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

EDIMAR MANICA

ORION: uma abordagem eficaz e robusta para aquisição de valores de atributos de entidades do mundo real

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

Orientadora: Profa. Dra. Renata Galante

Porto Alegre
2017

CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Manica, Edimar

ORION: uma abordagem eficaz e robusta para aquisição de valores de atributos de entidades do mundo real / Edimar Manica. – Porto Alegre: PPGC da UFRGS, 2017.

127 f.: il.

Tese (doutorado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2017. Orientadora: Renata Galante.

1. Descoberta de páginas-entidade. 2. Extração de dados independente de domínio. 3. Banco de dados orientado a grafos. 4. Cypher. 5. Funções de similaridade. I. Galante, Renata. II. Título.

“Nós não podemos mudar nossas memórias,
mas nós podemos mudar o sentido e o poder que elas têm sobre nós.”
— David Seamands

AGRADECIMENTOS

Meus sinceros agradecimentos a todos que me incentivaram e contribuíram de alguma forma durante esta longa caminhada. Em especial, gostaria de agradecer:

À minha mãe, Neiva Perin Manica, que sempre foi minha parceira nos estudos, me incentivando das mais diversas formas, com carinho e dedicação. Mesmo nos momentos mais difíceis sempre manteve a fé e a esperança. Muito obrigado mãe!

Ao meu pai, Leonir Manica, pelo incentivo aos estudos muito importante para minha formação e por colocar a nossa família sempre em primeiro lugar. Muito obrigado pai!

Ao meu irmão, Edinei Manica, pelo apoio nos momentos mais difíceis.

À minha namorada, Diane Botega, pelo apoio, carinho, amizade e companheirismo.

À minha orientadora, Renata Galante, por me apoiar, incentivar, orientar e guiar durante todos esses anos com muita sabedoria e sensibilidade. Não tenho como traduzir em palavras a admiração, o carinho e o respeito que sinto por você. Obrigado de coração!

À minha co-orientadora, Carina F. Dorneles, pela parceria e incentivo desde a graduação.

Ao meu orientador de Trabalho de Conclusão de Graduação, Cristiano R. Cervi, por me incentivar a seguir a carreira acadêmica.

Aos professores membros da banca: Profa. Dra. Viviane Pereira Moreira, Prof. Dr. Altigran Soares da Silva e Prof. Dr. Ricardo da Silva Torres, por terem aceitado o convite e pelas sugestões que contribuíram no aprimoramento do texto final.

À professora que me alfabetizou, Maria Ieda Pogere, pelo ensino personalizado que me propiciou. Em seu nome, agradeço a todos os meus professores pelo incentivo e dedicação.

Aos meus colegas de laboratório no Instituto de Informática da UFRGS e no IFRS – *Campus* Ibirubá, pelas amizades criadas que certamente serão carregadas para a vida toda. Especialmente, gostaria de agradecer aos meus colegas Ana Dionéia Wouters, Danny Suarez Vargas, Dionéia Magda Everling, Lisiane Cézar de Oliveira, Luis Cláudio Gubert e Solange Pertile pela palavra amiga nos momentos de incerteza.

Ao Instituto de Informática da UFRGS, pela infraestrutura disponibilizada e ao IFRS – *Campus* Ibirubá pela concessão do meu afastamento.

Por fim, agradeço imensamente a Deus por ter me guiado e conduzido durante mais esta importante etapa de minha vida.

RESUMO

Página-entidade é uma página Web que publica dados que descrevem uma entidade de um tipo particular. Adquirir os valores dos atributos de entidades do mundo real publicados nessas páginas é uma tarefa estratégia para diversas empresas. Essa aquisição envolve as tarefas de encontrar as páginas-entidade nos sites e extrair os valores dos atributos publicados nessas páginas. Os trabalhos que discorrem sobre como realizar as tarefas de descoberta das páginas-entidade e de extração dos dados de forma integrada possuem aplicação limitada porque são específicos para um domínio de aplicação ou porque requerem anotações *a priori*. Tendo em vista essa lacuna, esta Tese apresenta **Orion**, uma abordagem para aquisição de valores de atributos de entidades do mundo real a partir de páginas-entidade baseadas em *template*. **Orion** descobre as páginas-entidade nos sites e extrai os valores dos atributos publicados nessas páginas. A principal originalidade da abordagem **Orion** é realizar as tarefas de descoberta das páginas-entidade e de extração dos dados de forma integrada, independentemente de domínio de aplicação e de anotação *a priori*. A abordagem **Orion** inclui uma etapa de descoberta de páginas-entidade que combina características de HTML e URL sem a necessidade de intervenção do usuário para definição dos limiares de similaridade entre as páginas. A etapa de descoberta utiliza uma nova função de similaridade entre páginas baseada na URL que atribui diferentes pesos para os termos de URL de acordo com a capacidade de distinção de páginas-entidade das demais páginas. A abordagem **Orion** também inclui uma etapa de extração de valores de atributos a partir de consultas Cypher em um banco de dados orientado a grafos. Essa etapa infere as consultas automaticamente. A abordagem **Orion** é robusta porque inclui uma etapa adicional de reforço que realiza o tratamento de atributos com variação de *template*. Esse reforço é realizado por meio de uma combinação linear de diferentes funções de similaridade. A fim de avaliar a eficácia de cada etapa da abordagem isoladamente e da abordagem de forma integral, foram realizados experimentos exaustivos utilizando sites reais. Nesses experimentos, a abordagem **Orion** foi numérica e estatisticamente mais eficaz que os *baselines*.

Palavras-chave: Descoberta de páginas-entidade. Extração de dados independente de domínio. Banco de dados orientado a grafos. Cypher. Funções de similaridade.

ORION: an effective and robust approach for acquiring attribute values of real-world entities

ABSTRACT

Entity-page is a Web page which publishes data that describe an entity of a specific type. Acquiring the attribute values of the real-world entities that are published in these pages is a strategic task for various companies. This acquisition involves the tasks of discovering the entity-pages in the websites and extracting the attribute values that are published in them. However, the current approaches that carry out the tasks of discovering entity-pages and extracting data in an integrated way have limited applications because they are restricted to a particular application domain or require an *a priori* annotation. This thesis presents **Orion**, which is an approach to acquire the attribute values of real-world entities from template-based entity-pages. **Orion** discovers the entity-pages in the websites and extracts the attribute values that are published in them. What is original about the **Orion** approach is that it carries out the tasks of discovering entity-pages and extracting data in a way that is integrated, domain-independent, and independent of any *a priori* annotation. The **Orion** approach includes an entity-page discovery stage that combines the HTML and URL features without requiring the user to define the similarity threshold between the pages. The discovery stage employs a new URL-based similarity function that assigns different weights to the URL terms in accordance with their capacity to distinguish entity-pages from other pages. **Orion** also includes a stage during which the attribute values are extracted by means of Cypher queries in a graph database. This stage automatically induces the queries. It should be noted that the **Orion** approach is robust because it includes an additional reinforcement stage for handling attributes with template variations. This stage involves exploring a linear combination of different similarity functions. We carried out exhaustive experiments through real-world websites with the aim of evaluating the effectiveness of each stage of the approach both in isolation and in an integrated manner. It was found that the **Orion** approach was numerically and statistically more effective than the baselines.

Keywords: Entity-page discovery. Domain-independent data extraction. Graph databases. Cypher. Similarity functions.

LISTA DE ABREVIATURAS E SIGLAS

API *Application Programming Interface*

BNF *Backus Normal Form*

CALA *ClAssifying Links Automatically based on their URL*

CSS *Cascading Style Sheets*

DEPTA *Data Extraction based on Partial Tree Alignment*

DIADEM *Domain-centric, intelligent, automated data extraction methodology*

DOM *Document Object Model*

EIHDK *Extraction and Integration of web sources with Humans and Domain Knowledge*

EOC *Entity-Oriented Crawler*

FN Falsos negativos

FP Falsos positivos

GPP *Growing Parallel Paths*

HCUD *Homepage Classification with Unlabeled Data*

HMM *Hidden Markov Model*

HTML *HyperText Markup Language*

ICQ *Identification of Complementary Queries*

IFRS Instituto Federal do Rio Grande do Sul

ISBN *International Standard Book Number*

MDL *Minimum Description Length*

NCQ *Next Complementary Query*

NP Número de páginas-entidade

ONDUX *On Demand Unsupervised Information Extraction*

PEBL *Positive Example Based Learning*

RTDM *Restricted Top-Down Mapping*

SDC *Structure-Driven Crawler*

SQL *Structured Query Language*

SSM *Stacked Skews Model*

SVM *Support Vector Machine*

SWDE *Structured Web Data Extraction*

URL *Uniform Resource Locator*

VP Verdadeiros positivos

XML *eXtensible Markup Language*

XPath *XML Path Language*

LISTA DE FIGURAS

Figura 1.1 Exemplo de página-entidade baseada em *template*. 14

Figura 3.1 Fluxo de execução da abordagem **Orion**..... 44
Figura 3.2 Três exemplos de sites. 45
Figura 3.3 Detalhes de site sobre um campeonato de futebol. 48
Figura 3.4 Fluxo de execução da etapa de *Descoberta*. 49
Figura 3.5 Exemplo de *tokens*, *sub-tokens* e termos de URL. 51
Figura 3.6 Uma página-entidade do site FCBarcelona.com. 59
Figura 3.7 HTML da página-entidade apresentada na Figura 3.6..... 59
Figura 3.8 S-graph do site FCBarcelona.com. 59
Figura 3.9 Um exemplo de consulta Cypher. 62
Figura 3.10 Fluxo de execução da etapa de *Extração*. 63
Figura 3.11 Exemplos de instruções Cypher para criar um S-graph..... 63
Figura 3.12 Instruções Cypher que definem a propriedade *function*. 64
Figura 3.13 Exemplo de S-graph. 70
Figura 3.14 Consultas que extraem o atributo *time* no S-graph apresentado na Figura 3.13... 70
Figura 3.15 Fluxo de execução da etapa de *Reforço*. 72

Figura 4.1 F1 da abordagem **Orion** para cada valor de H 84
Figura 4.2 F1 da abordagem INDESIT para cada configuração de parâmetros..... 85
Figura 4.3 F1 da abordagem GPP para cada valor de K 86
Figura 4.4 Comparação entre *SSM*, *SWDE* e **Orion**. 96
Figura 4.5 Configuração dos parâmetros da etapa de *Reforço*. 100

LISTA DE TABELAS

Tabela 1.1 Valores dos atributos extraídos da página apresentada na Figura 1.1. 15

Tabela 2.1 Comparação entre os trabalhos apresentados neste capítulo. 37

Tabela 3.1 Exemplo de execução do Algoritmo filter. 55
Tabela 3.2 Valores dos atributos publicados na página apresentada na Figura 3.6. 60
Tabela 3.3 Representação das consultas inferidas a partir do S-graph apresentado na Figura 3.13. 71

Tabela 4.1 Composição das bases de dados. 79
Tabela 4.2 Detalhes das bases de dados criadas. 80
Tabela 4.3 Revocação e precisão das abordagens *INDESIT* e **Orion**. 87
Tabela 4.4 Tempo de processamento e número de páginas baixadas por *INDESIT* e **Orion**. 88
Tabela 4.5 Revocação e precisão das abordagens *GPP* e **Orion**. 89
Tabela 4.6 Tempo de processamento e número de páginas baixadas por *GPP* e **Orion**. 90
Tabela 4.7 Características dos *baselines* da etapa de *Extração* 92
Tabela 4.8 Comparação entre **Orion** e *Trinity*. 94
Tabela 4.9 Ocorrência de atributos com formato variante. 99
Tabela 4.10 Eficácia da abordagem **Orion** com e sem a etapa de *Reforço*. 101
Tabela 4.11 Comparação entre *Trinity* e **Orion**. 102
Tabela 4.12 Comparação entre as estratégias de identificação de consultas complementares. 103
Tabela 4.13 Parâmetros adotados. 104
Tabela 4.14 F1 da abordagem **Orion** por atributo. 105
Tabela 4.15 F1 da abordagem **Orion** por site. 106
Tabela 4.16 Comparação entre *INDESIT +Trinity* e **Orion** em termos de eficácia. 106
Tabela 4.17 Resumo dos experimentos. 108

LISTA DE GRAMÁTICAS

3.1 BNF das consultas Cypher. 61

SUMÁRIO

1 INTRODUÇÃO 14
2 TRABALHOS RELACIONADOS 19
2.1 Trabalhos que descobrem páginas-entidade 19
2.2 Trabalhos que classificam ou agrupam as páginas Web 23
2.3 Trabalhos que extraem os valores dos atributos publicados na Web 26
2.3.1 Páginas-entidade baseadas em *template* 26
2.3.2 Outras categorias de páginas 31
2.4 Trabalhos que combinam descoberta e extração 33
2.5 Análise dos trabalhos relacionados 36
2.6 Diferencial da abordagem proposta 41
3 ABORDAGEM ORION 44
3.1 Visão geral 44
3.2 Descoberta 46
3.2.1 Definições preliminares 47
3.2.2 Visão geral 49
3.2.3 Funções de similaridade entre páginas 51
3.2.3.1 Função $ssim_{URL}$ 51
3.2.3.2 Função sim_{HTML} 53
3.2.4 Identificar o caminho de exemplo 53
3.2.5 Percorrer a árvore-entidade 54
3.3 Extração 56
3.3.1 Definições preliminares 58
3.3.2 Visão geral 62
3.3.3 Construção do S-Graph 63
3.3.4 Reconhecimento do template 64
3.3.5 Inferência das consultas 65
3.3.6 Seleção das consultas 67
3.3.7 Execução das consultas 68
3.4 Reforço 69
3.4.1 Definições preliminares 69
3.4.2 Visão geral 72
3.4.3 Escore de similaridade 73
3.4.4 Algoritmos 75
4 AVALIAÇÃO EXPERIMENTAL 77
4.1 Métricas de avaliação 77
4.2 Bases de dados 78
4.3 Avaliação da etapa de Descoberta 81
4.3.1 Configuração dos experimentos 81
4.3.1.1 Baselines 81
4.3.1.2 Metodologia 82
4.3.2 Descrição dos experimentos de calibragem 83
4.3.2.1 Melhor configuração da abordagem **Orion** 83
4.3.2.2 Melhor configuração da abordagem INDESIT 84
4.3.2.3 Melhor configuração da abordagem GPP 85
4.3.3 Descrição dos experimentos de avaliação 86
4.3.3.1 **Orion** *versus* INDESIT 87
4.3.3.2 **Orion** *versus* GPP 89

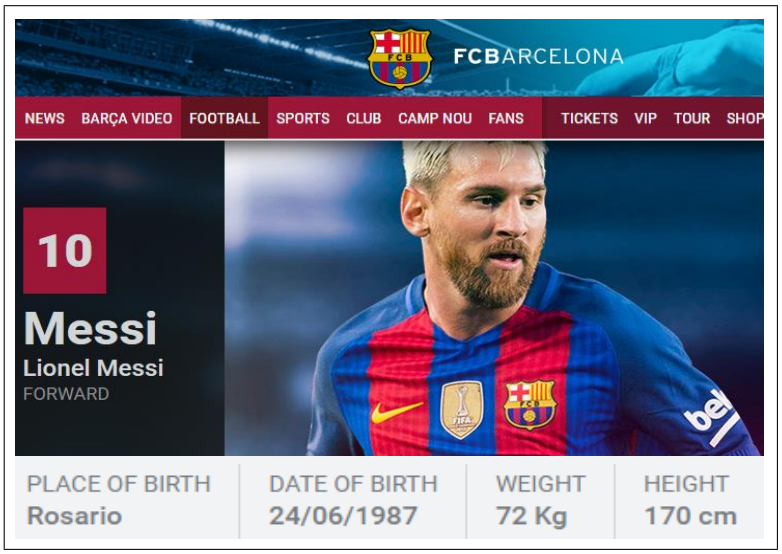
4.4 Avaliação da etapa de Extração **91**
4.4.1 Configuração dos experimentos 91
4.4.1.1 Baselines 91
4.4.1.2 Metodologia 92
4.4.2 Descrição dos experimentos 93
4.4.2.1 **Orion** *versus* Trinity 94
4.4.2.2 **Orion** *versus* SSM e SWDE 95
4.5 Avaliação da etapa de Reforço **97**
4.5.1 Configuração dos experimentos 97
4.5.1.1 Baselines 97
4.5.1.2 Metodologia 97
4.5.2 Descrição dos experimentos 98
4.5.2.1 Ocorrência de atributos com formato variante 99
4.5.2.2 Melhor configuração da etapa de Reforço 100
4.5.2.3 Impacto da etapa de Reforço 101
4.5.2.4 **Orion** *versus* Trinity 101
4.5.2.5 Eficácia da identificação automática de consultas complementares 102
4.6 Avaliação da abordagem Orion integral **103**
4.6.1 Configuração dos experimentos 103
4.6.1.1 Baselines 103
4.6.1.2 Metodologia 104
4.6.2 Descrição dos experimentos 105
4.6.2.1 Eficácia 105
4.6.2.2 **Orion** *versus* INDESIT+Trinity 106
4.7 Análise geral dos experimentos **107**
4.8 Análise dos casos de falha **109**
4.8.1 Descoberta das páginas-entidade 109
4.8.2 Extração dos valores dos atributos 111
5 CONCLUSÃO **113**
REFERÊNCIAS **119**
APÊNDICE A — **124**
A.1 Funções f_l e f_x **124**
A.2 Função f_a **124**
A.3 Função f_e **126**

1 INTRODUÇÃO

A Web é um vasto repositório de dados sobre entidades do mundo real, tais como pessoas, produtos, livros e organizações. Uma página Web que publica dados que descrevem uma entidade de um determinado tipo é denominada página-entidade (BLANCO et al., 2008). As páginas-entidade são uma das formas mais comuns de dados disponíveis na Web (CRESTAN; PANTEL, 2011). Uma fração significante das páginas-entidade é gerada dinamicamente a partir da população de *templates* HTML (*HyperText Markup Language* - Linguagem de Marcação de Texto) fixos com conteúdo de um banco de dados (GULHANE et al., 2010).

De acordo com um estudo de Gibson, Puneira e Tomkins (2005), entre 40 e 50% do total de bytes da Web são conteúdos de *templates* e não há sinais de diminuição desse percentual. Como exemplo, apresenta-se a Figura 1.1, que mostra um fragmento de uma página-entidade baseada em *template* coletada do site oficial do Barcelona[1]. Quanto ao seu conteúdo, essa página-entidade publica atributos (*nome*, *posição*, *número*, *local de nascimento*, *data de nascimento*, *peso* e *altura*) de uma entidade do tipo *jogador de futebol*.

Figura 1.1 – Exemplo de página-entidade baseada em *template*.



As páginas HTML, incluindo as páginas-entidade baseadas em *template*, são árvores DOM (*Document Object Model* - Modelo de Objetos do Documento). Essas páginas podem ser representadas diretamente em um banco de dados orientado a grafos. Dentre as vantagens dessa representação, destacam-se o suporte a algoritmos de grafos eficientes e a disponibilização de linguagens de consulta declarativas (CATTELL, 2011; ANGLES; GUTIERREZ, 2008).

[1]Disponível em: <https://www.fcbarcelona.com/football/first-team/staff/players/2016-2017/messi/>. Último acesso em: 20/10/2017.

Tabela 1.1 – Valores dos atributos extraídos da página apresentada na Figura 1.1.

Atributo	Valor
Nome	Lionel Messi
Posição	FORWARD
Número	10
Local de nascimento	Rosario
Data de nascimento	24/06/1987
Peso	72 Kg
Altura	170 cm

Não há uma linguagem de consulta padrão para os bancos de dados orientados a grafos. No entanto, Holzschuher e Peinl (2013) argumentam que a linguagem Cypher (Neo Technology, 2016) é uma candidata promissora para se tornar o padrão por três principais motivos. Primeiro, Cypher é uma linguagem declarativa para grafos, que permite consultas e atualizações expressivas e eficientes. Segundo, porque de uma perspectiva de legibilidade e manutenibilidade, Cypher é adequada, haja vista sua sintaxe ser de fácil entendimento para desenvolvedores familiarizados com SQL (*Structured Query Language* - Linguagem de Consulta Estruturada). Terceiro, porque, para dados com estrutura de grafo (como as páginas-entidade), o código em Cypher é adicionalmente mais compacto e fácil de entender do que o código em SQL.

O problema tratado nesta Tese é a aquisição de valores de atributos a partir de páginas-entidade baseadas em *template*. Esse problema envolve duas etapas principais, a saber: a descoberta das páginas-entidade e a extração de seus dados. A descoberta das páginas-entidade é responsável por coletar, nos sites, as páginas que descrevem entidades de um determinado tipo de interesse. Por exemplo, coletar as páginas-entidade sobre jogadores de futebol no site oficial do Barcelona. A extração dos dados é responsável por extrair os valores dos atributos publicados nas páginas-entidade coletadas. A Tabela 1.1, por exemplo, apresenta os valores dos atributos extraídos da página-entidade ilustrada na Figura 1.1.

A aquisição de valores de atributos a partir de páginas-entidade baseadas em *template* é uma tarefa estratégica para diversas empresas. Os motores de busca Google[2] e Bing[3], por exemplo, usam dados sobre entidades para fornecer consultas diretas[4]. O Assistente de Voz Siri[5], por sua vez, utiliza dados sobre entidades para responder a consultas do usuário. Outros

[2]Disponível em: <https://www.google.com/>. Último acesso em: 20/10/2017.

[3]Disponível em: <http://www.bing.com/>. Último acesso em: 20/10/2017.

[4]Consulta direta é um recurso de busca onde conteúdo relevante é fornecido diretamente na página de resultado (CHILTON; TEEVAN, 2011).

[5]Disponível em: <http://www.apple.com/ios/siri/>. Último acesso em: 20/10/2017.

casos incluem o uso de dados sobre entidades para analisar mídias sociais, realizar buscas na Web oculta e recomendar produtos (DESHPANDE et al., 2013).

Os trabalhos que realizam as tarefas de descoberta das páginas-entidade e de extração dos dados de forma integrada possuem aplicação limitada porque são específicos para um domínio de aplicação ou porque requerem anotações *a priori* (FURCHE et al., 2014; QIU et al., 2015). Anotações *a priori* são trabalhosas, demoradas e sujeitas a erros (HERNÁNDEZ et al., 2016). As abordagens que focam exclusivamente na descoberta das páginas-entidade possuem problemas que limitam sua aplicação em cenários reais. *INDESIT* (BLANCO; CRESCENZI; MERIALDO, 2005) e *SDC* (VIDAL et al., 2006), por exemplo, requerem a intervenção humana para a definição de limiares de similaridade entre páginas, o que é custoso uma vez que exige um usuário especialista, que processe algumas páginas para encontrar o limiar adequado. *GPP* (WENINGER; JOHNSTON; HAN, 2013), por sua vez, necessita da renderização das páginas em um navegador Web, fato que aumenta significativamente o tempo de processamento.

A literatura fornece diversos trabalhos com foco exclusivo na extração de valores de atributos de páginas-entidade baseadas em *template*. A maioria desses trabalhos é baseada em expressões XPath (ORTONA et al., 2016; QIU; LUCE, 2014; HAO et al., 2011; GULHANE et al., 2010) e não suporta atributos com formato variante (quando há variação de *template* na publicação dos valores do mesmo atributo em páginas-entidade diferentes). Não foi encontrado na literatura nenhum trabalho especificamente que modele a extração de dados como consultas Cypher em um banco de dados orientado a grafos. Fato esse que representa uma lacuna no que tange ao cenário em questão. Os bancos de dados orientados a grafos possuem algoritmos de grafos eficientes que são necessários nas buscas por caminhos no HTML das páginas, que são realizadas por diversos métodos de extração de dados (ORTONA et al., 2016; QIU; LUCE, 2014; HAO et al., 2011; GULHANE et al., 2010). Nesse contexto, a linguagem Cypher é mais robusta que XPath porque permite percorrer, consultar e atualizar o grafo, enquanto XPath é específica para percorrer um documento XML (*eXtensible Markup Language* - Linguagem de Marcação Estendida) ou equivalente.

Tendo em vista essas considerações, esta Tese apresenta **Orion**[6], uma abordagem para aquisição de valores de atributos de entidades do mundo real a partir de páginas-entidade baseadas em *template*. **Orion** descobre as páginas-entidade nos sites e extrai os valores dos atributos publicados nessas páginas. A principal originalidade da abordagem **Orion** é realizar a descoberta das páginas-entidade e a extração dos seus dados de forma integrada, independentemente de domínio de aplicação e de anotação *a priori*.

[6]Na mitologia grega, Órion foi um grande caçador. Esse nome foi adotado, uma vez que a abordagem proposta é uma grande "caçadora" de dados.

Ao analisar como cada uma dessas tarefas é realizada pela abordagem **Orion**, podem-se elencar outras originalidades. A descoberta das páginas-entidade é feita sem a necessidade de intervenção humana para a definição de limiares de similaridade ou renderização das páginas em um navegador Web. A extração dos dados é inspirada nos bancos de dados orientados a grafos e nas linguagens de consulta de grafos. Essa inspiração permitiu que a extração de dados da abordagem **Orion** possuísse semântica declarativa, ou seja, o usuário não precisa se preocupar com detalhes de implementação (tanto o caminhamento quanto a atualização do grafo é realizada por meio de uma linguagem declarativa). A abordagem **Orion** é robusta porque inclui um tratamento adicional para atributos com formato variante e eficaz porque supera os *baselines* em termos de eficácia em experimentos utilizando sites reais.

A questão de pesquisa desta Tese é: **combinar características de HTML e URL para encontrar as páginas-entidade nos sites e utilizar consultas Cypher em um banco de dados orientado a grafos para extrair os valores dos atributos publicados nessas páginas pode aumentar a eficácia da aquisição de valores de atributos de entidades do mundo real a partir de páginas-entidade baseadas em template?** Com a questão de pesquisa definida, o problema de pesquisa delineado e observados os estudos que compõem as pesquisas na área, a seguir, discorre-se em forma de tópicos sobre as principais contribuições desta Tese. São elas:

- uma etapa eficaz de descoberta de páginas-entidade que combina características de HTML e URL sem a necessidade de intervenção do usuário para definição dos limiares de similaridade entre as páginas. Essa etapa inclui a definição de uma função de similaridade entre páginas baseada na URL, que atribui diferentes pesos para os termos de URL de acordo com a capacidade de distinção de páginas-entidade das demais páginas;
- uma etapa eficaz de extração de valores de atributos a partir de consultas Cypher em um banco de dados orientado a grafos, que infere as consultas automaticamente. Essa etapa inclui - (i) uma forma de representação de páginas-entidade baseadas em *template* em um banco de dados orientado a grafos; e (ii) uma estratégia automática de inferência de consultas Cypher que extraem valores de atributos publicados em páginas-entidade;
- uma alternativa para tratamento de atributos com formato variante por meio de uma combinação linear de diferentes funções de similaridade.

No que diz respeito à organização do trabalho, além desta introdução, a Tese conta com três capítulos e uma conclusão. Ao final, são indicadas as referências, bem como um apêndice. Assim, a seguir, em tópicos, descreve-se brevemente a organização:

- O Capítulo 2 discute os trabalhos relacionados à temática da Tese, bem como posiciona as contribuições desta Tese em si com relação à literatura atual;
- No Capítulo 3, é apresentada a abordagem **Orion** para aquisição de valores de atributos de entidades do mundo real a partir de páginas-entidade baseadas em *template*;
- O Capítulo 4 apresenta um conjunto exaustivo de experimentos que avaliam a eficácia da abordagem **Orion**. Cada etapa da abordagem foi analisada individualmente e depois a abordagem **Orion** foi avaliada de forma integral. Foram utilizados sites reais, que descrevem entidades de diferentes tipos a fim de demonstrar que a abordagem é independente de domínio de aplicação. Os resultados obtidos são comparados com *baselines* reais;
- Após, é apresentada a conclusão. As principais contribuições desta Tese e os resultados obtidos no decorrer do doutorado são sintetizados neste item, bem como as publicações resultantes do doutorado são apresentadas. Também são discutidos alguns pontos interessantes que podem ser explorados em trabalhos futuros;
- Por fim, o Apêndice A detalha as funções de similaridade utilizadas nos experimentos.

2 TRABALHOS RELACIONADOS

Este capítulo apresenta os trabalhos relacionados ao contexto no qual esta Tese está inserida. A Seção 2.1 apresenta quatro trabalhos que visam exclusivamente descobrir as páginas-entidade nos sites. A Seção 2.2 introduz um conjunto de trabalhos que realiza a classificação ou agrupamento de páginas Web. Esses trabalhos são incluídos porque podem ser adaptados para a descoberta de páginas-entidade. A Seção 2.3 apresenta uma seleção de trabalhos com o objetivo de extrair valores de atributos publicados na Web. Essa seleção cobre diversos aspectos relacionados à extração: diferentes técnicas (trabalhos baseados em XPath, expressões regulares, alinhamento parcial de árvore e modelos estatísticos), diferentes graus de supervisão (supervisão antes da execução do trabalho, após a execução do trabalho e sem supervisão) e diferentes recursos (bases de conhecimento e contribuição colaborativa). A Seção 2.4 destaca dois trabalhos que realizam tanto a descoberta de páginas-entidade quanto a extração dos valores dos atributos publicados nessas páginas. A Seção 2.5 sintetiza e compara as principais características dos trabalhos analisados. Por fim, a Seção 2.6 diferencia a abordagem proposta, **Orion**, dos trabalhos diretamente relacionados.

