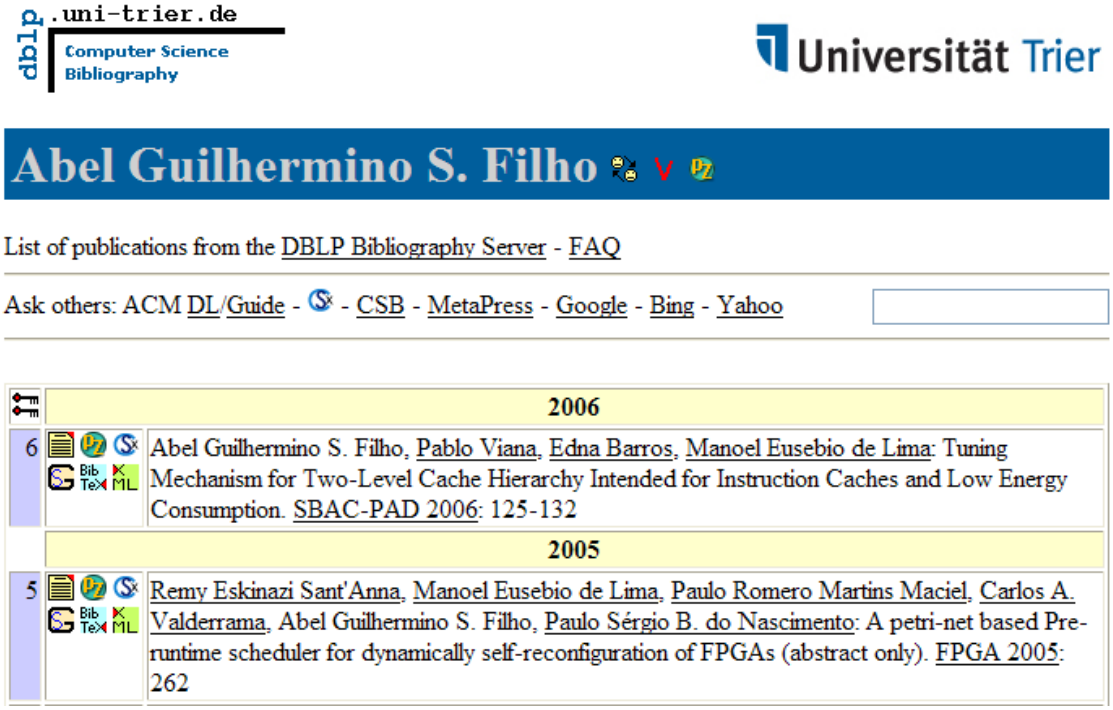


Figura 2.1: Tela da DBLP com publicações do autor Abel Guilhermino S. Filho

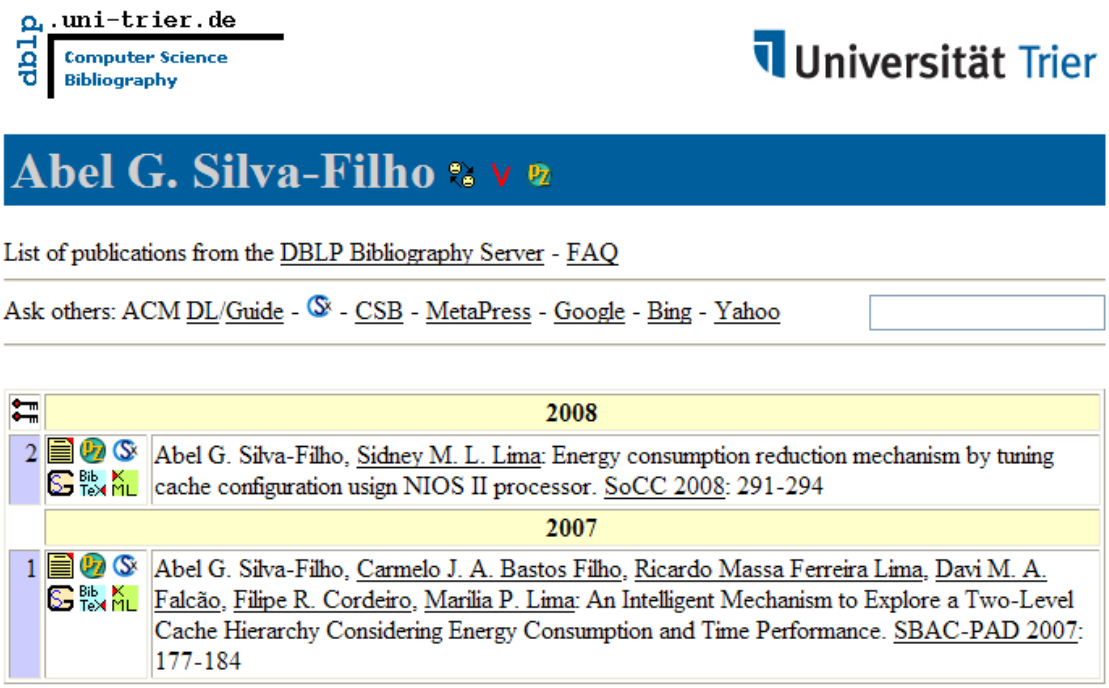


Figura 2.2: Tela da DBLP com publicações do autor Abel G. Silva-Filho

Já o problema de citações misturadas ocorre quando diferentes autores possuem o mesmo nome ou utilizam uma mesma denominação na biblioteca digital. Assim, suas

publicações são apresentadas juntas e um usuário tem dificuldade em dizer quem é o real autor de uma publicação. Na Figura 2.3 temos uma tela da DBLP com publicações do autor "A. Gupta". Porém, na realidade estas publicações referem-se a diferentes autores. A publicação número 12, de 2004, pertence ao autor Apoorv Gupta, enquanto a publicação número 10, de 1999, pertence ao autor Apurba Gupta. Isso dificulta a busca pelo trabalho de um autor, pois não se consegue saber com certeza quais publicações pertencem ao autor procurado.

A tarefa de resolver o problema de ambigüidade de nomes é conhecida na literatura como desambiguação de nomes (*name disambiguation).* Métodos tradicionais de desambiguação comparam a informação sintática dos atributos dos objetos ambíguos e, utilizando funções de similaridade e outras heurísticas, determinam se estes objetos representam a mesma entidade real ou não. Em desambiguação de nomes de autores, a sintaxe dos seus nomes é comparada. Em alguns métodos, como (COTA; GONÇALVES; LAENDER, 2007), outros atributos, como local de publicação e título da publicação, são comparados e utilizados como evidências de que dois autores são (ou não são) o mesmo autor real. Este trabalho argumenta que informações semânticas, especificamente os relacionamentos entre autores, pode ser utilizado em conjunto com métodos baseados em sintaxe para aumentar a qualidade dos resultados da deduplicação.

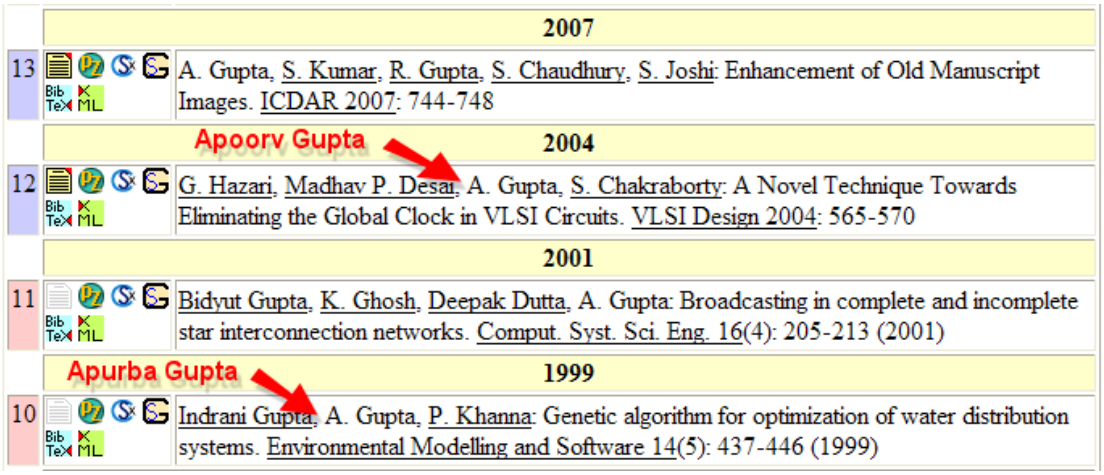


Figura 2.3: Tela da DBLP com publicações de A. Gupta

Neste trabalho nós utilizamos a análise de redes sociais como evidência de que dois autores em duas citações diferentes em um conjunto de dados têm mais chance de ser o mesmo autor real. Uma rede social é uma coleção de pessoas – ou atores – onde cada ator é ligado a um subconjunto dos demais (NEWMAN, 2001). Em redes de colaboração de produção científica, atores são autores que são ligados quando eles trabalharam em conjunto em alguma publicação científica. A colaboração entre dois autores implica uma afinidade entre eles: eles podem atuar em uma mesma área de interesse ou serem afiliados a uma mesma organização (MENEZES et al., 2009). Se a distância entre estes dois autores na rede é pequena, eles têm uma chance maior de possuírem os mesmos interesses e de serem afiliados a uma mesma organização. A distância é um indicador importante devido a um fenômeno identificado pela primeira vez em (MILGRAM, 1967) e conhecido como o efeito do "mundo pequeno" (*"small*

*world" effect*), que diz que quaisquer dois indivíduos estão separados por um caminho de até sete indivíduos. Como nossos experimentos demonstram, caminhos a partir de três indivíduos começam a perder importância no problema de desambiguação de nomes. Adicionalmente, se existe mais de um caminho conectando os autores, isso significa que eles são mais fortemente conectados. Assim, se dois autores em um conjunto de citações são fortemente relacionados e possuem algum grau de similaridade sintática, podemos assumir, com alta confiança, que eles são duplicatas do mesmo autor real, como demonstraremos em nossos experimentos.

Neste trabalho nós avaliamos o impacto de adicionar análise de redes sociais a métodos de desambiguação de autores baseados na similaridade sintática dos atributos das citações, demonstrando que há um aumento significativo na qualidade dos resultados. No primeiro conjunto de experimentos, é analisado o impacto da adição de análise de redes sociais a funções de similaridade de nomes. No segundo conjunto de experimentos, demonstramos que o uso de redes sociais melhora os resultados de métodos baseados não só em similaridade de nomes, mas em outras evidências como título e local da publicação. Para combinar estas evidências e as medidas de redes sociais, nós utilizamos uma abordagem de aprendizagem de máquina para criar funções de casamento para o processo de desambiguação. Esta abordagem é baseada em Programação Genética (PG), que já foi utilizada com sucesso em desambiguação de nomes de autores em (CARVALHO et al., 2008). O uso de PG para a geração de funções de casamento se faz necessário pelo fato de que, conforme aumentamos o número de evidências utilizadas, torna-se difícil definir pesos e limiares para cada uma delas. Além de demonstrar que o uso de redes sociais aumenta a qualidade dos resultados, nós comparamos os resultados obtidos pelas funções geradas por nossa abordagem de PG com resultados obtidos por outro método (COTA; GONÇALVES; LAENDER, 2007), mostrando que as funções criadas por nosso algoritmo são capazes de competir com o estado da arte.

## 1.1 Contribuições

A principal contribuição desse trabalho é demonstrar que o uso de redes sociais em desambiguação de nomes de autores aumenta a qualidade dos resultados obtidos, mostrando como as redes sociais podem ser utilizadas para este fim.

Como contribuições secundárias, podemos citar:

- Apresentação de um conjunto de Medidas de Relacionamento para avaliar a importância de conexões entre autores em uma Rede Social de Autores;
- Apresentação um conjunto de funções de casamento de autores, utilizando evidências tradicionais como nome, título e local de publicação, em conjunto com as Medidas de Relacionamento;
- Avaliação do impacto de adicionar análise de redes sociais a métodos tradicionais baseados em atributos utilizando conjuntos de dados reais;
- Demonstração experimental de que somente conexões entre autores com distância dois ou menor na rede social devem ser consideradas para melhorar o processo de desambiguação;
- Apresentação de um método automático de geração de funções de casamento de autores baseado em Programação Genética; e

- Demonstração da eficácia do mesmo, ao comparar os resultados obtidos pelas funções de casamento geradas com os resultados obtidos pelo estado da arte.

## 1.2 Organização do Texto

Esta dissertação está estruturada da seguinte forma:

- Capítulo 2, Desambiguação de Nomes de Autores: apresenta os principais conceitos relativos à desambiguação de autores, bem como métodos tradicionais de desambiguação, baseados em sintaxe de atributos, e métodos mais recentes, baseados em semântica;
- Capítulo 3, Utilização de Redes Sociais na Desambiguação de Nomes de Autores: apresenta o método de utilização de redes sociais em desambiguação de nomes de autores;
- Capítulo 4, Geração de Funções de Casamento com Programação Genética: apresenta o algoritmo de aprendizagem de máquina baseado em Programação Genética para a geração de funções de casamento;
- Capítulo 5, Análise Experimental: descreve os procedimentos experimentais realizados e os resultados obtidos;
- Capítulo 6, Conclusão: apresenta as conclusões deste trabalho e possíveis trabalhos futuros.

# 2 DESAMBIGUAÇÃO DE NOMES DE AUTORES

O problema de identificar representações ambíguas ou duplicadas é bastante antigo, tendo sido definido pela primeira vez em 1959 por Newcombe em (NEWCOMBE, et al., 1959) e formalizado dez anos mais tarde por Fellegi e Sunter em (FELLEGI; SUNTER, 1969). Desde então, o problema vem sendo objeto de pesquisa de um grande volume de trabalhos e publicações. Entretanto, o próprio problema de desambiguação vem recebendo denominações diferentes ao longo dos anos, conforme observado por (WEIS; NAUMANN, 2006-a). *Record linkage* (CHAUDHURI, et al., 2003), *merge/purge* (HERNANDEZ; STOLFO, 1995), *duplicate detection* (WEIS; NAUMANN, 2006-a) e *reference desambiguation* (KALASHNIOV; MEHROTRA, 2006) são algumas das denominações que vêm sendo utilizadas para referenciar este mesmo problema.

A pesquisa em desambiguação de informações pode ser dividida em duas categorias gerais: técnicas focadas na qualidade dos resultados e técnicas focadas na rapidez dos resultados (WEIS; NAUMANN, 2006-a). Trabalhos centrados no primeiro problema propõem métodos que buscam encontrar o maior número possível de duplicatas com o menor índice de erro possível. Para isso, são elaboradas sofisticadas medidas de similaridade de objetos como em (WEIS; NAUMANN, 2004). Essas medidas são utilizadas para definir se dois objetos encontram-se em duplicidade na base de dados: se dois objetos são suficientemente similares, são considerados duplicados. Medidas muito rígidas possuem uma margem de erro pequena, porém deixam de detectar um número elevado de duplicatas (falsos negativos). Já medidas menos rígidas costumam detectar a maioria das duplicatas, porém consideram como duplicatas objetos que, na realidade, são diferentes (falsos positivos).

A segunda categoria de trabalhos preocupa-se com o custo do processo de deduplicação, ou seja, o tempo necessário para a obtenção dos resultados. Neste caso, assume-se certo método de detecção de duplicatas e desenvolve-se um algoritmo que evita realizar todas as comparações possíveis entre pares de objetos, trocando qualidade por eficiência, como em (CHAUDHURI, et al., 2003) e (HERNANDEZ; STOLFO, 1995). Um dos processos mais utilizados para esse fim é a blocagem, ou seja, os dados são divididos em blocos (um dado pode pertencer a diferentes blocos, dependendo da técnica) e cada objeto é comparado somente com objetos que se encontram no mesmo bloco que o seu. Aqui, enfrenta-se um problema semelhante ao das medidas de similaridade: blocos muito pequenos necessitam de menos comparações, mas produzem um número elevado de falsos negativos. Já blocos muito grandes necessitam de um maior número de comparações, reduzindo o ganho de eficiência resultante da blocagem.

Este trabalho encaixa-se na primeira categoria, ou seja, preocupa-se com a qualidade dos resultados do processo de desambiguação. Além disso, foca um domínio específico, que é o domínio das bibliotecas digitais. Entretanto, o problema de entidades ambíguas aparece em diversos domínios. Em redes de franquias, por exemplo, os cadastros de clientes são descentralizados. Para realizar uma análise de clientes, a franqueadora deve realizar algum processo de deduplicação ao unificar os cadastros de clientes das diversas franqueadas, pois um cliente pode estar cadastrado em diferentes lojas. Outro exemplo é de um *website* de compras que vende produtos de uma rede de fornecedores. Cada fornecedor possui um catálogo de produtos e diferentes fornecedores podem fornecer o mesmo produto. Para que o *website* possa identificar estes produtos iguais e, por exemplo, escolher o fornecedor com o menor preço, é necessário algum processo de identificação de duplicatas.

A seguir, apresentaremos diversos métodos existentes que procuram resolver o problema de desambiguação. Na Seção 2.1 trataremos de métodos baseados em sintaxe de atributos e na Seção 2.2 abordaremos métodos que utilizam informações semânticas.

## 2.1 Métodos Baseados em Sintaxe de Atributos

Tradicionalmente, os métodos de desambiguação de entidades baseiam-se na similaridade sintática dos seus atributos para encontrar entidades duplicadas. A Tabela 2.1 mostra um conjunto de dados hipotético a ser desambiguado. Cada registro corresponde a um cliente, com seu nome, sobrenome e ano de nascimento. Os registros 1, 2 são duplicatas do cliente Antônio Schnider enquanto os registros 3, 4 e 5 são duplicatas da cliente Giovana Torres.

Tabela 2.1: Tabela de Clientes

| Id | Nome | Sobrenome | Ano de Nasc. |
|---|---|---|---|
| 1 | Antônio | Schnider | 1970 |
| 2 | Antônio | Schider | 70 |
| 3 | Giovana | Torres | 1980 |
| 4 | Geovana | Tores | 1980 |
| 5 | Giovana | Torres | 80 |

Pode parecer simples à primeira vista encontrar os dados correspondentes no exemplo acima. Porém, construir um algoritmo que encontre duplicatas de forma automática e com precisão em grandes volumes de dados não é uma tarefa trivial. Normalmente, algoritmos de detecção de dados em duplicidade utilizam uma função de similaridade que compara objetos dois a dois, produzindo um valor entre zero e um. Este valor indica o grau de semelhança entre esses objetos. Quanto mais próximo de zero, maior a diferença entre os objetos, ao passo que quanto mais perto de um, mais semelhantes eles são. Quando este valor encontra-se entre um limiar k e um, o algoritmo assume que os dois objetos são duplicatas.

Assim, o que vai determinar a qualidade dos resultados produzidos pelo algoritmo é a função de similaridade. Alguns trabalhos utilizam funções específicas a determinados domínios, como detecção de similaridade entre endereços e nomes. Outros, como

(COHEN; RICHMAN, 2002) e (SARAWAGI; BHAMIDIPATY, 2002), são baseados no aprendizado de funções de similaridade a partir de dados de treinamento, utilizando algoritmos de classificação para encontrar os dados correspondentes. Já trabalhos como (HERNANDEZ; STOLFO, 1995) e (NAVARRO, et al., 2001) baseiam-se na distância de edição entre *strings* (*string edit distance*), que é uma forma genérica para calcular a similaridade entre dados textuais. No caso específico de desambiguação de autores em bibliotecas digitais, utiliza-se a similaridade de atributos como os nomes dos autores e os títulos de suas publicações para encontrar autores em duplicidade (COTA; GONÇALVES; LAENDER, 2007).