2.1 Trabalhos que descobrem páginas-entidade

Os trabalhos apresentados nesta seção têm como objetivo exclusivo a descoberta de páginas-entidade nos sites. Os três primeiros trabalhos destinam-se à Web de superfície (páginas obtidas por meio da navegação por links), enquanto o último é voltado para a Web oculta (páginas obtidas por meio da submissão de formulários HTML).

INDESIT (BLANCO; CRESCENZI; MERIALDO, 2005)

Blanco, Crescenzi e Merialdo (2005) apresentam um algoritmo, denominado *INDESIT*, para coletar automaticamente as páginas-entidade dos sites. Esse algoritmo parte da premissa de que as páginas-entidade presentes no mesmo site possuem uma estrutura similar. A principal contribuição do *INDESIT* é caracterizar a estrutura das páginas por meio do conjunto de propriedades de leiaute e apresentação associadas com os links contidos nas páginas.

Dada uma página-entidade (página de exemplo) como entrada, *INDESIT* retorna o conjunto das páginas-entidade do mesmo tipo presentes no site. Cada página é representada pelo seu conjunto de *link-paths*, denominado `esquema`. Um *link-path* é um caminho, através da árvore DOM, que inicia na raiz e termina em um nodo âncora. A similaridade entre duas pá-

ginas é calculada pelo coeficiente de Jaccard (TAN; STEINBACH; KUMAR, 2005) entre seus esquemas. *INDESIT* tem quatro etapas principais. Na primeira etapa, as páginas apontadas pela página de exemplo são agrupadas. Cada grupo contém apenas as páginas com o mesmo esquema. A fim de minimizar o número de páginas baixadas, quando mais de n links com mesmo *link-path* na mesma página apontam para páginas do mesmo grupo, os links restantes do *link-path* não são seguidos. Na segunda etapa, os grupos são avaliados e ranqueados. O grupo com o maior escore concentra, em um pequeno número de páginas, o maior número de links distintos que apontam para páginas similares à página de exemplo. Na terceira etapa, as etapas 1 e 2 são reexecutadas com o grupo com o maior escore como entrada em vez da página de exemplo inicial, retornando as páginas similares ao grupo com maior escore. Os links nessas páginas são analisados a fim de encontrar páginas similares à página de exemplo. Na última etapa, outros grupos são eventualmente considerados se eles entregam novas páginas similares à página de exemplo. Uma página é considerada nova na primeira vez que ela é encontrada.

SDC (VIDAL et al., 2006)

Vidal *et al.* (2006) propõem uma abordagem, denominada nesta Tese *SDC* (***S**tructure-**D**riven **C**rawler* - Coletor Dirigido pela Estrutura), para encontrar automaticamente as páginas-entidade nos sites. Essa abordagem parte de duas premissas: (i) as páginas-entidade presentes no mesmo site são estruturalmente similares; e (ii) existe um padrão de navegação em cada site que permite encontrar as páginas-entidade. A principal contribuição do *SDC* é caracterizar o padrão de navegação exclusivamente por meio de características de URL.

Dada uma página-entidade (página de exemplo) e um ponto de entrada para o site, *SDC* retorna as páginas-entidade do mesmo tipo presentes no site. Um site é considerado como um grafo dirigido $S = \langle P, L \rangle$, em que P é o conjunto de páginas do site S e L é um conjunto de pares $\langle x, y \rangle$ tal que existe um link da página x para a página y em S. *SDC* possui duas fases: *mapeamento do site* e *geração do padrão de navegação*. A fase de *mapeamento do site* realiza uma busca em largura no grafo do site e armazena os caminhos que iniciam no ponto de entrada e terminam em uma página alvo. Uma `página alvo` é uma página estruturalmente similar à página de exemplo. A similaridade estrutural entre duas páginas é calculada utilizando o RTDM (*Restricted Top-Down Mapping*) (REIS et al., 2004). O RTDM é um algoritmo baseado na distância de edição entre árvores. A fase de *geração do padrão de navegação* cria uma lista de expressões regulares, em que cada expressão regular representa os links que ocorrem em uma página que a abordagem deve seguir para alcançar as páginas alvo. As URLs similares nos caminhos armazenados na fase anterior são generalizadas por meio de expressões regulares. O

caminho que entrega o maior número de páginas alvo é selecionado. Uma URL é considerada uma string formada por várias substrings separadas por um caractere "/". Cada substring é denominada `nível`. Duas URLs são consideradas similares se elas satisfazem as seguintes condições: (i) elas têm o mesmo número de níveis; (ii) elas possuem a mesma substring no primeiro nível; e (iii) elas têm no máximo K níveis na mesma posição que não são iguais.

GPP (WENINGER; JOHNSTON; HAN, 2013)

Weninger, Johnston e Han (2013) apresentam um *framework*, denominado nesta Tese *GPP* (***G**rowing **P**arallel **P**aths* - Estendendo Caminhos Paralelos), para descobrir páginas-entidade nos sites. Esse *framework* parte da premissa de que é possível combinar características de HTML e visuais (posições X e Y, largura e altura) para descobrir as páginas-entidade nos sites. A principal contribuição é identificar caminhos paralelos nos sites que permitem encontrar as páginas que descrevem entidades do mesmo tipo.

GPP recebe como entrada a página principal e uma página-entidade do site. A saída é composta pelo conjunto das páginas-entidade do mesmo tipo presentes no site. Um site é representado como um grafo dirigido com raiz, no qual os nodos são as páginas, as arestas representam os links entre as páginas e a raiz é a página principal do site. As páginas são representadas como árvores DOM. *GPP* tem quatro etapas principais. Na primeira etapa, o caminho de página é encontrado usando uma busca em largura. O `caminho de página` é o menor caminho da página principal até a página-entidade (fornecida como entrada) no grafo do site. Na segunda etapa, os *link-paths* são encontrados. Considerando duas páginas p_x e p_{x+1} pertencentes a um caminho de página, em que p_x aponta para p_{x+1}, o *link-path* de p_{x+1} é o caminho da raiz até o nodo âncora que contém o link para p_{x+1} na árvore DOM de p_x. Na terceira etapa, *GPP* procura por outras páginas-entidade em caminhos paralelos ao caminho de página. A fim de analisar se dois caminhos são paralelos, *GPP* usa um algoritmo de alinhamento parcial de árvores e um algoritmo de extração de listas Web. O algoritmo de alinhamento parcial de árvore (ZHAI; LIU, 2006) e a identificação de listas Web (FUMAROLA et al., 2011) exploram a árvore DOM e as características visuais das páginas-entidade. A identificação das listas Web é realizada por meio do reconhecimento de padrões visuais a fim de extrair listas mais genéricas do que as listas HTML, que contêm marcações específicas (*<UL>*, *<OL>* ou *<DL>*). Na última etapa, novas iterações são realizadas, nas quais outros caminhos são considerados de acordo com o número de iterações fornecido como parâmetro.

EOC (HE et al., 2013)

He *et al.* (2013) apresentam um sistema, denominado nesta de Tese *EOC* (***E**ntity-**O**riented **C**rawler* - Coletor Orientado a Entidades), para coletar páginas-entidade na Web oculta. Esse sistema parte da premissa de que os usuários de motores de busca pesquisam por entidades e em seguida clicam em resultados que remetem a sites que publicam páginas descrevendo essas entidades. A principal contribuição de *EOC* é utilizar *logs* de consulta e bases de conhecimento para preencher formulários de busca a fim de retornarem páginas-entidade.

Dada uma lista de sites como entrada, *EOC* retorna as páginas-entidade desses sites. *EOC* tem quatro módulos principais. O primeiro módulo extrai e analisa os formulários HTML das páginas principais dos sites (de acordo com Madhavan *et al.* (2008)), e produz *templates* de URL. Um *template* de URL é gerado para cada formulário de busca. Na prática, o campo de texto do formulário é preenchido com uma *string* x e os demais campos mantêm o valor padrão. O *template* de URL é a URL obtida a partir da submissão do formulário substituída a *string* x pela marca $\{query\}$. O segundo módulo gera valores para substituir a marca $\{query\}$ no *template* de URL a fim de produzir URLs finais (URLs de páginas-entidade). Os valores são gerados para cada site w de duas formas: (i) *logs* de consulta - palavras-chave de buscas Web com um alto número de clicks para o site w; e (ii) base de conhecimento - entidades presentes no Freebase[1] do mesmo tipo que as entidades descobertas por meio dos *logs* de consulta. Por exemplo, através da análise de consultas é verificado que muitos usuários que pesquisam por "*Coração Valente*" clicam no link para o site do IMDb[2]. *Coração Valente* é escolhido como um valor para substituir a marca $\{query\}$ no site IMDb. Verifica-se que *Coração Valente* está no Freebase como uma entidade do tipo *filmes*. Outras entidades desse tipo presentes no Freebase são utilizadas como valores para substituir a marca $\{query\}$ no site IMDb. O terceiro módulo remove páginas errôneas ou vazias, ou seja, que não possuem entidades. Para cada site, *EOC* força a geração de páginas errôneas por meio da submissão de caracteres arbitrários que não possuem nenhum significado semântico (por exemplo, "zzzzzzzzzz"). *EOC* remove as páginas coletadas que são similares às páginas errôneas geradas. A similaridade é calculada utilizando o coeficiente de Jaccard entre as assinaturas das páginas. A assinatura é essencialmente um conjunto de *tokens* que são descritivos do conteúdo da página. O último módulo descobre novas URLs de páginas-entidade por meio da análise dos links das páginas-entidade encontradas.

[1]Disponível em: <https://www.freebase.com/>. Último acesso em: 10/10/2017.
[2]Disponível em: <http://www.imdb.com/>. Último acesso em: 10/10/2017.

2.2 Trabalhos que classificam ou agrupam as páginas Web

Esta seção apresenta os trabalhos que classificam ou agrupam as páginas Web. Essa seção também descreve como esses trabalhos podem ser adaptados para a descoberta de páginas-entidade e as limitações de tal adaptação.

PEBL (YU; HAN; CHANG, 2004)

Yu, Han e Chang (2004) propõem um *framework*, denominado *PEBL* (***P**ositive **E**xample **B**ased **L**earning* - Aprendizado baseado em Exemplos Positivos), para classificação binária de páginas. Sua principal contribuição é requerer apenas exemplos positivos para o treinamento, enquanto classificadores binários tradicionais necessitam de exemplos positivos e negativos.

PEBL aprende a partir de um conjunto de páginas positivas e uma coleção de amostras aleatórias de páginas, em que as classes dessas amostras não são conhecidas. A fim de construir um classificador de páginas pessoais, por exemplo, o usuário necessita coletar exemplos de páginas pessoais e um conjunto de páginas sem identificação da classe que pertencem, ou seja, sem informar se são páginas pessoais ou não. *PEBL* usa dois classificadores. O primeiro classificador (por exemplo, um classificador baseado em regras) encontra as páginas com maior probabilidade de serem exemplos negativos. Os exemplos positivos (fornecidos pelo usuário) e os exemplos negativos (obtidos por meio da aplicação do primeiro classificador) são utilizados para treinar um segundo classificador. Por exemplo, o algoritmo de classificação de Máquina de Vetores de Suporte (*Support Vector Machine* - SVM). Esse classificador é aplicado para as páginas que permanecem sem classificação. *PEBL* extrai características da URL e do conteúdo das páginas. As características do conteúdo são divididas, de acordo com sua localização, em título, cabeçalhos, links, textos de âncora, textos normais e *metatags*. Cada característica é um predicado que indica se um termo ou um caractere especial aparece em determinada parte da página. Por exemplo, o símbolo "~" aparece na URL ou o termo "*homepage*" aparece no título.

PEBL pode ser adaptado para realizar a descoberta de páginas-entidade. É necessário fornecer como entrada um conjunto de páginas-entidade anotadas como exemplos positivos e as demais páginas do site como amostras aleatórias de páginas. A partir dessa entrada, *PEBL* classifica as páginas do site em páginas-entidade ou não. As limitações dessa adaptação incluem: (i) a necessidade de baixar todas as páginas do site; e (ii) a necessidade de fornecer vários exemplos de páginas-entidade anotados manualmente, pois um exemplo apenas não é suficiente para *PEBL* aprender as características das páginas-entidade.

MDL-UC (BLANCO; DALVI; MACHANAVAJJHALA, 2011)

Blanco, Dalvi e Machanavajjhala (2011) apresentam um *framework*, denominado *MDL-UC*, para agrupar páginas de um determinado site de modo que as páginas geradas pelo mesmo *template* fiquem no mesmo grupo. A principal contribuição desse *framework* é utilizar características de URL para agrupar as páginas com estrutura similar.

Dado um conjunto de páginas de um site, *MDL-UC* agrupa as páginas que possuem o mesmo *template*. *MDL-UC* extrai características da URL, do HTML e do conteúdo das páginas. O HTML e o conteúdo são representados pelos pares (`termo`, `posição`), em que `termo` é um nodo textual e `posição` é uma expressão XPath (sobre a árvore DOM da página) que inicia no nodo raiz e termina no nodo textual. A URL é representada como um conjunto de pares (`termo`, `posição`), em que `termo` é uma sequência de caracteres da URL separadas por um critério de tokenização e `posição` é um número que representa a localização do termo com relação aos demais termos na URL. *MDL-UC* infere como parte de um *template* os termos que estão em uma posição cujo conjunto de valores distintos (considerando todas as páginas do site) é pequeno. No entanto, *templates* e dados podem ocorrer na mesma posição em diferentes páginas. Nesses casos, *MDL-UC* infere como parte de um *template* os termos cuja frequência em uma posição específica é muito maior que os outros termos que também ocorrem naquela posição. A fim de encontrar o conjunto de *templates* que melhor descrevem o site, *MDL-UC* utiliza o princípio do comprimento mínimo de descrição (MDL - ***M**inimum **D**escription **L**ength*) (GRÜNWALD, 2007).

MDL-UC pode ser adaptado para realizar a descoberta de páginas-entidade, pois as páginas-entidade do mesmo tipo presentes no mesmo site são geradas pelo mesmo *template*, logo ficam em um mesmo grupo. É necessário que o usuário anote o grupo que contém as páginas-entidade após a execução do *framework*. As limitações dessa adaptação incluem: (i) a necessidade de baixar todas as páginas do site, pois *MDL-UC* requer um grande conjunto de treinamento que deve incluir tanto as páginas-entidade quanto as demais páginas do site; e (ii) a necessidade de anotação do grupo que contém as páginas-entidade.

HCUD (GOLLAPALLI et al., 2015)

Gollapalli *et al.* (2015) apresentam um *framework*, denominado nesta Tese *HCUD* (***H**omepage **C**lassification with **U**nlabeled **D**ata* - Classificação de páginas pessoais com dados não rotulados), que tem como objetivo classificar páginas pessoais de pesquisadores em sites acadêmicos. A principal contribuição desse *framework* é realizar o treinamento utilizando uma pequena quantidade de páginas anotadas e uma grande quantidade de páginas não anotadas.

HCUD requer dois conjuntos de exemplos de treinamento: (i) um pequeno conjunto anotado; e (ii) um grande conjunto que não está anotado. Dois tipos de características (conteúdo e URL) são extraídos de cada conjunto de treinamento. Um classificador é treinado para cada tipo de característica usando os exemplos anotados. Após, os dois classificadores são utilizados para classificar os exemplos não anotados, e as classificações com maior confiança de cada classificador são adicionadas ao conjunto dos exemplos anotados. Esse processo é repetido iterativamente até que todos os exemplos tenham sido anotados. As características de URL verificam a presença de: (i) termos (unigramas e bigramas) mais frequentes na coleção; (ii) termos que pertencem a WordNet (MILLER, 1995); (iii) termos hifenizados ou sublinhados; (iv) padrões alfanuméricos; (v) termos com mais de 30 caracteres; e (vi) determinados símbolos (por exemplo, ~). As características de conteúdo incluem: (i) os termos mais frequentes e discriminatórios; (ii) o número de tabelas; (iii) o número de links; (iv) o número de imagens; e (v) os termos comumente encontrados em nodos âncoras de páginas pessoais.

HCUD pode ser adaptado para realizar a descoberta de páginas-entidade sobre pesquisadores. É necessário fornecer como entrada: (i) um conjunto de páginas anotadas como páginas-entidade; (ii) um conjunto de páginas anotadas como não sendo páginas-entidade; e (iii) um conjunto com as páginas do site não anotadas. A partir dessa entrada, *HCUD* classifica as páginas do site em páginas-entidade ou não. As limitações dessa adaptação incluem: (i) a necessidade de baixar todas as páginas do site; (ii) a necessidade de anotar um conjunto de páginas; e (iii) a restrição de aplicação ao tipo de entidade *pesquisadores*.

CALA (HERNÁNDEZ et al., 2016)

Hernández *et al.* (HERNÁNDEZ et al., 2016) propõem uma ferramenta, denominada *CALA* (***Cl**Assifying **L**inks **A**utomatically based on their URL* - Classificando links automaticamente com base em suas URLs), para classificação de páginas da Web oculta. A principal contribuição é realizar a classificação com base exclusivamente em características de URL.

A entrada dessa ferramenta é uma página Web que contém um formulário de busca baseado em palavras-chave que pode ser preenchido e submetido para recuperar páginas *hub*. Páginas *hub* são páginas que resultam da submissão de um formulário de busca e que fornecem resumos e links para as páginas-entidade. O formulário é preenchido e submetido com palavras extraídas do próprio site para obter um conjunto de páginas *hub*. Esse conjunto é utilizado para construir uma lista de padrões que representam as URLs do site. Os padrões são construídos analisando a frequência de prefixos compartilhados. Um usuário atribui um rótulo semântico para cada padrão (por exemplo, *autores*, *publicações*). Os padrões anotados são utilizados para

classificar novas páginas do site, encontrando o padrão que corresponde à URL.

CALA pode ser adaptada para realizar a descoberta de páginas-entidade. É necessário fornecer como entrada um site que permita o acesso às suas páginas-entidade por meio da submissão de formulários. A partir dessa entrada, *CALA* preenche e submete o formulário, encontra as páginas *hub* e constrói uma lista de padrões que representam as URLs do site. Um usuário deve anotar o padrão que corresponde às páginas-entidade. As limitações dessa adaptação incluem: (i) a necessidade de uma estratégia mais robusta de submissão de formulários para encontrar todas as páginas-entidade do site, uma vez que *CALA* realiza uma submissão simplista, por meio de termos da página que contém o formulário, apenas para encontrar um conjunto de páginas *hub* do qual se possa extrair os padrões; e (ii) a necessidade de anotação.

2.3 Trabalhos que extraem os valores dos atributos publicados na Web

Os trabalhos apresentados nesta seção visam exclusivamente à extração de valores de atributos publicados na Web. A Subseção 2.3.1 descreve os trabalhos que se destinam à extração a partir de páginas-entidade baseadas em *template*. Esses trabalhos são os mais detalhados, uma vez que são os mais próximos da Tese. A Subseção 2.3.2 apresenta os trabalhos que realizam a extração em outras categorias de páginas Web. Esses trabalhos são apresentados em um nível de detalhe menor, por estarem menos relacionados com a Tese. Sua inclusão se justifica para mostrar as diferenças entre os trabalhos conforme a categoria de página alvo da extração.

2.3.1 Páginas-entidade baseadas em *template*

SSM (CARLSON; SCHAFER, 2008)

Carlson e Schafer (2008) propõem uma abordagem, denominada *SSM* (*Stacked Skews Model*), para extrair valores de atributos publicados em páginas-entidade. A principal contribuição dessa abordagem é realizar a extração de valores de atributos em sites não anotados a partir do conhecimento aprendido em um conjunto de sites (que descrevem entidades do mesmo tipo) que foram previamente anotados.

SSM recebe como entrada um conjunto de sites anotados e um site não anotado. Cada site é formado pelo conjunto de suas páginas-entidade. A saída é composta pelos valores dos atributos publicados nas páginas-entidade do site não anotado. *SSM* aprende características de

cada atributo (por exemplo, *altura*) do tipo de entidade (por exemplo, *jogador*) a partir dos sites anotados. Essas características são utilizadas para extrair os valores de atributos no site não anotado. *SSM* é baseado em três conceitos principais: (i) `campo de dados` - um campo de dados em um site é uma localização dentro do *template* das páginas-entidade do site (um nodo dentro da árvore DOM); (ii) `valor de dados` - um valor de dados é uma instância de um campo de dados em uma página-entidade do site; e (iii) `contexto` - o contexto de um valor de dados é o conteúdo do nodo textual que precede o valor de dados na árvore DOM da página. *SSM* identifica as regiões alinhadas entre as páginas-entidade que possuem potenciais valores de dados por meio de um algoritmo de Alinhamento Parcial de Árvores (ZHAI; LIU, 2006). Essas regiões são consideradas campos de dados candidatos. Cada campo de dados candidato recebe um escore, para cada atributo do tipo de entidade, que indica a probabilidade do campo de dados conter valores do atributo. Para cada atributo, *SSM* seleciona o campo de dados no site não rotulado que possui o maior escore (desde que esse escore satisfaça um determinado limiar). O escore é obtido levando em conta quatro tipos de características: (i) valores de contexto tokenizados utilizando 3-gramas; (ii) valores de dados tokenizados utilizando espaço em branco; (iii) valores de dados tokenizados utilizando 3-gramas; (iv) tipos genéricos presentes nos valores de dados (por exemplo, dígitos, todas as letras maiúsculas). As características de um campo de dados do site não rotulado são comparadas com as características de um determinado atributo (aprendidas nos sites rotulados) utilizando uma função de similaridade que foi adaptada a partir da divergência de Skew (LEE, 1999). *SSM* calcula um escore de similaridade para cada tipo de característica. Os escores de similaridade dos diferentes tipos de características são combinados usando um modelo de regressão linear.

FindAttrPos (GULHANE et al., 2010)

Gulhane *et al.* (2010) apresentam uma abordagem, denominada *FindAttrPos* (*Find Attribute Position* - Encontre a Posição do Atributo), para extrair os valores dos atributos em páginas-entidade baseadas em *template*. Sua principal contribuição é utilizar a redundância de conteúdo em nível de entidade entre um site e uma base de conhecimento para eliminar a necessidade de anotação do usuário. A redundância de conteúdo em nível de entidade ocorre quando duas ou mais fontes publicam entidades em comum. Por exemplo, dois sites que publicam dados sobre um mesmo jogador. Outra contribuição é a definição de uma função de similaridade que considera a existência de *templates* nos valores dos atributos publicados na Web.

FindAttrPos recebe como entrada uma base de conhecimento com exemplos de entidades de um determinado tipo e o conjunto de páginas-entidade de um site. Cada registro da

base de conhecimento fornecida como entrada contém os valores dos atributos de uma entidade. O site deve descrever entidades do mesmo tipo que as entidades presentes na base de conhecimento e ter entidades compartilhadas com a base de conhecimento. A saída é composta pelos valores dos atributos publicados nas páginas-entidade. *FindAttrPos* analisa a árvore DOM das páginas-entidade para gerar regras de extração. As regras de extração são representadas como expressões XPath a partir do nodo raiz até um nodo textual. Por exemplo, a regra "*/HTML[1]/UL[1]/LI[2]/DIV[2]/text()*". Para cada atributo da base de conhecimento, é selecionada a regra que: (i) extrai valores similares aos valores do atributo na base de conhecimento em mais páginas; e (ii) cada página que a regra extrai valores similares também possui valores de outros atributos da mesma entidade (redundância de conteúdo em nível de entidade) extraídos por outras regras. Um algoritmo baseado no Apriori (AGRAWAL; SRIKANT, 1994) é utilizado para podar regras com suporte inadequado de forma eficiente.

SWDE (HAO et al., 2011)

Hao *et al.* (2011) propõem um sistema, denominado nesta Tese *SWDE* (***S**tructured **W**eb **D**ata **E**xtraction* - Extração de Dados Estruturados da Web), que extrai os valores dos atributos publicados nas páginas-entidade baseadas em *template* de um site. A principal contribuição desse sistema é realizar a extração em um site não anotado a partir de características aprendidas em outro site (que descreve entidades do mesmo tipo) que foi previamente anotado. Outro importante diferencial é considerar a dependência visual entre os atributos.

SWDE recebe como entrada dois sites que descrevem entidades do mesmo tipo: (i) um site anotado; e (ii) um site não anotado. Os sites são formados pelo conjunto de suas páginas-entidade. O site anotado deve ter os valores de todos os atributos anotados em todas as páginas-entidade. Não há necessidade de compartilhamento de entidades entre os sites. A saída é composta pelos valores dos atributos publicados nas páginas-entidade do site não anotado. Inicialmente, *SWDE* extrai características de conteúdo, contexto e leiaute do site anotado. Com base nessas características, *SWDE* adquiri conhecimento sobre os atributos do tipo de entidade que é descrito no site. Então, são geradas regras de extração para o site não anotado a partir das suas próprias características. Para cada atributo de interesse do usuário, é selecionada uma regra de extração por meio do conhecimento adquirido a partir do site anotado. São extraídas características para cada nodo textual da árvore DOM das páginas-entidade. As *características de conteúdo* são: conjunto de *tokens*, número de *tokens*, número de caracteres e tipo de caracteres. As *características de contexto* são: prefixo, sufixo e rótulo. As *características de leiaute* são: a expressão XPath a partir do nodo raiz até o nodo textual; a posição visual (X, Y) e o

tamanho visual (largura e altura).

EIHDK (QIU; LUCE, 2014)

Qiu e Luce (2014) propõem um *framework*, denominado *EIHDK* (***E**xtraction and **I**ntegration of Web sources with **H**umans and **D**omain **K**nowledge* - Extração e integração de sites com conhecimento de domínio e humano), que tem como objetivo extrair valores de atributos de páginas-entidade para expandir uma base de conhecimento. A principal contribuição desse *framework* é utilizar a redundância de conteúdo em nível de entidade, uma base de conhecimento e a contribuição colaborativa para evitar a necessidade de um usuário especialista.

EIHDK recebe como entrada uma base de conhecimento e um conjunto de sites que descrevem entidades do mesmo tipo. Os sites são formados pelo conjunto de suas páginas-entidade. A saída é formada pela base de conhecimento expandida com os valores dos atributos extraídos das páginas-entidade dos sites. *EIHDK* gera regras de extração baseadas em XPath. Para cada atributo da base de conhecimento, é selecionada uma regra de extração. Essa seleção explora a redundância de conteúdo em nível de entidade entre os sites e a base de conhecimento fornecida como entrada. *EIHDK* considera que os sites compartilham entidades com uma base de conhecimento (por exemplo, Wikipedia[3], Freebase[4]). Quando não há um número suficiente de entidades compartilhadas entre os sites e a base de conhecimento, utiliza-se a contribuição colaborativa (por exemplo, a plataforma Amazon Mechanical Turk[5]). Para minimizar o número de questões necessárias e, com isso, diminuir o custo com a contribuição colaborativa, são adotadas técnicas de aprendizagem ativa (CRESCENZI; MERIALDO; QIU, 2013). Essas técnicas permitem selecionar as melhores questões de modo que apenas poucas questões sejam necessárias. As pessoas que respondem às questões podem fornecer respostas diferentes ou erradas, uma vez que não são especialistas. Por isso, são utilizadas técnicas para encontrar as melhores respostas considerando o voto da maioria (AHN; DABBISH, 2004). Por fim, a base de conhecimento é expandida a partir dos valores de atributos extraídos dos sites.

Trinity (SLEIMAN; CORCHUELO, 2014)

Sleiman e Corchuelo (2014) apresentam uma técnica, denominada *Trinity*, que extrai os valores dos atributos publicados em páginas-entidade baseadas em *template* de um site. Essa

[3]Disponível em: <http://www.wikipedia.org/>. Último acesso em: 10/10/2017.

[4]Disponível em: <https://www.freebase.com/>. Último acesso em: 10/10/2017.

[5]Amazon Mechanical Turk (disponível em: <https://www.mturk.com/mturk/welcome> - último acesso em: 10/10/2017.) é uma plataforma de contribuição colaborativa que coordena tarefas que exigem conhecimento humano. Nessa plataforma, os *requerentes* postam tarefas, como, por exemplo, identificar o cantor de uma determinada música. Os *produtores* podem navegar entre tarefas existentes, completá-las e receber pagamento do requerente pelas tarefas realizadas.

técnica parte da premissa de que o *template* introduz alguns padrões compartilhados que não fornecem dados relevantes e podem ser ignorados. A principal contribuição da técnica *Trinity* é construir uma árvore ternária de prefixos, separadores e sufixos, que é utilizada para inferir uma expressão regular que extrai os valores dos atributos nas páginas-entidade.

Trinity recebe como entrada um conjunto de páginas-entidade de um site. A saída é composta pelos valores dos atributos publicados nessas páginas. *Trinity* analisa os padrões compartilhados entre as páginas-entidade para construir uma árvore ternária de prefixos, separadores e sufixos. Sempre que encontra um padrão compartilhado entre as páginas-entidade, *Trinity* particiona as páginas em prefixos, separadores e sufixos. Os prefixos, separadores e sufixos são analisados recursivamente até que nenhum padrão seja encontrado. Prefixos, separadores e sufixos são organizados em uma árvore ternária. Essa árvore é percorrida para inferir uma expressão regular, com grupos de captura, que representa o *template* que foi utilizado para gerar as páginas-entidade. O usuário deve mapear os grupos de captura resultantes que extraem os valores dos atributos de interesse para as estruturas adequadas.

WADaR (ORTONA et al., 2016)

Ortona *et al.* (2016) propõem o método, denominado *WADaR* (***W**rapper **A**nd **D**ata **R**epair in **W**eb **D**ata **E**xtraction - Wrapper* e Reparação de Dados na Extração de Dados da Web), para extração de valores de atributos publicados em páginas-entidade baseadas em *template*. A principal contribuição desse método é utilizar cadeias de Markov para reparar regras de extração baseadas em XPath a fim de corrigir erros na extração.

WADaR recebe como entrada um conjunto de páginas-entidade de um site e o esquema de uma tabela. Esse esquema define os atributos que devem ser extraídos. A saída é composta pelos valores dos atributos (definidos no esquema) que estão publicados nas páginas-entidade. *WADaR* gera regras de extração baseadas em XPath utilizando um extrator de dados, que analisa a árvore DOM das páginas e encontra padrões. Através de um conjunto de reconhecedores de entidades (CHEN et al., 2013), *WADaR* seleciona uma regra de extração para cada atributo do esquema. As regras selecionadas são analisadas a fim de encontrar ruído, valores deslocados e valores segmentados incorretamente. Cadeias de Markov são utilizadas para segmentar os valores extraídos pelas regras de forma apropriada. A partir dessas segmentações, *WADaR* infere expressões regulares que são incorporadas às expressões XPath das regras de extração de modo que a extração realizada pelas regras modificadas reflitam as segmentações.

2.3.2 Outras categorias de páginas

DEPTA (ZHAI; LIU, 2006)

Zhai e Liu (2006) propõem um método, denominado *DEPTA* (*Data Extraction based on Partial Tree Alignment* - Extração de Dados baseada em Alinhamento Parcial de Árvore), para extrair valores de atributos publicados em páginas-multi-entidade baseadas em *template*. Uma página-multi-entidade é uma página que publica dados referentes a várias entidades de um determinado tipo. As principais contribuições do método *DEPTA* são: (i) utilizar as informações visuais (posições X e Y, altura e largura) para segmentar dados de diferentes entidades; e (ii) utilizar uma técnica de alinhamento parcial (baseado em casamento de árvores) para extrair valores do mesmo atributo que pertencem a diferentes entidades.

DEPTA recebe como entrada uma página que descreve várias entidades. A saída é composta pelos valores dos atributos das entidades publicadas na página. *DEPTA* segmenta a página para identificar cada região que contém valores de atributos de uma entidade. Essa segmentação é realizada analisando a árvore DOM em conjunto com as informações visuais. As informações visuais são obtidas a partir da renderização da página em um navegador Web. As informações visuais são utilizadas para corrigir erros no HTML e para identificar lacunas na página. Lacunas entre valores de uma mesma entidade são geralmente menores que lacunas entre valores de entidades diferentes. Depois que as regiões que publicam as entidades são identificadas, as subárvores de cada entidade são reorganizadas em uma única árvore porque: (i) mais de uma subárvore da árvore DOM original podem conter valores de atributos da mesma entidade; e (ii) os valores dos atributos de cada entidade podem não ser contínuos. As árvores de todas as entidades são alinhadas usando a técnica de alinhamento parcial. O resultado do alinhamento permite extrair os valores dos atributos das entidades publicadas na página.

Tubo Syncer (CHUANG; CHANG; ZHAI, 2007)

Chuang, Chang e Zhai (2007) propõem a abordagem *Tubo Syncer* para extrair valores de atributos publicados em páginas-multi-entidade de diferentes sites. Esse trabalho é um dos pioneiros em considerar a redundância de conteúdo entre múltiplos sites (que descrevem entidades do mesmo tipo) em vez de extrair os valores de cada site de forma isolada. *Tubo Syncer* explora apenas a redundância de conteúdo em nível de atributo. A redundância de conteúdo em nível de atributo ocorre quando duas ou mais fontes publicam atributos em comum. Por exemplo, dois sites que publicam o *nome*, o *time*, a *data de nascimento*, a *altura* e o *peso* de jogadores de futebol. Esse tipo de redundância ocorre entre múltiplos sites que descrevem entidades do

mesmo tipo e entre um site e uma base de conhecimento.

Tubo Syncer recebe como entrada páginas-multi-entidade de diferentes sites (que descrevem entidades do mesmo tipo) e uma base de conhecimento. A saída é formada pelos valores dos atributos publicados nessas páginas. *Tubo Syncer* identifica as áreas que descrevem cada entidade nas páginas utilizando um extrator não supervisionado que explora páginas baseadas em *template*, como, por exemplo, RoadRunner (CRESCENZI; MECCA; MERIALDO, 2001). É utilizada uma versão adaptada desse extrator que não separa os atributos da mesma entidade, apenas separa as entidades. Então, cada site é processado de modo a separar os segmentos de texto que se referem a diferentes atributos. Essa separação é realizada utilizando a base de conhecimento e modelos estatísticos, como, por exemplo, *Hidden Markov Models* (HMMs) (RABINER, 1989). Por fim, as extrações dos múltiplos sites são analisadas em conjunto a fim de corrigir segmentações errôneas com base no algoritmo *Expectation Maximization*.

ONDUX (CORTEZ et al., 2010)

Cortez *et al.* (2010) apresentam uma abordagem, denominada *ONDUX* (***On D**emand **U**nsupervised Information **Ex**traction* - Extração de informação não supervisionada sob demanda), que tem como objetivo identificar os valores disponíveis em um conjunto de registros textuais implícitos e associar esses valores com os atributos adequados. *ONDUX* é um dos pioneiros em explorar a informação disponível em bases de conhecimento para associar segmentos da entrada com atributos usando estratégias de casamento (funções de similaridade) em vez de estratégias explícitas de aprendizado (por exemplo, tamanho do segmento).

A abordagem *ONDUX* recebe como entrada uma base de conhecimento e registros implicitamente semiestruturados disponíveis em fontes textuais (por exemplo, endereços e anúncios). A saída é composta pelos valores dos atributos contidos nesses registros. A abordagem *ONDUX* separa a entrada textual em blocos (substrings) com base na coocorrência dos termos em um mesmo valor de atributo de acordo com a base de conhecimento. Os blocos são associados aos atributos contidos na base de conhecimento. Essa associação é realizada utilizando uma função de similaridade que compara o valor de cada bloco com os valores dos atributos presentes na base de conhecimento e determina o atributo que é mais provável que o bloco pertença. Finalmente, as associações entre blocos e atributos são reavaliadas utilizando um modelo de grafo probabilístico baseado em *Hidden Markov Models* (HMMs) (RABINER, 1989) que explora informações de posição e sequência. Essa reavaliação permite corrigir associações errôneas e unir blocos separados incorretamente.

JTOE (GUPTA; SARAWAGI, 2011)

Gupta e Sarawagi (2011) propõem um *framework*, denominado nesta Tese *JTOE* (***J****oint* ***T****raining for* ***O****pen-domain* ***E****xtraction on the Web* - Treinamento conjunto para extração em domínio aberto na Web), para extrair valores de atributos de listas HTML de diferentes sites que possuem segmentos de texto redundantes. Cada lista descreve várias entidades, ou seja, *JTOE* é voltado para páginas-multi-entidade. As listas podem estar em páginas sem *template*. A principal contribuição de *JTOE* é utilizar segmentos de texto compartilhados entre as listas e com uma base de conhecimento para eliminar a necessidade de anotação do usuário.

JTOE recebe como entrada um conjunto de listas HTML de diferentes sites que descrevem entidades do mesmo tipo e uma base de conhecimento com exemplos de entidades. A saída é composta pelos valores dos atributos contidos nas listas. *JTOE* utiliza modelos estatísticos (por exemplo, *Conditional Ramdom Fields* (LAFFERTY; MCCALLUM; PEREIRA, 2001)), para converter os elementos das listas em registros estruturados. São exploradas características de listas HTML e de redundância de conteúdo. As características de listas HTML incluem presença de delimitadores, marcações HTML, frequência das palavras, ordem de rótulos, entre outras. A redundância de conteúdo é explorada por meio do compartilhamento de segmentos de texto entre diferentes sites e entre cada site e a base de conhecimento. *JTOE* explora essa característica no treinamento dos modelos de extração, em que cada site tem um número limitado de segmentos redundantes com a base de conhecimento, porém entre os sites há muitos segmentos de texto compartilhados.

2.4 Trabalhos que combinam descoberta e extração

Esta seção apresenta as abordagens que realizam tanto a descoberta das páginas-entidade quanto a extração dos valores dos atributos publicados nessas páginas.

DIADEM (FURCHE et al., 2014)

Furche *et al.* (2014) propõem uma metodologia, denominada *DIADEM* (***D****omain-centric,* ***i****ntelligent,* ***a****utomated* ***d****ata* ***e****xtraction* ***m****ethodology* - Metodologia para Extração de Dados Automática, Inteligente e Centrada no Domínio), que explora e analisa sites que descrevem entidades de um tipo específico para extrair os valores dos atributos dessas entidades. A principal contribuição dessa metodologia é utilizar conhecimento extensivo de ontologia e fenomenologia para eliminar a necessidade de anotação de determinadas páginas dos sites. A ontologia abrange o conhecimento sobre as entidades e seus relacionamentos. A fenomenologia

incorpora o conhecimento sobre as representações das entidades na linguagem textual, estrutural e visual dos sites que descrevem entidades de um determinado tipo. Por exemplo, alguns atributos são obrigatórios, alguns atributos aparecem juntos, outros valores são opções comuns em determinados campos de formulários.

A metodologia *DIADEM* recebe como entrada a URL da página principal de um site. A saída é composta pelos valores dos atributos publicados nas páginas-multi-entidade do site. A metodologia *DIADEM* explora o site de entrada para descobrir exemplos de páginas-multi-entidade. As páginas descobertas são analisadas para identificar valores de atributos. Por fim, são geradas regras que permitem a extração dos valores de atributos publicados nas páginas-multi-entidade do site. As regras de extração são definidas por meio da linguagem OXPath (FURCHE et al., 2011). A linguagem OXPath estende a linguagem XPath incluindo a possibilidade de: (i) simular a iteração do usuário; e (ii) especificar características visuais.

A metodologia *DIADEM* explora o site a partir de uma combinação de coletor focado e preenchimento de formulários. A fenomenologia e a ontologia são utilizadas para preencher os formulários do site a fim de obter as páginas-multi-entidade. A metodologia *DIADEM* identifica paginação, menus de navegação e dados irrelevantes, como, por exemplo, propagandas. As páginas são agrupadas de acordo com sua similaridade estrutural e visual para guiar a estratégia de exploração e evitar a análise de páginas similares.

A metodologia *DIADEM* identifica os valores de atributos publicados nas páginas-multi-entidade analisando os padrões presentes dentro das páginas e entre as páginas. Como o foco da metodologia *DIADEM* são páginas-multi-entidade, essa identificação deve contemplar os limites entre as entidades e entre os atributos. A ontologia e a fenomenologia são utilizadas para distinguir ruído de dados relevantes. A principal contribuição dessa etapa é o conceito de atributo pivô. Atributos pivô (por exemplo, *preço*) são atributos obrigatórios de um tipo que é fácil de detectar por meio da fenomenologia. A metodologia *DIADEM* localiza esses atributos pivôs para descartar estruturas regulares com dados irrelevantes ou ruído entre estruturas regulares com dados relevantes. Por exemplo, uma propaganda entre a descrição de duas entidades.

DEXTER (QIU et al., 2015)

Qiu *et al.* (2015) propõem o sistema *DEXTER*, que encontra sites sobre produtos na Web, bem como detecta e extrai as especificações de produtos publicadas nesses sites. A especificação de um produto é um conjunto de pares atributo-valor. As principais contribuições desse sistema são: (i) uma abordagem para descobrir sites que contêm especificações de produtos; (ii) a adaptação de uma técnica de coleta de fóruns em sites para descobrir páginas-entidade com

especificações de produtos; (iii) uma técnica para encontrar e extrair especificações de produtos; e (iv) um sistema completo para construir uma grande coleção de especificações de produtos.

DEXTER recebe como entrada um conjunto semente de especificações de produtos de uma determinada categoria. A saída é formada por uma coleção de especificações de produtos dessa categoria, que são extraídas de sites sobre produtos (que são localizados pelo sistema). A partir do conjunto semente de especificações de produtos, *DEXTER* localiza sites sobre produtos que contêm especificações. Essa localização é feita por um coletor focado que explora consultas a um motor de busca e links de entrada. Uma coleta intra-site é realizada em cada site, localizado pelo sistema, a fim de encontrar as páginas que contém especificações de produtos. A partir das páginas com especificações de produtos, *DEXTER* detecta o fragmento HTML das páginas que contém a especificação e extrai os valores dos atributos que a compõe.

A coleta intra-site de páginas com especificações de produtos corresponde a tarefa de descoberta de páginas-entidade sobre produtos. *DEXTER* realiza essa tarefa com base na premissa de que os sites consistem de uma página de entrada para conteúdo sobre produtos de uma determinada categoria, páginas-índice que apontam para páginas-entidade e as páginas-entidade. Dado um site, *DEXTER* descobre a página de entrada relacionada à determinada categoria de produto. Essa descoberta é realizada combinando duas estratégias: (i) os links da página principal do site recebem um escore que leva em consideração os termos contidos em seus textos de âncora; e (ii) as páginas do site recebem um escore atribuído por um motor de busca a partir da consulta formada pelo nome da categoria. Essas duas estratégias retornam escores independentes, que são combinados por meio de uma média harmônica. A página com o maior escore combinado é selecionada como página de entrada. Em seguida, as páginas apontadas pela página de entrada são analisadas para identificar as páginas-índice. Para realizar essa identificação, diversos classificadores supervisionados são treinados, um para cada categoria. As características utilizadas pelo classificador são os termos contidos nos textos de âncora dos links presentes nas páginas que são avaliadas. Finalmente, as páginas-entidade são descobertas classificando os links contidos nas páginas-índice a partir de seus textos de âncora.

A detecção e a extração de especificações de produtos correspondem a tarefa de extração de valores de atributos. *DEXTER* realiza essa tarefa com base na observação de que a maioria das especificações de produtos está em tabelas ou listas HTML. *DEXTER* encontra tabelas e listas nas páginas-entidade (obtidas na coleta intra-site) e as classifica como descrevendo especificações ou não. O classificador utilizado considera diversas características estruturais, como, por exemplo, número de links, número de itens e tamanho do texto. A partir das tabelas classificadas como descrevendo especificações, *DEXTER* extrai os valores dos atributos. Duas estra-

tégias são adotadas: (i) uma estratégia baseadas em heurísticas que consideram pontos comuns entre especificações em listas e tabelas; e (ii) uma estratégia híbrida que combina a estratégia baseada em heurísticas com a abordagem proposta por Dalvi, Kumar e Soliman (2011), que gera regras de extração baseadas em XPath a partir de uma anotação que inclui ruído.

2.5 Análise dos trabalhos relacionados

Esta seção apresenta uma análise comparativa dos trabalhos apresentados neste capítulo. Primeiramente, são apresentados os critérios utilizados para comparação. Em seguida, é feita a análise dos critérios, que é resumida na Tabela 2.1. Nessa tabela, os seguintes símbolos denotam que: (i) ✓ - o trabalho suporta o critério; (ii) × - o trabalho não suporta o critério; (iii) *N/A* - o critério não é aplicável ao trabalho; e (iv) ~ - o trabalho suporta parcialmente o critério.

Os critérios foram selecionados com o intuito de avaliar se os trabalhos podem ser efetivamente aplicados em cenários reais, que é o objetivo desta Tese. Os seguintes critérios são analisados para realizar a comparação, que indicam:

- C_1 - se o trabalho descobre as páginas que descrevem as entidades no site;
- C_2 - se o trabalho extrai os valores dos atributos das entidades publicadas nas páginas;
- C_3 - se o trabalho inclui uma estratégia de coleta que evite navegar por regiões improdutivas do site, ou seja, uma estratégia que coleta o máximo possível de páginas que descrevem entidades visitando o menor número possível de páginas do site;
- C_4 - se o trabalho é independente de limiares manuais de similaridade entre páginas, ou seja, se o usuário não precisa definir limiares de similaridade entre páginas;
- C_5 - se o trabalho é independente de supervisão, ou seja, se o usuário não necessita fazer anotações;
- C_6 - se o trabalho é independente de renderização, ou seja, se não é necessário renderizar as páginas em um navegador Web;
- C_7 - se o trabalho é independente de domínio de aplicação, ou seja, se diferentes tipos de entidade são suportados;
- C_8 - se o trabalho é independente de marcação HTML específica, ou seja, se as páginas que descrevem as entidades não precisam publicar os valores de atributos com marcações HTML específicas (por exemplo, <*TABLE*>);
- C_9 - se o trabalho é independente de um motor de busca, ou seja, se não é utilizado um motor de busca;

Tabela 2.1 – Comparação entre os trabalhos apresentados neste capítulo.

Trabalhos	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
Descoberta de páginas-entidade												
INDESIT	✓	X	✓	X	✓	✓	✓	✓	✓	✓	✓	N/A
SDC	✓	X	~	X	✓	✓	✓	✓	✓	✓	✓	N/A
GPP	✓	X	~	✓	✓	X	✓	✓	✓	✓	✓	N/A
EOC	✓	X	N/A	X	✓	✓	✓	✓	X	X	✓	N/A
Classificação de páginas												
PEBL	~	X	X	✓	X	✓	✓	✓	✓	✓	✓	N/A
MDL-UC	~	X	X	X	~	✓	✓	✓	✓	✓	✓	N/A
HCUD	~	X	X	✓	X	✓	X	✓	✓	X	✓	N/A
CALA	~	X	N/A	✓	~	✓	✓	✓	✓	✓	✓	N/A
Extração de valores de atributos: páginas-entidade baseadas em *template*												
SSM	X	✓	N/A	N/A	X	✓	✓	✓	✓	✓	✓	X
FindAttrPos	X	✓	N/A	N/A	✓	✓	✓	✓	✓	X	✓	X
SWDE	X	✓	N/A	N/A	X	X	✓	✓	✓	✓	✓	X
EIHDK	X	✓	N/A	N/A	~	✓	✓	✓	✓	X	X	X
Trinity	X	✓	N/A	N/A	~	✓	✓	✓	✓	✓	✓	✓
WADaR	X	✓	N/A	N/A	✓	✓	✓	✓	✓	X	✓	X
Extração de valores de atributos: outros tipos de páginas												
DEPTA	X	✓	N/A	N/A	✓	X	✓	✓	✓	✓	✓	X
TurboSyncer	X	✓	N/A	N/A	✓	✓	✓	✓	✓	X	✓	✓
ONDUX	X	✓	N/A	N/A	✓	✓	✓	✓	✓	X	✓	✓
JTOE	X	✓	N/A	N/A	✓	✓	✓	X	✓	X	✓	✓
Descoberta e extração												
DIADEM	✓	✓	N/A	✓	X	X	✓	✓	✓	X	✓	~
DEXTER	✓	✓	✓	✓	X	✓	X	X	X	✓	✓	X

Nota: C_1 - descoberta das páginas-entidade; C_2 - extração dos valores dos atributos; C_3 - estratégia de coleta; C_4 - independente de limiares manuais de similaridade entre páginas; C_5 - independente de supervisão; C_6 - independente de renderização; C_7 - independente de domínio de aplicação; C_8 - independente de marcação HTML específica; C_9 - independente de motor de busca; C_{10} - independente de base de conhecimento; C_{11} - independente de contribuição colaborativa; C_{12} - suporte a atributos com formato variante.

- C_{10} - se o trabalho é independente de base de conhecimento, ou seja, se o trabalho não necessita utilizar dicionários, ontologias, bases lexicais ou bases de conhecimento;
- C_{11} - se o trabalho é independente de contribuição colaborativa, ou seja, se o trabalho não submete tarefas a alguma plataforma de contribuição colaborativa; e
- C_{12} - se o trabalho suporta a extração de valores de atributos com formato variante.

A seguir os critérios são discutidos:

C_1 **e** C_2 - Os trabalhos *INDESIT*, *SDC*, *GPP* e *EOC* realizam a descoberta de páginas-entidade. Esses trabalhos descobrem as páginas que descrevem entidades nos sites, porém não extraem os valores dos atributos publicados nessas páginas. Dessa forma, um trabalho que realize a extração precisa ser aplicado posteriormente. Os trabalhos *PEBL*, *MDL-UC*, *HCUD* e *CALA* classificam ou agrupam as páginas Web. Esses trabalhos podem ser adaptados para a descoberta de páginas-entidade. No entanto, essa adaptação requer um algoritmo que colete as páginas do site, bem como a anotação do usuário. Além disso, um trabalho que realize a extração também precisa ser aplicado posteriormente. Os trabalhos *SSM*, *FindAttrPos*, *SWDE*, *EIHDK*, *Trinity*, *WADaR*, *DEPTA*, *TurboSyncer*, *ONDUX* e *JTOE* realizam a extração de valores de atributos em páginas que descrevem entidades. No entanto, esses trabalhos requerem que as páginas que descrevem as entidades sejam fornecidas como entrada. Dessa forma, um trabalho que encontre as páginas que descrevem entidades no site precisa ser empregado previamente. Os trabalhos *DIADEM* e *DEXTER* realizam tanto a descoberta das páginas que descrevem entidades nos sites quanto a extração dos valores dos atributos publicados nessas páginas.

C_3 - Os trabalhos *PEBL*, *MDL-UC* e *HCUD* apenas classificam ou agrupam as páginas do site. Dessa forma, um coletor extensivo deve ser aplicado ao site sobre análise. Esse coletor pode interferir na operação normal do site (HERNÁNDEZ et al., 2016), o que não é desejável. Os trabalhos *SDC* e *GPP* possuem uma estratégia de coleta de páginas que realiza buscas em largura a partir da página principal do site até as páginas-entidade. Essas buscas em largura fazem com que diversos caminhos improdutivos sejam percorridos. Os trabalhos *EOC*, *CALA* e *DIADEM* são específicos para a Web oculta, logo requerem a submissão de formulários em vez de um coletor que explore as páginas do site. Os trabalhos que realizam apenas a extração (*SSM*, *FindAttrPos*, *SWDE*, *EIHDK*, *Trinity*, *WADaR*, *DEPTA*, *TurboSyncer*, *ONDUX* e *JTOE*) precisam que as páginas que descrevem as entidades sejam fornecidas como entrada. Os trabalhos *INDESIT* e *DEXTER* encontram as páginas-entidade levando em consideração a forma como os sites organizam as páginas-entidade a fim de minimizar a navegação por regiões improdutivas.

C_4 - Os trabalhos *INDESIT*, *SDC*, *EOC* e *MDL-UC* requerem a intervenção humana para definição de limiares de similaridade entre páginas. Essa é uma tarefa custosa uma vez que requer um usuário especialista que processe um conjunto de páginas para encontrar o limiar adequado. Além disso, cada site possui um limiar de similaridade adequado diferente (conforme observado nos experimentos). Esse critério não se aplica aos trabalhos que realizam apenas a extração. Os demais trabalhos não requerem a intervenção humana para definição de limiares de similaridade entre páginas.

C_5 - *PEBL*, *HCUD*, *SSM*, *SWDE* e *DEXTER* requerem supervisão *a priori*, ou seja, o usuário precisa realizar algum tipo de anotação antes da execução do trabalho. *DIADEM* também necessita de supervisão *a priori* uma vez que, antes de sua execução, o usuário precisa descrever a fenomenologia dos sites. *MDL-UC*, *CALA*, *EIHDK* e *Trinity* requerem supervisão *a posteriori*, ou seja, o usuário precisa realizar algum tipo de anotação após a execução do trabalho. Anotações são trabalhosas e sujeitas a erros, por isso não são adequadas para cenários reais (MADHAVAN et al., 2008). *INDESIT*, *SDC*, *EOC*, *GPP*, *FindAttrPos*, *WADaR*, *DEPTA*, *TurboSyncer*, *ONDUX* e *JTOE* não requerem nenhum tipo de anotação.

C_6 - Os trabalhos *GPP*, *SWDE*, *DEPTA* e *DIADEM* utilizam características visuais. Para extrair essas características, as páginas são renderizadas em um navegador Web, carregando imagens, CSS e JavaScripts. Essa renderização aumenta significativamente o tempo de processamento (conforme observado nos experimentos). Por isso, renderizar as páginas não é adequado para cenários reais. Os demais trabalhos não requerem a renderização das páginas.

C_7 - *HCUD* é específico para o tipo de entidade *pesquisadores* enquanto que *DEXTER* é específico para o tipo de entidade *produtos*. Esses trabalhos não são genericamente aplicáveis, uma vez que podem ser utilizados apenas para um determinado tipo de entidade. Os demais trabalhos podem ser aplicados para diferentes tipos de entidades, o que expande a quantidade de aplicações do mundo real em que eles podem ser empregados.

C_8 - *JTOE* é específico para listas HTML enquanto que *DEXTER* identifica especificações de produtos em tabelas ou listas HTML. A aplicação desses trabalhos é restrita às páginas cujos valores dos atributos estão organizados em um número limitado de padrões HTML específicos. Além disso, a linguagem CSS (*Cascading Style Sheets*) habilitou os designers a substituir os leiautes baseados em tabelas por outras marcações (por exemplo, <*DIV*>) e, com isso, muitas características estruturais usadas nos trabalhos baseados em tabelas e listas HTML foram erradicadas do HTML das páginas-entidade (MUROLO; NORRIE, 2016). Os demais trabalhos são independentes de marcações HTML específicas.

C_9 - *EOC* necessita de um *log* de consultas de um motor de busca. No entanto, é difícil para pessoas que não trabalham em motores de busca comerciais obter um grande conjunto de *logs* de consultas (YOSHINAGA; TORISAWA, 2007). *DEXTER* submete consultas a um motor de busca. Porém, os motores de busca restringem o número de buscas diárias gratuitas. Os demais trabalhos não utilizam motores de busca.

C_{10} - Os trabalhos *EOC*, *HCUD*, *FindAttrPos*, *EIHDK*, *WADaR*, *TurboSyncer*, *ONDUX*, *JTOE* e *DIADEM* utilizam algum tipo de base de conhecimento. Diferentes tipos de bases de conhecimento são utilizados, como, por exemplo, uma base léxica (WordNet), uma ontologia,

reconhecedores de entidade, uma base com exemplos de entidades e uma base com exemplos de valores de atributos. A necessidade de uma base de conhecimento limita a aplicação do trabalho a sites em um idioma que possua tal recurso (HERNÁNDEZ et al., 2016). Além disso, as bases de conhecimento existentes apresentam falta de cobertura de entidades (DONG et al., 2014). As bases de conhecimento existentes também podem estar desatualizadas, o que afeta os trabalhos baseados em redundância de conteúdo em nível de entidade. Os demais trabalhos não utilizam bases de conhecimento.

C_{11} - *EIHDK* utiliza contribuição colaborativa (*crowdsourcing*). O principal problema de utilizar contribuição colaborativa em cenários reais está relacionado ao seu custo. Esse custo envolve o pagamento pela utilização da contribuição colaborativa, bem como o tempo de espera entre a requisição e a resposta. Além disso, um cuidado especial é necessário na formulação das tarefas a serem submetidas aos colaboradores uma vez que eles não são especialistas (KITTUR; CHI; SUH, 2008). Os demais trabalhos não utilizam contribuição colaborativa.

C_{12} - Os trabalhos *FindAttrPos*, *SWDE*, *EIHDK*, *WADaR*, *DEXTER* não suportam atributos com formato variante porque geram regras de extração baseadas em XPath e selecionam apenas uma regra para cada atributo. Os trabalhos *DEPTA* e *SSM* não suportam atributos com formato variante porque utilizam uma técnica de alinhamento parcial de árvore que requer que as marcações correspondam exatamente e estejam no mesmo nível. *DIADEM* suporta parcialmente atributos com formato variante uma vez que utiliza uma extensão de XPath. *Trinity*, *TurboSyncer*, *ONDUX* e *JTOE* suportam atributos com formato variante.

A lacuna deixada pelos trabalhos relacionados está em não realizar as tarefas de descoberta das páginas-entidade nos sites e de extração dos valores dos atributos publicados nessas páginas de forma integrada. Cada uma dessas tarefas é um desafio significante por si só, mas essas tarefas, geralmente, têm sido tratadas na literatura de forma isolada (FURCHE et al., 2014). Além disso, considerando apenas a tarefa de descoberta de páginas-entidade, não há um trabalho que combine todas as seguintes características: (i) inclusão de uma estratégia de coleta que vise baixar o menor número possível de páginas do site; (iii) independência de limiares de similaridade entre páginas; (ii) independência de supervisão; (iv) independência de renderização; (v) independência de domínio de aplicação; (vi) independência de marcações específicas; (vii) independência de motores de busca; (viii) independência de base de conhecimento; e (iv) independência de contribuição colaborativa. Considerando apenas a tarefa de extração dos valores dos atributos publicados nas páginas-entidade baseadas em *template*, não há um trabalho que combine todas as seguintes características: (i) independência de supervisão; (ii) independência de renderização; (iii) independência de domínio de aplicação; (iv) independência de marcações

HTML específicas; (v) independência de motores de busca; (vi) independência de base de conhecimento; (vii) independência de contribuição colaborativa; e (viii) suporte a atributos com formato variante. O objetivo da abordagem proposta, **Orion**, é preencher essas lacunas.