A seguir descreveremos alguns métodos de deduplicação baseados em sintaxe de atributos e como eles podem ser utilizados em desambiguação de nomes de autores.

### 2.1.1 Similaridade de Levenshtein

O cálculo de similaridade de *strings* utilizando a distância de edição baseia-se no número mínimo de transformações (inserção, exclusão e substituição) necessário para transformar uma string *a* em uma string *b*. Existem diversas variações de distância de edição na literatura, sendo que uma das mais utilizadas e conhecidas é a distância de Levenshtein (LEVENSHTEIN, 1966).

A Figura 2.4 mostra um exemplo onde é calculado o número de transformações entre *John Smith* em *Jon Schmidt.* São necessárias quatro transformações, logo a distância de edição é quatro. Quanto menor a distância de edição, mais similares são os *strings.* Como tipicamente a similaridade entre dois objetos é calculada como um número entre zero e um, sendo que um indica que eles são idênticos, a distância de edição é normalizada pela maior *string* e o resultado é subtraído de um. Em nosso exemplo, o tamanho máximo das *strings* é onze e a distância de edição normalizada é 4/11 = 0,36. Assim, a similaridade entre *John Smith* e *Jon Schmidt* é de 1 – 0,36 = 0,64. Existem diversas variações deste método: alguns atribuem pesos diferentes para cada operação e outros utilizam uma herística diferente da distância de Levenshtein, como a distância de Hamming, que considera somente substituições. A maioria dessas funções de similaridade oferece melhores resultados quando utilizadas sobre *strings* de tamanho pequeno, como nomes de pessoas.

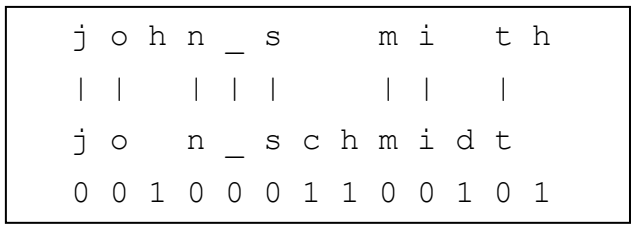


Figura 2.4: Transformação de strings

No problema de desambiguação de autores em bibliotecas digitais, a similaridade de Levenshtein pode ser utilizada para calcular a similaridade dos nomes dos autores, definindo como duplicatas aqueles autores com alto grau de similaridade.

### 2.1.2 Similaridade de *N-Grams*

Outro método para calcular a similaridade de *strings* é o cálculo de similaridade de *n-grams* (ELGARAMID; IPEIROTIS; VERYKIOS, 2007). Um *n-gram* é um conjunto

de *n* caracteres adjacentes. Dessa forma, um *bigram* é um conjunto de dois caracteres adjacentes e um *trigram* é um conjunto de três caracteres adjacentes.

Para calcular a similaridade de *n-grams* entre dois *strings*, é necessário contar o número de *n-grams* que aparecem em ambos os *strings*, dividindo este total pelo número total de *n-grams* distintos.

A Figura 2.5 mostra o cálculo de similaridade de *trigrams* entre os nomes *John Smith* e *Jon Schmidt*. Cada nome é dividido em seus respectivos *trigrams*, sendo que o caractere # é utilizado para designar o inicio ou o fim da palavra e o caractere _ é utilizado para designar o espaço em branco entre as palavras. A similaridade é dada pelo número de *trigrams* que aparecem em ambos os nomes (a intersecção dos dois conjuntos de *trigrams*) dividido pelo número de *trigrams* de ambas as palavras, sem contar repetições (a união dos dois conjuntos). O valor de similaridade de *trigrams* entre esses dois nomes é de 0,10.

Este método também pode ser utilizado para o cálculo da similaridade de nomes de autores, assim como a similaridade de Levenshtein. Porém, os resultados obtidos pelos dois métodos são muito diferentes, como podemos ver em nosso exemplo, ao comparar *John Smith* com *Jon Schmidt*. A escolha do melhor método vai depender do domínio em que o cálculo de similaridade será aplicado.

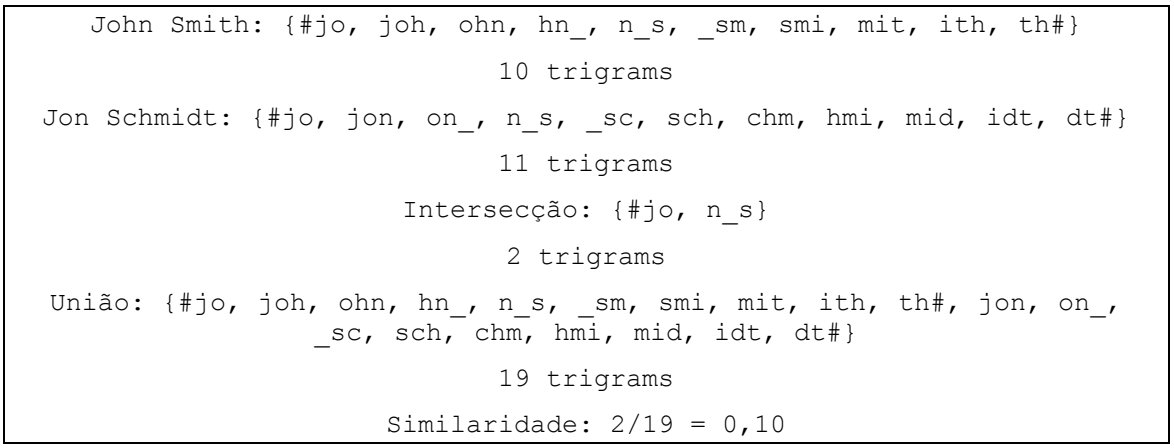


Figura 2.5: Similaridade de Trigrams

### 2.1.3 Similaridade de Registros

Embora o cálculo de similaridade de *strings* seja importante, no processo de integração de dados é mais comum que o objetivo da deduplicação de dados seja o casamento de registros inteiros. Esses registros costumam possuir campos com tipos de dados distintos (*strings*, números, datas) e conteúdos diferentes (nomes, descrições, textos longos). Assim, pode ser necessário utilizar funções de similaridade diferentes para comparar cada campo. Além disso, é necessário combinar os resultados das diferentes funções para indicar se dois registros casam ou não.

Em muitos casos, o usuário irá definir manualmente a forma de combinar os resultados, dependendo do domínio, utilizando funções de similaridade conhecidas. Por exemplo, na Tabela 2.1, um usuário poderia definir que dois clientes são duplicatas se o nome e o sobrenome possuem uma similaridade de Levenshtein superior a 0,75 e se os

dois últimos dígitos do ano de nascimento são iguais, caso contrário, os registros são diferentes. Usando esta fórmula, todas as duplicatas da Tabela 2.1 seriam identificadas com sucesso.

Um cálculo de similaridade definido dessa forma costuma ser mais eficaz do que cálculos de similaridade genéricos, porém é necessário um conhecimento profundo dos dados, o que nem sempre é possível. Em (DORNELES, et al., 2007) é proposto um método de casamento de registros mais genérico, onde os escores gerados pelas funções de similaridade utilizadas em cada campo são combinados em um escore único. Nesta proposta, cada função de similaridade é utilizada em um conjunto de dados reduzido em uma fase de treinamento. Os escores de similaridade retornados pela função são mapeados para valores de precisão correspondentes, criando um *escore ajustado* em termos de precisão. Uma vez tendo os escores das diferentes funções definidos em termos de precisão, fica mais fácil e intuitivo combiná-los e definir limiares de corte.

### 2.1.4 Agrupamento Hierárquico Heurístico

O Agrupamento Hierárquico Heurístico (*Heuristical Hierarchical Clustering* – HHC), apresentado em (COTA; GONÇALVES; LAENDER, 2007), é um método específico para desambiguação de nomes de autores, baseado nos atributos de citações presentes em bibliotecas digitais (e.g., nomes dos co-autores, título, local de publicação). Além do nome do autor, utilizam-se evidências como o título da publicação, o local de publicação e a lista de co-autores.

O objetivo de HHC é agrupar citações em grupos que correspondem ao mesmo autor. Para isso, ele utiliza diversas heurísticas e funções de similaridade sobre os atributos das citações. O agrupamento é dividido em fases, e em cada fase uma heurística diferente é utilizada, fundindo os grupos existentes e provendo mais informações para as fases subseqüentes. É exatamente este modelo hierárquico de agrupamento, dividido em fases, que faz com que este método tenha ótimos resultados, superando diversos outros métodos, conforme demonstrado em (COTA; GONÇALVES; LAENDER, 2007).

## 2.2 Métodos que Utilizam Informações Semânticas

Um ponto em comum entre a maioria dos métodos de deduplicação de dados é que a similaridade é baseada em informações intrínsecas dos objetos. É avaliada a proximidade sintática de *strings*, a proximidade numérica de valores, mas são avaliados somente os atributos de cada objeto. Entretanto, a informação extrínseca, ou seja, o relacionamento entre os objetos, também pode trazer um dado valioso a ser utilizado no processo de deduplicação.

A Tabela 2.2 traz um exemplo semelhante ao do capítulo anterior. Aqui, é mostrada uma lista de filmes, cada qual com título, diretor e ano de lançamento. Embora um analista humano seja capaz de concluir que os três primeiros itens da lista se referem ao mesmo filme *O Iluminado*, algoritmos baseados em sintaxe têm mais chance de concluir que o registro de número 3 tem uma semelhança maior com o registro 4 do que com os registros 1 e 2.

Uma solução para o problema seria, em um primeiro passo, deduplicar os atributos individualmente. Uma função de similaridade poderia concluir que “Stanley Kubric” é o mesmo que “S. Kubric” e que “1980” é o mesmo que “80”. Em um segundo passo, os registros seriam deduplicados e os relacionamentos entre os registros seriam levados em

conta. Ao comparar os registros 2 e 3, a função de comparação identificaria que esses registros, embora possuam um título com baixa similaridade sintática, são fortemente relacionados, possuem o mesmo diretor e o mesmo ano, e, portanto, tem grandes chances de serem duplicatas. Neste exemplo, a informação semântica, representada pelos relacionamentos entre as entidades, fez uma diferença substancial.

Tabela 2.2: Conjunto de Filmes Duplicados

| Id | Título | Diretor | Ano |
|---|---|---|---|
| 1 | O Iluminado | Stanley Kubric | 1980 |
| 2 | Iluminado, O | S. Kubrick | 80 |
| 3 | Shining, The | S. Kubrick | 1980 |
| 4 | Shine | S. Hicks | 1996 |
| 5 | Shine - Brilhante | Scott Hicks | 96 |

Recentemente, a deduplicação de informações vem sendo estudada em conjuntos de dados mais complexos como estruturas XML ou conjuntos de tabelas relacionadas entre si. A Figura 2.6 mostra um conjunto de dados desse tipo, em forma de grafo.

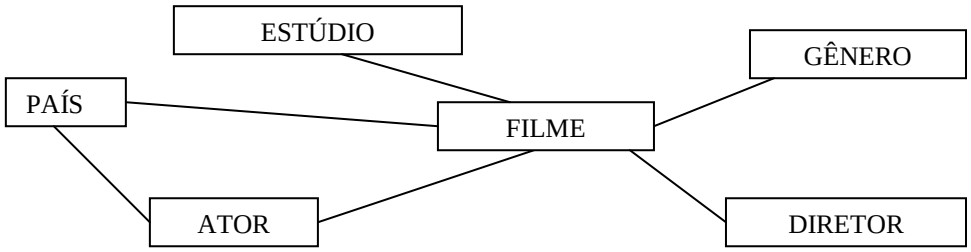


Figura 2.6: Esquema de dados sobre filmes

No conjunto de dados acima, além de utilizar a sintaxe dos atributos do filme na deduplicação, podem ser utilizados os relacionamentos entre os objetos. Por exemplo, se os filmes “O Iluminado” e “The Shining” se relacionam com os mesmos atores, gênero, país e estúdio, é possível identificar que eles representam o mesmo objeto no mundo real, mesmo que seus títulos sejam completamente diferentes sintaticamente. Com isso, a análise dos relacionamentos nos traz uma maior qualidade no resultado da desambiguação. Além disso, levando em conta os relacionamentos entre entidades pode-se limitar o número de comparações somente a entidades relacionadas entre si. Em nosso exemplo, poderíamos comparar somente filmes que possuem o mesmo estúdio ou que compartilhem no mínimo um ator, resultando em um ganho elevado no tempo de execução.

Desta forma, considerar o relacionamento semântico entre entidades abre novas possibilidades para técnicas de deduplicação de informações. A teoria que embasa o uso

de relacionamentos na deduplicação é explicada através do Princípio da Atração Contextual (*Contextual Atraction Principle* - CAP) (KALASHNIOV; MEHROTRA, 2006). Este princípio diz que duas ocorrências no banco de dados que podem ser duplicatas possuem mais chances de serem duplicatas se estiverem fortemente relacionadas. O modo de relacionar as entidades varia dependendo das fontes de dados (NIN, et al., 2007). Em um banco relacional, os relacionamentos podem ser obtidos a partir das chaves estrangeiras, enquanto em um documento XML eles são definidos pela hierarquia entre os elementos do mesmo. A seguir, serão apresentados diferentes métodos de deduplicação que utilizam relacionamentos entre entidades e redes de co-autores no processo de deduplicação.

### 2.2.1 *Semantic Graph Blocking*

*Semantic Graph Blocking* (SGB) é uma abordagem apresentada em (NIN, et al., 2007) que propõe uma nova família de algoritmos de blocagem que constroem blocos baseados no contexto. A finalidade da blocagem em deduplicação de informações, como explicado anteriormente, é, através de algum critério, agrupar os dados em conjuntos de dados semelhantes entre si, com o intuito de restringir o número de comparações entre objetos.

Dois dos métodos de blocagem mais utilizados, *Standard Blocking* (JARO, 1989) e *Sorted Neighborhood* (HERNANDEZ; STOLFO, 1995), são baseados em similaridade de atributos. A diferença entre SGB e os outros métodos de blocagem é que SGB não se baseia em atributos, mas nos relacionamentos entre os dados. Com base no Princípio da Atração Contextual, (KALASHNIOV; MEHROTRA, 2006), somente são comparados dados relacionados entre si.

Para encontrar as duplicatas de um determinado objeto, o algoritmo SGB constrói um grafo que representa os relacionamentos deste objeto com os demais. Primeiro é criado o nodo que representa o objeto em questão, depois, são criados os nodos de cada objeto relacionado a este. Para cada novo nodo no grafo, é realizado o mesmo processo, até que o grafo atinja uma profundidade p a partir do objeto inicial. A partir do bloco formado, são utilizadas funções de similaridade de atributos para encontrar as duplicatas. A Figura 2.7 mostra um bloco de profundidade dois. Na figura, representando um banco de dados de autores e artigos, o bloco foi montado a partir do autor *Lyons, Don*, construindo relacionamentos entre pares de autores que escreveram um mesmo artigo. Os nodos em cinza estão dentro do bloco, enquanto os nodos em branco estão fora.

No exemplo, os autores *Smith, John* e *Xmith, John*, de acordo com CAP, tem grande chance de ser o mesmo autor, pois ambos escreveram artigos com *Lyons, Don*, e realmente o são. Já *Smith, Jehn*, por estar a uma distância quatro dos *Johns*, tem mais chances de representar uma entidade diferente, como realmente ocorre, apesar de a distância de edição entre os três ser a mesma. Note que um método de blocagem baseado em sintaxe, como SB, que só compararia nomes que iniciam com as mesmas letras, colocaria *Xmith, John* em um bloco diferente, por começar com a letra X, enquanto *Smith, John* e *Smith, Jehn*, que possuem uma alta similaridade sintática, seriam considerados duplicatas, pois estariam no mesmo bloco. Somente um método que utiliza informação semântica traria um resultado correto neste exemplo.

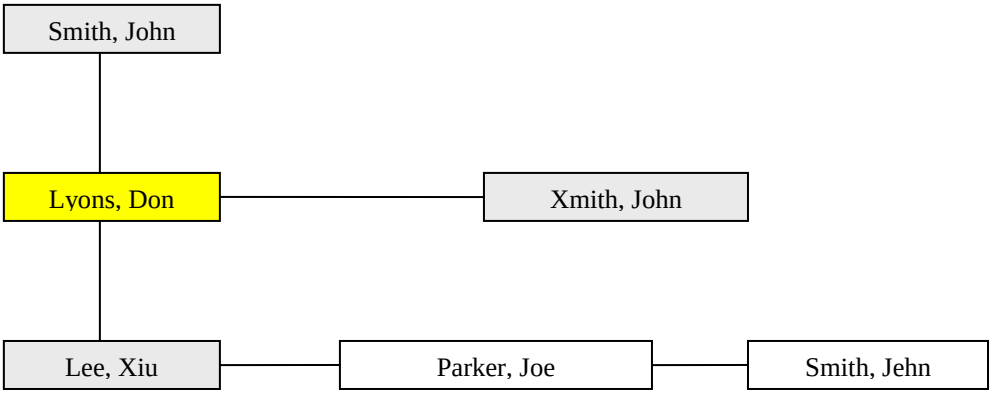


Figura 2.7: Blocagem em SGB

### 2.2.2 RelDC

RelDC é uma técnica proposta em (KALASHNIOV; MEHROTRA, 2006) como uma forma de aprimorar resultados produzidos por funções de similaridade tradicionais utilizando o relacionamento entre entidades. Ele trata de um problema análogo à desambiguação de informações que é denominado desambiguação de referências. Dado um conjunto de dados de entrada A e um conjunto de dados de referência B, o método busca encontrar quais os objetos de B são referidos na entrada A. Por exemplo, dada uma lista de autores como a referência A, como a lista da Figura 2.8, e uma lista de artigos com seus autores como a entrada B, como a lista da Figura 2.9, o algoritmo percorre os autores de B encontrado, para cada um, o autor apropriado na referência A. Ou seja, o método assume que os dados em B estão duplicados em A e busca casar esses dados. Nesta abordagem, é utilizado um método baseado em similaridade sintática de atributos para, para cada autor de A, encontrar o autor correspondente em B. Se a similaridade entre os autores ultrapassa certo limiar, eles são considerados o mesmo objeto. No entanto, é possível que mais de um autor em B ultrapasse esse limiar de similaridade com um autor de A, resultando em diversos candidatos. Como os autores da entrada A devem referenciar somente um autor da referência B, é necessário uma forma de desempate. E é esse o problema que RelDC se propõe a resolver, utilizando os relacionamentos entre entidades para encontrar, dentre estes candidatos, qual o mais adequado.

```
<A1, 'Dave White'>
<A2, 'Don White'>
<A3, 'John Black'>
<A4, 'Susan Grey'>
```

Figura 2.8: Registros de Autores

No exemplo ilustrado pelas Figuras 2.8 e 2.9, os autores *Susan Grey*, *John Black*, e *Don White* na lista de publicações seriam facilmente encontrados na base de dados de referência. Entretanto, o autor D. White poderia se referir tanto a Dave White quanto a Don White na referência de autores. É nesse momento que o RelDC entra, utilizando os relacionamentos entre entidades para encontrar a referência correta.

```
<P1, 'Título1', 'Susan Grey', 'D. White'>
<P2, 'Título2', 'John Black', 'Don White'>
<P3, 'Título3', 'Susan Grey', 'John Black'>
```

Figura 2.9: Registros de Publicações

Um grafo como o da Figura 2.10 é criado, relacionando autores e artigos. No caso do artigo P1, não sabemos se um de seus autores se refere a *Dave White* ou a *Don White*. Assim, é criado um *nodo de escolha* ligando P1 a esses dois autores.