2.6 Diferencial da abordagem proposta

Esta seção discute as diferenças entre a abordagem **Orion** e os trabalhos diretamente relacionados. Primeiramente, são apresentadas as características da abordagem **Orion**. É realizada uma diferenciação entre **Orion** e os trabalhos que realizam a descoberta de páginas-entidade e a extração de valores de atributos de forma integrada. São discutidas as diferenças entre **Orion** e os trabalhos que realizam apenas a descoberta de páginas-entidade. São elencadas as diferenças entre **Orion** e os trabalhos que realizam apenas a extração de valores de atributos nas páginas-entidade. Finalmente, as contribuições da abordagem **Orion** são enfatizadas.

A abordagem **Orion** realiza tanto a descoberta de páginas-entidade quando a extração dos valores dos atributos publicados nessas páginas. A tarefa de descoberta de páginas-entidade é realizada por meio de uma estratégia de coleta que visa baixar o menor número possível de páginas do site. Essa estratégia é independente de: (i) limiares manuais de similaridade entre páginas; (ii) supervisão do usuário; (iii) renderização; (iv) domínio de aplicação; (v) marcação específica; (vi) motor de busca; (vii) base de conhecimento; e (vii) contribuição colaborativa. A tarefa de extração dos valores dos atributos é realizada por meio de consultas Cypher. A inferência das consultas é realizada de forma independente de: (i) supervisão do usuário; (ii) renderização; (iii) domínio de aplicação; (iv) marcação específica; (v) motor de busca; (vi) base de conhecimento; e (vii) contribuição colaborativa. A abordagem **Orion** descreve cinco alternativas para a seleção das consultas que extraem os valores dos atributos: (i) um usuário especialista selecione a consulta adequada para cada atributo; (ii) colaboradores engajados por meio de uma plataforma de contribuição colaborativa selecionam a consulta adequada para cada atributo; (iii) um algoritmo supervisionado seleciona a consulta adequada para cada atributo com base em um conjunto de treinamento; (iv) um algoritmo não supervisionado seleciona a consulta adequada para cada atributo a partir de uma base de conhecimento; e (v) um algoritmo não supervisionado seleciona a consulta adequada para cada atributo com base na redundância de conteúdo entre diferentes sites que descrevem entidades do mesmo tipo. A abordagem **Orion** ainda inclui uma etapa adicional para tratamento de atributos com formato variante.

DIADEM e *DEXTER* são os trabalhos mais relacionados à abordagem **Orion** porque realizam tanto a descoberta de páginas-entidade quando a extração dos valores dos atributos

publicados nessas páginas. No entanto, eles não são diretamente comparáveis com **Orion**. *DIADEM* é voltado à Web oculta enquanto **Orion** é voltado à Web de superfície. *DIADEM*, ao contrário de **Orion**, requer anotação *a priori*. *DEXTER* é específico para o tipo de entidade *produtos* enquanto que **Orion** é independente de tipo de entidade, logo, **Orion** pode ser empregado em um número maior de aplicações reais.

Com relação à descoberta de páginas-entidade, os trabalhos diretamente relacionados com a abordagem **Orion** são: (i) *INDESIT*- que explora apenas características de HTML; (ii) *GPP*- que explora características de HTML e visuais; e (iii) *SDC* - que explora características de HTML e URL. A abordagem **Orion** explora características de HTML e URL. **Orion** reusa as definições de *INDESIT* para representar o HTML das páginas. A contribuição de **Orion** nessa tarefa é como utilizar as características de URL e combiná-las com as características de HTML. Além disso, **Orion** determina automaticamente os limiares de similaridade de HTML e de URL nos sites, enquanto que *INDESIT* precisa que o usuário defina o limiar de similaridade de HTML. A abordagem **Orion** não utiliza características visuais, ao contrário de *GPP*, porque a extração dessas características requer a renderização das páginas em um navegador Web, o que aumenta significativamente o tempo de processamento. **Orion** e *SDC* combinam características de URL com características de HTML. *SDC* calcula a similaridade de HTML usando a distância de edição entre árvores. **Orion** calcula a similaridade de HTML aplicando uma função de similaridade sintática (coeficiente de Jaccard) sobre os caminhos na árvore DOM da página que iniciam na raiz e terminam em nodos âncoras. A técnica utilizada por **Orion** é menos custosa em termos de tempo de processamento e é menos influenciada por desalinhamentos entre as árvores DOM das páginas-entidade. **Orion** atribui diferentes pesos para os termos de URL de acordo com sua capacidade de discriminar páginas-entidade das demais páginas enquanto *SDC* não atribui pesos para os termos de URL. *SDC* requer a definição de um limiar de similaridade de HTML, enquanto **Orion** determina esse limiar automaticamente em cada site.

Com relação à extração de valores de atributos, os trabalhos diretamente relacionados com a abordagem **Orion** são: (i) *SSM*- que é baseado em um algoritmo de alinhamento parcial de árvore; (ii) *SWDE*- que é baseado em expressões XPath; e (iii) *Trinity*- que infere uma expressão regular a partir de uma árvore ternária de prefixos, separadores e sufixos. A abordagem **Orion** é baseada em consultas Cypher em um banco de dados orientado a grafos. Hao *et al.* (2011) argumentam que: (i) *SSM* é projetado para um conjunto de treinamento que contemple inúmeros sites e é propenso a *overfit* quando o conjunto de treinamento é pequeno; (ii) *SSM* é facilmente afetado por desalinhamentos entre as árvores DOM das páginas-entidade; e (iii) o alinhamento de árvores DOM é bastante demorado. Por outro lado, **Orion** não é baseado em

um algoritmo de alinhamento parcial de árvore, assim como, ele não é dependente de um conjunto de treinamento. *SWDE* é suscetível à variação de *template* uma vez que ele requer que os valores de um atributo tenham o mesmo caminho DOM (da raiz até o nodo textual) em todas as páginas-entidade do site. Além disso, *SWDE* requer que as páginas-entidade sejam renderizadas em um navegador Web, o que aumenta significativamente o tempo de processamento. *SWDE* também requer um conjunto de treinamento. De acordo com Hernández *et al.* (2016), anotar um conjunto de treinamento é uma tarefa tediosa, demorada e suscetível a erros. Por outro lado, **Orion** é menos suscetível a variação de *template* que *SWDE* devido à forma como a abordagem realiza a extração dos valores (por meio de consultas Cypher) e a inclusão de uma etapa adicional para tratamento de atributos com formato variante. Além disso, **Orion** não depende nem da renderização das páginas-entidade nem de um conjunto de treinamento. *Trinity* é afetado por: (i) atributos omissos (ou seja, atributos onde os valores dos atributos são publicados em apenas algumas páginas-entidade do site); (ii) atributos publicados em tabelas horizontais (ou seja, o cabeçalho está na primeira linha da tabela); e (iii) páginas-ruído (ou seja, páginas que não descrevem uma entidade ou que descrevem uma entidade de um tipo diferente do tipo de entidade de interesse). **Orion** não é afetado por atributos publicados em tabelas horizontais, páginas-ruído nem atributos omissos que são publicados em elementos simples (ou seja, quando o elemento possui apenas um nodo textual).

A principal contribuição da abordagem **Orion** é tratar os problemas de descoberta de páginas-entidade e extração dos valores publicados nessas páginas de forma integrada, independente de domínio de aplicação e de anotação *a priori*. Como contribuições secundárias destacam-se: (i) o aprendizado automático dos limiares de similaridade de HTML e URL para a descoberta de páginas-entidade; (ii) a atribuição de pesos para os termos de URL de acordo com sua capacidade de distinguir páginas-entidade das demais páginas; (iii) a utilização de consultas Cypher sobre um banco de dados orientado a grafos para a extração dos valores de atributos, que atribui uma semântica declarativa a essa tarefa uma vez que tanto o caminhamento quanto a atualização do grafo é realizada por meio de uma linguagem declarativa; e (iv) o tratamento de atributos com formato variante.

3 ABORDAGEM ORION

Este capítulo apresenta a abordagem proposta nesta Tese para aquisição de valores de atributos de entidades do mundo real por meio da descoberta e da extração de dados de páginas-entidade baseadas em *template*. A Seção 3.1 apresenta a visão geral da abordagem e enfatiza o objetivo da Tese. Nas seções seguintes, cada etapa da abordagem proposta é detalhada.

3.1 Visão geral

O objetivo geral desta Tese é o **desenvolvimento de uma abordagem eficaz e robusta para aquisição de valores de atributos de entidades do mundo real por meio da descoberta e da extração de dados de páginas-entidade**. A Figura 3.1 apresenta as principais etapas da abordagem proposta, denominada **Orion**. A entrada é composta por um conjunto de páginas-entidade de diferentes sites. Apenas uma página-entidade é requerida para cada site, uma vez que a abordagem utiliza essa página como exemplo para encontrar as demais páginas-entidade que estão presentes no site. A saída é composta pelos valores dos atributos publicados nas páginas-entidades dos sites. **Orion** inclui as seguintes etapas: (1) *Descoberta* - encontra as páginas-entidade presentes nos sites (Seção 3.2); (2) *Extração* - extrai os valores dos atributos publicados nas páginas-entidade de cada site (Seção 3.3); e (3) *Reforço* - revisa a extração realizada na segunda etapa com o objetivo de extrair mais valores de atributos (Seção 3.4).

Figura 3.1 – Fluxo de execução da abordagem **Orion**.

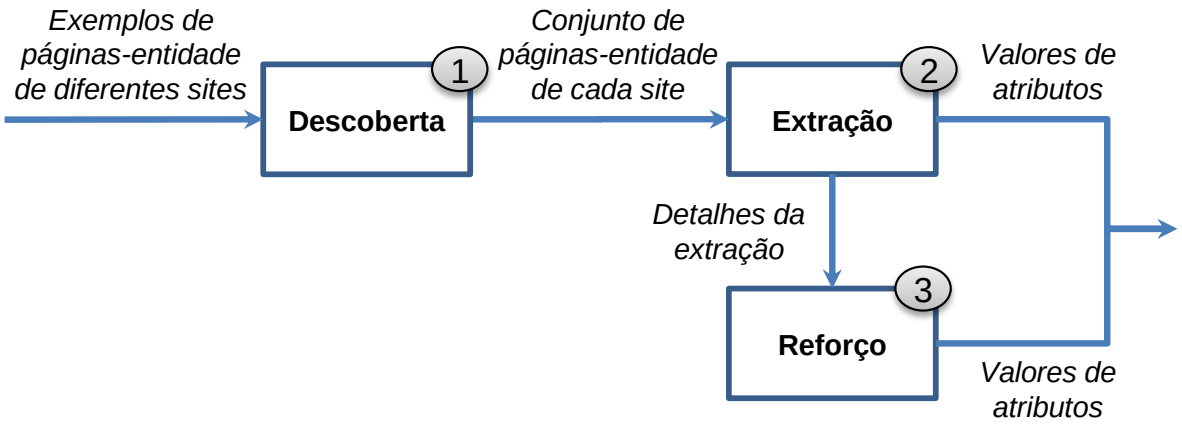
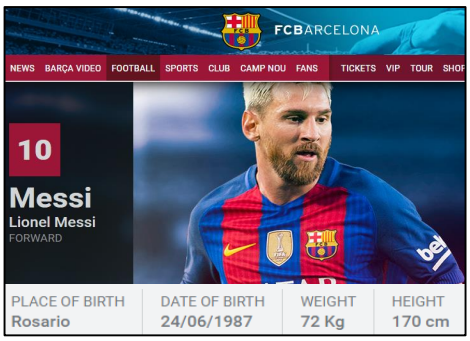
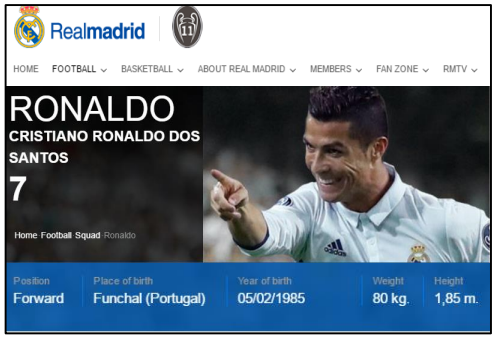


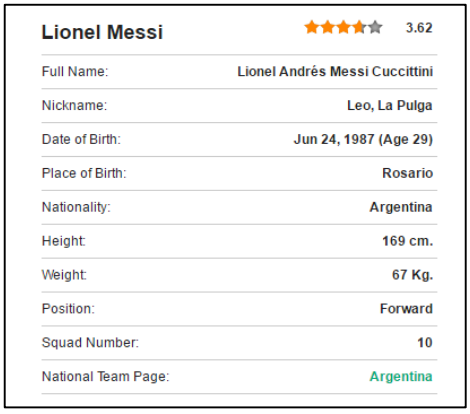
Figura 3.2 – Três exemplos de sites.



(a) FCBarcelona.com



(b) RealMadrid.com



(c) Goal.com

site	página	nome	nasc.	altura	...
FCBarcelona.com	http://...	Lionel Messi	24/06/1987	170 cm	...
	http://...	Andrés Iniesta	11/05/1984	171 cm	...
	...	...	...	...	...
RealMadrid.com	http://...	Cristiano ...	05/02/1985	1,85 m.	...
	http://...	Francisco ...	02/10/1986	191 cm	...
	...	...	...	...	...
Goal.com	http://...	Lionel Messi	Jun 24, 1987	169 cm.	...
	http://...	Cristiano ...	Feb 5, 1985	185 cm.	
	...	...	...	...	...

(d) Exemplo de saída

Por exemplo, a Figura 3.2 ilustra um fragmento da página-entidade que descreve o jogador: (a) *Lionel Messi* no site *FCBarcelona.com*[1]; (b) *Cristiano Ronaldo* no site *RealMadrid.com*[2]; e (c) *Lionel Messi* no site *Goal.com*[3]. Os jogadores são descritos nessas páginas por meio dos atributos: *nome*, *data de nascimento*, *altura*, entre outros. A Figura 3.2(d) apresenta a saída da abordagem **Orion** a partir de uma entrada composta pelas páginas-entidade ilustradas nas figuras 3.2(a)-(c). A abordagem **Orion** utiliza as páginas-entidade fornecidas como entrada para encontrar as demais páginas-entidade dos sites. Por exemplo, a página que descreve o jogador *Andrés Iniesta* é encontrada no site *FCBarcelona.com*. A abordagem **Orion** extrai e retorna os valores dos atributos publicados nas páginas-entidade dos sites.

[1]Disponível em: <https://www.fcbarcelona.com/>. Último acesso em: 10/07/2017.
[2]Disponível em: <http://www.realmadrid.com/>. Último acesso em: 10/07/2017.
[3]Disponível em: <http://www.goal.com/>. Último acesso em: 10/07/2017.

3.2 Descoberta

A etapa de *Descoberta* coleta automaticamente as páginas-entidade do mesmo tipo presentes nos sites. Essa etapa obtém as páginas que publicam os valores dos atributos a serem extraídos. A abordagem **Orion** parte da premissa que páginas-entidade do mesmo tipo em um site compartilham: (i) uma estrutura de HTML similar; e (ii) uma estrutura de URL similar.

A tarefa de encontrar as páginas-entidade envolve diversos desafios, tais como:

- *diferentes estruturas de páginas-entidade* - os sites utilizam diferentes estruturas para publicar os dados nas páginas-entidade: <*TABLE*>, <*UL*>, <*OL*>, <*DL*>, <*DIV*> e inúmeras outras. Diferentes estruturas de páginas-entidade influenciam a estratégia de identificação das páginas-entidade em cada site. A abordagem **Orion** utiliza uma estratégia independente de marcação HTML específica. Considera-se que todas as páginas-entidade do mesmo site têm uma estrutura de HTML similar, mas não é exigido que as páginas possuam uma marcação específica (por exemplo, <*TABLE*>) em todos os sites;
- *diferentes limiares de similaridade de HTML* - o limiar de similaridade de HTML que separa as páginas-entidade das demais páginas varia de site para site. A abordagem **Orion** determina automaticamente o limiar de similaridade de HTML em cada site. Em alguns sites, não há um limiar de similaridade de HTML capaz de separar as páginas-entidade das demais páginas, pois há páginas-entidade mais similares em termos de estrutura HTML às páginas que não são páginas-entidade do que às outras páginas-entidade do site. Por isso, a abordagem **Orion** diferencia as páginas-entidade das demais páginas analisando tanto características de HTML quanto características de URL; e
- *evolução dos sites* - os sites podem evoluir com o tempo de modo que os valores dos atributos publicados nas páginas-entidade podem mudar, a URL das páginas-entidade pode mudar e novas páginas-entidade podem ser disponibilizadas. Essa evolução exige que a descoberta de páginas-entidade seja reexecutada periodicamente para o mesmo site. Portanto, o tempo de processamento e o número de páginas baixadas são importantes métricas para escolher uma abordagem de descoberta de páginas-entidade efetiva. A abordagem **Orion** utiliza: (i) apenas características que não necessitem renderizar as páginas em um navegador Web, uma vez que essa renderização aumenta significativamente o tempo de processamento; e (ii) uma estratégia de coleta com o objetivo de baixar o menor número possível de páginas do site para encontrar as páginas-entidade.

O restante desta seção está organizado da seguinte forma. A Subseção 3.2.1 define os

conceitos necessários para a compreensão da etapa de *Descoberta*. A Subseção 3.2.2 apresenta uma visão geral da etapa. As subseções de 3.2.3 a 3.2.5 descrevem detalhes da etapa.

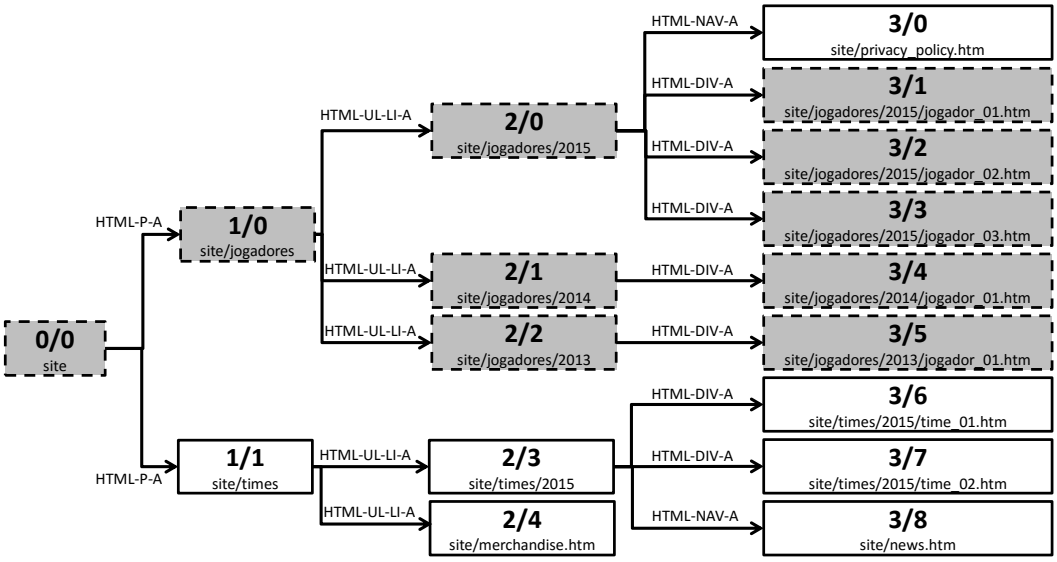
3.2.1 Definições preliminares

Esta Tese assume que parte de um site é projetada para permitir que o usuário navegue da página principal até as páginas que descrevem as entidades do mesmo tipo por meio de uma hierarquia lógica de tópicos (WENINGER; JOHNSTON; HAN, 2013), denominada `árvore-entidade`. As páginas que integram a árvore-entidade são classificadas em páginas-entidade e páginas-índice. As `páginas-entidade` são as páginas que publicam dados que descrevem uma determinada entidade de um tipo particular (BLANCO et al., 2008). As páginas-entidade são as folhas da árvore-entidade. As `páginas-índice` são as páginas que apontam para as páginas-entidade ou para outras páginas-índice. O papel das páginas-índice é agrupar páginas-entidade/páginas-índice a fim de permitir que o usuário navegue por meio da hierarquia lógica de tópicos. As páginas-índice são os nodos internos da árvore-entidade. As páginas do site que não pertencem à árvore-entidade são denominadas `páginas-ruído`. Um site pode ter diversas árvores-entidade. Por exemplo, um site sobre um campeonato de futebol pode ter uma árvore-entidade para jogadores e uma árvore-entidade para times. A página é formalizada a seguir.

Definição 1 *(`Página`): uma `página` é uma tripla $p = (u, \chi, i)$, em que u é a URL; $\chi = \{\chi_1, ..., \chi_n\}$ é o conjunto de todos os link-paths presentes no HTML da página; e i é o index-path. Um `link-path` de uma página é um caminho através da árvore DOM da página que inicia na raiz e termina em um nodo âncora (marcação HTML $< a >$). O `index-path` de uma página p é o link-path (do nodo âncora) da página-índice que aponta para a página p.*

Por exemplo, a Figura 3.3 detalha as páginas de um site fictício sobre um campeonato de futebol com enfoque na árvore-entidade de jogadores. Essa árvore-entidade é utilizada para exemplificar os conceitos e a forma como a abordagem **Orion** descobre as páginas-entidade. A página principal do site é a página $0/0$. As páginas são identificadas na árvore-entidade por i/j, em que i representa o nível da página com relação à página principal e j é um identificador que difere as páginas do mesmo nível. Por exemplo, a página $2/1$ é a página com identificador 1 no nível 2. O site possui páginas-entidade de dois tipos: jogadores e times. Há uma árvore-entidade para cada tipo. A árvore-entidade de jogadores está destacada com fundo sólido. As páginas $3/i$ ($1 \leq i \leq 5$) são páginas-entidade que descrevem um determinado jogador. As

Figura 3.3 – Detalhes de site sobre um campeonato de futebol.



Nota: a árvore-entidade de jogadores está destacada com fundo sólido.

páginas $2/j$ $(0 \leq j \leq 2)$ são páginas-índice que apontam para as páginas-entidade sobre jogadores que disputaram o campeonato em um determinado ano. A página $1/0$ é a página-índice que aponta para todas as páginas-índice de nível 2. A página $0/0$ é a página-índice que aponta para a página-índice de nível 1. As demais páginas são páginas-ruído no contexto de jogadores, pois não são páginas-entidade de jogadores nem fazem parte da hierarquia lógica de tópicos que permite a navegação do usuário a partir da página principal até as páginas-entidade de jogadores. O rótulo de cada aresta é o *link-path* do nodo âncora por meio do qual uma página aponta para outra, ou seja, é o *index-path* da página que é apontada. Por exemplo, a página $2/0$ é representada como: $(u = \text{“}site/jogadores/2015\text{”}, \chi = \{\text{“}HTML-NAV-A\text{”}, \text{“}HTML-DIV-A\text{”}\}, i = \text{“}HTML-UL-LI-a\text{”})$.

A abordagem **Orion** requer que o usuário forneça, como entrada, uma página-entidade de cada site do qual se deseja extrair os valores dos atributos publicados. Essa página é escolhida aleatoriamente e é denominada `página de exemplo`. O caminho da raiz até a página de exemplo na árvore-entidade é denominado `caminho de exemplo`. As páginas apontadas por uma determinada página p_x são denominadas `páginas-filhas` da página p_x. Considerando que: (i) $nivel(p_i)$ é uma função que retorna o nível de uma determinada página p_i na árvore-entidade; (ii) p_x e p_{x+1} são duas páginas que pertencem a árvore-entidade que contém a página de exemplo; (iii) p_{x+1} é uma página que integra o caminho de exemplo; e (iv) $nivel(p_{x+1}) - nivel(p_x) = 1$. A abordagem **Orion** calcula a similaridade entre cada página-filha de p_x e a página p_{x+1}. As páginas-filhas de p_x cujo escore de Similaridade de URL é maior que

o limiar $limiar_{URL}$ e o escore de Similariade de HTML é maior que o limiar $limiar_{HTML}$ são denominadas `páginas similares`. Os limiares $limiar_{HTML}$ e $limiar_{URL}$ são aprendidos automaticamente pela abordagem **Orion** em cada site.

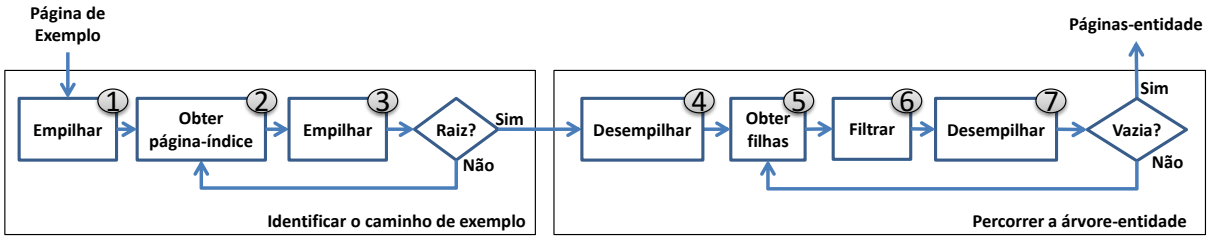
Por exemplo, considerando o site apresentado na Figura 3.3 e a página $3/2$ como página de exemplo. O caminho de exemplo é "$0/0 \rightarrow 1/0 \rightarrow 2/0 \rightarrow 3/2$". As páginas $3/0$, $3/1$, $3/2$ e $3/3$ são páginas-filhas da página $2/0$. As páginas $3/1$, $3/2$ e $3/3$ são páginas similares.

3.2.2 Visão geral

A etapa de *Descoberta* processa cada site de forma isolada. A página-entidade do site fornecida como entrada é utilizada para encontrar as demais páginas-entidade do mesmo tipo que estão presentes no site. A abordagem **Orion** assume que as páginas que devem ser encontradas são similares à página fornecida como entrada. Essa similaridade ocorre tanto entre o HTML das páginas quanto entre a URL.

A Figura 3.4 apresenta o fluxo de execução da etapa de *Descoberta* para um site. A entrada é uma página-entidade, denominada página de exemplo. A saída é composta pelas páginas-entidade que são do mesmo tipo que a página de exemplo e estão no mesmo site. Duas subetapas são executadas: (i) *identificar o caminho de exemplo* - encontra e retorna o caminho de exemplo, isto é, o caminho que inicia na raiz da árvore-entidade e termina na página de exemplo; e (ii) *percorrer a árvore-entidade* - percorre a árvore-entidade que contém a página de exemplo e retorna as suas páginas-entidade.

Figura 3.4 – Fluxo de execução da etapa de *Descoberta*.



A subetapa *identificar o caminho de exemplo* recebe como entrada uma página de exemplo. A página de exemplo é adicionada a uma pilha que representa o caminho de exemplo (1). A página-índice da página que está no topo da pilha (página de exemplo) é obtida (2) e adicionada à pilha (3). Se a página que está no topo da pilha (última página-índice obtida) não é a raiz da árvore-entidade, a página-índice da página que está no topo da pilha é obtida (2) e adicionada à

pilha (3). Se a página que está no topo da pilha é a raiz da árvore-entidade, a identificação do caminho de exemplo é concluída. A saída é uma pilha que representa o caminho de exemplo. O topo da pilha contém a raiz da árvore-entidade. A base da pilha contém a página de exemplo.

A subetapa *percorrer a árvore-entidade* recebe uma pilha que representa o caminho de exemplo. A raiz da árvore-entidade (topo) é removida da pilha (4). As páginas-filhas da raiz da árvore-entidade são obtidas (5). As páginas-filhas são filtradas de acordo com a sua similaridade de URL e HTML com a página que está no topo da pilha e apenas as páginas similares são mantidas (6). O topo da pilha é removido (7). Se a pilha não ficou vazia, as páginas-filhas das páginas similares são obtidas (5) e filtradas (6), assim como, o topo da pilha é removido (7). Se a pilha ficou vazia, as páginas similares obtidas na última iteração são retornadas.