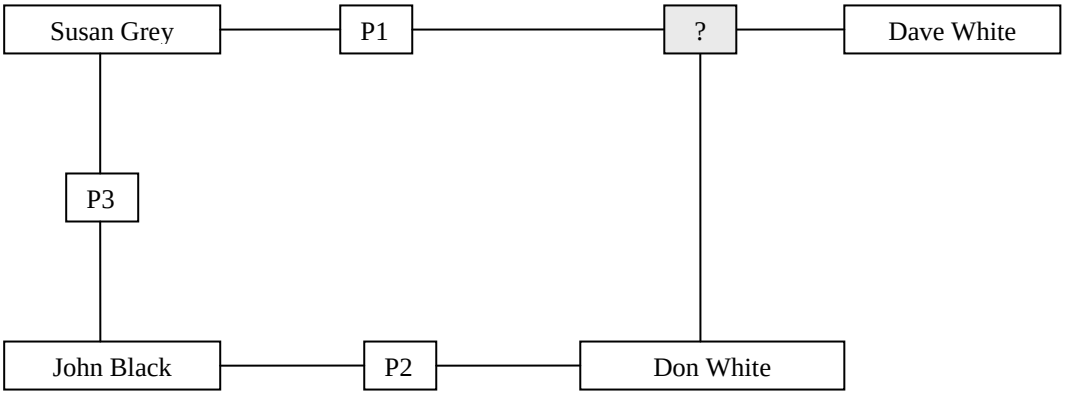


Figura 2.10: Relacionamentos entre Autores

O objetivo do algoritmo RelDC é remover os *nodos de escolha* do grafo e encontrar a referência correta para as entradas ambíguas. Baseado em CAP, a referência escolhida pelo algoritmo é aquela que está mais fortemente conectada à entrada ambígua. No exemplo da Figura 2.9, a referência escolhida para P1 seria *Don White*, por estar mais fortemente conectada a P1: enquanto *Dave White* não se liga a P1 por nenhum outro caminho no grafo, *Don White* também está conectado pelo caminho que passa por P2 e P3. A semântica passada pelo grafo é que há uma relação entre os autores *Susan Grey*, *John Black* e *Don White* que não passa pelo *nodo de escolha*, enquanto *Dave White* não está relacionado a estes autores por outros caminhos. Com isso, baseado em CAP, *D. White* tem mais chances de se referir a *Dave White* por estar mais fortemente relacionado a esta referência do que à referência *Don White*.

Em (KALASHNIOV; MEHROTRA, 2006) são apresentados experimentos que demonstram o ganho de precisão e revocação com o uso do RelDC em relação ao uso de funções de similaridade sintática para selecionar o objeto correto entre o conjunto de candidatos. Na realidade, RelDC já utiliza uma função de similaridade sintática para escolher o conjunto de candidatos, mais ou menos como um algoritmo de blocagem, embora aqui os blocos sejam mais numerosos e muitos deles possuem somente um

elemento. A inovação desse trabalho é utilizar os relacionamentos semânticos para aprimorar os resultados de funções que utilizam sintaxe.

### 2.2.3 Recona e Adama

Em (WEIS; NAUMANN, 2006-b), os autores apresentam os algoritmos Recona e Adama, e chamam a atenção para uma questão importante ao utilizar relacionamentos no processo de deduplicação: objetos duplicados influenciam na deduplicação de objetos dependentes. Por exemplo, a Figura 2.11 mostra um conjunto de dados formado por quatro artigos P1, P2, P3 e P4, cada um com dois ou três autores, sendo que P2 e P3 são duplicatas do mesmo artigo real. Um algoritmo como SGB utilizaria os relacionamentos através dos artigos para formar blocos de autores a serem deduplicados. Entretanto, se existem artigos duplicados, como é o caso de P2 e P3 na Figura 2.11, os autores desses artigos podem ficar em blocos diferentes e com isso, não seriam identificados como duplicatas. No exemplo, Maria Carneiro e M. Carneiro nunca seriam comparados, pois estariam em blocos diferentes. Para resolver o problema, os artigos teriam que ser deduplicados em um primeiro passo.

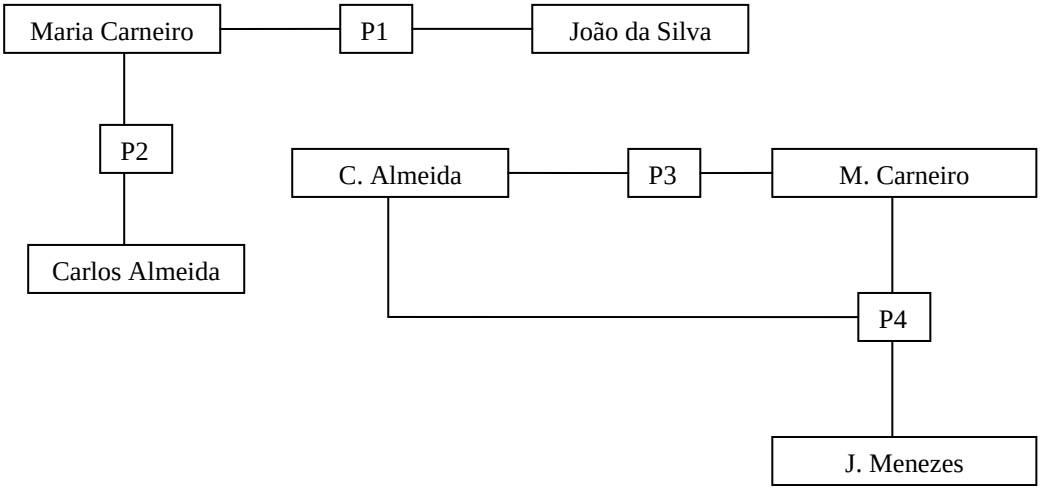


Figura 2.11: Dependência entre entidades

Com isso, a abordagem descrita em (WEIS; NAUMANN, 2006-b) sugere que todas as entidades envolvidas no processo sejam deduplicadas. Ao encontrar uma duplicata, as entidades relacionadas a ela devem ser comparadas novamente, de forma a garantir que problemas, como o da Figura 2.11, não ocorram. Além disso, (WEIS; NAUMANN, 2006-b) propõe que as entidades que possuem um maior número de dependentes sejam comparadas primeiro, reduzindo o número de recomparações.

Os algoritmos Recona e Adama, são semelhantes. Eles comparam objetos dois a dois, retirando os pares de uma lista e adicionando-os a uma lista de duplicatas caso sejam assim considerados. A diferença entre os dois é que Recona, ao encontrar um par duplicado, recompara os objetos que dependem desse par. Já Adama não realiza recomparações, mas realiza as comparações em uma ordem tal que objetos com maior número de dependências sejam comparados por último, sendo mais rápido que Recona,

mas resultando em uma qualidade menor dos resultados. O grande ganho desses algoritmos em relação a outros métodos é justamente a ordem das comparações. Recona consegue reduzir em 90% o número de recomparações devido a esta ordenação. Já o algoritmo Adama, que não realiza recomparações, tem a qualidade dos seus resultados aumentada pela ordenação.

### 2.2.4 Resolução Coletiva de Entidades

Em (BHATTACHARYA; GETOOR, 2007), é proposto um algoritmo de desambiguação de entidades que compara as entidades coletivamente. Ao invés de comparar sempre pares de entidades dois a dois, o algoritmo compara grupos de objetos duplicados (na fase inicial do algoritmo, estes grupos são unitários) e, à medida que os grupos casam, ou seja, os objetos de ambos os grupos representam a mesma entidade, os grupos são unidos.

Este algoritmo utiliza, além de atributos como nome, título, entre outros, os relacionamentos entre co-autores como evidência no processo de desambiguação. É gerado um valor de Similaridade de Vizinhança, onde autores, para serem considerados duplicatas, precisam ter um conjunto similar de co-autores.

Na proposta de (KANG et al., 2009) também é utilizada a similaridade de conjuntos de co-autores no processo de desambiguação e estes conjuntos de co-autores são obtidos através de extrações de dados da Web.

### 2.2.5 Desambiguação de Nomes via Similaridade de Grafos e Redes Sociais

Em (ON et al., 2006), é utilizada similaridade de grafos para a desambiguação de entidades. No caso de desambiguação de autores, para cada autor, é criado um grafo representando a sua rede de co-autores. Ao comparar dois autores, é calculada a similaridade de seus respectivos grafos e esta similaridade é utilizada para verificar se ambos os autores representam o mesmo autor real.

Na proposta de (MALIN, 2005), é criada uma rede social entre os nomes envolvidos no processo de deduplicação. Dois nomes se relacionam na rede social se eles aparecem em conjunto em uma mesma publicação, por exemplo. Para desambiguar dois nomes de autores, é calculada uma similaridade de redes sociais como a probabilidade de um autor chegar ao outro autor através de caminhos nesta rede.

Embora os trabalhos vistos nesta seção façam uso de informações extraídas dos relacionamentos dos autores, nenhum deles apresenta um estudo detalhado sobre como as características da rede social de autores podem afetar e contribuir com o processo de desambiguação. A seguir, veremos quais informações podem ser extraídas da rede de co-autoria para melhorar os resultados da desambiguação.

# 3 UTILIZAÇÃO DE REDES SOCIAIS NA DESAMBIGUAÇÃO DE NOMES DE AUTORES

Nesse capítulo, é proposto um método que utiliza informações contidas na rede social do autor para aprimorar o processo de desambiguação de nomes de autores. No capítulo anterior, mostramos diversos métodos de deduplicação que utilizam relacionamentos entre autores e até mesmo similaridade de redes sociais. Porém, nenhum desses métodos utiliza as características que podem ser extraídas da rede social de autores no processo de deduplicação e é exatamente esta a nossa proposta. Na Seção 3.1, é definida a Rede Social de Autores, apresentada pela primeira vez em (LEVIN; HEUSER, 2009), e, na Seção 3.2, apresentamos uma série de medidas que podem ser utilizadas sobre esta rede para avaliar a importância de um relacionamento entre dois autores. Já na Seção 3.3, mostramos como estas medidas podem ser utilizadas em funções de casamento de autores. Conforme é experimentalmente demonstrado no Capítulo 5, a utilização de tais medidas aumenta de forma significativa a qualidade dos resultados do processo de desambiguação.

## 3.1 Rede Social de Autores

Normalmente, a entrada para o processo de desambiguação de nomes de autores é uma lista de referências a publicações. Cada referência possui informações como o título da publicação, os seus autores, o periódico ou congresso onde foi publicado, o ano de publicação, etc. Na Tabela 3.1, temos uma lista fictícia de referências, onde cada referência possui um título e seus autores. A partir desta lista, podemos construir uma rede social dos autores que a compõe utilizando as informações contidas na própria lista.

Na Figura 3.1, é apresentada a Rede Social de Autores construída a partir das referências da Tabela 3.1. Esta rede, representada aqui como um grafo, é composta por dois tipos de vértices: autores (caixas retangulares) e publicações (caixas arredondadas). Existem também dois tipos de arestas, representadas como linhas contínuas ou pontilhadas. As linhas contínuas ligam uma publicação a seus autores, enquanto as linhas pontilhadas são utilizadas para criar caminhos na rede, ligando autores de diferentes publicações e estabelecendo relacionamentos entre autores. Em nossa heurística – outras poderiam ser utilizadas – são ligados autores com a mesma inicial e o mesmo sobrenome, por possuírem uma alta possibilidade de representarem o mesmo autor real. Esta heurística foi utilizada em nossos experimentos, porém, de modo a facilitar o entendimento, no exemplo deste capítulo foram ligados autores cujo nome é exatamente igual.

Ao comparar autores no processo de desambiguação, podemos utilizar informações presentes na Rede Social de Autores em conjunto com outras informações – como o

nome do autor – para avaliar se dois autores são a mesma pessoa real. Muitas vezes, somente o nome do autor não é suficiente para estabelecer que dois autores se refiram à mesma pessoa. Na Figura 3.1, os autores P1.1, *Robert Walker*, e P2.2, *Robert D. Walker*, possuem nomes com alto grau de similaridade, mas a Rede Social de Autores traz mais evidências de que estes autores podem ser a mesma pessoa: existe um caminho de comprimento dois entre estes dois autores passando por P1 e P2. O comprimento de um caminho é dado pelo número de ligações autor-publicação-autor no caminho em questão. Por exemplo, no caminho entre P1.1 e P2.2 que passa por P1 e P2, existem duas ligações autor-publicação-autor: Robert Walker-P1-Carl Parker e Carl T. Parker-P2-Robert D. Walker. Conforme será demonstrado no Capítulo 5, quando dois autores são ligados por um caminho de comprimento menor ou igual a dois, a probabilidade de que estes dois autores sejam o mesmo aumenta significativamente. Dado que P1.1 e P2.2 possuem nomes similares e um caminho de comprimento dois entre eles, a probabilidade de eles se referirem à mesma pessoa é muito grande.

Tabela 3.1: Lista de Referências a Publicações

| ***Cód.*** | ***Título*** | ***Autores*** |
|---|---|---|
| P1 | A Survey on Record Linkage | Robert Walker, Ben Goldman e Carl Parker |
| P2 | Record Linkage using Information from the Web | Carl T. Parker, Robert D. Walker e J. R. Anderson |
| P3 | Data Disambiguation in Digital Libraries | Ben Goldman e Ruth Adams |
| P4 | Web Crawling and Digital Libraries | Rob Walker, Ruth Adams e George Brown |
| P5 | People Search on the Web | John Anderson e George S. Brown |

O mesmo não é válido quando comparamos P1.1 e P4.1. Apesar de possuírem nomes similares – Robert Walker e Rob Walker – o caminho ligando estes dois autores possui comprimento três e, portanto, a ligação entre eles não é próxima o suficiente para considerarmos o relacionamento entre ambos como uma evidência de que ambos são duplicatas de um mesmo autor.

## 3.1 Medidas de Relacionamento

A seguir, apresentaremos um conjunto de medidas que extraem evidências da Rede Social do Autor para o processo de desambiguação. Estas medidas devem ser utilizadas em conjunto com outras evidências tipicamente usadas, como a similaridade dos nomes dos autores ou dos títulos das publicações, melhorando a qualidade dos resultados.

Em redes sociais, a distância entre dois atores é definida com o comprimento do menor caminho entre eles (NEWMAN, 2003). Como definimos na seção anterior, o comprimento de um caminho é dado pelo número de ligações autor-publicação-autor no

caminho. Assim, a Distância de Relacionamento (*Relationship Distance* - RD[1]) entre dois autores é definida como o valor de comprimento do menor caminho entre eles.

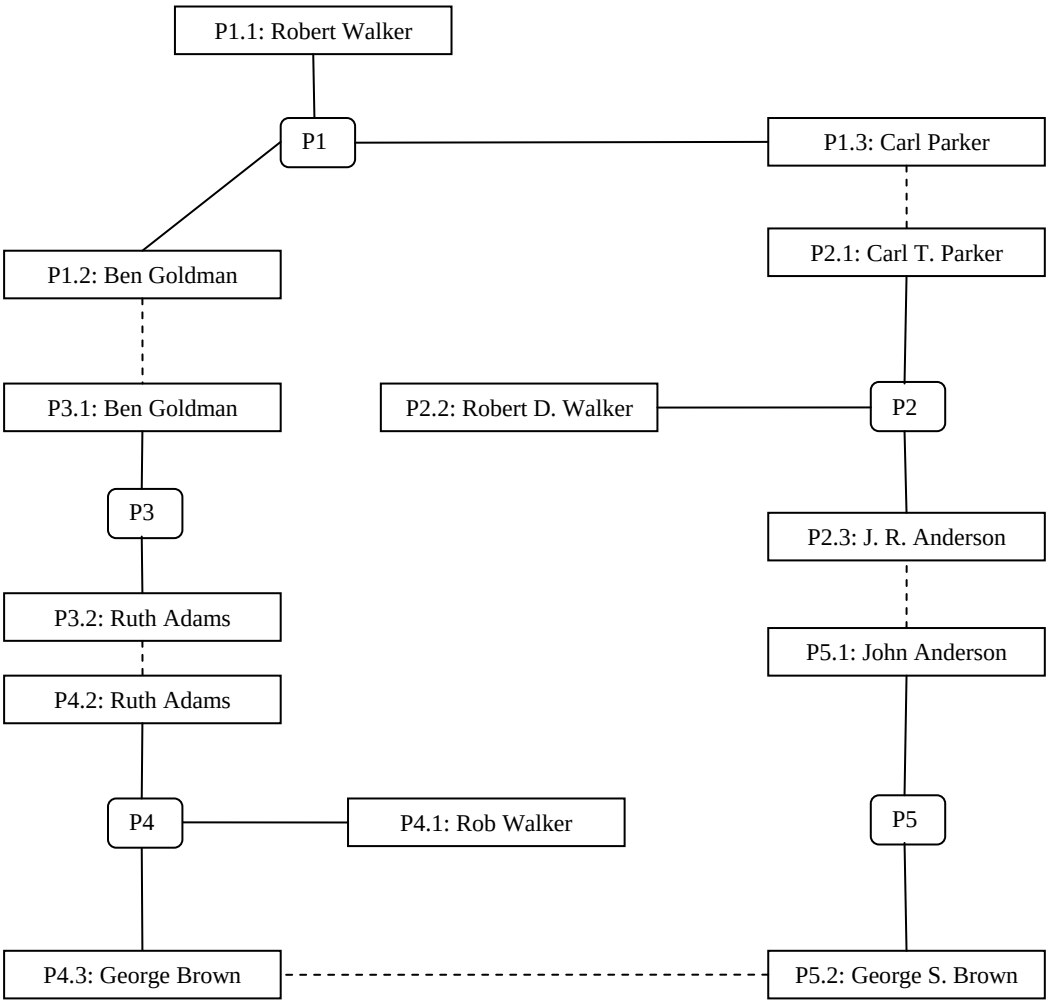


Figura 3.1: Rede Social de Autores

**Distância de Relacionamento (RD):** Sejam $a_1$ e $a_2$ dois autores. Então, RD($a_1$, $a_2$) é o comprimento de menor caminho entre eles, sendo infinito se nenhum caminho existir.

Em nosso exemplo, P1.1 e P2.2 estão ligados por dois caminhos. O primeiro passa por P1 e P2 e possui comprimento dois. Já o segundo passa por P1, P3, P4, P5 e P2, com comprimento cinco. Como a distância entre dois autores é o comprimento do menor caminho entre eles, RD(P1.1, P2.2) é dois. Esta medida é utilizada para representar a proximidade entre dois autores na rede social. Conforme será demonstrado no Capítulo 5, quanto mais próximos dois autores na Rede Social de Autores, maior a

[1] As abreviaturas das medidas de relacionamento foram mantidas de acordo com o nome da mesma em inglês, de forma a manter a mesma denominação apresentada em (LEVIN; HEUSER, 2009).

probabilidade de que ambos representem o mesmo autor e por isso a importância de se medir a proximidade dos autores, utilizando RD.

Assim como é importante medir a proximidade entre dois autores, também é importante saber se existe um relacionamento entre dois autores a uma dada distância *d*. Para isso, é utilizada a medida Existência de Relacionamento (*Relationship Existence* - RE), que retorna um valor booleano indicando se existe um caminho entre dois autores a uma distância máxima *d*.