A seguir, é apresentado um exemplo da execução da etapa de *Descoberta* sobre o site apresentado na Figura 3.3. A entrada é a página $3/2$. A página-índice da página $3/2$ é obtida $(2/0)$ e adicionada à pilha. A página-índice da página $2/0$ é obtida $(1/0)$ e adicionada à pilha. A página-índice da página $1/0$ é obtida $(0/0)$ e adicionada à pilha. O caminho de exemplo está completo e é formado por: "$0/0 \rightarrow 1/0 \rightarrow 2/0 \rightarrow 3/2$". As páginas-filhas da raiz são obtidas: $1/0$ e $1/1$. As páginas-filhas são filtradas de acordo com a similaridade de HTML e de URL com a página $1/0$. As duas páginas-filhas são similares, então são mantidas. As páginas-filhas das páginas $1/0$ e $1/1$ são obtidas: $2/i$ ($0 <= i <= 4$). As páginas-filhas são filtradas de acordo com a similaridade de HTML e de URL com a página $2/0$. As páginas $2/0$, $2/1$, $2/2$ e $2/3$ são similares e permanecem. A página $2/4$ é eliminada porque não é similar. As páginas-filhas das páginas $2/i$ ($0 <= i <= 3$) são obtidas: $3/j$ ($0 <= j <= 8$). As páginas-filhas são filtradas de acordo com a similaridade de HTML e de URL com a página $3/2$. As páginas $3/k$ ($1 <= k <= 5$) são similares e permanecem. As páginas $3/0$, $3/6$, $3/7$ e $3/8$ são eliminadas porque não são similares. Como as folhas da árvore-entidade foram atingidas, as páginas similares obtidas na última iteração ($3/1$, $3/2$, $3/3$, $3/4$ e $3/5$) são retornadas.

As próximas subseções detalham as duas subetapas da etapa de *Descoberta* da abordagem **Orion**. A Subseção 3.2.4 apresenta a subetapa *identificar o caminho de exemplo* enquanto que a Subseção 3.2.5 descreve a subetapa *percorrer a árvore-entidade*. Antes delas, a Subseção 3.2.3 descreve as funções utilizadas para calcular a similaridade entre duas páginas, as quais são utilizadas nas subetapas.

3.2.3 Funções de similaridade entre páginas

Nesta subseção, as funções que medem a similaridade entre duas páginas são definidas. Esta Tese propõe uma função baseada na URL, denominada $ssim_{URL}$, e utiliza uma função de similaridade baseada em HTML. Essa função, denominada sim_{HTML}, foi proposta por Blanco, Crescenzi e Merialdo (2005).

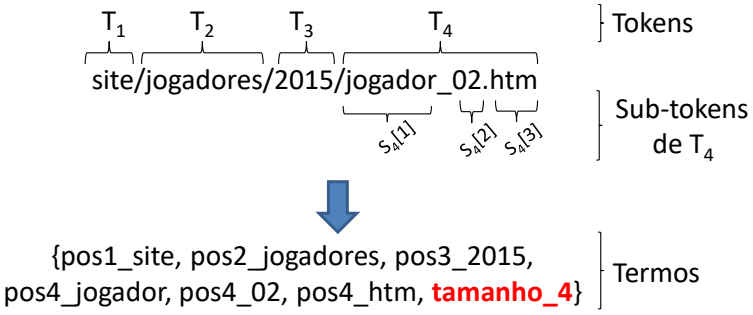
3.2.3.1 Função $ssim_{URL}$

A função de similaridade $ssim_{URL}$ compara duas páginas com base nos termos que pertencem a suas URLs. Essa função atribui um peso diferente para cada termo de URL de acordo com sua capacidade de distinguir páginas-entidade do mesmo tipo de outras páginas. A função $ssim_{URL}$ é utilizada tanto na subetapa *identificar o caminho de exemplo* quanto na subetapa *percorrer a árvore-entidade*. A função $ssim_{URL}$ é descrita na seguinte ordem: (i) como os termos são extraídos das URLs; (ii) como os termos de URL são ponderados; e (iii) como os pesos dos termos de URL são usados para calcular a similaridade entre duas páginas.

Extração de Termos. A URL é tokenizada para extrair os termos. Uma URL é uma sequência de substrings (denominadas `tokens`) separadas pelos caracteres "/", "?" ou "&". Cada *token* é um conjunto de substrings (`sub-tokens`) separados por símbolos, caracteres especiais, mudança de letra para dígito e vice-versa. Um termo de URL é um *sub-token* associado com a posição do *token* que o contém. O termo de URL é formalizado a seguir.

Definição 2 *(`Termo de URL`) Considere $T = \{T_1, ..., T_n\}$ sendo uma sequência de tokens de uma URL u, em que T_i ocorre em u antes de T_{i+1}. Cada token $T_i = \{s_i[1], ..., s_i[m]\}$ é um conjunto de sub-tokens em que cada $s_i[j]$ é um sub-token do token T_i. Cada sub-token $s_i[j]$ associado com i é um termo de URL. Um termo de URL adicional é o tamanho de T.*

Figura 3.5 – Exemplo de *tokens*, *sub-tokens* e termos de URL.



Por exemplo, a Figura 3.5 ilustra uma URL. Os *tokens*, os *sub-tokens* do *token* T_4 e os termos de URL são apresentados. O termo de URL adicional está destacado em vermelho.

Ponderação de Termos. O peso de um termo de URL é baseado em duas observações sobre páginas-entidade do mesmo tipo: (i) elas normalmente compartilham uma estrutura de URL comum; e (ii) elas normalmente possuem o mesmo *index-path*. A ponderação assume que os termos com a maior capacidade de discriminação são aqueles que ocorrem: **(i) na URL de muitas páginas com o mesmo *index-path* que a página de exemplo; e (ii) ocorrem na URL de poucas páginas com *index-path* diferente da página de exemplo**. O peso de um termo de URL t é definido como segue:

$$W_t(ip, sp) = TF(t, ip, sp) \times log(ILF(t, ip, sp))$$

em que t é um termo de URL; sp é uma página de exemplo; ip é a página índice da página de exemplo; $TF(t, ip, sp)$ é o número de páginas apontadas por ip por meio do mesmo *link-path* que sp e que contêm o termo de URL t; e $ILF(t, ip, sp)$ é o número de *link-paths* em ip dividido pelo número de *link-paths* em ip que apontam para alguma página com o termo de URL t.

Por exemplo, considerando o site apresentado na Figura 3.3 e a página $3/2$ como página de exemplo. Os termos "*pos2_jogadores*" e "*pos4_jogador*" devem receber o maior peso porque: (i) eles ocorrem na URL de três páginas (páginas $3/1$, $3/2$, $3/3$) apontadas pela página-índice (página $2/0$) por meio do mesmo *link-path* ("*HTML-DIV-A*") que a página de exemplo; e (ii) eles não ocorrem na URL de páginas apontadas pela página-índice por meio de outros *link-paths*. Os termos "*site*" e "*htm*" tem peso zero, pois ocorrem na URL de todas as páginas apontadas pela página-índice.

Cálculo da similaridade. A função $ssim_{URL}$ acumula o peso dos termos de URL que são compartilhados entre a página de exemplo e a `página-alvo` (uma página que é comparada com a página de exemplo). A função $ssim_{URL}$ é definida como segue:

$$ssim_{url}(ip, p_1, p_2) = \frac{\sum_{t \in \tau^1 \cap \tau^2} W_t(ip,p_1)}{\sum_{t \in \tau^1} W_t(ip,p_1)}$$

em que p_1 é uma página de exemplo; ip é a página-índice da página p_1; p_2 é uma página-alvo; τ^i denota o conjunto de termos de URL de uma página p_i; e $W_t(ip, p_1)$ é o peso do termo t.

Por exemplo, considerando o site apresentado na Figura 3.3 e a página $3/2$ como página de exemplo. A $ssim_{URL}$ entre as páginas $3/2$ e $3/1$ é 0,94 porque essas páginas compartilham os termos "*pos1_site*", "*pos2_jogadores*", "*pos3_2015*", "*pos4_jogador*", "*pos4_htm*" e "*tamanho_4*". A $ssim_{URL}$ entre as páginas $3/2$ e $3/7$ é 0,62 porque essas páginas compartilham os termos "*pos1_site*", "*pos3_2015*", "*pos4_02*", "*pos4_htm*" e "*tamanho_4*".

3.2.3.2 Função sim_{HTML}

A função de similaridade sim_{HTML}, proposta por Blanco, Crescenzi e Merialdo (2005), compara duas páginas com base em seus *link-paths*. Essa função dá a mesma importância para todos os *link-paths*. A função sim_{HTML} é utilizada na subetapa *percorrer a árvore-entidade* e é definida como segue:

$$sim_{html}(p_1, p_2) = \frac{|\chi^1 \cap \chi^2|}{|\chi^1 \cup \chi^2|}$$

em que $\chi^i = \{\chi^i_1, ..., \chi^i_n\}$ é o conjunto dos *link-paths* de uma página p_i.

É importante destacar que as funções $ssim_{URL}$ e sim_{HTML} usam o conceito de *link-path* em contextos diferentes. Na função $ssim_{URL}$, o *link-path* de uma página p é o *link-path* por meio do qual a página-índice aponta para a página p. Na função sim_{HTML}, o conjunto de *link-paths* de uma página p contém os *link-paths* presentes no HTML da página p.

3.2.4 Identificar o caminho de exemplo

O objetivo dessa subetapa é encontrar o caminho de exemplo, ou seja, o caminho da raiz até a página de exemplo na árvore-entidade. O caminho de exemplo é obtido na ordem inversa. **Orion** inicia com a página de exemplo, encontra a sua página-índice, descobre a página-índice da página-índice, e assim por diante até atingir a raiz da árvore-entidade.

Orion identifica que atingiu a raiz da árvore-entidade por meio de um parâmetro que estima a altura da árvore-entidade. Se esse parâmetro for menor que a altura real da árvore-entidade, algumas páginas-entidade podem não ser encontradas. Se esse parâmetro for maior que a altura real da árvore-entidade, algumas páginas desnecessárias são baixadas, mas a eficácia dos resultados não é afetada uma vez que as páginas-ruído são filtradas na subetapa *percorrer a árvore-entidade*. **Orion** tem um critério de parada adicional: quando a página-índice selecionada já está inserida na pilha que representa o caminho de exemplo.

Orion encontra a página-índice de uma determinada página com base na seguinte premissa: **a página-índice de uma página gp é a página que aponta para o maior número de páginas mais similares à página gp.** Esse número é representado pela função $index\_score$, que acumula a similaridade de URL entre as páginas apontadas pela possível página-índice e a página gp. A função $index\_score$ é definida como segue:

$$index\_score(cp, gp) = \sum_{c_i \in C} ssim_{url}(cp, gp, c_i)$$

em que cp é uma possível página-índice da página gp, $C = \{c_1, ..., c_n\}$ é o conjunto de páginas apontadas por cp por meio do mesmo *link-path* que gp e $ssim_{url}(cp, gp, c_i)$ é o escore de similaridade de URL entre c_i e gp.

Esta Tese define o Algoritmo 1 - **index-finder** - para encontrar as páginas-índice. O Algoritmo 1 recebe uma página como entrada, denominada página-alvo. A saída é a página-índice da página-alvo. O algoritmo coleta as páginas apontadas pela página-alvo (Linha 1) e as páginas apontadas pelas páginas coletadas (linhas 2-4). As páginas coletadas que não apontam para a página-alvo são removidas (Linha 5). Como saída, o algoritmo retorna a página coletada com o maior $index\_score$ (Linha 6). As linhas 2-4 foram adicionadas depois de constatar que algumas páginas-alvo não possuem um link direto para suas páginas-índice.

ALGORITHM 1: Algoritmo index-finder

Input: uma página-alvo (gp).
Output: a página-índice de gp.

1 P = A = obtémLinks(gp);
2 **for *each*** $a_i \in A$ **do**
3 adiciona obtémLinks(a_i) para P;
4 **end**
5 remove cada $p_i \in P$ em que p_i não aponta para gp;
6 **return** $p_i \in P$ com o maior $index\_score(p_i, gp)$;

3.2.5 Percorrer a árvore-entidade

O objetivo da subetapa *percorrer a árvore-entidade* é encontrar as páginas-entidade que pertencem à árvore-entidade cujo caminho de exemplo foi encontrado na subetapa anterior (*identificar o caminho de exemplo*). Essas páginas são obtidas percorrendo a árvore-entidade da raiz até as folhas. Em cada iteração, as páginas apontadas pelas páginas resultantes da iteração anterior (ou pela raiz, se for a primeira iteração) são obtidas e filtradas. As páginas são filtradas de acordo com a sua similaridade de HTML e de URL com a página que está no topo da pilha (ou seja, a página do caminho de exemplo que está no mesmo nível na árvore-entidade que as páginas que estão sendo analisadas).

Esta Tese define o Algoritmo 2 - **filter** - para filtrar as páginas obtidas em uma determinada iteração. A entrada é composta pelo conjunto de páginas obtidas na iteração corrente e uma página de exemplo (página que está no topo da pilha). A saída é composta pelo conjunto das páginas que são similares à página de exemplo.

ALGORITHM 2: Algoritmo filter

Input: um conjunto de páginas (L), uma página de exemplo (sp).
Output: o conjunto das páginas em L que são similares à sp.

1 adiciona sp para rs;
2 remove de L cada página l_i em que index-path(l_i) <> index-path(sp)
3 cria uma lista de grupos $G = \{G_1, ..., G_n\}$, onde cada grupo G_j contém cada página $l_i \in L$ (exceto sp) com o mesmo $ssim_{url}$ entre l_i e sp;
4 ordena G em ordem decrescente de $ssim_{url}$;
5 adiciona todas as páginas de G_1 para rs;
6 $limiar_{html} = \min_{l_i \in G_1} sim_{html}(sp, l_i)$;
7 remove G_1 de G ;
8 **for** ***each*** G_j ***IN*** G **do**
9 **if** $\max_{l_i \in G_j} sim_{html}(sp, l_i) >= limiar_{html}$ **then**
10 adiciona todas as páginas de G_j para rs;
11 **else**
12 break;
13 **end**
14 **end**
15 **return** rs ;

Tabela 3.1 – Exemplo de execução do Algoritmo filter.

ID	Index-path	$ssim_{URL}$	sim_{HTML}
3/0	HTML-NAV-A	-	-
3/1	HTML-DIV-A	0.94	0.85
3/2	**HTML-DIV-A**	**1.00**	**1.00**
3/3	HTML-DIV-A	0.94	0.75
3/4	HTML-DIV-A	0.75	0.80
3/5	HTML-DIV-A	0.75	0.82
3/6	HTML-DIV-A	0.56	0.45
3/7	HTML-DIV-A	0.62	0.40
3/8	HTML-NAV-A	-	-

(a)

Linha(s) no Algoritmo	L	G	$limiar_{html}$	rs
Inicialização	{3/0, 3/1, 3/2, 3/3, 3/4, 3/5, 3/6, 3/7, 3/8}	{}	-	{}
01	{3/0, 3/1, 3/2, 3/3, 3/4, 3/5, 3/6, 3/7, 3/8}	{}	-	**{3/2}**
02	**{3/1, 3/2, 3/3, 3/4, 3/5, 3/6, 3/7}**	{}	-	{3/2}
03-04	-	**G_1 {{3/1, 3/3}, G_2 {3/4, 3/5}, G_3 {3/7}, G_4 {3/6}}**	-	{3/2}
05	-	{{3/1, 3/3}, {3/4, 3/5}, {3/7}, {3/6}}	-	{3/2, **3/1, 3/3**}
06	-	{{3/1, 3/3}, {3/4, 3/5}, {3/7}, {3/6}}	**0.75**	{3/2, 3/1, 3/3}
07	-	**G_2 G_3 G_4 {{3/4, 3/5}, {3/7}, {3/6}}**	0.75	{3/2, 3/1, 3/3}
08-14 (1ª iteração)	-	{{3/4, 3/5}, {3/7}, {3/6}}	0.75	{3/2, 3/1, 3/3, **3/4, 3/5**}
08-14 (2ª iteração)	-	{{3/4, 3/5}, {3/7}, {3/6}}	0.75	{3/2, 3/1, 3/3, 3/4, 3/5}

(b)

Nota: (a) entrada; (b) passo a passo do algoritmo.

O Algoritmo 2 é descrito passo a passo por meio de um exemplo que utiliza as tabelas 3.1(a) e 3.1(b). A Tabela 3.1(a) descreve a entrada do exemplo que é composta por 9 páginas. Uma dessas páginas (destacada em negrito) é a página de exemplo, que foi escolhida aleatoriamente. As seguintes informações das páginas são fornecidas: (i) ID - é o identificador

da página; (ii) index-path - é o *link-path* por meio do qual a página-índice aponta para a referida página; (iii) $ssim_{URL}$ - é a similaridade de URL entre a página descrita na linha e a página de exemplo; e (iv) sim_{HTML} - é a similaridade de HTML entre a página descrita na linha e a página de exemplo. A Tabela 3.1(b) apresenta o valor de quatro variáveis (L, G, $limiar_{HTML}$ e rs) à medida que o Algoritmo 2 é executado. A primeira coluna da Tabela 3.1(b) - *Linha(s) no Algoritmo* - indica as linhas do Algoritmo 2 que estão sendo executadas.

Na linha 01 do Algoritmo 2 (acompanhe por meio da primeira coluna da Tabela 3.1(b)), a página de exemplo ($3/2$) é adicionada ao conjunto rs. Na linha 02, as páginas $3/0$ e $3/8$ são removidas da lista L, pois possuem *index-path* diferente da página de exemplo. Na Linha 03, quatro grupos são criados (G_1, G_2, G_3 e G_4). Cada grupo contém as páginas da lista L (exceto a página de exemplo) com o mesmo escore de similaridade de URL (calculado utilizando a função $ssim_{URL}$). Na Linha 04, os grupos são ordenados em ordem decrescente de escore de similaridade de URL. O Grupo G_1 tem as páginas com o maior escore. O Grupo G_4 tem as páginas com o menor escore. Na Linha 05, todas as páginas do Grupo G_1 ($3/1$ e $3/3$) são adicionadas ao conjunto rs. Na Linha 06, o menor escore de similaridade de HTML (calculado utilizando a função sim_{HTML}) entre a página de exemplo e as páginas do Grupo G_1 é definido como o limiar de similaridade de HTML ($limiar_{HTML}$). Nesse exemplo, o $limiar_{HTML}$ é o escore de similaridade de HTML entre a página de exemplo e a página $3/3$ (0,75). Na Linha 07, o Grupo G_1 é removido da lista de grupos G. A primeira iteração do "for" (linhas 8-14) analisa o Grupo G_2 uma vez que esse grupo passou a ter o maior escore de similaridade de URL depois que o Grupo G_1 foi removido. Todas as páginas do Grupo G_2 são adicionadas ao conjunto rs porque há pelo menos uma página nesse grupo que satisfaz o limiar de similaridade de HTML. A segunda iteração "for" (linhas 8-14) analisa o Grupo G_3. Todas as páginas do Grupo G_3 são descartadas porque nenhuma página desse grupo satisfaz o limiar de similaridade de HTML. Todos os demais grupos (G_4) também são descartados porque o escore de similaridade de URL do Grupo G_3 (0,62) representa o limiar de similaridade de URL ($limiar_{URL}$) de acordo com a seguinte heurística: **páginas-entidade do mesmo tipo tem similaridade de URL e de HTML maiores entre si do que com outras páginas**. Finalmente, o conjunto rs (com as páginas $3/1$, $3/2$, $3/3$, $3/4$, $3/5$) é retornado na Linha 15.

3.3 Extração

A etapa de *Extração* extrai os valores dos atributos publicados nas páginas-entidade de cada site. Na abordagem **Orion**, o problema de extrair os valores dos atributos é modelado

como consultas em um banco de dados orientado a grafos. Essa opção de modelagem foi adotada porque as páginas Web possuem estrutura de árvore DOM, logo podem ser diretamente tratadas como grafos. Além disso, os bancos de dados orientados a grafos possuem distribuição de índices, uso de memória e algoritmos de grafos eficientes. Um conjunto de consultas sobre um banco de dados orientado a grafos é gerado de modo que cada consulta extrai os valores de um atributo nas páginas-entidade baseadas em *template* do site. As consultas sobre o banco de dados orientado a grafos são expressas na linguagem Cypher[4]. Cypher foi escolhida porque é uma linguagem declarativa para grafos que permite consultas e atualizações expressivas e eficientes. Cypher é mais robusta que XPath (linguagem que é comumente utilizada para manipular páginas Web) porque permite percorrer, consultar e atualizar o grafo, enquanto XPath é uma linguagem específica para percorrer um documento XML ou equivalente.

A tarefa de extrair os valores dos atributos envolve diversos desafios, tais como:

- os formatos de apresentação e o conjunto de atributos que formam uma entidade estão sujeitos a variações (CHANG et al., 2006). A estratégia adotada por **Orion** não é afetada por atributos omissos (atributos cujos valores são publicados apenas em uma fração das páginas-entidade do site) ou atributos em múltiplas ordens (os valores dos atributos são publicados em ordens diferentes nas páginas-entidade do mesmo site) quando esses atributos são publicados em elementos simples (elementos com apenas um nodo textual);
- existem inúmeras formas distintas de publicar os valores de atributos nas páginas-entidade (tabelas horizontais e verticais, listas horizontais e verticais, marcações DIV e SPAN, etc.). **Orion** adota uma estratégia que suporta qualquer tipo de marcação HTML;
- a etapa de *Descoberta* não é totalmente precisa porque algumas páginas-ruído, em determinados sites, possuem características de HTML e de URL similares às páginas-entidade, então o conjunto de páginas-entidade de um site fornecido como entrada para a etapa de *Extração* pode conter páginas-ruído. A modelagem adotada não é afetada pela presença de páginas-ruído no conjunto de páginas-entidade fornecido como entrada porque não requer que as consultas extraiam valores em todas as páginas.

O restante desta seção está organizado da seguinte forma. A Subseção 3.3.1 define os conceitos necessários para a compreensão da etapa de *Extração*. A Subseção 3.3.2 apresenta uma visão geral da etapa. As demais subseções detalham as subetapas dessa etapa.

[4]Disponível em: <https://neo4j.com/developer/cypher-query-language/>. Último acesso em: 10/10/2017.

3.3.1 Definições preliminares

Esta Tese modela o problema de extrair os valores dos atributos como consultas em um banco de dados orientado a grafos. Esse tipo de bancos de dados modela os dados como uma estrutura de rede contendo nodos e arestas (KAUR; RANI, 2013). As consultas no banco de dados orientado a grafos são expressas, nesta Tese, usando Cypher (Neo Technology, 2016), que é uma linguagem declarativa para grafos que permite consultas e atualizações eficientes.

O conjunto de páginas-entidade baseadas em *template* de um site é representado como um grafo desconexo e acíclico (floresta), denominado `S-graph`. O S-graph contém um conjunto de árvores, uma para cada página-entidade. A árvore de uma página-entidade é uma versão modificada da árvore DOM. Essa versão modificada difere-se da árvore DOM porque ela inclui apenas os nodos de marcações (*tag nodes*) e os nodos textuais, assim como, ela armazena as seguintes propriedades de cada nodo: (i) *value* - o nome de um nodo de marcação ou o texto de um nodo textual; (ii) *url* - a URL da página-entidade; (iii) *type* - a literal "*TAG*" para nodos de marcação e a literal "*TEXT*" para nodos textuais; (iv) *position* - um número que representa a posição do nodo com relação a seus irmãos; e (v) *function* - a literal "*TEMPLATE*" para nodos textuais que fazem parte do *template* das páginas-entidade, a literal "*DATA*" para os demais nodos textuais e a ausência dessa propriedade para nodos de marcação. Um `nodo template` (*function="TEMPLATE"*) é um nodo textual com um valor que ocorre em, pelo menos, δ páginas-entidade. Os demais nodos textuais são denominados `nodos de dados` (*function="DATA"*). O S-graph é formalizado a seguir.

Definição 3 *Um `S-graph` é um par (N, E), em que N é um conjunto de nodos e E é um conjunto de relacionamentos (arestas) entre dois nodos. Há um nodo no S-graph para cada nodo de marcação ou nodo textual de cada página-entidade baseada em template. Há um relacionamento entre dois nodos (n_x e n_y) no S-graph se n_y é filho de n_x na árvore DOM da página-entidade. Cada nodo tem as seguintes propriedades: value, url, type, position e function (exclusiva para nodos textuais).*

Por exemplo, a Figura 3.6 apresenta uma página-entidade real do site FCBarcelona.com[5]. Essa página-entidade publica alguns atributos (*nome*, *posição*, *número*, entre outros) de uma entidade do tipo *jogador de futebol*. A Figura 3.7 mostra o HTML da página-entidade apresentada na Figura 3.6. A Figura 3.8 ilustra a representação visual do S-graph formado pelas

[5]Disponível em: <https://www.fcbarcelona.com/football/first-team/staff/players/2016-2017/messi/>. Último acesso em: 10/07/2017.

Figura 3.6 – Uma página-entidade do site FCBarcelona.com.

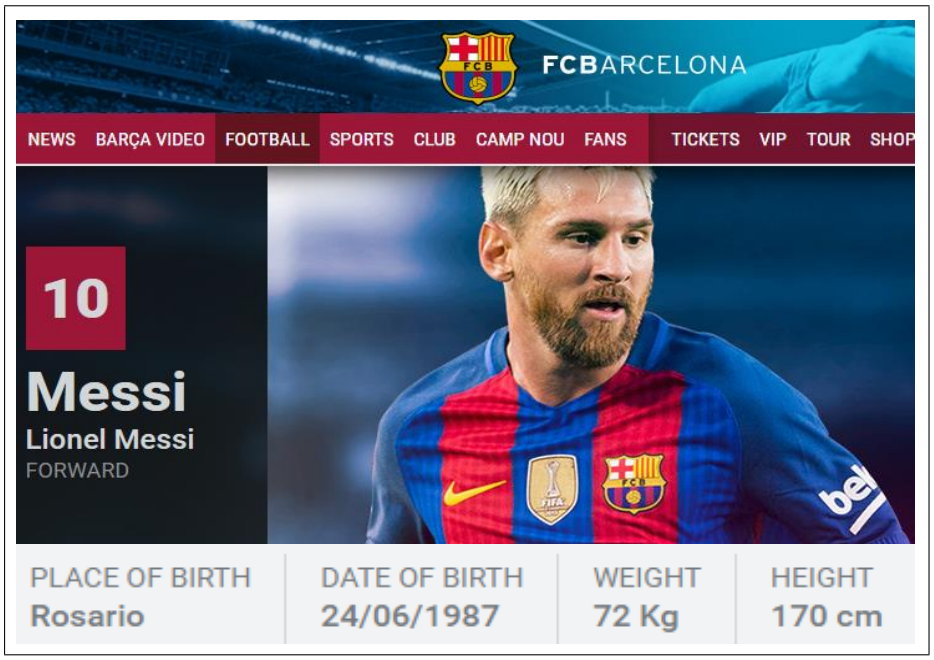
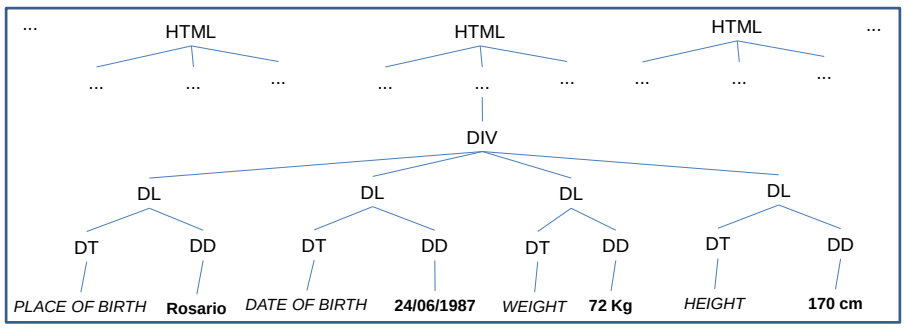


Figura 3.7 – HTML da página-entidade apresentada na Figura 3.6.

```
<HTML>
  …                                                    611     612
    <DIV>
        <DL><DT>PLACE OF BIRTH</DT><DD>Rosario</DD></DL>
        <DL><DT>DATE OF BIRTH</DT><DD>24/06/1987</DD></DL>
        <DL><DT>WEIGHT</DT><DD>72 Kg</DD></DL>
        <DL><DT>HEIGHT</DT><DD>170 cm</DD></DL>
    </DIV>
  ...
</HTML>
```

Figura 3.8 – S-graph do site FCBarcelona.com.



páginas-entidade sobre jogadores que estão publicadas no site FCBarcelona.com. Os nodos de dados estão destacados em negrito e os nodos *template* estão destacados em itálico.

Tabela 3.2 – Valores dos atributos publicados na página apresentada na Figura 3.6.

Atributo	Valor
Nome	Lionel Messi
Posição	FORWARD
Número	10
Local de nascimento	Rosario
Data de nascimento	24/06/1987
Peso	72 Kg
Altura	170 cm

Em extração de dados, um `wrapper` é um programa que extrai dados estruturados de uma coleção de páginas Web (ORTONA et al., 2016). A partir de uma coleção de páginas P, um *wrapper* produz uma relação R com esquema Σ. Uma `relação` R é um conjunto de n-tuplas $\{U_1, .., U_n\}$, em que n é o número de tuplas. Um `esquema` $\Sigma = \{A_1, .., A_m\}$ é uma tupla que representa os atributos de uma relação, em que m é o número de atributos. No contexto desta Tese, as páginas que pertencem à P são páginas-entidade de um site que seguem o mesmo *template*. O *wrapper* é modelado por meio de consultas Cypher sobre o S-graph e definido como:

Definição 4 *Um `Cypher wrapper` W é um conjunto de consultas $\{Q_1, .., Q_m\}$, em que cada Q_i é uma consulta Cypher sobre o S-graph que extrai os valores do atributo A_i nas páginas-entidade que estão no S-graph. A aplicação de um Cypher wrapper significa executar as consultas sobre o S-graph e associar os valores retornados pelas consultas aos atributos correspondentes. A aplicação de um Cypher wrapper W em uma coleção de páginas-entidade P produz uma relação com esquema $\{A_1, .., A_m\}$.*

Por exemplo, a aplicação de um Cypher *wrapper* na página-entidade ilustrada na Figura 3.6 retorna a tupla apresentada na Tabela 3.2. São extraídos os valores dos seguintes atributos: *nome*, *posição*, *número*, *local de nascimento*, *data de nascimento*, *peso* e *altura*.

As consultas Cypher, utilizadas nesta Tese, especificam um caminho no S-graph entre um nodo *template* e um nodo de dados. Essas consultas filtram: (i) a propriedade *position* de todos os nodos do caminho; (ii) a propriedade *type* de todos os nodos do caminho; e (iii) a propriedade *value* de todos os nodos do caminho, exceto o nodo de dados porque o valor da propriedade *value* do nodo de dados é o que se deseja extrair. As consultas Cypher retornam o valor das propriedades *value* e *url* do nodo de dados. A Gramática 3.1 apresenta a sintaxe das

⟨*Cypher query*⟩ ::= **MATCH** ⟨*template node*⟩ {'–''–'⟨*internal node*⟩}+'–''–'⟨*data node*⟩
WHERE ⟨*template node*⟩'.'**value** '=' ⟨*node value*⟩
AND ⟨*template node*⟩'.'**type** '=' 'TEXT'
AND ⟨*template node*⟩'.'**position** '=' ⟨*position value*⟩
{**AND** ⟨*internal node*⟩'.'**value** '=' ⟨*node value*⟩
AND ⟨*internal node*⟩'.'**type** '=' 'TAG'
AND ⟨*internal node*⟩'.'**position** '=' ⟨*position value*⟩ }+
AND ⟨*data node*⟩'.'**type** '=' 'TEXT'
AND ⟨*data node*⟩'.'**position** '=' ⟨*position value*⟩
RETURN ⟨*data node*⟩'.'**url**, ⟨*data node*⟩'.'**value**

⟨*template node*⟩ ::= ⟨*node identifier*⟩

⟨*internal node*⟩ ::= ⟨*node identifier*⟩

⟨*data node*⟩ ::= ⟨*node identifier*⟩

⟨*node value*⟩ ::= ⟨*string value*⟩

⟨*position value*⟩ ::= ⟨*integer number*⟩

⟨*node identifier*⟩ ::= '(' ⟨*letter*⟩ { ⟨*letter*⟩ | ⟨*digit*⟩ | '_'}* ')'

⟨*string value*⟩ ::= '"' { any character including blank } '"'

⟨*integer number*⟩ ::= {⟨*digit*⟩}+

⟨*digit*⟩ ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

⟨*letter*⟩ ::= A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z

Gramática 3.1 – BNF das consultas Cypher.

consultas Cypher utilizadas nesta Tese. Essa sintaxe é expressa por meio de uma BNF (*Backus Naur Form* ou *Backus Normal Form*) simplificada, utilizada para expressar gramáticas livres de contexto. A notação utilizada é a seguinte, sem recursão à esquerda:

- $< >$ para delimitar as metavariáveis;
- $|$ para separar duas alternativas;
- $\{\ \}*$ para itens repetitivos (zero ou mais vezes);
- $\{\ \}+$ para itens repetitivos (uma ou mais vezes);
- os símbolos são delimitados por um par de aspas simples;
- símbolos terminais e não terminais são diferenciados em negrito pela apresentação dos terminais;
- a precedência dos operadores é a seguinte - opcional, repetição, sequência e alternativas.

Por exemplo, a Figura 3.9 apresenta a consulta Cypher que extrai o valor do atributo *altura* no S-graph apresentado na Figura 3.8. Essa consulta Cypher especifica um caminho no S-graph, que: (i) inicia em um nodo *template* com *value="HEIGHT"*, *type="TEXT"* e *position=1*; (ii) segue para um nodo com *value="DT"*, *type="TAG"* e *position=1*; (iii) segue para um nodo com *value="DL"*, *type="TAG"* e *position=4*; (iv) segue para um nodo com *value="DD"*, *type="TAG"* e *position=2*; e (v) termina em um nodo com *type="TEXT"* e *position=1*. Essa consulta Cypher retorna o valor das propriedades *value* e *url* do nodo de dados.

Figura 3.9 – Um exemplo de consulta Cypher.

```
MATCH (n1)--(n2)--(n3)--(n4)--(n5)
WHERE n1.type="TEXT"  AND n1.position=1 AND n1.value="HEIGHT"
   AND n2.type="TAG"   AND n2.position=1 AND n2.value="DT"
   AND n3.type="TAG"   AND n3.position=4 AND n3.value="DL"
   AND n4.type="TAG"   AND n4.position=2 AND n4.value="DD"
   AND n5.type="TEXT"  AND n5.position=1
RETURN n5.url, n5.value
```

A etapa de *Extração* tem como foco a tarefa de inferir um Cypher *wrapper* a partir de uma coleção de páginas-entidade baseadas em *template* de um site. Essa inferência é realizada conforme descrito nas próximas subseções.

3.3.2 Visão geral

A etapa de *Extração* processa cada site de forma isolada[6]. As páginas-entidades coletadas na etapa anterior (*Descoberta*) são analisadas para extrair os valores dos atributos publicados nessas páginas. Para cada site, a abordagem **Orion** infere consultas Cypher sobre o S-graph (construído a partir das páginas-entidade do site). Cada consulta extrai os valores de um atributo nas páginas-entidade do site.

A Figura 3.10 apresenta o fluxo de execução da etapa de *Extração* para um site. A entrada é um conjunto de páginas-entidade baseadas em *template* de um site. A saída é o conjunto de valores de atributos publicados nessas páginas. Essa etapa possui cinco subetapas: (1) *construção do S-Graph* constrói um S-graph a partir das páginas-entidade fornecidas como entrada; (2) *reconhecimento do template* identifica os nodos no S-graph que são parte do *template*

[6]Exceto uma das alternativas da subetapa de *seleção de consultas* que processa os sites de forma conjunta.

Figura 3.10 – Fluxo de execução da etapa de *Extração*.

das páginas-entidade; (3) *inferência das consultas* infere consultas Cypher a partir dos nodos *template*; (4) *seleção das consultas* seleciona uma consulta Cypher para cada atributo de um esquema Σ, ou seja, constrói o Cypher *wrapper*; (5) *execução das consultas* extrai os valores dos atributos publicados nas páginas-entidade executando as consultas Cypher (Cypher *wrapper*) sobre o S-graph. Por exemplo, a Tabela 3.2 apresenta os valores dos atributos extraídos da página-entidade apresentada na Figura 3.6. As próximas subseções descrevem cada subetapa.

3.3.3 Construção do S-Graph

A subetapa de *construção do S-graph* analisa o HTML das páginas-entidade fornecidas como entrada para a etapa de *Extração* da abordagem **Orion** e constrói um S-graph, sem a definição da propriedade *function* para os nodos, que é adicionada na próxima subetapa (*reconhecimento do template*). O S-graph é construído usando a linguagem Cypher. Por exemplo, a Figura 3.11 apresenta as instruções Cypher que inserem no S-graph: (a) um nodo representando a marcação identificada na Figura 3.7 pelo identificador 611; (b) um nodo representando o conteúdo textual identificado na Figura 3.7 pelo identificador 612; e (c) um relacionamento representando que o nodo 612 é filho do nodo 611[7].

Figura 3.11 – Exemplos de instruções Cypher para criar um S-graph.

```
(a)  CREATE (n {value: "DD", url: "http://... ", type: "TAG", position:2})  RETURN ID(n)        //611
(b)  CREATE (n {value: "Rosario", url: "http://...", type: "TEXT", position:1}) RETURN ID(n)  //612
(c)  MATCH (a), (b) WHERE ID(a) = 611 AND ID(b) = 612 CREATE (a) - [r:contains] -> (b)
```

Por exemplo, se a entrada for composta pelas páginas-entidade sobre jogadores do site FCBarcelona.com, a subetapa de *construção do S-graph* constrói o S-graph ilustrado na Figura 3.8. Entretanto, os nodos textuais não são classificados como *template* (itálico) ou dados (negrito) nessa subetapa, uma vez que a propriedade *function* não está definida ainda.

[7] **Orion** não usa a direção e o rótulo dos relacionamentos, mas a linguagem Cypher requer tal informação para a criação dos relacionamentos.

3.3.4 Reconhecimento do template

A subetapa de *reconhecimento do template* avalia os nodos do S-graph e encontra aqueles que são parte do *template* das páginas-entidade. Esses nodos são denominados nodos *template*. Um nodo *template* é um nodo textual com um valor que ocorre em, pelo menos, δ páginas-entidade[8]. Os demais nodos textuais são denominados nodos de dados. A Figura 3.12(a) apresenta a instrução Cypher que seleciona os nodos *template* e define a propriedade *function* desses nodos. A Linha 1 seleciona todos os nodos textuais. A Linha 2 obtém os valores da propriedade *value* (dos nodos textuais) que ocorrem em, pelo menos, δ páginas-entidade distintas. A Linha 3 adiciona os valores obtidos por meio da Linha 2 em uma lista L. A Linha 4 seleciona todos os nodos textuais cujo valor da propriedade *value* pertence à lista L. A Linha 5 atribui a literal "*TEMPLATE*" para a propriedade *function* dos nodos obtidos na Linha 4. A Figura 3.12(b) apresenta a instrução Cypher que seleciona os nodos de dados e define a propriedade *function* desses nodos. Essa instrução deve ser executada após a instrução que define os nodos *template*. Na Figura 3.12(b), a Linha 1 seleciona os nodos textuais que não possuem a propriedade *function* (nesse momento, todos e apenas os nodos *template* possuem essa propriedade). A Linha 2 atribui a literal "*DATA*" para a propriedade *function* dos nodos obtidos na Linha 1.

Figura 3.12 – Instruções Cypher que definem a propriedade *function*.

(a)

```
1. MATCH (a) WHERE a.type= "TEXT"
2. WITH a.value AS v, COUNT(DISTINCT a.url) AS qt WHERE qt >= δ
3. WITH COLLECT(v) AS L
4. MATCH (b) WHERE b.type="TEXT" AND ANY(x IN L WHERE x=b.value)
5. SET b.function= "TEMPLATE"
```

(b)

```
1. MATCH (a) WHERE a.type= "TEXT" AND a.function IS NULL
2. SET a.function= "DATA"
```

Por exemplo, se o conjunto de entrada for composto pelas páginas-entidade sobre jogadores do site FCBarcelona.com, a subetapa de *reconhecimento do template* classifica os nodos textuais do S-graph (ilustrado na Figura 3.8) como *template* (destacado em itálico) ou dados (destacado em negrito). Ao final dessa subetapa, o S-graph está completo.

[8] O valor de δ é definido no Capítulo 4.

3.3.5 Inferência das consultas

Essa subetapa infere as consultas Cypher que permitem extrair os valores dos atributos publicados nas páginas-entidade. Para realizar essa inferência dois algoritmos foram definidos.

O Algoritmo 3 - Query Induction (Inferência de Consultas) - tem como objetivo inferir as consultas Cypher que extraem os valores dos atributos nas páginas-entidade do site. A entrada é o S-graph de um site. A saída é composta por uma lista W de consultas Cypher, que extraem dados nas páginas-entidade do site. A Linha 1 inicializa a lista W. O laço nas linhas 2-7 percorre cada nodo de dados (n_{data}) do S-graph. A Linha 3 realiza uma busca em largura para encontrar o nodo *template* mais próximo ($n_{template}$) do nodo n_{data}. A distância é o número de nodos entre o nodo *template* e o nodo de dados. A Linha 4 obtém o caminho do nodo $n_{template}$ até o nodo n_{data} no S-graph. A Linha 5 utiliza o Algoritmo 4 - **path2query** - para construir uma consulta Cypher (Q_i) a partir do caminho obtido na linha anterior. A Linha 6 adiciona a consulta Q_i à lista W. A Linha 8 remove as consultas de W que não extraem valores em, pelo menos, γ páginas-entidade diferentes. Finalmente, a Linha 9 retorna a lista W.

ALGORITHM 3: Query Induction - Inferência de Consultas

Input: S_{graph}

Output: W

1 $W = \emptyset$;
2 **for** $n_{data} \in S_{graph}$ **do**
3 $n_{template}$ = encontraNodoTemplateMaisPróximo(n_{data}, S_{graph});
4 P = obtémCaminho($n_{template}$, n_{data}, S_{graph});
5 Q_i = path2query(P);
6 adiciona Q_i para W;
7 **end**
8 remove cada $Q_i \in W$ em que nrExtrações(Q_i) < γ ;
9 **return** W;

O Algoritmo 4 - Path2Query (Caminho para Consulta) - tem como objetivo criar uma consulta Cypher a partir de um caminho no S-graph que inicia em um nodo *template* e termina em um nodo de dados. A entrada é um caminho $P = \{P_1, P_2, ..., P_n\}$ representado como um vetor de nodos do S-graph, cujo primeiro elemento é o nodo inicial do caminho, o último elemento é o nodo final do caminho e n é o número de nodos que compõem o caminho. A saída é uma consulta Cypher que extrai dados nas páginas-entidade do site representado pelo S-graph. As linhas 1-2 inicializam as variáveis $match$ e $where$. O laço nas linhas 3-10 percorre os nodos que

compõem o caminho. A Linha 4 adiciona o identificador do nodo corrente na cláusula *MATCH*. As linhas 5 e 6 adicionam condições de seleção sobre as propriedades *type* e *position* do nodo corrente na cláusula *WHERE*. Se o nodo corrente não é o último nodo do caminho (Linhas 7-10), então é adicionado: (i) na cláusula *MATCH*, o símbolo que indica um relacionamento direto entre o nodo corrente e o próximo nodo da iteração (Linha 8); e (ii) na cláusula *WHERE*, uma condição de seleção sobre a propriedade *value* (Linha 9). A Linha 12 adiciona as propriedades *url* e *value* do nodo de dados (último nodo do caminho) na cláusula *RETURN*. A Linha 13 constrói a consulta concatenando as cláusulas *MATCH*, *WHERE* e *RETURN*. Finalmente, a Linha 14 retorna a consulta construída.

ALGORITHM 4: Path2Query - Caminho para consulta

Input: P

Output: Q

```
1  match = 'MATCH ';
2  where = 'WHERE ';
3  for i = 1; i <= tamanho(P); i++ do
4      match += '(n' + i + ')';
5      where += 'n' + i + '.type="' + P[i].getProperty('type') + '" ';
6      where += 'AND n' + i + '.position=' + P[i].getProperty('position') + ' ';
7      if i < tamanho(P) then
8          match += '- -';
9          where += 'AND n' + i + '.value="' + P[i].getProperty('value') + '"AND';
10     end
11 end
12 return = 'RETURN n'+tamanho(P)+'.url, n'+tamanho(P)+'.value';
13 Q = match + ' ' + where + ' ' + return;
14 return Q;
```

Por exemplo, a Figura 3.9 apresenta a consulta Cypher gerada a partir do nodo de dados com *value="170 cm"* (no S-graph ilustrado na Figura 3.8). O nodo *template* mais próximo a esse nodo é o nodo *template* com *value="HEIGHT"* (há três nodos entre eles). Essa consulta extrai os valores do atributo *altura*.

Destaca-se que a subetapa de *inferência de consultas* gera um consulta para cada nodo de dados e remove as consultas que não extraem valores em, pelo menos, γ páginas-entidade diferentes. Algumas consultas não extraem valores de atributos, pois nem todos os nodos de dados são valores de atributo. É necessário selecionar as consultas que extraem valores de atributos. Essa seleção é realizada na próxima subetapa (*seleção de consultas*).

3.3.6 Seleção das consultas

A subetapa de *seleção das consultas* seleciona uma consulta Cypher para cada atributo de um esquema Σ. O esquema Σ é definido por um especialista e representa os atributos de interesse. O conjunto das consultas Cypher selecionadas forma o Cypher *wrapper*. Essa subetapa assume que a subetapa anterior (*inferência das consultas*) gera, para cada atributo, uma consulta Cypher que é capaz de extrair o valor do atributo nas páginas-entidade de cada site. Na prática, inúmeras consultas Cypher são geradas, algumas das quais extraem dados que não são atributos da entidade descrita na página. Por exemplo, propagandas, links, etc.

Cinco alternativas são identificadas para realizar a seleção das consultas:

- um usuário especialista analisa os valores extraídos por cada consulta e seleciona a consulta apropriada para cada atributo. Uma vantagem dessa alternativa é a precisão dos resultados. Uma desvantagem dessa alternativa é a necessidade de um usuário especialista para realizar as anotações;
- colaboradores engajados por meio de uma plataforma de contribuição colaborativa (por exemplo, Amazon Mechanical Turk[9]) analisam os valores extraídos por cada consulta e selecionam a consulta adequada para cada atributo (similar a Qui e Luce (2014)). Seleções incorretas podem ser fornecidas, uma vez que os colaboradores não são especialistas. Uma técnica de voto da maioria (NORDHEIMER et al., 2015) é empregada para encontrar as melhores seleções. Uma vantagem dessa alternativa é não necessitar de usuários especialistas. Entre as desvantagens se destacam o tempo de espera entre a submissão da tarefa e a obtenção da resposta por um número mínimo de colaboradores e o custo financeiro da utilização da plataforma (os colaboradores recebem para executar tarefas em diversas plataformas);
- um algoritmo supervisionado analisa os valores extraídos por cada consulta e seleciona a consulta adequada para cada atributo com base em um conjunto de treinamento (similar a Hao *et al.* (2011)). Um usuário anota alguns sites que descrevem entidades do mesmo tipo. O algoritmo aprende as características dos atributos publicados nos sites anotados. Então, uma consulta gerada a partir de um site não anotado é selecionada para cada atributo com base no características aprendidas. Uma vantagem dessa alternativa é que apenas alguns sites para cada tipo de entidade precisam ser anotados. Entre as desvantagens se destaca o fato que a tarefa de anotar um conjunto de treinamento é normalmente tediosa, demorada e suscetível a erros (HERNÁNDEZ et al., 2016);

[9]Disponível em: <https://www.mturk.com/mturk/welcome>. Último acesso em 10/10/2017.

- um algoritmo não supervisionado analisa os valores extraídos por cada consulta e seleciona a consulta adequada para cada atributo a partir de uma base de conhecimento (similar a Gulhane *et al.* (2010)). O algoritmo compara os valores extraídos por cada consulta com os valores dos atributos presentes em uma base de conhecimento. Essa comparação é realizada por um conjunto de funções de similaridade. Para cada atributo, o algoritmo seleciona a consulta com o maior escore de similaridade desde que esse escore satisfaça um determinado limiar. Uma vantagem dessa alternativa é não depender da anotação do usuário. Uma desvantagem dessa alternativa é a necessidade de uma base de conhecimento para cada tipo de entidade;
- um algoritmo não supervisionado analisa os valores extraídos por cada consulta e seleciona a consulta adequada para cada atributo com base na redundância de conteúdo entre diferentes sites que descrevem entidades do mesmo tipo (similar a Bronzi *et al.* (2013))[10]. O algoritmo compara as consultas geradas a partir dos S-graphs de diferentes sites por meio dos valores extraídos por elas. São comparados apenas os valores extraídos em páginas que descrevem a mesma entidade. Essa comparação é realizada por um conjunto de funções de similaridade. As consultas oriundas de diferentes sites com um escore de similaridade maior que um dado limiar são agrupadas em mapeamentos. Cada mapeamento representa um atributo. Uma vantagem dessa alternativa é não depender da anotação do usuário. Uma desvantagem dessa alternativa é a necessidade dos sites envolvidos na extração compartilharem um conjunto de entidades.

Por exemplo, se o conjunto de entrada for composto pelas páginas-entidade sobre jogadores do site FCBarcelona.com, a subetapa de *seleção de consultas* seleciona a consulta apresentada na Figura 3.9 para o atributo *altura*. Essa consulta é capaz de extrair o valor do atributo *altura* nas páginas-entidade do site.

3.3.7 Execução das consultas

A subetapa de *execução das consultas* aplica o Cypher *wrapper*, que executa todas as consultas Cypher que foram selecionadas na subetapa anterior (*seleção de consultas*) sobre o S-graph para extrair os valores dos atributos publicados nas páginas-entidade do site. As consultas Cypher podem ser executadas sobre o S-graph usado para gerá-las ou sobre outro S-graph construído com páginas-entidade que seguem o mesmo *template*.

[10]Essa é a única alternativa que processa os sites em conjunto.

Por exemplo, a execução da consulta apresentada na Figura 3.9 sobre a página-entidade ilustrada na Figura 3.6 retorna "*170 cm*". A aplicação do Cypher *wrapper* sobre a página-entidade apresentada na Figura 3.6 produz a saída exibida na Tabela 3.2.

3.4 Reforço

A etapa de *Reforço* revisa a extração realizada na etapa anterior (*Extração*) com o objetivo de extrair mais valores de atributos. Uma única consulta Cypher não é capaz de extrair os valores de um atributo em todas as páginas-entidade de um site quando os valores desse atributo são publicados com marcações diferentes. Por exemplo, um site de comércio eletrônico no qual apenas os preços dos produtos em desconto são publicados em negrito e na cor vermelha, e, por isso, apenas esses valores possuem as marcações <*B*> e <*FONT*>. A etapa de *Reforço* identifica automaticamente esses casos e seleciona consultas que são complementares às consultas selecionadas na etapa de *Extração* de modo a extrair os valores dos atributos nas páginas-entidade que as consultas anteriormente selecionadas não foram capazes de extrair. Essa tarefa envolve diversos desafios:

- determinar se duas consultas extraem valores do mesmo atributo;
- distinguir atributos cujos valores são publicados no site com diferentes marcações de atributos cujos valores são publicados apenas em algumas páginas-entidade do site; e
- distinguir valores de atributo da entidade que é descrita na página de qualquer outro valor publicado na página (por exemplo, distinguir o time do jogador de futebol descrito na página de outros nomes de times que aparecem na página).

O restante desta seção está organizado da seguinte forma. A Subseção 3.4.1 define os conceitos necessários para a compreensão da etapa de *Reforço*. A Subseção 3.4.2 apresenta uma visão geral da etapa. As demais subseções apresentam detalhes dessa etapa.

3.4.1 Definições preliminares

Um atributo cujos valores são publicados com diferentes marcações nas páginas-entidade do mesmo site é denominado `atributo com formato variante`. Por exemplo, a Figura 3.13 apresenta duas páginas-entidade (p_1 e p_2) de um site fictício e o S-graph do site. As páginas-entidade desse site descrevem os atributos *nome*, *time* e *salário* de jogadores

Figura 3.13 – Exemplo de S-graph.

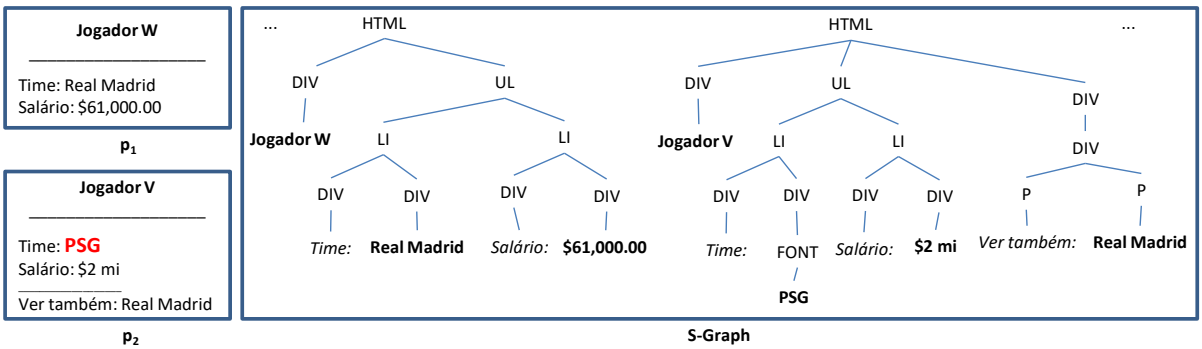


Figura 3.14 – Consultas que extraem o atributo *time* no S-graph apresentado na Figura 3.13.

```
MATCH (n1)--(n2)--(n3)--(n4)--(n5)
WHERE n1.type="TEXT"  AND n1.position=1 AND n1.value="Time:"
   AND n2.type="TAG"   AND n2.position=1 AND n2.value="DIV"
   AND n3.type="TAG"   AND n3.position=1 AND n3.value="LI"
   AND n4.type="TAG"   AND n4.position=2 AND n4.value="DIV"
   AND n5.type="TEXT"  AND n5.position=1
RETURN n5.url, n5.value
```

Q_1

```
MATCH (n1)--(n2)--(n3)--(n4)--(n5)--(n6)
WHERE n1.type="TEXT"  AND n1.position=1 AND n1.value="Time:"
   AND n2.type="TAG"   AND n2.position=1 AND n2.value="DIV"
   AND n3.type="TAG"   AND n3.position=1 AND n3.value="LI"
   AND n4.type="TAG"   AND n4.position=2 AND n4.value="DIV"
   AND n5.type="TAG"   AND n5.position=1 AND n5.value="FONT"
   AND n6.type="TEXT"  AND n6.position=1
RETURN n6.url, n6.value
```

Q_2

de futebol. O atributo *time* possui formato variante, uma vez que na página p_2 ele possui a marcação <*FONT*> e na página p_1 não. A Figura 3.14 apresenta duas consultas Cypher (sobre o S-graph ilustrado na Figura 3.13) que extraem o valor do atributo *time*: (Q_1) na página p_1; e (Q_2) na página p_2. A notação utilizada por **Orion** não possui expressividade suficiente para gerar uma única consulta capaz de extrair os valores do atributo *time* nas páginas p_1 e p_2.

A etapa de *Reforço* reanalisa as consultas Cypher geradas na subetapa *inferência de consultas* da etapa de *Extração*. Cada consulta é representada por meio: (i) do conteúdo textual do nodo *template*; (ii) das marcações que compõem o caminho do nodo *template* até o nodo de dados; e (iii) dos valores extraídos pela consulta. A consulta é formalizada a seguir.

Definição 5 *(Consulta): Uma* `consulta Cypher` *(ou* `consulta`*) é uma tripla* $Q = (l, x, \Omega)$ *em que* l *é o conteúdo textual do nodo template, denominado* `pivô`*;* x *é uma string que representa o caminho do nodo template até o nodo de dados por meio das marcações HTML, denominado* `caminho`*; e* $\Omega = \{\Omega_1, ..., \Omega_n\}$ *é o conjunto das extrações realizadas pela consulta. Cada* `extração` Ω_i *é uma tripla que tem a forma* (p, e, v) *em que* p *é a URL de uma página;* e *é o identificador da entidade que é descrita na página* p*; e* v *é o valor extraído pela consulta na página* p*.*

Tabela 3.3 – Representação das consultas inferidas a partir do S-graph apresentado na Figura 3.13.

ID	Pivô	Caminho	Extrações
Q_1	Time:	DIV-LI-DIV	{(p_1, “Jogador W”, “Real Madrid”)}
Q_2	Time:	DIV-LI-DIV-FONT	{(p_2, “Jogador V”, “PSG”)}
Q_3	Time:	DIV-LI-UL-HTML-DIV	{(p_1, “Jogador W”, “Jogador W”), (p_2, “Jogador V”, “Jogador V”)}
Q_4	Salário:	DIV-LI-DIV	{(p_1, “Jogador W”, “$61,000.00”), ($p_2$, “Jogador V”, “$2 mi”)}
Q_5	Veja também:	P-DIV-P	{(p_2, “Jogador V”, “Real Madrid”)}

Por exemplo, a Tabela 3.3 apresenta a representação das consultas inferidas na subetapa de *inferência de consultas* da etapa de *Extração* a partir do S-graph ilustrado na Figura 3.13.

As consultas são classificadas como:

- `consultas selecionadas` - as consultas que foram inferidas na subetapa *inferência de consultas* da etapa de *Extração* e que foram selecionadas para algum atributo na subetapa de *seleção de consultas* da etapa de *Extração*;
- `consultas não selecionadas` - as consultas que foram inferidas na subetapa *inferência de consultas* da etapa de *Extração* e que não foram selecionadas para nenhum atributo na subetapa de *seleção de consultas* da etapa de *Extração*;
- `consultas alinhadas` - considere Q_s sendo a consulta selecionada para o atributo a_y no site s_x, as consultas alinhadas com Q_s são as consultas de outros sites que também foram selecionadas para o atributo a_y;
- `consultas disjuntas` - considere P_i sendo um conjunto de páginas-entidade que tem valor extraído por uma consulta Q_i e P_j sendo o conjunto de páginas-entidade que tem valor extraído pela consulta Q_j, as consultas Q_i e Q_j são disjuntas se $P_i \cap P_j = \emptyset$;
- `consultas complementares` - considere Q_s sendo a consulta selecionada para o atributo a_y no site s_x, uma consulta complementar à Q_s deve extrair o valor do atributo a_y em uma fração de páginas-entidade do site s_x para qual nem a consulta selecionada nem as demais consultas complementares são capazes de extrair valores, ou seja, as consultas complementares são disjuntas entre si e disjuntas com relação à consulta selecionada;
- `consulta candidata` - é uma consulta não selecionada que é disjunta com relação à consulta selecionada e é avaliada na etapa de *Reforço* a fim de determinar se ela é ou não uma consulta complementar.