**Existência de Relacionamento (RE):** Sejam $a_1$ e $a_2$ dois autores sendo comparados e *d* um número inteiro. Então, RE($a_1$, $a_2$, d) é verdadeiro se RD($a_1$, $a_2$) $\leq$ d e falso caso contrário.

Em nossa rede social de exemplo, a existência de relacionamento entre P1.1 e P2.2 a uma distância dois, denotada como RE(P1.1, P2.2, 2), é verdadeira, pois, como visto no exemplo anterior a distância entre esses dois autores é dois e, portanto, existe um caminho menor ou igual a dois ligando-os. Já RE(P1.1, P4.1, 2) é falso, pois os dois caminhos ligando estes autores possuem comprimento maior que dois e, por conseqüência, a distância entre eles é maior que dois.

Outro fator importante a ser considerado é a conectividade do autor, ou seja, a quantidade de relacionamentos com outros autores que ele possui. A importância deste fator é que relacionamentos entre autores com baixa conectividade são mais significativos do que relacionamentos entre autores de alta conectividade. Isso ocorre por que autores com baixa conectividade têm uma baixa probabilidade de se relacionar com outro autor. Por isso, quando há um relacionamento entre autores desse tipo é um indicativo mais forte de que são duplicatas do que quando há um relacionamento entre autores de alta conectividade. Para medir a conectividade de um autor, é utilizada a Quantidade de Relacionamentos (*Relationship Quantity* - RQ), que é o número de relacionamentos que um autor possui a uma distância máxima *d*.

**Quantidade de Relacionamentos (RQ):** Sejam *a* um autor, A o conjunto de todos os autores na rede social e *d* um número inteiro. Então, RQ(a, d) = |B|, onde para cada autor $b \in$ B, RD(a, b) $\leq$ d e B $\subset$ A.

Em nosso exemplo, existem caminhos ligando todos os autores (nem sempre isso ocorre), porém a distâncias diferentes. O autor P1.1, considerando distância máxima um, se relaciona com quatro outros autores. Logo, RQ(P1.1, 1) = 4. Considerando uma distância dois, o número de conexões sobe para nove, sendo RQ(P1.1, 2) = 9.

Outro ponto importante a ser considerado é o número de conexões entre dois autores, considerando uma distância *d*. Quanto mais conexões entre dois autores, mais forte o relacionamento entre eles e maior a probabilidade de eles serem duplicatas. Para medir a intensidade desse relacionamento, é utilizada a Força de Relacionamento (*Relationship Strength* - RS) que mede o número de relacionamentos entre dois autores a uma distância *d*.

**Força de Relacionamento (RS):** Sejam $a_1$ e $a_2$ dois autores sendo comparados e *d* um número inteiro. Então, RS($a_1$, $a_2$, d) é o número de conexões entre $a_1$ e $a_2$ com comprimento *d* ou menor.

Na Figura 2, a força de relacionamento entre P1.1 e P4.1 a uma distância dois é zero, pois não há nenhum caminho com comprimento dois ou menor entre ambos. Já aumentando a distância para três, temos RS(P1.1, P4.1) = 1, pois há um caminho de

comprimento três entre os autores, passando por P1, P3 e P4. Considerando uma distância quatro, temos RS(P1.1, P4.1) = 2, pois além do caminho de comprimento três, há também um caminho de comprimento quatro, passando por P1, P2, P5 e P4.

## 3.2 Determinando Duplicatas com Medidas de Relacionamento

De forma a tirar proveito das Medidas de Relacionamento, é necessário utilizá-las na função de casamento que será usada no processo de desambiguação de autores. Uma função de casamento recebe dois autores como entrada e a sua saída é um valor booleano indicando se eles são o mesmo autor real ou não.

A Figura 3.2 apresenta um algoritmo simplificado para desambiguação de autores em português estruturado. Ele recebe como entrada uma lista de referências bibliográficas e uma função de casamento e retorna uma lista de grupos de autores, onde o grupo da posição i da lista de grupos corresponde aos autores duplicados do autor da posição i da lista de autores.

No primeiro passo do algoritmo, a função *ExtraiAutores* gera uma lista de autores extraídos de cada uma das referências bibliográficas. A cada autor são associados os dados que serão utilizados no processo de desambiguação, como o nome do autor e o título da publicação em que ele aparece. A seguir, a função *InicializaGrupos* gera a lista inicial de grupos de autores com um grupo para cada autor. O algoritmo entra então em um laço que compara todos os autores dois a dois. A função *Casa* compara dois autores através da função de casamento *func* e retorna um valor booleano indicando se os autores casam através daquela função de casamento ou não. Se eles casam, o autor *j* é adicionado ao grupo do autor *i* e o grupo do autor *j* passa a ser o grupo do autor *i.* Ao final, é retornada a lista com os grupos de cada autor.

```
1. Desambiguacao (ref: lista de refer, func: funcCasam)
2.   i, j: inteiros
3.   aut: lista de autores
4.   grupos: lista de grupoDeAutores
5. Inicio
6.   aut := ExtraiAutores(ref)
7.   grupos := InicializaGrupos(aut)
8.   Para i de 1 até Tamanho(aut)
9.     Para j de i + 1 até Tamanho(aut)
10.       Se Casa(aut[i], aut[j], func) então
11.       Inicio
12.          AdicionaAutor(aut[j], grupos[i])
13.          grupos[j] := grupos[i]
14.       Fim
15.    Próximo j
16.  Próximo i
17.  Retorna grupos
18.Fim
```

Figura 3.2: Algoritmo Simplificado de Desambiguação de Autores

O que este algoritmo faz é agrupar autores em grupos de autores que são considerados o mesmo autor real, de acordo com uma função de avaliação. É importante notar, porém, que a função Casa não necessariamente retornar verdadeiro ao comparar todos os autores de um grupo entre si. Isso por que se um autor A casa com um autor B e um autor B casa com um autor C, os autores A, B e C serão colocados em um mesmo grupo, mesmo que A não case com C.

Como visto anteriormente, o algoritmo possui duas entradas: a lista de referências e a função de casamento. Como a lista de referências é o objeto da desambiguação, fica claro que a função de casamento é o ponto chave na qualidade do resultado do algoritmo. Abaixo, temos um exemplo de uma função de casamento simples, denominada *SimNome*, que utiliza somente o nome do autor como evidência.

**SimNome:** Sejam $a_1$ e $a_2$ dois autores sendo comparados e *k* um número real. Se $Sim(a_1, a_2) \geq k$ então $SimNome(a_1, a_2, k)$ = verdadeiro, senão $SimNome(a_1, a_2, k)$ = falso.

Ao comparar dois autores, esta função verifica se a similaridade entre seus nomes é superior a um limiar *k.* Em caso afirmativo eles casam, do contrário não casam. É uma função simples, que considera somente os nomes dos autores no processo de desambiguação. Entretanto, para melhorar os resultados produzidos pela função de avaliação, podemos utilizar as Medidas de Relacionamento apresentadas na seção anterior. Abaixo, a função *SimNomeRE* utiliza além do nome do autor a Existência de Relacionamento como evidência.

**SimNomeRE:** Sejam $a_1$ e $a_2$ dois autores sendo comparados, *k* um número real e *d* um inteiro. Se $Sim(a_1, a_2) \geq k$ e $RE(a_1, a_2, d)$ então $SimNomeRE(a_1, a_2, k, d)$ = verdadeiro, senão $SimNomeRE(a_1, a_2, k, d)$ = falso.

A função *SimNomeRE* restringe a função *SimNome*, exigindo que além da similaridade de nomes os autores sejam relacionados. Abaixo apresentamos uma alternativa do uso de RE:

**SimNomeOuRE:** Sejam $a_1$ e $a_2$ dois autores sendo comparados, *k, l* dois números reais e *d* um inteiro. Se $Sim(a_1, a_2) \geq k$ ou ($Sim(a_1, a_2) \geq l$ e $RE(a_1, a_2, d)$) então $SimNomeRE(a_1, a_2, k, l, d)$ = verdadeiro, senão $SimNomeRE(a_1, a_2, k, l, d)$ = falso.

Através da função *SimNomeOuRE* é possível utilizar, além do limiar *k* para a similaridade de nomes de autores, um segundo limiar *l*, mais baixo, que é utilizado quando existe um relacionamento entre os autores a uma distância *d.* Esta função captura a noção de que o fato de dois autores estarem relacionados faz com que a probabilidade de eles serem o mesmo autor seja maior e, com isso, a similaridade de seus nomes não precisa ser tão grande.

A seguir, apresentamos uma terceira função, que utiliza a Quantidade de Relacionamentos como evidência.

**SimNomeRERQ:** Sejam $a_1$ e $a_2$ dois autores sendo comparados, *k, l, q* três números reais e *d* um número inteiro. Se ($Sim(a_1, a_2) \geq k$ e ($RQ(a_1, d) < q$ ou $RQ(a_2, d) < q$)) ou ($Sim(a_1, a_2) > l$ e $RE(a_1, a_2, d)$ e $RQ(a_1, d) >= q$ e $RQ(a_2, d) >= q$) então $SimNomeRERQ(a_1, a_2, k, l, q, d)$ = verdadeiro, senão $SimNomeRERQ(a_1, a_2, k, l, q, d)$ = falso.

Nesta função, a Quantidade de Relacionamentos é utilizada para decidir se deve ser utilizado o limiar mais alto *k* ou o limiar mais baixo *l.* Se um dos autores possui baixa

conectividade, a função não exige que os autores possuam um relacionamento, mas exige uma similaridade de nomes maior. Porém se ambos os autores possuem alta conectividade, a existência de um relacionamento entre eles é exigida, porém a similaridade de nomes não precisa ser tão alta.

Por fim, apresentamos um último exemplo, que utiliza a Força de Relacionamento como evidência.

**SimNomeRERQRS:** Sejam $a_1$ e $a_2$ dois autores sendo comparados, *k, l, q, m* quatro números reais e *d* e *f* dois números inteiro. Se ($Sim(a_1, a_2) \geq k$ e ($RQ(a_1, d) < q$ ou $RQ(a_2, d) < q$)) ou ($Sim(a_1, a_2) > l$ e $RE(a_1, a_2, d)$ e $RQ(a_1, d) >= q$ e $RQ(a_2, d) >= q$) ou ($Sim(a_1, a_2) > m$ e $RS(a_1, a_2, d) \geq f$) então SimNomeRERQRS($a_1$, $a_2$, k, l, m, q, d, f) = verdadeiro, senão SimNomeRERQRS($a_1$, $a_2$, k, l, m, q, d, f)= falso.

A função *SimNomeRERQRS* é semelhante à função anterior, porém ela possui um limiar adicional *m* que é utilizado quando a Força de Relacionamento entre dois autores é maior que um valor *f.* A idéia desta função é que, se o relacionamento entre os autores é muito forte, a probabilidade de eles serem o mesmo é ainda maior do que se eles possuíssem apenas um relacionamento simples. Com isso, a similaridade de seus nomes pode ser ainda menor.

Conforme demonstraremos em nossos experimentos, as funções de casamento *SimNomeRE, SimNomeRERQ* e *SimNomeRERQRS* apresentam melhores resultados do que a função SimNome, que não utiliza Medidas de Relacionamento, comprovando a relevância do uso das mesmas.

# 4 GERAÇÃO DE FUNÇÕES DE CASAMENTO COM PROGRAMAÇÃO GENÉTICA

Na Seção 3.3, apresentamos algumas funções de casamento criadas manualmente utilizando as Medidas de Relacionamento apresentadas na Seção 3.2. Entretanto, o problema de criar funções de casamento manualmente é que nem sempre utilizamos as evidências da melhor forma possível. Além disso, com um número grande de evidências, fica difícil estabelecer limiares e pesos para cada uma delas. Por este motivo, apresentamos neste capítulo um método automático para geração de funções de casamento utilizando Programação Genética.

Na Seção 4.1, falaremos sobre a estrutura das funções de casamento geradas pelo método. Na Seção 4.2, abordaremos os operadores evolucionários utilizados no algoritmo de geração de funções e, na Seção 4.3, explicaremos o algoritmo em si.

## 4.1 Estrutura das Funções de Casamento

De acordo com (KOZA, 1992), existem três requisitos para a correta utilização da técnica de Programação Genética:

(1) O problema deve ser modelado como uma estrutura em árvore;

(2) A qualidade da árvore modelada deve poder ser avaliada de forma automática;

(3) Os operadores evolucionários aplicados sobre uma árvore válida devem resultar em uma árvore válida.

Para atender o primeiro requisito, as Funções de Casamento devem ser modeladas com uma estrutura em árvore. A Figura 4.1 mostra a estrutura utilizada pelo nosso método. No nível mais baixo da estrutura, estão os componentes das evidências, sendo que cada evidência é decomposta em tipo de evidência, operador e valor. O tipo de evidência corresponde a qual evidência está sendo utilizada (e.g., similaridade de nomes, existência de relacionamento, similaridade de títulos). O operador varia entre maior, maior e igual, igual, menor e igual ou menor, enquanto o valor corresponde a um número cuja gama de valores dependerá do tipo de evidência. Um exemplo de uma evidência seria SimNome $\geq$ 0,7, sendo SimNome o tipo de evidência, $\geq$ o operador e 0,7 o valor.

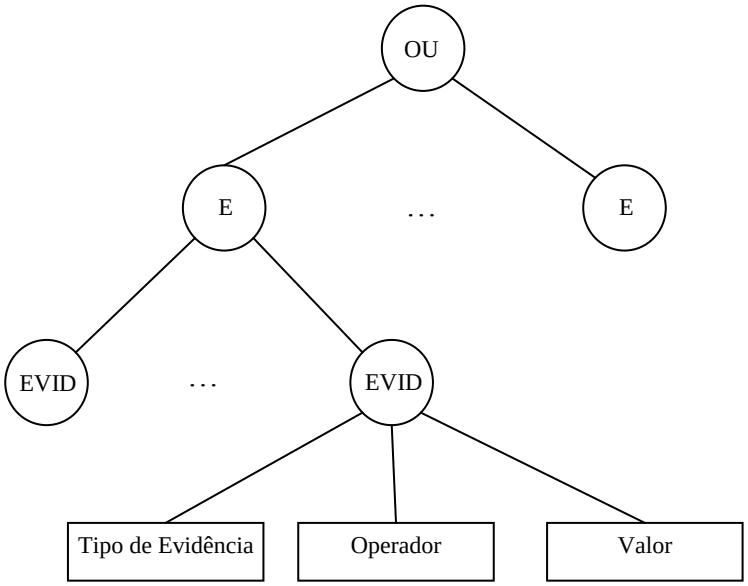


Figura 4.1: Estrutura de representação das Funções de Casamento

A composição de tipo de evidência, operador e valor formam uma evidência, representada pelo nodo *EVID* na Figura 4.1. Conforme a estrutura apresentada, as evidências são combinadas através do operador lógico *e*, representado pelo nodo *E* na Figura 4.1, formando um conjunto de evidências. Estes conjuntos de evidências podem ser por sua vez, combinados com outros conjuntos de evidências através do operador lógico *ou*, representado pelo nodo raiz *OU*.

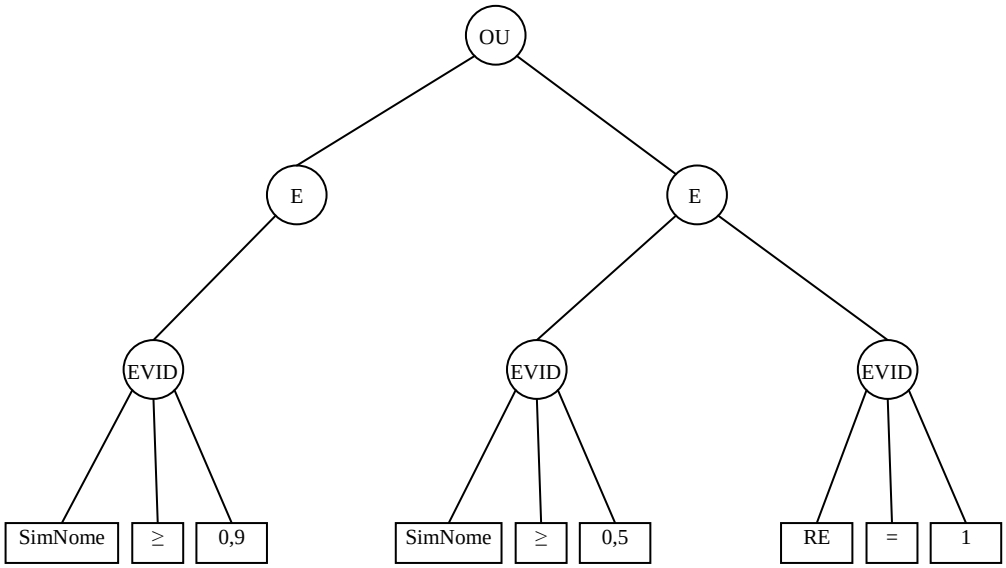


Figura 4.2: Exemplo de Função de Casamento

Para ilustrar o modo que uma função de casamento é representada, vamos considerar a seguinte função: (SimNome ≥ 0,9) OU (SimNome ≥ 0,5 E RE = 1). Esta função utiliza a similaridade de nomes (SimNome) e a Existência de Relacionamento (RE) como evidências. Autores sendo comparados através dessa função casarão se a similaridade de nomes for igual ou superior a 0,9 ou se existir um relacionamento entre eles e a similaridade de nomes for igual ou superior a 0,5. A Figura 4.2 mostra como essa função seria representada na estrutura utilizada por nosso método.

## 4.2 Operadores Evolucionários

Através da representação em árvore apresentada na seção anterior, é possível aplicar operadores evolucionários sobre as funções de casamento, alterando-as e permitindo que elas evoluam com o passar das gerações, em nosso algoritmo. Nosso método utiliza dois operadores evolucionários: *crossover* e mutação.

### 4.2.1 Crossover

O operador de *crossover* é utilizado para gerar novos indivíduos, isto é, novas funções de casamento para a próxima geração a partir de duas funções-pais. Nessa operação, são escolhidas aleatoriamente duas sub-árvores, uma de cada função-pai, e elas são trocadas, gerando dois novos indivíduos. A idéia é que se duas funções apresentam bons resultados, então algumas das sub-árvores que as compõem possuem algum mérito. Assim sendo, a recombinação aleatória pode gerar filhos melhores.

A Figura 4.3 mostra um exemplo de *crossover,* onde temos duas funções-pai que geram duas novas funções. Como a figura mostra, é selecionada uma sub-árvore de cada função e estas sub-árvores são trocadas, gerando as novas funções.

De acordo com os três requisitos para a correta utilização de PG, da Seção 4.1, o uso de operadores evolucionários deve resultar em árvores válidas. Assim sendo, se faz necessário impor uma limitação à operação de *crossover* no que diz respeito às sub-árvores escolhidas: sempre devem ser escolhidas sub-árvores do mesmo nível na estrutura da árvore. No exemplo da Figura 4.3, foram escolhidas duas sub-árvores que possuem um nodo representando o operador *e* como raiz e que, conseqüentemente, possuem o mesmo nível. A troca dessas duas sub-árvores resulta em duas novas árvores válidas, seguindo a mesma estrutura de árvore estabelecida na Seção 4.1. Se fossem, entretanto, escolhidas sub-árvores de níveis diferentes, teríamos como resultado árvores inválidas, pois elas não seguiriam a estrutura. Ao assegurar que sempre serão escolhidas sub-árvores do mesmo nível para o *crossover*, estamos garantindo que o terceiro requisito para a correta utilização de PG está sendo cumprido.