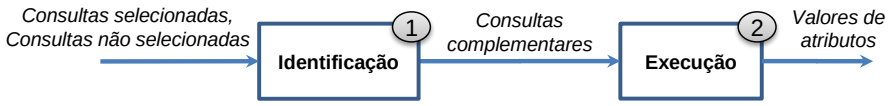
Por exemplo, considerando as consultas apresentadas na Tabela 3.3, as consultas Q_1 e Q_2 são disjuntas. Se, na subetapa de *seleção de consultas* da etapa de *Extração*, forem selecionadas as consultas Q_1, Q_3 e Q_4 para os atributos *time*, *nome* e *salário*, respectivamente, então: (i) Q_1, Q_3 e Q_4 são consultas selecionadas; (ii) Q_2 e Q_5 são consultas não selecionadas e candidatas; (iii) Q_2 é uma consulta complementar à consulta Q_1; e (iv) uma consulta Q_x que foi selecionada para o atributo *time* em outro site é uma consulta alinhada à consulta Q_1.

3.4.2 Visão geral

A etapa de *Reforço* processa os sites em conjunto. As consultas não selecionadas são reanalisadas a fim de identificar as consultas complementares às consultas selecionadas. Dessa forma, a etapa de *Reforço* realiza o tratamento de atributos com formato variante.

A Figura 3.15 apresenta o fluxo de execução da etapa de *Reforço*. A entrada é composta pelas consultas selecionadas e pelas consultas não selecionadas. A saída é composta pelos valores dos atributos extraídos pelas consultas complementares às consultas selecionadas, ou seja, valores de atributo que a etapa de *Extração* não é capaz de extrair porque o atributo possui formato variante. Duas subetapas são executadas: (i) *identificação* - identifica as consultas complementares; e (ii) *execução* - executa as consultas complementares.

Figura 3.15 – Fluxo de execução da etapa de *Reforço*.



A subetapa de *identificação* assume que diferentes consultas que extraem valores do mesmo atributo são similares. Essa subetapa analisa as consultas não selecionadas e verifica a similaridade delas com as consultas selecionadas. As consultas mais similares são consideradas complementares. Para medir essa similaridade, é proposta a função q_{score} (Subseção 3.4.3), que leva em consideração diferentes características das consultas por meio de uma combinação linear de diferentes funções de similaridade. Dois algoritmos - **ICQ** e **NCQ** - (Subseção 3.4.4) foram propostos para determinar as consultas que são complementares.

A subetapa de *execução* da etapa de *Reforço* é realizada da mesma forma que a subetapa de *execução* da etapa de *Extração*. As consultas complementares são executadas sobre o S-graph e os valores retornados são associados aos atributos correspondentes (mesmo atributo associado com a consulta selecionada que é complementada).

Por exemplo, considerando que o S-graph ilustrado na Figura 3.13 e as consultas apresentadas na Tabela 3.3. Se a consulta Q_1 for selecionada para o atributo *time* na subetapa de *seleção de consultas* da etapa de *Extração*, então a etapa de *Reforço* deve identificar que a consulta Q_2 é complementar a ela, executar essa consulta sobre o S-graph e associar os valores retornados ao atributo *time*.

3.4.3 Escore de similaridade

A etapa de *Reforço* necessita atribuir um escore de similaridade entre as consultas candidatas e as consultas selecionadas. Esse escore é utilizado para verificar quais consultas candidatas são complementares. Para atribuir um escore de similaridade entre uma consulta candidata e uma consulta selecionada, é proposta a função q_{score}. Essa função assume que as consultas que extraem valores do mesmo atributo são similares. Essa similaridade ocorre entre consultas do **mesmo site** em termos: (i) dos valores que elas extraem - porque eles são valores do mesmo atributo; (ii) de seus pivôs - porque o pivô de uma consulta, em alguns casos, coincide com o rótulo do atributo no site, como, por exemplo, a consulta Q_4 (Tabela 3.3) tem o pivô "*Salário:*" e extrai valores do atributo *salário*; e (iii) de seus caminhos - porque as consultas complementares normalmente compartilham diversas marcações com suas consultas selecionadas, como, por exemplo, os caminhos das consultas Q_1 e Q_2 (Tabela 3.3) diferem-se apenas em uma marcação. Consultas (que extraem valores do mesmo atributo) de **diferentes sites** também podem ser similares. Essa similaridade é encontrada principalmente entre os valores que elas extraem em páginas que descrevem a mesma entidade.

A função q_{score} é uma combinação linear de diferentes funções de similaridade, que leva em conta: (i) a similaridade entre a consulta candidata e a consulta selecionada (que pertence ao mesmo site) em termos de seus pivôs, caminhos e valores de suas extrações; e (ii) a similaridade entre a consulta candidata e as consultas alinhadas (que pertencem a sites diferentes) em termos dos valores de suas extrações para a mesma entidade.

Antes de explicar as funções de similaridade, é essencial definir algumas variáveis comuns que são usadas nelas: (i) Q_c é uma consulta candidata; (ii) Q_s é uma consulta selecionada; e (iii) $A = \{a_1, ..., a_n\}$ é o conjunto das consultas alinhadas com a consulta selecionada Q_s.

Similaridade de Pivô: atribui um escore de similaridade entre uma consulta candidata e uma consulta selecionada comparando seus pivôs, e é definida como segue:

$$sim_{pivot}(Q_c, Q_s) = f_l(l^c, l^s) \tag{3.1}$$

em que l^c e l^s são os pivôs das consultas Q_c e Q_s, respectivamente, e $f_l(l^c, l^s)$ é um escore de similaridade entre l^c e l^s.

Similaridade de Caminho: atribui um escore de similaridade entre uma consulta candidata e uma consulta selecionada comparando seus caminhos, e é definida como segue:

$$sim_{path}(Q_c, Q_s) = f_x(x^c, x^s) \tag{3.2}$$

em que x^c e x^s são os caminhos das consultas Q_c e Q_s, respectivamente, e $f_x(x^c, x^s)$ é um escore de similaridade entre x^c e x^s.

Similaridade de Valor em Nível de Atributo: atribui um escore de similaridade entre uma consulta candidata e uma consulta selecionada comparando os valores de suas extrações. O valor de cada extração da consulta candidata é comparado com o valor de todas as extrações da consulta selecionada ($n \times n$). Essa função é definida como segue:

$$sim_{attr}(Q_c, Q_s) = \frac{\sum_{v_i \in \Omega^c} fa(v_i, \Omega^s)}{|\Omega^c|} \tag{3.3}$$

em que Ω^c e Ω^s são os conjuntos de extrações das consultas Q_c e Q_s, respectivamente; v_i é um valor que pertence ao conjunto Ω^c; e $fa(v_i, \Omega^s)$ é um escore de similaridade entre v_i e todos os valores que pertencem ao conjunto Ω^s.

Similaridade de Valor em Nível de Entidade: atribui um escore de similaridade entre uma consulta candidata e as consultas alinhadas (que são de sites diferentes) a uma consulta selecionada comparando os valores de suas extrações. O valor de cada extração da consulta candidata é comparado apenas com o valor da extração (das consultas alinhadas) que tem o mesmo identificador de entidade (1×1). Essa função é definida como segue:

$$sim_{entity}(Q_c, A) = \max_{a_i \in A} \sum_{e_i \in \Omega^c \cap \Omega^{a_i}} fe(v_c^{e_i}, v_a^{e_i}) \tag{3.4}$$

em que a_i é uma consulta alinhada que pertence ao conjunto A; Ω^c e Ω^{a_i} são os conjuntos de extrações das consultas Q_c e a_i, respectivamente; e_i é um identificador de entidade que pertence aos conjuntos Ω^c e Ω^{a_i}; $v_c^{e_i}$ e $v_a^{e_i}$ são valores das extrações que pertencem aos conjuntos Ω^c e Ω^{a_i}, respectivamente, e cujo identificador de entidade é e_i; e $fe(v_c^{e_i}, v_a^{e_i})$ é um escore de similaridade entre $v_c^{e_i}$ e $v_a^{e_i}$.

A função q_{score}: é uma combinação linear das funções de similaridade apresentadas anteriormente e é definida como segue:

$$q_{score}(Q_c, Q_s, A) = w_1 \times sim_{pivot}(Q_c, Q_s) + w_2 \times sim_{path}(Q_c, Q_s) + w_3 \times sim_{attr}(Q_c, Q_s) + w_4 \times sim_{entity}(Q_c, A) + w_5 \times nr_{pages} \quad (3.5)$$

em que w_i (1 <= i <= 5) representa os pesos[11]; sim_{pivot} é a Similaridade de Pivô; sim_{path} é a Similaridade de Caminho; sim_{attr} é a Similaridade de Valor em Nível de Atributo; sim_{entity} é a Similaridade de Valor em Nível de Entidade; e nr_{pages} é o número de páginas distintas que pertencem ao conjunto de extrações da consulta Q_c dividido pelo número de páginas-entidade do site. O componente nr_{pages} impulsiona o escore das consultas que extraem valores em mais páginas-entidade.

3.4.4 Algoritmos

Esta subseção define os algoritmos que compõem a solução para identificar as consultas complementares. O Algoritmo 5 - **ICQ** - analisa as consultas selecionadas. Se uma consulta selecionada não extrai os valores do atributo em todas as páginas-entidade do site (porque o atributo tem formato variante), o Algoritmo 5 utiliza o Algoritmo 6 - **NCQ** - para obter as consultas complementares. O Algoritmo 6 obtém as consultas complementares ranqueando as consultas candidatas por meio do escore de similaridade retornado pela função q_{score}.

O Algoritmo 5 - ICQ (*Identification of Complementary Queries* - Identificação de Consultas Complementares) - tem como objetivo identificar as consultas complementares a uma determinada consulta selecionada (quando necessário). A entrada é composta de uma consulta selecionada Q_s e um site s_x. A consulta selecionada Q_s extrai valores de um atributo a_y nas páginas-entidade do site s_x. A saída é composta por uma lista contendo as consultas que extraem o valor do atributo a_y no site s_x em diferentes páginas-entidade, ou seja, a consulta selecionada e suas consultas complementares (quando necessário). A Linha 1 adiciona a consulta selecionada para uma lista L. A Linha 2 adiciona as páginas-entidade que têm valor extraído pela consulta selecionada para um conjunto P. O laço nas linhas 3-10 procura pela próxima consulta complementar enquanto o número de páginas-entidade em P é menor que o número de páginas-entidade do site s_x. A Linha 4 obtém a próxima consulta complementar, de acordo com o Algoritmo 6 - **NCQ**. Se o valor *null* for retornado, o laço é interrompido porque não há mais consultas complementares (linhas 5-7). Caso contrário, a consulta complementar retornada é adicionada à lista L (Linha 8) e as páginas-entidade que possuem valor extraído por essa

[11] O peso de cada componente da função q_{score} é definido no Capítulo 4.

consulta são adicionadas ao conjunto P (Linha 9). Finalmente, a Linha 11 retorna a lista L.

ALGORITHM 5: ICQ (Identification of Complementary Queries - Identificação de Consultas Complementares

Input: Q_s, s_x

Output: L

```
1  adiciona Q_s para L;
2  adiciona páginasExtraídas(Q_s) para P;
3  while nrPáginas(P) < nrPáginas(s_x) do
4      Q_c = NCQ(s_x, L);
5      if Q_c == null then
6          break;
7      end
8      adiciona Q_c para L;
9      adiciona páginasExtraídas(Q_c) para P;
10 end
11 return L;
```

ALGORITHM 6: NCQ (Next Complementary Query - Próxima Consulta Complementar)

Input: s_x, L

Output: r_i

```
1  P = ∅;
2  for l_i ∈ L do
3      adiciona páginasExtraídas(l_i) para P;
4  end
5  Q = obtémConsultasNãoSelecionadas(s_x);
6  remove cada Q_i ∈ Q onde páginasExtraídas(Q_i)
     ∩ P <> ∅;
7  return Q_i ∈ Q com o maior q_score e q_score > β;
```

O Algoritmo 6 - NCQ (*Next Complementary Query* - Próxima Consulta Complementar) tem como objetivo buscar a próxima consulta complementar a uma determinada consulta selecionada para um atributo a_y em um site s_x. A entrada é composta por um site s_x e uma lista L formada por uma consulta selecionada e as consultas complementares que já foram identificadas em iterações prévias. A saída é a próxima consulta complementar. A Linha 1 inicializa o conjunto P. As linhas 2-4 adicionam ao conjunto P as páginas-entidade que têm valor extraído pelas consultas presentes na lista L. A Linha 5 obtém as consultas não selecionadas do site s_x e adiciona ao conjunto Q. Essas consultas são fornecidas como entrada para a etapa de *Reforço*. A Linha 6 remove cada consulta Q_i do conjunto Q que tem interseção não vazia entre o conjunto P e o conjunto de páginas-entidade que possuem valor extraído pela consulta Q_i. Essa remoção garante que o conjunto Q mantenha apenas as consultas que são disjuntas à consulta selecionada e às consultas complementares já identificadas. A Linha 7 retorna a consulta Q_i do conjunto Q que contém o maior q_{score} e que tem um q_{score} maior que β. Quando nenhuma consulta do conjunto Q satisfaz o limiar β, o valor *null* é retornado.

4 AVALIAÇÃO EXPERIMENTAL

Este capítulo descreve os experimentos realizados com o intuito de avaliar a eficácia das etapas da abordagem **Orion**, propostas nesta Tese. Cada etapa é, primeiramente, avaliada de forma isolada. Por fim, a abordagem **Orion** é avaliada integralmente.

Este capítulo está organizado da seguinte forma. A Seção 4.1 define as métricas utilizadas. A Seção 4.2 apresenta as bases de dados utilizadas. As Seções 4.3, 4.4 e 4.5 descrevem, respectivamente, os experimentos realizados com o objetivo de avaliar as etapas de *Descoberta*, *Extração* e *Reforço* de forma isolada. A Seção 4.6 descreve os experimentos realizados com o objetivo de avaliar a abordagem **Orion** integralmente. A Seção 4.7 detalha a análise geral dos experimentos. Finalmente, a Seção 4.8 discute os casos de falha da abordagem **Orion**.

4.1 Métricas de avaliação

As seguintes métricas foram utilizadas para avaliar a eficácia das abordagens: revocação, precisão e F1. Essas métricas são tradicionalmente utilizadas pela comunidade de recuperação de informação (BAEZA-YATES; RIBEIRO-NETO, 2011). Os valores extraídos pelas abordagens foram comparados com os valores do gabarito para calcular o número de verdadeiros positivos (VP), falsos negativos (FN) e falsos positivos (FP), uma vez que por meio deles é possível calcular a precisão como $P = \frac{VP}{VP+FP}$, a revocação como $R = \frac{VP}{VP+FN}$ e a F1 como $F1 = 2\frac{P \times R}{P+R}$. A definição de verdadeiros positivos, falsos negativos e falsos positivos é apresentada junto com a metodologia dos experimentos, uma vez que ela varia conforme o objetivo de cada experimento. Quanto maior a revocação, a precisão e a F1, maior a eficácia da abordagem.

As seguintes métricas foram utilizadas para avaliar a eficiência: tempo de processamento e número de páginas baixadas. O tempo de processamento indica o tempo total gasto na execução de uma determinada etapa. O número de páginas baixadas indica o número de páginas que tiveram seu conteúdo acessado durante determinada etapa. Essas páginas precisam ser baixadas localmente para que a abordagem acesse seu conteúdo. Quanto menor o tempo de processamento e o número de páginas baixadas, maior a eficiência da abordagem. O número de páginas baixadas é uma métrica importante porque quanto menos páginas são baixadas de um site, menor a sobrecarga desse site.

A significância estatística foi confirmada utilizando o Teste T (*Student's t-test*) (ANDERSON; FINN, 1996) com o limiar padrão de significância ($\alpha = 0,05$). Quando o valor de p bicaudal, calculado pelo Teste T, for menor que α, existe uma diferença estatisticamente

significante entre as abordagens comparadas, com uma confiança de 95%.

Os experimentos foram realizados em um computador com processador Intel Core 2 Quad 2,66GHz, 8GB de memória principal e 5TB de espaço em disco. Foi utilizado o sistema operacional Ubuntu 14.04.

4.2 Bases de dados

Foram utilizadas seis bases de dados. As bases $DATASET\_S$[1] e $DATASET\_W$[2] são coleções de páginas-entidade de sites reais categorizadas por tipo de entidade, criadas por Hao *et al.* (2011) e Bronzi *et al.* (2013), respectivamente. Essas bases de dados incluem um gabarito que descreve os valores dos atributos publicados em cada página-entidade. Os gabaritos possuem granularidade elemento da árvore DOM, ou seja, os nodos textuais não estão segmentados. Se o rótulo e o valor de um atributo são publicados em um determinado site no mesmo nodo textual, então o gabarito inclui tanto o rótulo quando o valor. As bases $DATASET\_S$ e $DATASET\_W$ foram utilizadas para avaliar diversos métodos de extração de dados (HAO et al., 2011; BRONZI et al., 2013; GENTILE et al., 2013; ORTONA et al., 2016).

As bases de dados $DATASET\_C$, $DATASET\_D1$, $DATASET\_D2$ e $DATASET\_M$ foram criadas nesta Tese e contêm as páginas-entidades de um conjunto de sites reais. Para cada site s_x: (i) as páginas do site foram baixadas; (ii) um tipo de entidade de interesse d_y (por exemplo, piloto em um site sobre automobilismo) foi definido; e (iii) o gabarito (conjunto de páginas-entidade do tipo d_y no site s_x) foi obtido por meio de regras criadas manualmente. O gabarito da base de dados $DATASET\_D2$ ainda inclui os valores dos atributos publicados em cada página-entidade.

As características das seis bases de dados estão sumarizadas na Tabela 4.1, que apresenta o número de sites, o número total de páginas-entidade e os atributos para cada tipo de entidade. Os detalhes das bases de dados $DATASET\_C$, $DATASET\_D1$, $DATASET\_D2$ e $DATASET\_M$ são apresentados na Tabela 4.2, que apresenta um identificador, a URL da página principal, o tipo de entidade de interesse e o número de páginas-entidade para cada site. Os sites que compõem as bases de dados $DATASET\_S$ e $DATASET\_W$ podem ser verificados em Hao *et al.* (2011) e Bronzi *et al.* (2013), respectivamente.

As bases de dados $DATASET\_C$, $DATASET\_D1$, $DATASET\_D2$ e $DATASET\_M$ foram criadas porque não foi encontrada na literatura nenhuma base de

[1]Disponível em: <http://swde.codeplex.com/>. Último acesso em: 10/10/2017.
[2]Disponível em: <http://www.dia.uniroma3.it/db/weir/>. Último acesso em: 10/10/2017.

dados que atendesse aos requisitos necessários para avaliar a etapa de *Descoberta* da abordagem **Orion**. Para a avaliação dessa etapa, a URL e o HTML de todas as páginas do site são necessários, uma vez que a abordagem **Orion** navega pelo site, analisando a URL e o HTML das páginas, para encontrar as páginas-entidade. Além disso, as páginas-entidade precisam estar identificadas a fim de formar o gabarito. A maioria dos sites que compõem as bases de dados criadas também foi utilizada pelos *baselines*. No entanto, os resultados não são diretamente comparáveis porque os sites evoluíram no período compreendido entre a realização dos experimentos dos *baselines* e a realização dos experimentos desta Tese, ou seja, o HTML e a URL das páginas desses sites mudaram nesse período.

Tabela 4.1 – Composição das bases de dados.

Base de Dados	Tipo de entidade	Sites	Páginas-entidade	Atributos
DATASET_S	automóveis	10	17.923	modelo, preço, motor, consumo
	câmeras	10	5.258	modelo, preço, fabricante
	empregos	10	20.000	título, companhia, localização, data de publicação
	filmes	10	20.000	título, diretor, gênero, classificação
	jogadores da NBA	10	4.405	nome, time, altura, peso
	livros	10	20.000	título, autor, ISBN, editor, data de publicação
	restaurantes	10	20.000	nome, endereço, telefone, culinária
	universidades	10	16.705	nome, telefone, site, tipo
DATASET_W	ações	10	4.646	valor, variação, percentual de variação, valor de abertura, maior valor, menor valor, volume, menor valor no ano, maior valor no ano
	jogadores de futebol	10	5.745	número, posição, altura, peso, data de nascimento, seleção, time, local de nascimento, nacionalidade
	livros	10	1.315	título, autor, ISBN, editor, data de publicação, formato e edição
	videogames	10	12.329	distribuidor, gênero, desenvolvedor, classificação
DATASET_C	Vereadores	18	558	-
DATASET_D1	Pilotos	10	1.976	-
DATASET_D2	Pilotos	10	963	Nome, equipe, data e local de nascimento
DATASET_M	Associação	2	413	-
	Banda	1	32.318	-
	Jogador de Futebol	1	28	-
	Pessoa	1	1.032	-
	Esporte	1	56	-
	Projeto de Software	1	382	-
	Senador	1	121	-
	Funcionário	1	473	-
	Carro	1	512	-

É importante destacar alguns pontos da metodologia adotada para a criação das bases de dados. Para a criação da base de dados $DATASET\_C$, todos os sites oficiais das câmaras municipais de vereadores das 26 capitais de estados brasileiros foram analisados. Foram excluídos: (i) quatro sites porque não possuíam uma página-entidade para cada vereador; (ii) três sites porque todas as páginas-entidade eram *frames* internos de uma única página; e (iii) um site porque não permitia coletar suas páginas.

Tabela 4.2 – Detalhes das bases de dados criadas.

Base de dados	ID do site	Página principal do site	Tipo de Entidade	NP
DATASET_C	Belém	http://cmb.pa.gov.br/	Vereador	22
	Belo Horizonte	http://www.cmbh.mg.gov.br/	Vereador	42
	Campo Grande	http://camara.ms.gov.br/	Vereador	29
	Cuiabá	http://camaracba.mt.gov.br/	Vereador	25
	Curitiba	http://www.cmc.pr.gov.br/	Vereador	38
	Florianópolis	http://cmf.sc.gov.br/	Vereador	23
	Fortaleza	http://cmfor.ce.gov.br/	Vereador	50
	Goiânia	http://www.camara.go.gov.br/	Vereador	35
	João Pessoa	http://cmjp.pb.gov.br/	Vereador	27
	Macapá	http://cmm.ap.gov.br/	Vereador	23
	Natal	http://cmnat.rn.gov.br/	Vereador	29
	Porto Velho	http://portovelho.ro.leg.br/	Vereador	21
	Recife	http://www.recife.pe.leg.br/	Vereador	39
	Rio Branco	http://riobranco.ac.leg.br/	Vereador	17
	Rio de Janeiro	http://www.camara.rj.gov.br/	Vereador	51
	Salvador	http://www.cms.ba.gov.br/	Vereador	43
	Teresina	http://teresina.pi.leg.br/	Vereador	29
	Vitória	http://www3.cmv.es.gov.br/	Vereador	15
DATASET_D1	Champ	http://champcar-ws.com/	Piloto	20
	F Truck	http://www.formulatruck.com.br/	Piloto	26
	F1	http://www.formula1.com/	Piloto	22
	F3	http://www.formula3.co/	Piloto	31
	GP2	http://www.gp2series.com/	Piloto	34
	Indycar	http://www.indycar.com/	Piloto	39
	Motor GP	http://www.motogp.com/	Piloto	1288
	Nascar	http://www.nascar.com/	Piloto	349
	Stock car	http://www.stockcar.com.br/	Piloto	34
	WRC	http://www.wrc.com/	Piloto	133
DATASET_D2	Euro	http://www.eurosport.com/	Piloto	62
	F1	http://www.formula1.com/	Piloto	22
	F3	http://www.formula3.co/	Piloto	31
	FE	http://www.fiaformulae.com/en/	Piloto	20
	GP2	http://www.gp2series.com/	Piloto	34
	GPUPDATE	http://www.gpupdate.net/en/	Piloto	349
	Indycar	http://www.indycar.com/	Piloto	39
	Nascar	http://www.nascar.com/	Piloto	349
	Sky	http://www.skysports.com/f1/	Piloto	23
	Stock car	http://www.stockcar.com.br/	Piloto	34
DATASET_M	Fifa	http://www.fifa.com/	Associação	209
	Guitar	http://www.guitaretab.com/	Banda	32318
	Inter	http://www.inter.it/en/hp/	Jogador	28
	MIT EECS	http://www.eecs.mit.edu/	Pessoa	1032
	Olympic-A	http://www.olympic.org/	Associação	204
	Olympic-S	http://www.olympic.org/	Esporte	56
	Pgfoundry	http://pgfoundry.org/	Projeto de Software	382
	Senado	http://www.senado.gov.br/	Senador	121
	Stanford EE	http://engineering.stanford.edu/	Funcionário	473
	Supercar	http://www.supercarsite.net/	Carro	512

Nota: NP = Número de páginas-entidade.

O site *MIT EECS* da base de dados $DATASET\_M$ inclui páginas-entidade sobre professores, técnico-administrativos e alunos. Essas páginas foram consideradas como sendo do mesmo tipo (*pessoa*) uma vez que compartilham os atributos *posição*, *telefone*, *e-mail*, entre outros. O site *Stanford* da base de dados $DATASET\_M$ inclui páginas-entidade sobre professores e técnico-administrativos. Essas páginas foram consideradas como sendo do mesmo tipo (*funcionário*) porque compartilham os atributos *grau acadêmico*, *departamento*, *e-mail*, *telefone*, entre outros. Os sites *F1*, *F3* e *GP2* das bases de dados $DATASET\_D1$ e $DATASET\_D2$ incluem páginas-entidade sobre pilotos ativos e pilotos inativos porque essas páginas compartilham os atributos *nome*, *equipe*, *data de nascimento*, entre outros.

4.3 Avaliação da etapa de Descoberta

O objetivo desta seção é apresentar os experimentos realizados para avaliar a eficácia e a eficiência da etapa de *Descoberta* da abordagem **Orion**. A Subseção 4.3.1 apresenta a configuração dos experimentos. A Subseção 4.3.2 descreve os experimentos realizados para definir a configuração de parâmetros da abordagem **Orion** e dos *baselines*. A Subseção 4.3.3 descreve os experimentos realizados para comparar a abordagem **Orion** com os *baselines*.

4.3.1 Configuração dos experimentos

Esta subseção descreve os *baselines* e a metodologia adotada nos experimentos sobre a etapa de *Descoberta*.

4.3.1.1 Baselines

A etapa de *Descoberta* da abordagem **Orion** foi comparada com duas abordagens: (i) *INDESIT* (BLANCO; CRESCENZI; MERIALDO, 2005) - que utiliza apenas características de HTML; e (ii) *Growing Parallel Paths* (referenciado nesta Tese como *GPP*) (WENINGER; JOHNSTON; HAN, 2013) - que combina características de HTML e visuais (posições X e Y, largura e altura).

INDESIT e *GPP* foram escolhidos como *baselines* porque eles: (i) possuem o mesmo objetivo, entrada e saída que a etapa de *Descoberta* da abordagem **Orion**; (ii) não requerem que o usuário realize anotações; (iii) não exigem que as páginas-entidade em todos os sites possuam marcações HTML específicas (por exemplo, <*TABLE*> e <*UL*>); e (iv) não dependem

de enciclopédias manualmente compiladas (por exemplo, Wikipedia).

4.3.1.2 Metodologia

Os experimentos desta seção avaliam a eficácia e a eficiência da descoberta de páginas-entidade realizada pela abordagem **Orion**. Para cada site, é fornecida uma página-entidade como entrada (denominada página de exemplo) e as páginas que são retornadas, como saída, são comparadas com um gabarito que contém as páginas-entidade do site.

Os experimentos foram divididos em dois grupos: (i) *calibragem* - com o objetivo de definir a melhor configuração de parâmetros da abordagem **Orion** e dos *baselines*; e (ii) *avaliação* - com o objetivo de comparar a abordagem **Orion** com os *baselines*. Para cada site, 10% das páginas-entidade foram selecionadas aleatoriamente para serem utilizadas como páginas de exemplo para os experimentos de calibragem e outros 10% para os experimentos de avaliação. As abordagens foram executadas com diferentes páginas de exemplo para evitar que a escolha da página de exemplo influenciasse nos resultados. Os resultados apresentados correspondem à média das execuções com as diferentes páginas de exemplo.