### 4.2.2 Mutação

O operador de mutação é utilizado para realizar alterações aleatórias em uma função de casamento, de forma a aumentar a diversidade da população. Sem a mutação, as gerações subseqüentes não passariam de recombinações das funções de casamento existentes, portanto a mutação é utilizada para incluir características novas na população.

Diferente do *crossover,* na mutação, existe somente uma função-pai. É escolhida aleatoriamente uma sub-árvore nesta função e ela é substituída por uma sub-árvore aleatória. Entretanto, para que a operação de mutação resulte em uma árvore válida, a nova sub-árvore gerada deve possuir a mesma hierarquia da sub-árvore substituída, mantendo a estrutura da Seção 4.1.

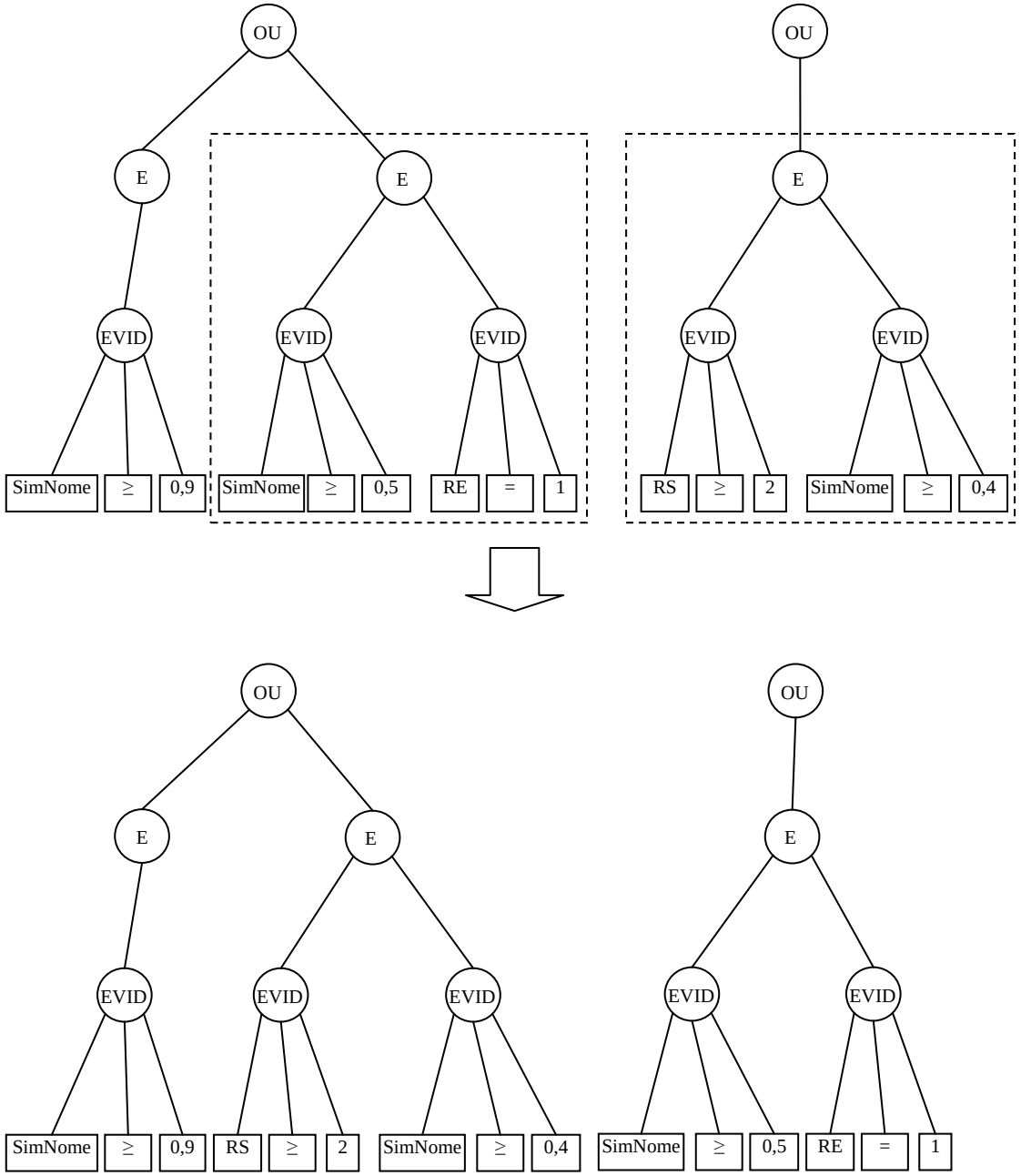


Figura 4.3: Exemplo de *Crossover*

Na Figura 4.4, temos dois exemplos de mutação. No primeiro exemplo, a sub-árvore escolhida é um nodo-folha de valor. Como a árvore resultante deve ser uma árvore válida, a sub-árvore aleatória gerada deve ser composta também por um nodo de valor. No exemplo, o valor 0,4 é substituído pelo valor 0,6.

No segundo exemplo da Figura 4.4, a sub-árvore escolhida pelo operador de mutação possui um nodo raiz do tipo *EVID*. Desta forma, a sub-árvore gerada deve também possuir o mesmo tipo de nodo raiz. Dessa forma, em nosso exemplo a evidência RE = 1 é substituída pela evidência RQ ≤ 2, mantendo a hierarquia da árvore e resultando em uma função de casamento válida.

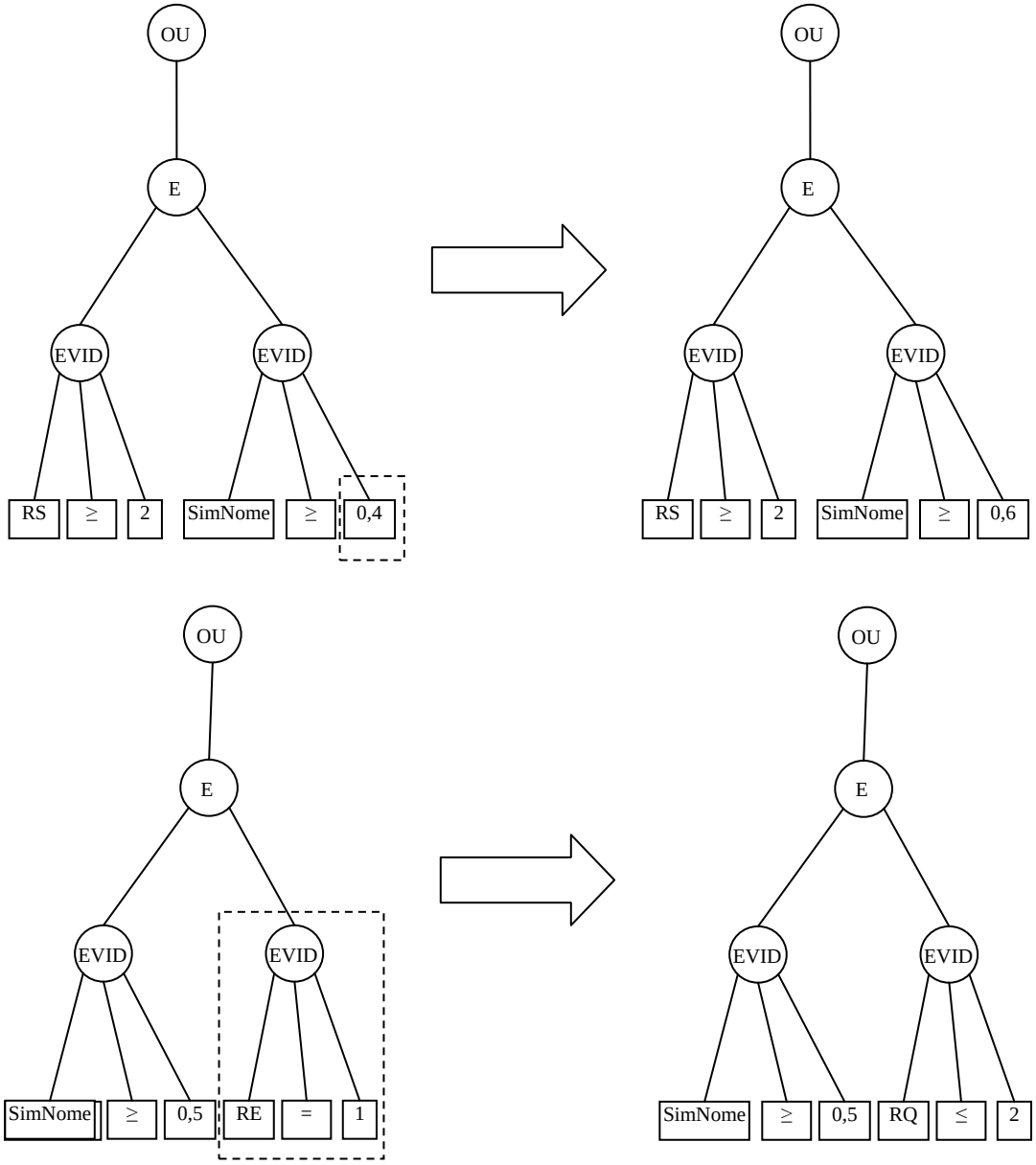


Figura 4.4: Exemplo de Mutação

## 4.3 Algoritmo para Geração de Funções de Casamento

A Figura 4.5 mostra o algoritmo para geração de Funções de Casamento. Ele recebe como parâmetros o tamanho da população inicial de funções e o número de gerações da execução. O primeiro passo do algoritmo, representado pela função *GeraPopulacaoInicial*, é gerar uma população inicial de funções. Cada função segue a estrutura de árvore apresentada na Seção 4.1 e é gerada de forma aleatória de acordo com os tipos de evidência disponíveis e os operadores e valores válidos para cada tipo. A Tabela 4.1 mostra os tipos de evidência utilizados em nossos experimentos, com os valores e operadores válidos para cada um. Estes tipos de evidência serão explicados com mais detalhe no Capítulo 5. A função *GeraPopulacaoInicial* (linha 6) estabelece de forma aleatória, para cada função gerada, o número de componentes *E* que cada função

terá, o número de evidências que cada componente *E* terá e, para cada componente, o tipo de evidência, o operador e o valor.

```
1. GeraFC (tamanhoPop: inteiro, numGer: inteiro)
2.   i, j: inteiros
3.   pop: lista de funcaoDeCasamento
4.   aval: lista de real
5. Inicio
6.   pop := GeraPopulacaoInicial(tamanhoPop)
7.   Para i de 1 até numGer
8.     Para j de 1 até Tamanho(pop)
9.       aval[j] := AvaliaFuncao(pop[j])
10.    Próximo j
11.    Selecao(pop, aval)
12.    Crossover(pop)
13.    Mutacao(pop)
14.  Próximo i
15.  Para j de 1 até Tamanho(pop)
16.    aval[j] := AvaliaFuncao(pop[j])
17.  Próximo j
18.  Retorna Ordena(pop, aval)
19.Fim
```

Figura 4.5: Algoritmo de Geração de Funções de Casamento

Tabela 4.1: Tipos de Evidência Utilizados

| ***Tipo de Evidência*** | ***Operadores*** | ***Valores*** |
|---|---|---|
| Similaridade de Nomes (SimNome) | ≥, ≤ | 0 a 1 |
| Similaridade de Títulos (SimTitulo) | ≥, ≤ | 0 a 1 |
| Similaridade de Local de Publicação (SimLocal) | ≥, ≤ | 0 a 1 |
| Casamento de Inicial e Sobrenome (IniSobrenome) | = | 0 ou 1 |
| Número de Palavras do Título que Casam (NumPalTitulo) | ≥, ≤ | 0 a 8 |
| Similaridade de Palavras do Título (SimPalTitulo) | ≥, ≤ | 0 a 1 |
| Primeiro Nome Abreviado (Abrev) | = | 0 ou 1 |
| Existência de Relacionamento (RE) | = | 0 ou 1 |
| Força de Relacionamento (RS) | ≥, ≤ | 0 a 10 |
| Quantidade de Relacionamentos Mínima (MinRQ) | ≥, ≤ | 0 a 10 |
| Quantidade de Relacionamentos Máxima (MaxRQ) | ≥, ≤ | 0 a 10 |

Depois de inicializada a população, cada Função de Casamento é avaliada utilizando uma função de *fitness* (linhas 8 e 9). Em PG, uma função de *fitness* é utilizada para avaliar indivíduos, para que os melhores possam ser selecionados para a próxima geração e os demais descartados. De acordo com o segundo requisito da Seção 4.1, a qualidade de cada árvore deve poder ser medida de forma automática. Em nosso método, para avaliar as Funções de Casamento, utilizamos com cada uma das FCs o algoritmo de desambiguação do Capítulo 3 sobre um conjunto de dados cujo resultado da desambiguação é conhecido. Assim, os resultados obtidos pela FC são comparados com o resultado conhecido e uma métrica de qualidade é utilizada para atribuir um valor de *fitness* para a Função de Casamento. Todo esse processo de avaliação é realizado de forma automática, atendendo o segundo requisito da Seção 4.1.

A etapa seguinte é a Seleção (linha 11), que utiliza os resultados de *fitness* de cada FC para escolher as mais aptas a seguir para a próxima geração. Dessa forma, as FCs com os melhores valores de *fitness* são selecionadas e as demais são descartadas. Este processo, inspirado no processo de Seleção Natural da biologia, garante que os melhores indivíduos da população vão sobreviver, enquanto os piores serão eliminados. Como as próximas gerações são baseadas nos melhores indivíduos, a tendência é que com o passar das gerações os indivíduos tornem-se cada vez melhores.

Nas etapas de *Crossover* (linha 12) e de Mutação (linha 13), novas FCs são geradas a partir de funções-pai, conforme explicamos em detalhe na Seção 4.2. Com isso a população se diversifica e evolui com o passar das gerações.

As fases de Seleção, *Crossover* e Mutação são executadas em um laço até que a máxima geração seja alcançada. Ao final, a população resultante é avaliada e apresentada em ordem decrescente de *fitness.* Conforme demonstraremos no Capítulo 5, nosso algoritmo foi capaz de gerar Funções de Casamento que competem o com Estado da Arte em desambiguação de autores.

# 5 ANÁLISE EXPERIMENTAL

Neste capítulo são apresentados os experimentos realizados para avaliar a utilização de redes sociais em desambiguação de autores, conforme proposta no Capítulo 3, bem como os experimentos realizados para avaliar o algoritmo de geração de Funções de Casamento utilizando Programação Genética, conforme apresentado no Capítulo 4. São descritos um primeiro conjunto de experimentos realizados sobre as FCs criadas manualmente e apresentadas no Capítulo 3 e um segundo conjunto de experimentos realizados com FCs geradas automaticamente através do algoritmo do Capítulo 4.

Na Seção 5.1 são apresentados os conjuntos de dados utilizados nos experimentos. A Seção 5.2 trata das métricas utilizadas para avaliar os resultados, enquanto a Seção 5.3 explica as evidências utilizadas no processo de deduplicação. Tanto os procedimentos realizados quanto os resultados obtidos nos dois conjuntos de experimentos são apresentados na Seção 5.4.

## 5.1 Conjuntos de Dados

De forma a realizar experimentos sobre dados com problemas reais de ambigüidade, todos os experimentos foram realizados sobre dados extraídos de bibliotecas digitais reais. A seguir apresentaremos detalhadamente os conjuntos de dados utilizados.

### 5.1.1 CORA

O conjunto de dados CORA foi criado por Andrew McCallum (MCCALLUM, 2000), consistindo em 1878 citações a publicações reais. Estas publicações foram desambiguadas manualmente pelos autores e colocadas em grupos que se referem à mesma publicação. Como nosso objetivo é desambiguar autores e não publicações, nós agrupamos manualmente os conjunto de autores que se referem ao mesmo autor real, utilizando informações disponíveis nas páginas pessoais dos autores e outras informações disponíveis na Web. O conjunto CORA apresenta 178 autores distintos e 1341 pares duplicados.

### 5.1.2 BDBComp

O segundo conjunto de dados, denominado BDBComp, é um subconjunto da biblioteca digital BDBComp[2] que foi manualmente avaliado e utilizado em (OLIVEIRA; LAENDER; GONÇALVES, 2005) e nos foi disponibilizado pelos autores do artigo. Este conjunto de dados é composto por 361 publicações cujo primeiro autor

[2] http://www.lbd.dcc.ufmg.br/bdbcomp/

possui um dos sobrenomes mais freqüentes na BDBComp, contendo 674 pares de autores duplicados. Como somente o primeiro autor de cada publicação foi avaliado pelos autores de (OLIVEIRA; LAENDER; GONÇALVES, 2005), em nossos experimentos apenas o primeiro autor de cada publicação foi utilizado no processo de desambiguação. Entretanto, todos os co-autores foram utilizados para criar a Rede Social do Autor, conforme descrita no Capítulo 3.

### 5.1.3 DBLP

A biblioteca digital DBLP[3] é uma das bibliotecas mais utilizadas pela comunidade científica de Ciência da Computação. Para nossos experimentos foram extraídos 12 conjuntos de dados diferentes desta biblioteca digital. Um destes conjuntos de dados, denominado *asilva*, foi criado por nós a partir de publicações da DBLP cujo primeiro nome do primeiro autor inicia com a letra *a* e que possui *Silva* como um de seus sobrenomes, resultando em 371 publicações com 773 pares de autores duplicados. Os autores que contém o sobrenome *Silva* foram considerados como os nomes ambíguos e foram sujeitos ao processo de desambiguação. Aqui também foram usados todos os co-autores para criar a Rede Social do Autor. O conjunto *asilva* também foi avaliado manualmente utilizando informações nas páginas pessoais dos autores e na Web.

Tabela 5.1: Conjuntos de Dados da DBLP

| ***Conjunto de Dados*** | ***Publicações*** |
|---|---|
| agupta | 576 |
| akumar | 243 |
| cchen | 801 |
| djohnson | 368 |
| jmartim | 112 |
| jrobinson | 171 |
| jsmith | 924 |
| ktanaka | 280 |
| mbrown | 153 |
| mjones | 260 |
| mmiller | 405 |

Os outros 11 conjuntos de dados da DBLP foram utilizados em (COTA; GONÇALVES; LAENDER, 2007) e cedidos a nós pelos autores do artigo. Estes conjuntos de dados, apresentados na Tabela 5.1, são compostos por citações a publicações cujo primeiro autor possui um dos sobrenomes mais freqüentes na

[3] http://dblp.uni-trier.de/

biblioteca. Cada primeiro autor em um conjunto de dados possui a mesma inicial e o mesmo sobrenome. Por exemplo, o conjunto *agupta* é composto por autores cuja inicial é a letra *a* e o sobrenome é *Gupta*. Como no conjunto *asilva* e no conjunto *bdbcomp* somente o primeiro autor foi desambiguado e todos os co-autores foram utilizados para criar a Rede Social do Autor.

## 5.2 Métricas

No primeiro conjunto de experimentos foram utilizadas métricas tradicionais de Recuperação de Informações: precisão, revocação e *F1-Measure* (SALTON; MCGILL, 1983). Em desambiguação de autores, a precisão identifica a fração dos pares de autores duplicados encontrados pelo método de desambiguação que correspondem a pares duplicados reais. Abaixo temos a fórmula para precisão:

$$precisão = \frac{\{pares\ duplicados\ encontrados\} \cap \{pares\ duplicados\ reais\}}{\{pares\ encontrados\}}$$

A revocação corresponde à fração dos pares de autores duplicados reais que foram encontrados pelo método de desambiguação. Abaixo temos a fórmula para a revocação:

$$revocação = \frac{\{pares\ duplicados\ encontrados\} \cap \{pares\ duplicados\ reais\}}{\{pares\ duplicados\ reais\}}$$

A *F1-Measure* combina precisão e revocação, atribuindo o mesmo peso para ambas e retornando um número entre 0 e 1. Segue abaixo a fórmula para *F-Measure:*

$$F1 - Measure = \frac{2 \cdot precisão \cdot revocação}{precisão + revocação}$$

Em Recuperação de Informações, funções de similaridade costumam ser avaliadas utilizando curvas de precisão e revocação (MANNING; RAGHAVAN; SCHÜTZE, 2008). Para estas avaliações, primeiramente um objeto de consulta (um nome de autor, por exemplo) é comparado a todos os objetos em um conjunto de dados utilizando uma função de similaridade e os resultados são classificados em ordem decrescente de similaridade. Começando no topo da lista, a precisão é computada em pontos de revocação específicos (tradicionalmente são utilizados 11 pontos, de 0% a 100% com intervalos de 10%). Os resultados são, então, interpolados, o que significa que se um ponto de revocação *a* possui uma precisão menor que um ponto de revocação *b* e *b* é um ponto de revocação mais alto que *a*, então *a* assume o mesmo valor de precisão de *b*. Esse processo é realizado para diversas consultas (em nosso caso, para diferentes nomes de autor) e a cada ponto de revocação é calculada a média aritimética da precisão interpolada das diversas consultas no mesmo ponto de revocação. Finalmente, a curva de precisão e revocação para a função de similaridade é construída e comparada a curvas de outras funções de similaridade. Desta forma, é possível comparar a precisão

em diferentes pontos de revocação, apresentando uma avaliação que independe dos limiares escolhidos ao comparar as duas funções.

Porém, enquanto funções de similaridade retornam valores reais entre 0 e 1, uma função de casamento retorna um valor booleano e assim é impossível classificar os resultados. Para comparar funções de casamento da Seção 3.3 utilizando curvas de precisão e revocação nós utilizamos um método diferente para criá-las. Foram testados diferentes limiares para cada FC, começando com limiares mais altos, que resultaram em alta precisão, até limiares mais baixos, que resultaram em alta revocação. Todos os testes foram feitos utilizando todo o conjunto de dados em questão. A curva foi criada plotando os valores obtidos de precisão para cada ponto de revocação. Os dados foram interpolados, de forma que se nossa menor revocação obtida foi de 21,3%, com 100% de precisão, então os pontos de revocação de 10% e 20% assumirão 100% de precisão. Para ilustrar isso, a Tabela 5.2 mostra resultados de testes hipotéticos ao avaliar alguma função de casamento e a tabela 5.3 mostra os resultados interpolados em cinco pontos de revocação. Nós utilizamos 11 pontos de revocação em nossos experimentos.

Tabela 5.2: Resultados de Testes Hipotéticos

| ***Limiar*** | ***Revocação*** | ***Precisão*** |
|---|---|---|
| 0,9 | 21,3% | 100,0% |
| 0,7 | 54,5% | 89,2% |
| 0,5 | 73,1% | 62,3% |
| 0,3 | 92,3% | 31,2% |
| 0,1 | 100,0% | 1,7% |

Tabela 5.3: Resultados Interpolados utilizando cinco Pontos de Revocação

| ***Revocação*** | ***Precisão*** |
|---|---|
| 0% | 100,0% |
| 25% | 89,2% |
| 50% | 89,2% |
| 75% | 31,2% |
| 100% | 1,7% |

No segundo conjunto de experimentos, foi utilizada a métrica de agrupamento K, definida em (LAPIDOT, 2002), que é a média geométrica de duas outras métricas de agrupamento definidas no mesmo trabalho: Pureza Média por Grupo (PMG) e Pureza Média por Autor (PMA). PMG avalia a pureza dos grupos gerados, ou seja, o valor

percentual que os grupos gerados incluem de autores que se referem ao mesmo autor real. Esta métrica trata do problema de citações misturadas. Assim, quanto mais puros os grupos gerados, mais perto de um será o valor de PMG e menos misturadas estarão as citações. A fórmula de PMG é:

$$PMG = \frac{1}{N}\sum_{i=0}^{q}\sum_{j=0}^{R}\frac{{n_{ij}}^2}{n_i}$$

onde R é o número de grupos reais;

N é o número total de citações no conjunto de dados;

q é o número de grupos gerados;

$n_{ij}$ é o número de elementos no grupo gerado i pertencendo ao grupo real j;

$n_i$ é o número de elementos no grupo gerado i.

PMA mede o nível de fragmentação dos grupos gerados em comparação aos grupos reais. Esta métrica trata do problema das citações separadas. Quanto mais próximo de um estiver o valor, menos fragmentado os grupos gerados estarão e menos separadas estarão as citações. A fórmula para PMA é:

$$PMA = \frac{1}{N}\sum_{i=0}^{R}\sum_{j=0}^{q}\frac{{n_{ij}}^2}{n_j}$$

onde R é o número de grupos reais;

N é o número total de citações no conjunto de dados;

q é o número de grupos gerados;

$n_{ij}$ é o número de elementos no grupo gerado i pertencendo ao grupo real j;

$n_j$ é o número de elementos no grupo gerado j.

A métrica K combina PMG e PMA calculando a média geométrica de ambos, retornando um valor entre zero e um, sendo que a melhor situação ocorre quando PMG e PMA são iguais a 1. Abaixo a equação para o cálculo de K:

$$K = \sqrt{PMG \cdot PMA}$$

## 5.3 Evidências Utilizadas

Nesta seção trataremos de forma mais detalhada das evidências utilizadas em nossos experimentos, referidas brevemente no Capítulo 4. Tipicamente o processo de deduplicação é realizado sobre um conjunto de citações a publicações, que possuem dados como: título, local de publicação, nomes dos autores, ano de publicação, número de páginas, entre outros. Destes dados, o principal para nós são os nomes dos autores,

que são o objeto do nosso processo de deduplicação e a partir dos quais é criada a Rede Social de Autores.

Entretanto, o título e o local de publicação também são informações importantes e que costumam ser utilizados por diversos métodos de deduplicação de autores. Títulos similares, por exemplo, podem ser um indicativo de que dois autores são o mesmo, pois autores tendem a publicar diversos artigos com assuntos e títulos semelhantes. Já o local de publicação é importante por que autores tendem a publicar artigos em diferentes edições de um mesmo congresso ou revista. Logo, a similaridade dos locais de publicação é uma evidência que não pode ser desprezada.

Na Tabela 5.4 estão listados os tipos de evidência utilizados. Esta tabela é idêntica à Tabela 4.1, e mostra os operadores e valores possíveis para cada tipo de evidência. A seguir explicaremos em detalhe cada um destes tipos.

**Similaridade de Nomes (SimNome):** Aqui é calculada a similaridade dos nomes dos dois autores que estão sendo comparados. A função retorna um valor de similaridade entre 0 e 1, sendo que o valor 1 significa que os nomes são iguais. Aqui foi utilizada a função de similaridade de Trigrams (ELGARAMID; IPEIROTIS; VERYKIOS, 2007).

Tabela 5.4: Tipos de Evidência Utilizados

| ***Tipo de Evidência*** | ***Operadores*** | ***Valores*** |
|---|---|---|
| Similaridade de Nomes (SimNome) | ≥, ≤ | 0 a 1 |
| Similaridade de Títulos (SimTitulo) | ≥, ≤ | 0 a 1 |
| Similaridade de Local de Publicação (SimLocal) | ≥, ≤ | 0 a 1 |
| Casamento de Inicial e Sobrenome (IniSobrenome) | = | 0 ou 1 |
| Número de Palavras do Título que Casam (NumPalTitulo) | ≥, ≤ | 0 a 8 |
| Similaridade de Palavras do Título (SimPalTitulo) | ≥, ≤ | 0 a 1 |
| Primeiro Nome Abreviado (Abrev) | = | 0 ou 1 |
| Existência de Relacionamento (RE) | = | 0 ou 1 |
| Força de Relacionamento (RS) | ≥, ≤ | 0 a 10 |
| Quantidade de Relacionamentos Mínima (MinRQ) | ≥, ≤ | 0 a 10 |
| Quantidade de Relacionamentos Máxima (MaxRQ) | ≥, ≤ | 0 a 10 |

**Similaridade de Títulos (SimTitulo):** Esta evidência calcula a similaridade dos dois títulos de publicação em que cada um dos dois autores sendo comparados aparece. Aqui também é utilizada a similaridade de Trigrams.

**Similaridade de Locais de Publicação (SimLocal):** Aqui é calculada a similaridade do nome dos dois locais de publicação das citações em que cada um dos

dois autores sendo comparados aparece. Novamente é utilizada a similaridade de Trigrams.

**Casamento de Inicial e Sobrenome (IniSobrenome):** Esta evidência retorna o valor 1 (verdadeiro) se a letra inicial e o último sobrenome dos nomes dos autores comparados são exatamente iguais e 0 (falso) caso contrário. Por exemplo, ao comparar "João C. da Silva" e "J. Carlos da Silva", *IniSobrenome* retorna verdadeiro. Mas ao comparar "João C. da Silva" e "João C. da Silva Mello", *IniSobrenome* retorna falso.

**Número de Palavras do Título que Casam (NumPalTitulo):** O valor retornado por esta evidência é o número de palavras iguais entre os dois títulos de publicação em que cada um dos dois autores sendo comparados aparece. Não são consideradas s*topwords* (BAEZA-YATES; RIBEIRO-NETO, 1999), que são aquelas palavras que aparecem com freqüência em diversos títulos e que não indicam uma relação entre eles (e.g., artigos, preposições). Por exemplo, comparando os títulos "Evaluating the use of social networks in author name disambiguation in digital libraries" e "Using Genetic Programming to Evaluate the Impact of Social Network Analysis in Author Name Disambiguation", *NumPalTitulo* é igual a 4, pois ambos os títulos contém as palavras "of", "in", "the", "social", "author", "name" e "disambiguation", sendo que as palavras "of", "in" e "the" são consideradas *stopwords* e não entram na soma.

**Similaridade de Palavras do Título (SimPalTitulo):** Esta evidência é semelhante a *SimPalTitulo,* porém o valor é normalizado pelo número de palavras do título com o maior número de palavras, descontadas as *stopwords.* No exemplo do parágrafo anterior, o valor 4 seria dividido por 11, que é o número de palavras do segundo título descontando as *stopwords,* resultado em *SimPalTitulo* = 0,36.

**Primeiro Nome Abreviado (Abrev):** Aqui, o valor retornado é 1 (verdadeiro) quando a primeira letra do nome de qualquer um dos dois autores é seguida por um espaço em branco ou por um ponto, e 0 (falso) caso contrário. Assim, para os nomes "J. Carlos da Silva" e "A F Menezes", *Abrev* é verdadeiro, enquanto para os nomes "Marcos S. Almeida" e "Júlia Antunes", *Abrev* é falso.

**Existência de Relacionamento (RE):** Esta é a mesma evidência *RE* descrita no Capítulo 3. Retorna 1 quando *RE* entre os dois autores é verdadeiro e 0 quando é falso.

**Força de Relacionamento (RS):** Esta também é a mesma evidência *RS* descrita no Capítulo 3. Foram utilizados os valores inteiros de 0 a 10 nas funções de casamento geradas automaticamente.

**Quantidade de Relacionamentos Mínima (MinRQ):** Esta evidência corresponde ao menor valor de *RQ* entre os dois autores sendo comparados.

**Quantidade de Relacionamentos Máxima (MaxRQ):** Aqui, é retornado o maior valor de *RQ* entre os dois autores sendo comparados.

## 5.4 Experimentos

A seguir são apresentados os experimentos que demonstram a melhora na qualidade do processo de desambiguação de autores proporcionada pelo uso da Rede Social de Autores e das Medidas de Relacionamento apresentadas no Capítulo 3.

### 5.4.1 Objetivos

Os experimentos apresentados nesta seção possuem os seguintes objetivos:

(1) Avaliar o impacto de adicionar análise de Redes Sociais a métodos tradicionais de desambiguação baseados em similaridade de nomes.
(2) Encontrar a distância máxima de relacionamento entre autores que deve ser considerada no processo de desambiguação.
(3) Avaliar o impacto de adicionar análise de Redes Sociais a Funções de Casamento mais complexas que utilizam diversas evidências.
(4) Avaliar a qualidade das Funções de Casamento geradas utilizando Programação Genética.

### 5.4.2 Experimentos sobre Funções de Casamento Geradas Manualmente

O primeiro conjunto de experimentos utiliza as FCs apresentadas na Seção 3.3 e visa atender os objetivos 1 e 2 da Subseção 5.4.1. No primeiro deles, comparamos os resultados das FCs sobre três conjuntos de dados: CORA, BDBComp e o conjunto *asilva* da DBLP. Para os parâmetros das funções, foram testados diversos limiares, com intervalos de 0,25, e a distância $d = 2$ foi utilizada para as funções baseadas na Rede Social de Autores. A Tabela 5.4 apresenta os melhores resultados dos experimentos utilizando a função de similaridade Levenshtein e a Tabela 5.5 apresenta os melhores resultados utilizando similaridade de Trigrams.

Tabela 5.4: Resultados das Funções de Casamento utilizando Levenshtein

| | ***CORA*** | | | ***BDBComp*** | | | ***DBLP - asilva*** | | |
|---|---|---|---|---|---|---|---|---|---|
| ***FC*** | ***Prec.*** | ***Rev.*** | ***F1*** | ***Prec.*** | ***Rev.*** | ***F1*** | ***Prec.*** | ***Rev.*** | ***F1*** |
| SimNome | 0,780 | 0,702 | 0,739 | 0,710 | 0,522 | 0,602 | 0,822 | 0,753 | 0,786 |
| SimNomeRE | 0,996 | 0,203 | 0,337 | 0,938 | 0,430 | 0,590 | 1,000 | 0,380 | 0,511 |
| SimNomeOuRE | 0,783 | 0,716 | 0,748 | 0,694 | 0,783 | 0,736 | 0,974 | 0,771 | 0,861 |
| SimNomeRERQ | 0,781 | 0,708 | 0,743 | 0,766 | 0,760 | 0,763 | 0,973 | 0,759 | 0,853 |
| SimNomeRERQRS | 0,783 | 0,720 | **0,750** | 0,768 | 0,811 | **0,789** | 0,976 | 0,894 | **0,933** |

Tabela 5.5: Resultados das Funções de Casamento utilizando Sim. de Trigrams

| | ***CORA*** | | | ***BDBComp*** | | | ***DBLP - asilva*** | | |
|---|---|---|---|---|---|---|---|---|---|
| ***FC*** | ***Prec.*** | ***Rev.*** | ***F1*** | ***Prec.*** | ***Rev.*** | ***F1*** | ***Prec.*** | ***Rev.*** | ***F1*** |
| SimNome | 0,992 | 0,887 | 0,936 | 0,578 | 0,730 | 0,646 | 0,968 | 0,746 | 0,843 |
| SimNomeRE | 1,000 | 0,310 | 0,473 | 0,762 | 0,543 | 0,634 | 0,994 | 0,413 | 0,583 |
| SimNomeOuRE | 0,936 | 0,899 | 0,917 | 0,721 | 0,838 | 0,775 | 0,953 | 0,955 | 0,954 |
| SimNomeRERQ | 0,991 | 0,911 | **0,949** | 0,814 | 0,774 | 0,793 | 0,963 | 0,920 | 0,941 |
| SimNomeRERQRS | 0,908 | 0,916 | 0,912 | 0,812 | 0,835 | **0,824** | 0,957 | 0,955 | **0,956** |

Como podemos ver em ambas as tabelas, ao compararmos a função *SimNomeRE* com a função de referência *SimNome,* existe uma melhora significativa em precisão, o que confirma nossa hipótese de que autores relacionados têm uma maior probabilidade de serem duplicatas do que autores não relacionados. Na BDBComp, utilizando Levenshtein, a precisão foi melhorada em 0,23 e na DBLP em 0,18, alcançando 100% de precisão. Entretanto, ao utilizar *SimNomeRE* os casamentos são limitados a autores relacionados e com isso a revocação cai drasticamente, resultando em uma redução de F1-measure ao comparar esta função com a função de referência. Isso mostra que, embora autores relacionados possuam uma chance maior de serem duplicatas, existem muitos autores duplicados que não são relacionados e que acabam sendo ignorados pela função *SimNomeRE.*

A função *SimNomeOuRE* tenta resolver este problema ao estabelecer dois limiares, um, mais alto, para quaisquer autores e outro, mais baixo, para autores relacionados. Como os resultados mostram, isso melhora de forma significativa a revocação em relação à função de referência com pouca redução – e em alguns casos com ganho – de precisão, aumentando a F1-measure em todos os cenários, exceto na base Cora com Trigrams, onde houve uma redução de 0,02. Houve melhora de revocação em todos os cenários, com uma melhora de 0,13 em F1-measure na BDBComp com Levenshtein e de 0,11 na DBLP com trigrams.

Na função *SimNomeRERQ,* conforme explicado na Seção 3.3, a Quantidade de Relacionamentos (RQ) é utilizada para a escolha do limiar a ser utilizado. Em nossos experimentos, utilizamos um valor maior para $k$ e um valor menor para $l$ e dessa forma autores com alta RQ devem ser relacionados, mas necessitam de uma similaridade de nomes menor, enquanto autores com RQ baixa não precisam ser relacionados, mas necessitam de uma similaridade maior. Utilizamos $q = 3$ para os conjuntos de dados da DBLP e da BDBComp. Para CORA, que possui publicações duplicadas e, por conseqüência, um número maior de conexões por autor, foi usado $q = 25$. Nossos resultados mostram que *SimNomeRERQ* melhora revocação e F1-measure em todos os cenários em comparação à função de referência. A precisão também é melhorada em quatro de seis cenários.

A última função, *SimNomeRERQRS*, é similar à anterior, mas utiliza a Força de Relacionamento (RS) para adicionar um terceiro limiar, que é utilizado para pares de autores com uma Força de Relacionamento mínima entre eles. Utilizamos $s = 2$ para todos os cenários. Os resultados mostram que *SimNomeRERQRS* possui a melhor F1-measure para cinco dos seis cenários ao compará-la com as demais funções.

O gráfico da Figura 5.1 mostra os resultados em termos de F1-measure em cada conjunto de dados com cada FC utilizando Levenshtein, enquanto a Figura 5.2 mostra a mesma comparação utilizando Trigrams. Como podemos observar nos gráficos, nossos experimentos mostram que ao utilizar as evidências extraídas da Rede Social do Autor o desempenho de ambas as funções de similaridade foi melhorada de forma significativa na DBLP e na BDBComp, com *SimNomeRERQRS* apresentando os melhores resultados.

No conjunto de dados CORA, a melhora não foi tão significativa e as funções de casamento *SimNomeRERQRS* e *SimNomeOuRE* tiveram um desempenho pior que a função de referência ao utilizar Trigrams. Uma das razões para este resultado é o fato de que a função de referência com Trigrams obteve ótimos resultados e, portanto não havia muito a ser melhorado. Além disso, muitos nomes de autores na base CORA contêm

erros de digitação e, como existem publicações duplicadas utilizando diferentes padrões de abreviação, os nomes são abreviados de diversas formas, resultando em mais variações do que na DBLP e na BDBComp, o que afeta negativamente a Rede Social de Autores. Por exemplo, em um dos registros da base CORA, a autora Carla Brodley estava descrita como “Carla E Brodley”, enquanto em outro registro ela estava descrita como “Brodley C E”. Como a Rede Social de Autores é criada ligando autores com a mesma letra inicial e o mesmo último nome, esses dois nomes não foram ligados. Na DBLP e na BDBComp existem menos variações do mesmo autor e o sobrenome sempre aparece no final do nome, portanto a Rede Social de Autores será mais informativa. Entretanto, mesmo com esses problemas, o uso de Redes Sociais não diminuiu a qualidade dos resultados de forma significativa e ainda melhorou a qualidade na maioria dos casos.

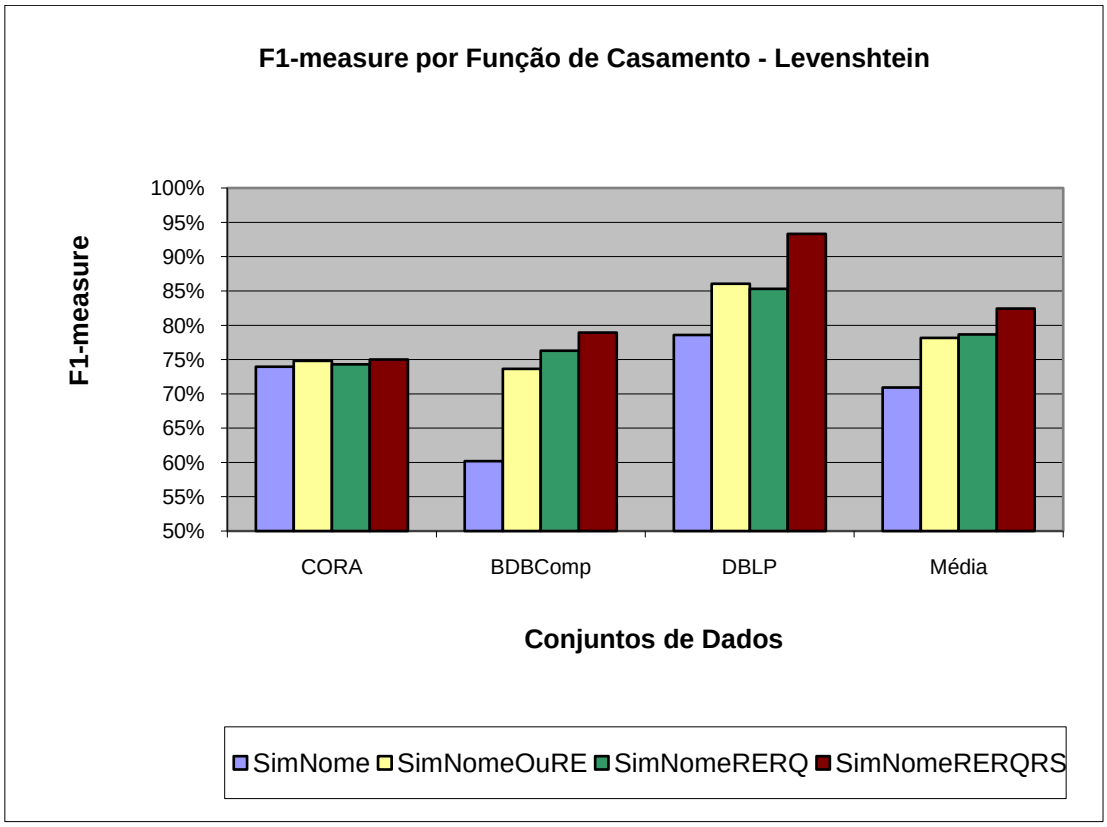


Figura 5.1: F1-measure por Função de Casamento utilizando Levenshtein

Na Figura 5.3, os resultados das funções *SimNome*, *SimNomeOuRE, SimNomeRERQ* e *SimNomeRERQRS* utilizando similaridade de Trigrams são comparados utilizando curvas de precisão e revocação. Para a base CORA, o comportamento é similar em todas as funções e mesmo a função de referência mantém 100% de precisão até o ponto de 80% de revocação. No ponto de 90% de revocação, as *SimNomeOuRE* e *SimNomeRERQRS* apresentam melhores resultados quando comparadas a *SimNome*, demonstrando que mesmo quando a função de similaridade original tem um bom desempenho, o uso da Rede Social de Autores ainda pode melhorar os resultados. Nos conjuntos de dados da DBLP e da BDBComp, entretanto, a melhoria é mais evidente. Até o ponto de 70% de revocação os resultados apresentam alta precisão na DBLP para

todas as funções, porém após 80% de revocação há uma queda acentuada em precisão para a função de referência *SimNome*, enquanto as funções baseadas em Redes Sociais mantêm uma alta precisão. Na BDBComp, as funções baseadas em Redes Sociais possuem maior qualidade em todos os pontos de revocação, com exceção do ponto de 50% de revocação, em que a função de referência possui um desempenho melhor do que *SimNomeOuRE*.

Como a média dos três conjuntos de dados demonstra, na Figura 5.3, *SimNomeRERQRS* apresentou os melhores resultados em nosso experimento em todos os pontos de revocação. *SimNomeOuRE* apresentou a segunda melhor precisão ao trabalhar com altos níveis de revocação, enquanto *SimNomeRERQ* apresentou a segunda melhor precisão ao trabalhar com baixos níveis de revocação.

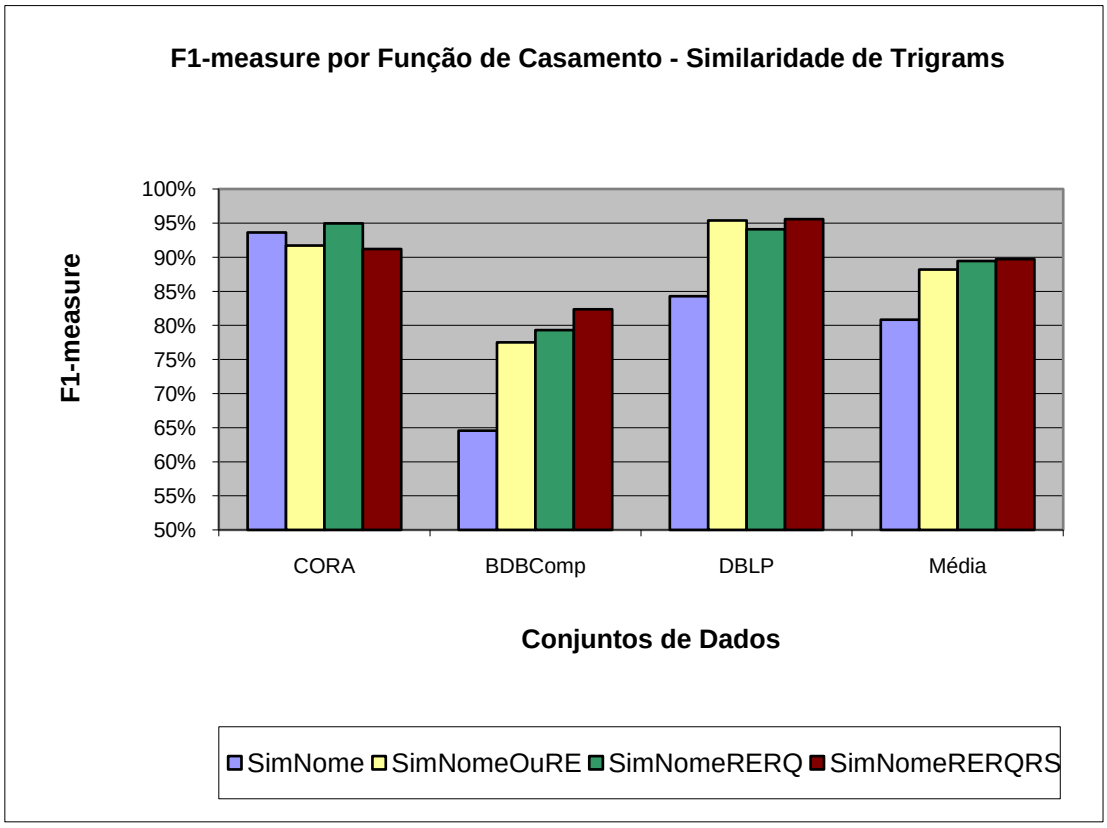


Figura 5.2: F1-measure por Função de Casamento utilizando Trigrams

De forma a demonstrar que a melhora apresentada pelas funções baseadas em Redes Sociais em relação à função de similaridade original são estatisticamente significativas, nós utilizamos o Teste de Wilcoxon (WILCOXON, 1945), comparando os melhores resultados de *SimNomeOuRE*, *SimNomeRERQ* e *SimNomeRERQRS* com os melhores resultados de *SimNome* no conjunto de dados da DBLP utilizando Trigrams, conforme apresentado anteriormente na Tabela 5.5. O Teste de Wilcoxon é uma alternativa ao Teste-T de Student quando as amostras não possuem uma distribuição normal, como é o caso aqui. Foram utilizadas 1000 amostras para cada teste e os valores de $p$ foram menores que 0,0001 em todos os testes. Como estes valores foram menores que o limiar estatístico de 0,01, os resultados do teste demonstram que as três funções possuem um

desempenho estatisticamente superior quando comparadas à função de referência *SimNome*.

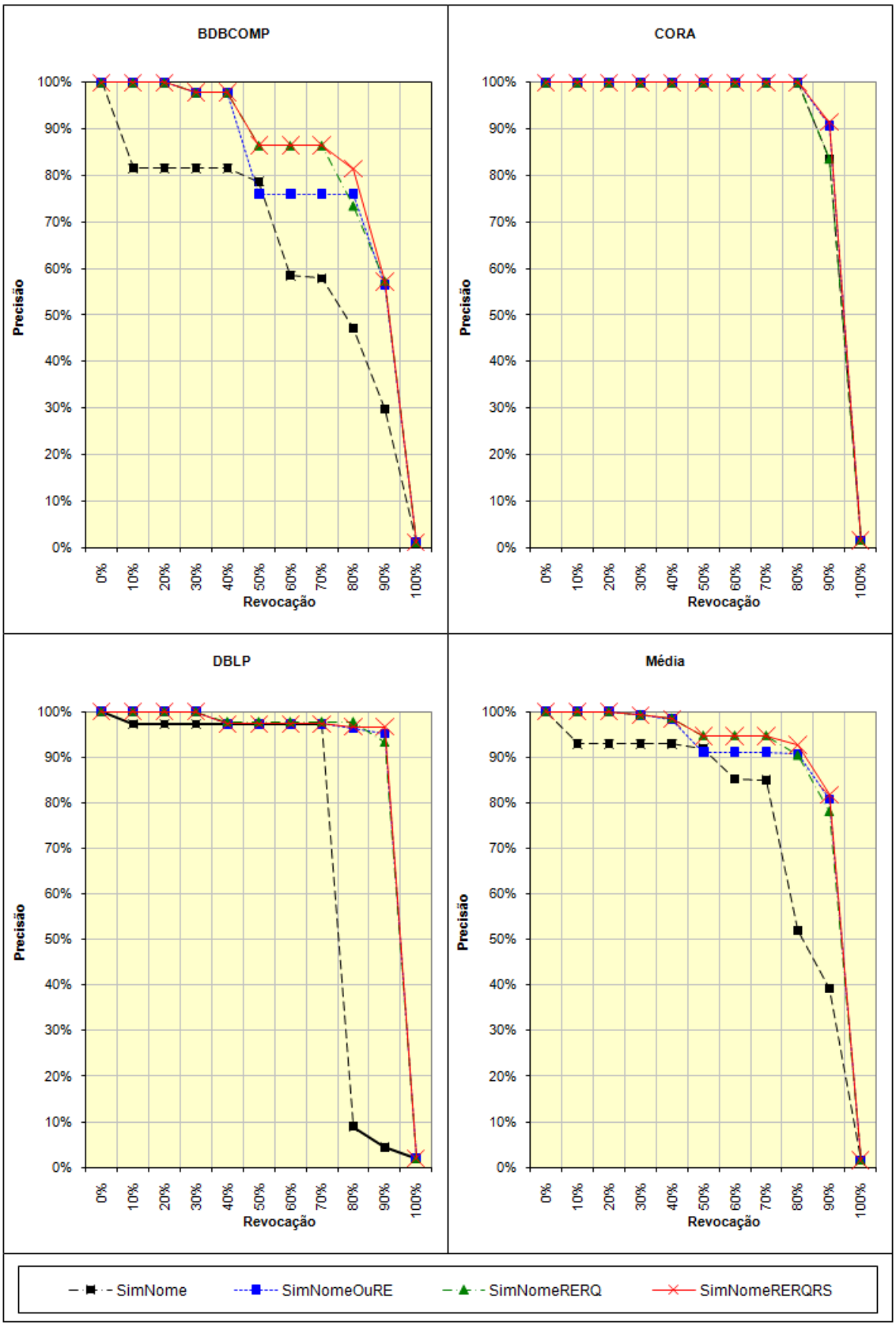


Figura 5.3: Curvas de Precisão e Revocação por Função utilizando Trigrams

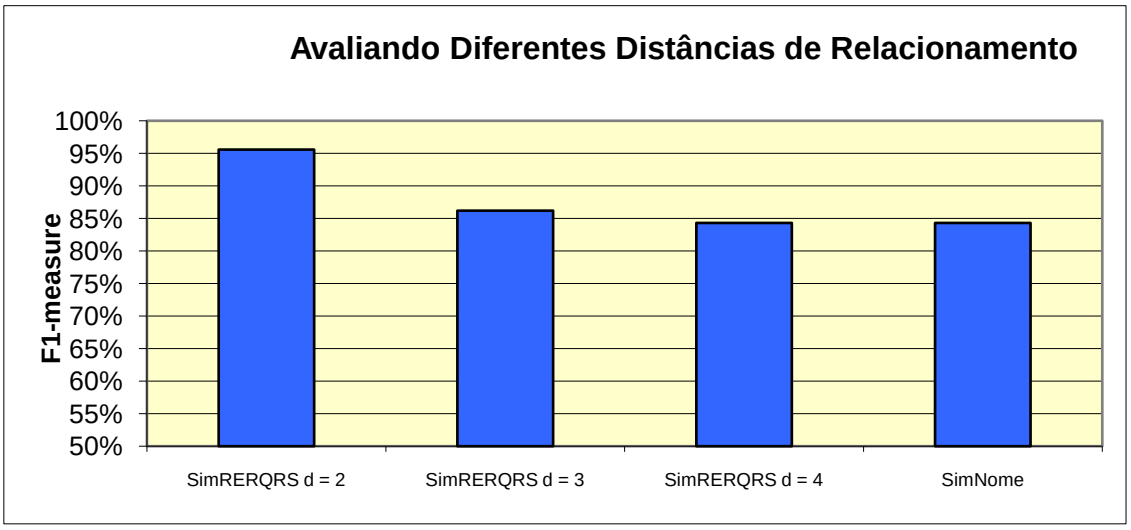


Figura 5.4: Avaliação de Diferentes Distâncias Máximas na DBLP

Por fim, no último experimento as Funções de Casamento foram comparadas utilizando diferentes valores *d*, que representa a distância entre dois autores. Conforme descrito no Capítulo 3, a distância entre dois autores é o comprimento do menor caminho entre eles e o comprimento do caminho é calculado como o número de segmentos autor-publicação-autor no caminho. Foram realizados experimentos com o valor de *d* variando entre dois e quatro. A distância um não foi testada, pois seriam considerados relacionamentos entre autores e co-autores, o que não traz melhorias aos resultados a não ser que um autor apareça mais de uma vez em uma mesma citação.

Tabela 5.6: Percentual de Pares de Autores Conectados no Conjunto de Dados DBLP

| ***Distância Máxima*** | ***% de Pares Conectados*** |
|---|---|
| 2 | 2,05% |
| 3 | 18,15% |
| 4 | 84,35% |

Os resultados deste experimento mostraram que à medida que a distância máxima aumenta além de dois, a revocação aumenta, mas a precisão cai de forma drástica. Para aumentar a precisão, os limiares precisam ser elevados, o que faz com que a revocação caia de forma significativa. Ao final, $d = 2$ apresentou os melhores resultados em termos de F1-measure, como é demonstrado na Figura 5.4, que mostra os resultados de *SimNomeRERQRS* no conjunto de dados da DBLP. F1-measure diminui à medida que a distância aumenta e utilizando $d = 4$, a qualidade dos resultados de *SimNomeRERQRS* torna-se idêntica à da função de referência *SimNome.* A explicação para este comportamento é simples: à distância quatro cada autor está conectado, em média, a 84,35% dos outros autores no conjunto de dados da DBLP, conforme demonstrado na Tabela 5.6, indicando a presença do efeito do "mundo pequeno". Como praticamente

todos os pares de autores estão conectados a $d = 4$, o uso das Medidas de Relacionamento torna-se irrelevante.

### 5.4.3 Experimentos sobre Funções de Casamento Geradas Automaticamente

Neste conjunto de experimentos, que visa atender os objetivos 3 e 4, foram geradas três funções de casamento utilizando o algoritmo de Programação Genética do Capítulo 4 para avaliar o impacto nos resultados ao utilizar as evidências encontradas na Rede Social de Autores. A primeira função gerada, denominada *SemRedeSocial*, não utiliza as Medidas de Relacionamento do Capítulo 3 e é utilizada como a função de referência. A segunda função, *SomenteRE*, utiliza somente a medida de relacionamento *RE*, enquanto a terceira função, *TodasMedRel*, utiliza todas as evidências que fazem uso das Medidas de Relacionamento. A Tabela 5.7 mostra quais tipos de evidência foram utilizadas com quais funções de casamento.

Tabela 5.7: Evidências Utilizadas para Gerar cada Função de Casamento

| ***Tipo de Evid.*** | ***SemRedeSocial*** | ***SomenteRE*** | ***TodasMedRel*** |
|---|---|---|---|
| SimNome | Sim | Sim | Sim |
| SimTitulo | Sim | Sim | Sim |
| SimLocal | Sim | Sim | Sim |
| IniSobrenome | Sim | Sim | Sim |
| NumPalTitulo | Sim | Sim | Sim |
| SimPalTitulo | Sim | Sim | Sim |
| Abrev | Sim | Sim | Sim |
| RE | Não | Sim | Sim |
| RS | Não | Não | Sim |
| MinRQ | Não | Não | Sim |
| MaxRQ | Não | Não | Sim |

Para gerar as funções de casamento, foram utilizados os conjuntos de dados *akumar*, *jsmith*, *ktanaka* e *BDBComp* como conjunto de treinamento, com um total de 1852 registros. Já os conjuntos *agupta*, *cchen*, *djohnson*, *jmartim*, *jrobinson*, *mbrown*, *mjones* e *mmiller* foram utilizados como conjunto de avaliação, com um total de 2846 registros. Ao utilizar o algoritmo de PG para gerar as funções, foi utilizada a métrica K como função de *fitness* sobre o conjunto de treinamento. A métrica K foi escolhida por combinar PMG e PMA e, dessa forma, tratar dos dois problemas que a desambiguação de autores se propõe a resolver: citações misturadas e citações separadas. Ao utilizar K como a função de *fitness*, o algoritmo de PG gerará funções que buscam minimizar estes dois problemas. Como parâmetros, foram utilizados 20 como o tamanho da população

inicial e 300 como o número de gerações. Estes parâmetros foram escolhidos com base em experimentos preliminares que mostraram que valores maiores do que estes tanto para a população e quanto para o número de gerações somente tornavam o algoritmo mais custoso sem trazer melhoras importantes para os resultados. Como o algoritmo realiza escolhas aleatórias e é não-determinístico, cada função de casamento foi gerada cinco vezes e aquela que apresentou os melhores resultados no conjunto de treinamento foi escolhida.

A seguir, temos a função *SemRedeSocial* gerada pelo algoritmo:

(SimNome ≥ 0.57 e SimLocal ≥ 0.85 e IniSobrenome = 1) ou (NumPalTitulo ≥ 7 e SimLocal ≥ 0.43 e IniSobrenome = 1) ou (SimNome ≥ 0.94)

A função gerada mostra que o nome do autor é a evidência mais importante e é utilizado em todos os conjuntos de evidências ligados pelo operador *e* na função de casamento através dos tipos de evidência *IniSobrenome* e *SimNome*. Quando os nomes são praticamente idênticos (similaridade maior do que 0,94) nenhuma outra evidência é utilizada, do contrário o local e o título são utilizados em conjunto com o nome.

A seguir, temos a função *SomenteRE* gerada:

(SimPalTitulo ≥ 0.72 e IniSobrenome = 1) ou (SimNome ≥ 0.4 e RE = 1 e IniSobrenome = 1) ou (SimNome ≥ 0.97 e Abrev = 0 e IniSobrenome = 1) ou (Abrev = 1 e IniSobrenome = 1 e SimLocal ≥ 0.39 e SimNome ≥ 0.71 e RE = 0 e SimPalTitulo ≥ 0.28) ou (SimPalTitulo ≥ 0.49 e SimNome ≥ 0.45 e Abrev = 1) ou (SimTitulo ≥ 0.59 e RE = 0 e IniSobrenome = 1) ou (SimTitulo ≥ 0.22 e Abrev = 1 e IniSobrenome = 1 e SimLocal ≥ 0.39 e SimNome ≥ 0.7 e RE = 1)

Nesta função, o nome continua a ser a evidência mais importante. Porém, ao adicionar *RE* podemos ver que autores relacionados precisam de um menor grau de similaridade de nomes para casar, enquanto autores não relacionados precisam de uma maior similaridade de nomes ou de outras evidências como locais e títulos similares. Além disso, a evidência *Abrev*, que não havia sido selecionada pelo algoritmo ao gerar a função anterior, aparece nesta função. Em *SomenteRE*, quando nomes não são abreviados e são praticamente idênticos (similaridade maior do que 0,97), eles casam, mas quando são abreviados necessitam de outras evidências, como *RE*, para casar.

Abaixo, apresentamos a função *TodasMedRel* gerada:

(Abrev = 1 e SimNome ≥ 0.94 e MinRQ ≤ 1) ou (SimTitulo ≥ 0.39 e MaxRQ ≤ 5 e SimNome ≥ 0.87 e NumPalTitulo ≥ 2 e IniSobrenome = 1) ou (SimPalTitulo ≥ 0.23 ou SimNome ≥ 0.87 e Abrev = 1 e SimLocal ≥ 0.35 e MaxRQ ≤ 3) ou (SimLocal ≥ 0.67 e MaxRQ ≤ 3 e SimNome ≥ 0.98) ou (Abrev = 1 e SimNome ≥ 0.45 e NumPalTitulo ≥ 4 e MinRQ ≤ 2 e IniSobrenome = 1 e RE = 0) ou (IniSobrenome = 1 e RE = 1 e MaxRQ ≤ 9) ou (RE = 1 e SimNome ≥ 0.72) ou (RS ≥ 2 e IniSobrenome = 1)

Em *TodasMedRel* as demais Medidas de Relacionamento foram utilizadas, juntamente com *RE*. Nesta função, os autores casam com *RS* maior ou igual a 2 e com seus nomes possuindo a mesma inicial e sobrenome. Isso mostra que, com um alto valor de *RS*, um número menor de evidências é necessário para que os autores casem. Também são utilizados *MinRQ* e *MaxRQ* na função. Por exemplo, um caso em que os

autores casam ocorre quando os nomes são similares, existe um relacionamento entre eles e *MaxRQ* é menor do que 9. Isso mostra que, à medida que *RQ* aumenta *RE* perde seu valor. Isso decorre do fato de que um autor com muitos relacionamentos possui uma maior chance de estar relacionado a uma pessoa diferente e que possui um nome similar, um falso positivo, do que um autor com poucos relacionamentos.

Na Tabela 5.8 é apresentado um comparativo de qualidade utilizando a métrica K nos conjuntos de treinamento e de avaliação. Como nossos resultados mostram, ao adicionar a evidência *RE* à função de casamento a qualidade dos resultados aumenta de forma significativa. Ao comparar *SemRedeSocial* a *SomenteRE* nós temos um grande aumento em K em todos os conjuntos de dados. No conjunto de treinamento nós temos um aumento médio de 0,15, enquanto no conjunto de avaliação o aumento médio é de mais de 0,14.

A diferença de qualidade de *SomenteRE* para *TodasMedRel* não é tão significativo e em alguns conjuntos de dados *SomenteRE* apresentou um melhor desempenho, mas na média os resultados demonstram que utilizar todas as Medidas de Relacionamento traz uma melhora significativa de qualidade. No conjunto de avaliação, esta melhora foi de mais de 0,04.

Tabela 5.8: Comparação das Funções de Casamento usando a Métrica K

| ***Conjunto de Dados*** | ***SemRedeSocial*** | ***SomenteRE*** | ***TodasMedRel*** |
|---|---|---|---|
| akumar | 0,770 | 0,877 | 0,864 |
| jsmith | 0,561 | 0,773 | 0,836 |
| ktanaka | 0,666 | 0,918 | 0,903 |
| bdbcomp | 0,900 | 0,932 | 0,937 |
| **Média Treinamento** | **0,724** | **0,875** | **0,885** |
| agupta | 0,608 | 0,699 | 0,880 |
| cchen | 0,523 | 0,569 | 0,573 |
| djohnson | 0,601 | 0,719 | 0,765 |
| jmartin | 0,728 | 0,826 | 0,872 |
| jrobinson | 0,522 | 0,858 | 0,808 |
| mbrown | 0,614 | 0,809 | 0,734 |
| mjones | 0,564 | 0,655 | 0,738 |
| mmiller | 0,656 | 0,806 | 0,911 |
| **Média Avaliação** | **0,602** | **0,743** | **0,785** |

Para mostrar que o nosso método de geração de funções de casamento é capaz de gerar funções que competem com o estado da arte, nós comparamos a função

*TodasMedRel*, que foi a melhor função gerada (as funções geradas por PG que utilizaram as Medidas de Relacionamento tiveram resultados melhores que as funções criadas manualmente), com o método HHC (COTA; GONÇALVES; LAENDER, 2007), utilizando o conjunto de avaliação. Como mostram os dados da Tabela 5.9, os resultados dos dois métodos foram semelhantes. Apenas em dois conjuntos de dados houve uma diferença maior que 0,02: em *agupta* e em *jrobinson* a função *TodasMedRel* venceu por 0,103 e 0,048, enquanto em *mbrown* HHC venceu por 0,121. Em média, *TodasMedRel* venceu por 0,002, o que podemos considerar como um empate entre os dois métodos. Com isso fica demonstrado que o nosso algoritmo de geração de funções utilizando PG é capaz de gerar funções de casamento eficientes.

Tabela 5.9: Comparação entre HHC e TodasMedRel

| ***Conjunto de Dados*** | ***HHC*** | ***TodasMedRel*** | ***Diferença*** |
|---|---|---|---|
| agupta | 0,777 | 0,880 | 0,103 |
| cchen | 0,588 | 0,573 | -0,015 |
| djohnson | 0,748 | 0,765 | 0,017 |
| jmartin | 0,885 | 0,872 | -0,013 |
| jrobinson | 0,760 | 0,808 | 0,048 |
| mbrown | 0,855 | 0,734 | -0,121 |
| mjones | 0,742 | 0,738 | -0,004 |
| mmiller | 0,911 | 0,911 | 0,000 |
| **Média Avaliação** | **0,783** | **0,785** | **0,002** |

# 6 CONCLUSÃO

Nesta dissertação, foi avaliado o uso de redes sociais como evidência para resolver o problema de desambiguação de nomes de autores em bibliotecas digitais e demonstrado, através de experimentos sobre dados reais, que o uso deste tipo de evidência trás um ganho significativo para a qualidade dos resultados. Através das Medidas de Relacionamento, foi apresentada uma heurística para a utilização das informações contidas nas redes sociais em desambiguação de nomes e foi mostrado como estas medidas podem ser utilizadas em funções de casamento em conjunto com outras evidências baseadas em atributos. Adicionalmente, foi proposto um método de geração automática de funções de casamento baseado em Programação Genética a partir de um conjunto de evidências.

Os experimentos realizados e apresentados nesta dissertação demonstraram que:

(1) O uso de redes sociais, através das Medidas de Relacionamento, combinado com outras evidências baseadas em atributos, como similaridade de nomes de autores, trás um ganho significativo na qualidade dos resultados.

(2) Ao utilizar evidências baseadas em redes sociais, devem ser utilizadas somente conexões com distância máxima igual a dois, pois ao considerar distâncias maiores que dois, o ganho de qualidade trazido por estas evidências diminui.

(3) O método de geração de funções de casamento baseado em Programação Genética proposto nesta dissertação produz funções de qualidade, pois os resultados obtidos pelas mesmas são equivalentes aos resultados obtidos por um método do estado da arte HHC (COTA; GONÇALVES; LAENDER, 2007).

O trabalho apresentado nesta dissertação resultou em uma publicação, (LEVIN; HEUSER, 2009), em congresso de âmbito nacional e uma publicação, (LEVIN; HEUSER, 2010), em congresso de âmbito internacional.

## 6.1 Trabalhos Futuros

Como trabalhos futuros relacionados a esta dissertação, as seguintes propostas podem ser citadas:

- Este trabalho utiliza redes sociais para o problema de desambiguação de nomes de autores em bibliotecas digitais. A generalização desta abordagem

para o problema de desambiguação de informações em geral é uma idéia que poderia ser explorada.

- Nos experimentos realizados, trabalhou-se com um conjunto reduzido de dados. A questão de escalabilidade poderia ser trabalhada no futuro.
- O método de geração de funções de casamento apresentado nesta dissertação poderia ser utilizado para medir o impacto de outras evidências em desambiguação de nomes, como origem do nome do autor (e.g., indiano, chinês) ou assunto da publicação.

# REFERÊNCIAS

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Modern Information Retrieval**. Boston, MA, EUA: Addison-Wesley, 1999.

BHATTACHARYA, I.; GETOOR, L. Collective entity resolution in relational data. In: **ACM Transactions on Knowledge Discovery from Data**, Vol. 1, No. 1. [S.l.]: 2007, p. 1-36.

BORGMAN, C. L: What are Digital Libraries? Competing Visions. In: **Information Processing and Management: an International Journal**, Vol. 35, No. 3. [S.l.]: 1999, p. 227-243.

CARVALHO, M. G.; LAENDER, A. H. F; GONÇALVES, M. A.; SILVA, A. S. Replica Identification Using Genetic Programming. In: **ACM SAC 2008**. Fortaleza, Brasil: 2008, p. 1801-1806.

CHAUDHURI, S.; GANJAM, K.; GANTI, V.; MOTWANI, R. Robust and Efficient Fuzzy Match for Online Data Cleaning. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, 2003, San Diego, CA, EUA. **Proceedings…** New York, NY, EUA: ACM, 2003, p. 313, 324.

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

COTA, R.; GONÇALVES, M.A.; LAENDER, A.H.F. A Heuristic-based Hierarchical Clustering Method for Author Name Disambiguation in Digital Libraries. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, 12., 2007, João Pessoa, Brasil. **Proceedings...** [S.l.:s.n], 2007, p. 20-34.

DORNELES, C. F.; NUNES, M. F.; HEUSER, C. A.; ORENGO, V. M.; SILVA, A. S., MOURA, E. S. A strategy for allowing meaningful and comparable scores in approximate matching. In: ACM CONFERENCE ON INFORMATION KNOWLEDGE AND MANAGEMENT, 16., 2007, Lisboa, Portugal. **Proceedings…** New York, NY, EUA: 2007, p. 673-689.

ELGARAMID, A.K.; IPEIROTIS, P.G.; VERYKIOS, V.S. Duplicate Record Detection: A Survey. In: **IEEE Transactions on Knowledge and Data Engineering**, Vol. 19, No. 1. Washington, DC, EUA: IEEE Computer Society, 2007, p. 1-16.

FELLEGI, L. P.; SUNTER, A. B. A theory for record linkage. In: **Journal of the American Statistical Association**, Vol. 64. [S.l.:s.n.], 1969, p.1183-1210.

HERNANDEZ, M; STOLFO, S. The merge/purge problem for large databases. **ACM SIGMOD Record**, Vol. 24, No. 2. New York, NY, EUA: ACM, 1995, p.127-138.

KALASHNIKOV, D.; MEHROTRA, S. Domain-Independent Data Cleaning via Analysis of Entity-Relationship Graph. In: **ACM Transactions on Database Systems**, Vol. 31, No. 2. New York, NY, EUA: ACM, 2006, p. 716-767.

KANG, I.-S., NA, S.-H., LEE, S., JUNG, H., KIM, P., SUNG, W.-K., LEE, J.-H. On co-authorship for author disambiguation. In: **Information Proc. and Management**, Vol. 45, No. 1. [S.l.:s.n.], 2009, p. 84-97.

KOZA, J.R. **Genetic Programming: On the programming of computers by means of natural selection**. Cambridge, MA, EUA: MIT Press, 1992.

LAPIDOT, I. Self-Organizing-Maps with BIC for Speaker Clustering. **IDIAP Research Report 02-60, IDIAP Research Institute**. Martigny, Suiça: [s.n.], 2002.

LEE, D.; ON, B.-W.; KANG, J.: Effective and scalable solution for mixed and split citation problems. In: INTERNATIONAL WORKSHOP IN INFORMATION QUALITY IN INFORMATION SYSTEMS, 2., 2005, Baltimore, MA, EUA. **Proceedings…** New York, NY, EUA: 2005 p. 69-76.

LEVENSHTEIN, V. I. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. In: **Soviet Physics Doklady**, Vol. 10, No. 8. URSS: 1966, 707-710.

LEVIN, F. H.; HEUSER, C. A. Evaluating the Use of Social Networks in Author Name Disambiguation in Digital Libraries. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, 14., 2009, Fortaleza, Brasil. **Proceedings…** [S.l.:s.n], 2009, p. 46-60.

LEVIN, F. H.; HEUSER, C. A. Using Genetic Programming to Evaluate the Impact of Social Network Analysis in Author Name Disambiguation. In: ALBERTO MENDELZON INTERNATIONAL WORKSHOP ON FOUNDATIONS OF DATA MANAGEMENT, 4., 2010, Buenos Aires, Argentina. **Proceedings...** [S.l.:s.n], 2010.

MANNING, C.; RAGHAVAN, P.; SCHÜTZE, H. **An Introduction to Information Retrieval**, Chapter 8. New York, NY, EUA: Cambridge University Press, 2008.

MALIN, B. Unsupervised name disambiguation via social network similarity. In: WORKSHOP ON LINK ANALYSIS, COUNTERTERRORISM, AND SECURITY, 2005, Newport Beach, CA, EUA. **Proceedings...** [S.l.:s.n], 2005, p. 93-102.

MCCALLUM, A.K. Efficient clustering of high-dimensional data sets with application to reference matching. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 6., 2000, Boston, MA, EUA. **Proceedings…** New York, NY, EUA: ACM, 2000, p. 169–178.

MENEZES, G. V.; ZIVIANI, N.; LAENDER, A. H. F.; ALMEIDA, V. A Geographical Analysis of Knowledge Production in Computer Science. In: INTERNATIONAL CONFERENCE ON WORLD WIDE WEB, 18., 2009, Madri, Espanha. **Proceedings...** New York, NY, EUA: ACM, 2009, p. 1041-1050.

MILGRAM, S. The small world problem. In: **Psychology Today**, Vol. 1, No. 1. [S.l.:s.n.], 1967, p. 60-67.

NAVARRO, G.; BAEZA-YATES, R.; SUTIEN, E.; TARHIO, J. Indexing methods for approximate string matching. In: **IEEE Data Engineering Bulletin**, Vol. 24, No. 4. [S.l.:s.n.], 2001, p. 19-27.

NEWCOMBE, H.; KENNEDY, J.; AXFORD, S.; JAMES, A. Automatic Linkage of Vital Records. In: **Science**, Vol. 130, No. 3381. [S.l.:s.n.], 1959, p. 954-959.

NEWMAN, M.E. The structure of scientific collaboration networks. In: **Proceedings of the National Academy of Sciences of the United States of America,** Vol. 98, No. 2. [S.l.]: 2001, p.404-409.

NEWMAN, M.E. The structure and function of complex networks. In: **SIAM Review**, Vol. 45, No. 2. [S.l.:s.n.], 2003, p. 167-256.

NIN, J.; MUNTÉS-MULERO, V.; MARTINEZ-BAZAN, N.; LARRIBA-PEY, J. On the Use of Semantic Blocking Techniques for Data Cleansing and Integration. In: INTERNATIONAL DATABASE ENGINEERING AND APPLICATIONS SYMPOSIUM, 11., 2007, Banff, Canadá. **Proceedings…** Washington, DC, EUA: IEEE Computer Society, 2007, p.190-198.

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 BANCO DE DADOS, 10., 2005, Uberlândia, Brasil. **Proceedings…** [S.l.:s.n], 2005, p.205-219.

ON, B.-W.; ELMACIOGLU, E.; LEE, D.; KANG, J.; PEI, J. An effective approach to entity resolution problem using quasi-clique and its application to digital libraries. In: ACM/IEE JOINT CONFERENCE ON DIGITAL LIBRERIES, 6., 2006, Chapel Hill, NC, EUA. **Proceedings...** New York, NY, EUA: ACM, 2006, p. 51–52.

SALTON, G.; MCGILL, M. **Introduction to Modern Information Retrieval**. New York, NY, EUA: McGraw-Hill, 1983.

SARAWAGI, S.; BHAMIDIPATY, A. Interactive deduplication using active learning. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 8., 2002, Edmonton, Canadá. **Proceedings…** New York, NY: ACM, 2002, p.267-278.

WEIS, M.; NAUMANN, F. Duplicate detection in XML. In: INTERNATIONAL WORKSHOP IN INFORMATION QUALITY IN INFORMATION SYSTEMS, 1., 2004, Paris, França. **Proceedings...** New York, NY: ACM, 2004, p. 10-19.

WEIS, M.; NAUMANN, F. Relationship Based Duplicate Detection. **Technical Report Nr. HU-IB-206.** [S.l.:s.n.], 2006.

WEIS, M.; NAUMANN, F. Detecting duplicates in complex XML data. In: INTERNATIONAL CONFERENCE ON DATA ENGENEERING, 22., 2006, Atlanta, GA, EUA. **Proceedings…** Washington, DC, EUA: IEEE Computer Society, 2006, p. 109.

WILCOXON, F. Individual comparisons by ranking methods. **Biometrics Bulletin**, Vol. 1, No. 6. [S.l.:s.n.], 1945, p. 80-83.