Os verdadeiros positivos (VP), falsos negativos (FN) e falsos positivos (FP), que são utilizados para calcular as métricas de eficácia (revocação, precisão e F1), foram calculados da seguinte forma: (VP) quantidade de páginas recuperadas que são relevantes; (FN) - quantidade de páginas relevantes que não foram recuperadas; e (FP) - quantidade de páginas recuperadas que não são relevantes. Uma página é `relevante` se ela é uma página-entidade do mesmo tipo que a página de exemplo. Uma página é `irrelevante` se: (i) não é uma página-entidade; ou (ii) é uma página-entidade de um tipo diferente da página de exemplo. As páginas de cada site foram previamente baixadas a fim de evitar a interferência da rede no tempo de processamento. Foram utilizadas as bases de dados $DATASET\_M$, $DATASET\_C$ e $DATASET\_D1$ porque incluem a URL e o HTML de todas as páginas do site e a abordagem **Orion** navega pelo site, analisando a URL e o HTML das páginas, para encontrar as páginas-entidade. As demais bases de dados não foram utilizadas porque: (i) as bases de dados $DATASET\_S$ e $DATASET\_W$ só incluem as páginas-entidade dos sites; e (ii) a base de dados $DATASET\_D2$, de forma geral, só difere da base de dados $DATASET\_D1$ por incluir a anotação dos valores dos atributos publicados nas páginas-entidade, que não contribui para a avaliação realizada nesta seção.

A abordagem *INDESIT* foi implementada como descrito por Blanco, Crescenzi e Merialdo (2005). As abordagens **Orion** e *INDESIT* foram implementadas em Java usando a biblioteca JSoup[3] para manipular o HTML das páginas. O módulo de extração das características

[3]Disponível em: <http://jsoup.org/>. Último acesso em: 10/10/2017.

visuais da abordagem *GPP* foi fornecido pelos autores. Esse módulo está implementado em Java e usa a API (*Application Programming Interface*) Selenium Web Driver[4].

4.3.2 Descrição dos experimentos de calibragem

Esta subseção apresenta os experimentos realizados com o objetivo de encontrar a melhor configuração de parâmetros para as abordagens (**Orion** e *baselines*). Essa calibragem se faz necessária porque a melhor configuração varia conforme a base de dados utilizada. Experimentos mostraram que as configurações dos *baselines* utilizadas nos experimentos descritos nos artigos que os propõem não eram adequadas para algumas das bases de dados utilizadas nos experimentos desta Tese.

*4.3.2.1 Melhor configuração da abordagem **Orion***

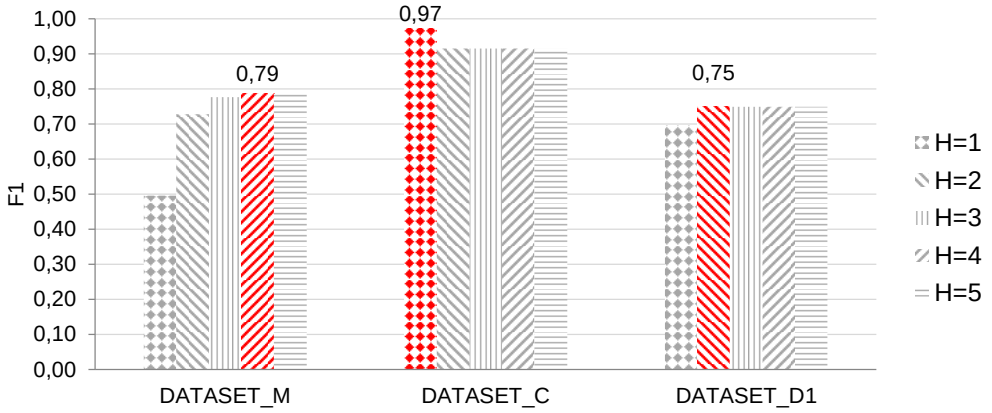
O objetivo desse experimento é responder a seguinte questão: *qual é a melhor configuração de parâmetros da etapa de Descoberta da abordagem **Orion** em cada base de dados?* A etapa de *Descoberta* da abordagem **Orion** possui um único parâmetro, H, que define a estimativa de altura da árvore-entidade. Os seguintes valores para H foram testados 1, 2, 3, 4 e 5. Uma análise manual verificou que nenhum site das bases de dados possui uma árvore-entidade com altura maior que cinco.

O melhor valor de H foi definido por meio da eficácia, que foi medida utilizando a métrica F1. Quando dois ou mais valores de H produzem o mesmo valor de F1, o melhor valor de H é o menor, pois possui o menor tempo de processamento. A Figura 4.1 apresenta a média da F1 para cada valor de H em cada base de dados. O melhor valor de H na base de dados $DATASET\_M$ foi quatro, na base de dados $DATASET\_C$ foi um e na base de dados $DATASET\_D1$ foi dois. Esses valores estão destacados em vermelho na figura e correspondem aos valores de H utilizados nos demais experimentos.

O melhor valor de H na base de dados $DATASET\_M$ foi maior que nas demais bases de dados porque os sites dessa base de dados possuem o maior número de páginas-entidade. Quanto maior o número de páginas-entidade um site possuir, maior a altura da árvore-entidade a fim de facilitar a navegação do usuário. Nas bases de dados $DATASET\_M$ e $DATASET\_D1$, um valor de H maior que a altura real da árvore-entidade não afetou a eficácia da abordagem porque as páginas irrelevantes foram filtradas corretamente por meio das

[4]Disponível em: <http://docs.seleniumhq.org/projects/webdriver/>. Último acesso em: 10/10/2017.

Figura 4.1 – F1 da abordagem **Orion** para cada valor de H.



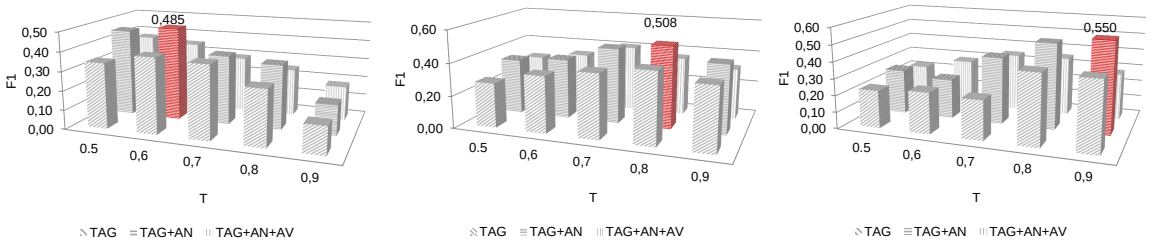
características de HTML e URL. Na base de dados $DATASET\_C$, um valor de H maior que a altura real da árvore-entidade afetou a eficácia da abordagem porque quatro sites dessa base de dados possuem páginas irrelevantes (páginas de notícias, seções, pautas, etc.) com HTML e URL similares às páginas relevantes (páginas-entidade de vereadores), logo a abordagem **Orion** não filtrou as páginas irrelevantes corretamente. Por exemplo, no site *Campo Grande*, a URL das páginas com notícias ou pautas difere da URL das páginas sobre vereadores apenas por um valor de parâmetro e a estrutura HTML também é similar.

4.3.2.2 Melhor configuração da abordagem INDESIT

O objetivo desse experimento é responder a seguinte questão: *qual é a melhor configuração de parâmetros da abordagem INDESIT em cada base de dados?* A abordagem *INDESIT* possui dois parâmetros: (i) T - define o limiar de similaridade de HTML entre duas páginas, que varia de 0 a 1; e (ii) *notação* - define os componentes de um *link-path*. Os seguintes valores de T foram testados 0,1; 0,2; ...; 0,9. Apenas os resultados para valores de T maiores ou iguais a 0,5 são apresentados, uma vez que a eficácia de *INDESIT* foi sempre menor quando $T < 0,5$. As seguintes notações foram testadas: (i) TAG - o *link-path* é formado apenas por marcações; (i) TAG+AN - o *link-path* é formado por marcações e nomes de atributos; e (iii) TAG+AN+AV - o *link-path* é formado por marcações, nomes de atributos e valores de atributos.

A melhor configuração de parâmetros é aquela que produz a maior eficácia, que foi medida utilizando a métrica F1. A Figura 4.2 apresenta a média da F1 para cada combinação de valor de T e notação em cada base de dados. A melhor configuração na base de dados $DATASET\_M$ foi $T = 0,6$ e notação utilizando marcações e nomes de atributos. A melhor configuração na base de dados $DATASET\_C$ foi $T = 0,8$ e notação utilizando marcações e

Figura 4.2 – F1 da abordagem INDESIT para cada configuração de parâmetros



(a) $DATASET\_M$ (b) $DATASET\_C$ (c) $DATASET\_D1$

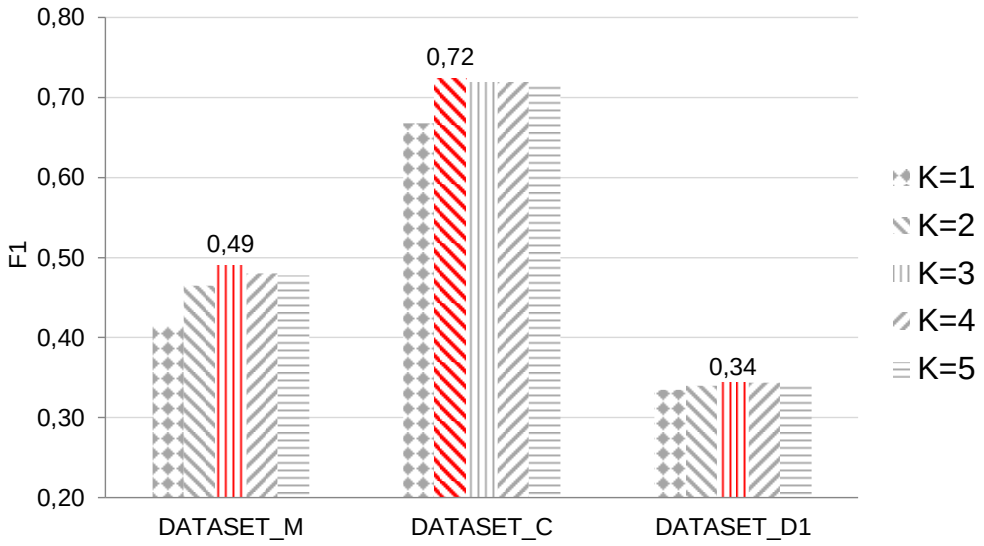
nomes de atributos. A melhor configuração na base de $DATASET\_D1$ foi $T = 0,9$ e notação utilizando marcações e nomes de atributos. Essas configurações estão destacadas em vermelho e correspondem à configuração utilizada nos demais experimentos.

A abordagem *INDESIT* obteve o maior valor de F1 com o maior valor de T na base de dados $DATASET\_D1$ porque as páginas-entidade nos sites dessa base de dados seguem um *template* rígido. A melhor notação nas três bases de dados foi aquela que inclui marcações e nomes de atributos. Essa notação tem uma capacidade de discriminação (capacidade de distinguir páginas relevantes de páginas irrelevantes) maior que a notação que utiliza apenas marcações uma vez que pode distinguir, por exemplo, a marcação <*TD*> com o atributo *colspan* de uma marcação <*TD*> com o atributo *rowspan*. A notação que inclui marcações, nomes de atributos e valores de atributos tem a maior capacidade de discriminação. No entanto, essa notação tem sua revocação afetada em sites que possuem um identificador da entidade no valor de um atributo. Por exemplo, todas as páginas-entidade do site *Olympic - A* possuem um identificador da entidade no valor do atributo *class* da marcação <*BODY*> (<*BODY class="country PageID30785 klp"*>). A utilização da notação que inclui marcações, nomes de atributos e valores de atributos no site *Olympic - A* faz com que a similaridade de HTML entre páginas-entidade do mesmo tipo seja zero, pois, nesse caso, elas não compartilham nenhum *link-path*.

4.3.2.3 Melhor configuração da abordagem GPP

O objetivo desse experimento é responder a seguinte questão: *qual é a melhor configuração de parâmetros da abordagem GPP em cada base de dados?* A abordagem *GPP* possui um único parâmetro, K, que define o número de iterações que a abordagem deve realizar. Os seguintes valores de K foram testados: 1, 2, 3, 4 e 5. Cinco foi o número de iterações que os autores utilizaram no artigo original.

Figura 4.3 – F1 da abordagem GPP para cada valor de K .



O melhor valor de K é aquele que produz a maior eficácia, que foi medida utilizando a métrica F1. Quando dois ou mais valores de K produzem o mesmo valor de F1, o melhor valor de K é o menor, pois possui o menor tempo de processamento. A Figura 4.3 apresenta a média da F1 para cada valor de K em cada base de dados. O melhor valor de K na base de dados $DATASET\_M$ foi três, na base de dados $DATASET\_C$ foi dois e na base de dados $DATASET\_D1$ foi três. Esses valores estão destacados em vermelho e correspondem aos valores de K utilizados nos demais experimentos.

O melhor valor de K nas bases de dados $DATASET\_M$ e $DATASET\_D1$ foi maior que na base de dados $DATASET\_C$ porque alguns sites dessas bases de dados possuem links na página principal do site que apontam para algumas páginas-entidade, os quais funcionam como atalhos. Por exemplo, um site sobre automobilismo possui links na página principal que apontam para as páginas-entidade dos três pilotos com maior pontuação. A primeira iteração ($K = 1$) encontrou esse caminho na busca em largura realizada pela abordagem *GPP*. Outras iterações foram necessárias para encontrar o caminho que entrega todas as páginas-entidades sobre pilotos presentes no site.

4.3.3 Descrição dos experimentos de avaliação

Esta subseção apresenta os experimentos que foram executados para comparar as abordagens em termos de eficácia e eficiência. Dois experimentos foram realizados: (i) Subseção 4.3.3.1 - compara a abordagem **Orion** com a abordagem *INDESIT*; e (ii) Subseção 4.3.3.2

- compara a abordagem **Orion** com a abordagem *GPP*. Cada experimento compara **Orion** com apenas um *baseline* a fim de fornecer uma análise profunda dos resultados, que descreve o impacto das características utilizadas pela abordagem em termos de eficácia e eficiência.

4.3.3.1 ***Orion*** *versus INDESIT*

O objetivo desse experimento é responder as seguintes questões: (i) *qual abordagem é mais eficaz:* ***Orion*** *ou INDESIT?*; e (ii) *qual abordagem é mais eficiente:* ***Orion*** *ou INDESIT?* A avaliação da eficácia foi realizada utilizando as seguintes métricas: revocação e precisão. A Tabela 4.3 apresenta a média da revocação e da precisão das abordagens em cada base de dados e a média geral considerando os sites das três bases de dados.

Tabela 4.3 – Revocação e precisão das abordagens *INDESIT* e **Orion**.

Bases de dados	**Revocação**			**Precisão**		
	INDESIT	**Orion**	**%**	**INDESIT**	**Orion**	**%**
DATASET_C	0,73	0,97	32	0,57	0,99	**73**
DATASET_D1	0,53	0,68	28	0,76	0,90	19
DATASET_M	0,52	0,82	**57**	0,60	0,93	54
Média	0,62	0,85	37	0,63	0,95	51

Em média, as abordagens **Orion** e *INDESIT* obtiveram uma revocação de 0,85 e 0,62, respectivamente. O ganho médio de revocação da abordagem **Orion** foi de 37%. O Teste T realizado mostra que esse ganho é estatisticamente significante, pois o valor do p bicaudal calculado ($\approx 2,23 \times 10^{-03}$) é menor que o coeficiente de significância adotado ($\alpha = 0,05$). Esse ganho ocorreu porque a abordagem **Orion** determina os limiares de similaridade de HTML e URL em cada site automaticamente. O limiar de similaridade de HTML da abordagem *INDESIT* é definido pelo usuário. O melhor valor de T para cada base de dados foi adotado, mas esse valor não era o valor apropriado para todos os sites da base de dados. Algumas páginas-entidade não satisfizeram o limiar e foram eliminadas erroneamente pela abordagem *INDESIT*. O maior ganho de revocação da abordagem **Orion** foi na base de dados $DATASET\_M$ (57%) porque essa base de dados possui sites que publicam diferentes tipos de entidade, logo a utilização de um limiar de similaridade de HTML fixo (caso da abordagem *INDESIT*) não foi efetiva.

Em média, as abordagens **Orion** e *INDESIT* obtiveram precisão de 0,95 e 0,63, respectivamente. O ganho médio de precisão da abordagem **Orion** foi de 51%. O Teste T realizado mostra que esse ganho é estatisticamente significante, pois o valor do p bicaudal calculado ($\approx 1,24 \times 10^{-06}$) é menor que o coeficiente de significância adotado ($\alpha = 0,05$). Esse ganho

ocorreu porque a abordagem **Orion** combina características de HTML com características de URL para distinguir páginas relevantes de páginas irrelevantes. Por outro lado, a abordagem *INDESIT* utiliza apenas características de HTML, que não foram suficientes para separar páginas relevantes de páginas irrelevantes em determinados sites. O maior ganho de precisão da abordagem **Orion** foi na base de dados $DATASET\_C$ (73%) porque as páginas-entidade na maioria dos sites dessa base de dados possuem um termo na URL (por exemplo, "*vereador*") que ocorre apenas nas páginas relevantes. Nesses sites, a similaridade de URL desempenhou um importante papel para distinguir as páginas relevantes das páginas irrelevantes.

A avaliação da eficiência foi realizada utilizando as seguintes métricas: tempo de processamento e número de páginas baixadas. A Tabela 4.4 apresenta a média do tempo de processamento em segundos (*Tempo*) e do número de páginas baixadas (*Nr. Downloads*) das abordagens considerando os sites das três bases de dados. O tempo de processamento e o número de páginas baixadas apresentados são valores relativos, ou seja, correspondem aos valores absolutos divididos pelo número de páginas relevantes que a abordagem recuperou.

Tabela 4.4 – Tempo de processamento e número de páginas baixadas por *INDESIT* e **Orion**.

Métrica	**INDESIT**	**Orion**	**%**
Tempo* (segundos)	3,98	0,66	-83
Nr. Downloads*	5,48	1,73	-68

Nota: * o valor foi dividido pelo número de páginas relevantes recuperadas.

Em média, as abordagens **Orion** e *INDESIT* obtiveram tempo de processamento de 0,66 e 3,98 segundos por página relevante recuperada, respectivamente. O tempo de processamento da abordagem **Orion** foi 83% menor. O Teste T realizado mostra que essa diferença não é estatisticamente significante, pois o valor do p bicaudal calculado ($\approx 0,36$) é maior que o coeficiente de significância adotado ($\alpha = 0,05$). A abordagem **Orion** obteve um tempo de processamento menor que a abordagem *INDESIT* nos sites que conseguiu filtrar a maioria das páginas irrelevantes apenas pelas características de URL, uma vez que o processamento da URL é mais rápido que o do HTML. Nos demais sites, a abordagem *INDESIT* obteve um tempo de processamento menor porque possui uma estratégia para minimizar o número de páginas que têm seu HTML acessado com base no *link-path* dos nodos âncoras que apontam para as páginas.

Em média, as abordagens **Orion** e *INDESIT* baixaram 1,73 e 5,48 páginas por página relevante recuperada, respectivamente. O número de páginas baixadas pela abordagem **Orion** foi 68% menor. O Teste T realizado mostra que essa diferença não é estatisticamente significante, pois o valor do p bicaudal calculado ($\approx 0,39$) é maior que o coeficiente de significância

adotado ($\alpha = 0,05$). A abordagem **Orion** baixou menos páginas nos sites que conseguiu filtrar a maioria das páginas irrelevantes apenas pelas características de URL, uma vez que essas páginas não precisaram ter seu conteúdo acessado. Nos demais sites, a abordagem *INDESIT* baixou menos páginas porque possui uma estratégia para minimizar o número de páginas que têm seu HTML acessado com base no *link-path* dos nodos âncoras que apontam para as páginas.

Os resultados mostram que a abordagem **Orion** é mais eficaz que a abordagem *INDESIT* porque **Orion** foi capaz de: (i) recuperar mais páginas relevantes (revocação); e (ii) recuperar menos páginas irrelevantes (precisão). A abordagem **Orion** foi tão eficiente quanto a abordagem *INDESIT* uma vez que não houve diferença estatisticamente significante entre as duas abordagens com relação ao tempo de processamento e ao número de páginas baixadas.

*4.3.3.2 **Orion** versus GPP*

O objetivo desse experimento é responder as seguintes questões: (i) *qual abordagem é mais eficaz: **Orion** ou GPP?*; e (ii) *qual abordagem é mais eficiente: **Orion** ou GPP?* A avaliação da eficácia foi realizada utilizando as seguintes métricas: revocação e precisão. A Tabela 4.5 apresenta a média da revocação e da precisão das abordagens em cada base de dados e a média geral considerando os sites das três bases de dados.

Tabela 4.5 – Revocação e precisão das abordagens *GPP* e **Orion**.

Bases de dados	**Revocação**			**Precisão**		
	GPP	**Orion**	%	**GPP**	**Orion**	**%**
DATASET_C	0,82	0,97	18	0,72	0,99	37
DATASET_D1	0,52	0,68	29	0,27	0,90	**236**
DATASET_M	0,56	0,82	**47**	0,60	0,93	54
Média	0,67	0,85	27	0,57	0,95	66

Em média, as abordagens **Orion** e *GPP* obtiveram revocação de 0,85 e 0,67, respectivamente. O ganho médio de revocação da abordagem **Orion** foi de 27%. O Teste T realizado mostra que esse ganho é estatisticamente significante, pois o valor do p bicaudal calculado ($\approx 4,18 \times 10^{-03}$) é menor que o coeficiente de significância adotado ($\alpha = 0,05$). Esse ganho ocorreu porque a abordagem **Orion**, ao contrário da abordagem *GPP*, não necessita que os links para as páginas-entidade estejam em listas Web. A revocação da abordagem *GPP* foi afetada tanto nos sites em que o caminho da página principal até as páginas-entidade não era composto por links em listas Web quanto nos sites em que os links estavam em listas aninhadas. O maior ganho de revocação da abordagem **Orion** foi na base de dados $DATASET\_M$ (47%) porque

essa base de dados possui o maior número de sites cujo caminho da página principal até as páginas-entidade não é composto por links em listas Web.

Em média, as abordagens **Orion** e *GPP* obtiveram precisão de 0,95 e 0,57, respectivamente. O ganho médio de precisão da abordagem **Orion** foi de 66%. O Teste T realizado mostra que esse ganho é estatisticamente significante, pois o valor do p bicaudal calculado ($\approx 1,37 \times 10^{-08}$) é menor que o coeficiente de significância adotado ($\alpha = 0,05$). Esse ganho ocorreu porque a abordagem **Orion** avalia as características de URL e de HTML das páginas-índice e das páginas-entidade, enquanto que a abordagem *GPP* avalia as características visuais e as características de HTML apenas das páginas-índice. Páginas irrelevantes cujas páginas-índice possuíam características visuais e de HTML similares à página-índice da página de exemplo foram retornadas pela abordagem *GPP*. Essa situação ocorreu com páginas com fotos, vídeos, documentos, etc. O maior ganho de precisão da abordagem **Orion** ocorreu na base de dados $DATASET\_D1$ (236%) porque a maioria dos sites nessa base de dados possui um *template* rígido que faz com que páginas-índice de páginas-entidade sobre pilotos e páginas-índice de páginas com fotos ou estatísticas tenham características de HTML e características visuais similares. **Orion**, ao contrário de *GPP*, não foi influenciada por essa situação.

A avaliação da eficiência foi realizada utilizando as seguintes métricas: tempo de processamento e número de páginas baixadas. A Tabela 4.6 apresenta a média do tempo de processamento em segundos (*Tempo*) e do número de páginas baixadas (*Nr. Downloads*) das abordagens considerando os sites das três bases de dados. O tempo de processamento e o número de páginas baixadas apresentados são valores relativos, ou seja, correspondem aos valores absolutos divididos pelo número de páginas relevantes que a abordagem recuperou.

Tabela 4.6 – Tempo de processamento e número de páginas baixadas por *GPP* e **Orion**.

Métrica	**GPP**	**Orion**	**%**
Tempo* (segundos)	14,73	0,66	-96
Nr. Downloads*	0,50	1,73	245

Nota: * o valor foi dividido pelo número de páginas relevantes recuperadas.

Em média, as abordagens **Orion** e *GPP* obtiveram um tempo de processamento de 0,66 e 14,73 segundos por página relevante recuperada, respectivamente. O tempo de processamento da abordagem **Orion** foi 96% menor. O Teste T realizado mostra que essa diferença é estatisticamente significante, pois o valor do p bicaudal calculado ($\approx 8,93 \times 10^{-04}$) é menor que o coeficiente de significância adotado ($\alpha = 0,05$). A abordagem **Orion** obteve um tempo de processamento menor que a abordagem *GPP* porque não precisa renderizar as páginas em um

navegador Web para extrair as características utilizadas para analisar as páginas. A abordagem *GPP* utiliza características visuais que são extraídas por meio da renderização das páginas em um navegador Web, carregando imagens, CSS e JavaScripts. A renderização aumentou significativamente o tempo de processamento da abordagem *GPP*.

Em média, as abordagens **Orion** e *GPP* baixaram 1,73 e 0,50 páginas por página relevante recuperada, respectivamente. O Teste T realizado mostra que essa diferença não é estatisticamente significante, pois o valor do p bicaudal calculado ($\approx 0,87$) é maior que o coeficiente de significância adotado ($\alpha = 0,05$). Em média, *GPP* baixou menos páginas que **Orion** porque *GPP* filtra as páginas pelas características visuais e de HTML das páginas-índice. Dessa forma, não é necessário baixar as páginas-entidade. Essa é a razão que faz com que a abordagem *GPP* baixe menos páginas que o número de páginas relevantes recuperadas.

Os resultados mostram que **Orion** é mais eficaz que *GPP* porque **Orion** foi capaz de: (i) recuperar mais páginas relevantes (revocação); e (ii) recuperar menos páginas irrelevantes (precisão). **Orion** foi mais eficiente que *GPP* porque: (i) **Orion** obteve um tempo de processamento menor que *GPP*; e (ii) não houve diferença estatisticamente significante entre as duas abordagens com relação ao número de páginas baixadas.

4.4 Avaliação da etapa de Extração

O objetivo desta seção é apresentar os experimentos realizados para avaliar a eficácia da etapa de *Extração* da abordagem **Orion**. A Subseção 4.4.1 apresenta a configuração dos experimentos. A Subseção 4.4.2 descreve os experimentos realizados e os resultados obtidos.

4.4.1 Configuração dos experimentos

Esta subseção está organizada da seguinte forma. Os *baselines* são apresentados na Subseção 4.4.1.1. A Subseção 4.4.1.2 descreve a metodologia adotada.

4.4.1.1 Baselines

A etapa de *Extração* da abordagem **Orion** foi comparada com três abordagens: (i) *Trinity* (SLEIMAN; CORCHUELO, 2014) - que infere uma expressão regular a partir de uma árvore ternária de prefixos, separadores e sufixos; (ii) *SSM* (CARLSON; SCHAFER, 2008) - que é baseada em um algoritmo de alinhamento parcial de árvores; e (iii) *SWDE* (HAO et al.,

2011) - que é baseada em expressões XPath. A Tabela 4.7 resume as características de *Trinity*, *SSM* e *SWDE* que motivaram a escolha dessas abordagens como *baselines*. As seguintes características foram consideradas, que indicam:

- *Independência de marcações específicas* - se a abordagem não exige que as páginas-entidade em todos os sites possuam marcações HTML específicas;
- *Independência de bases de conhecimento* - se a abordagem não depende de enciclopédias manualmente compiladas (por exemplo, Wikipedia);
- *Independência de supervisão a priori* - se o usuário não necessita fazer anotações antes da execução da abordagem; e
- *Independência de supervisão a posteriori* - se o usuário não necessita fazer anotações após a execução da abordagem.

Tabela 4.7 – Características dos *baselines* da etapa de *Extração*.

Característica	**Trinity**	**SSM**	**SWDE**
Independência de marcações específicas	Sim	Sim	Sim
Independência de bases de conhecimento	Sim	Sim	Sim
Independência de supervisão *a priori*	Sim	Não	Não
Independência de supervisão *a posteriori*	Não	Sim	Sim

4.4.1.2 Metodologia

Os experimentos desta seção avaliam a extração de valores de atributos realizada pela abordagem **Orion**. Para cada site, é fornecido um conjunto de páginas-entidade do site como entrada e os valores que são retornados pela abordagem, como saída, são comparados com um gabarito que contém os valores dos atributos publicados nas páginas-entidade do conjunto.

Os verdadeiros positivos (VP), falsos negativos (FN) e falsos positivos (FP), que são utilizados para calcular as métricas de eficácia (revocação, precisão e F1), foram calculados da seguinte forma: (VP) - quantidade de valores extraídos que estão de acordo com o gabarito; (FN) - quantidade de valores que constam no gabarito e não foram extraídos; e (FP) - quantidade de valores extraídos que não constam no gabarito. Cada ocorrência de atributo multivalorado (por exemplo, um livro com vários autores) teve o tratamento definido por Hao *et al.* (2011): (i) se pelo menos um valor foi extraído, foi contabilizado como um verdadeiro positivo; (ii) se nenhum dos valores foi extraído, foi contabilizado como um falso negativo. Foram escolhidas as bases de dados $DATASET\_S$ e $DATASET\_W$ uma vez que essas bases de dados foram

utilizadas na literatura (HAO et al., 2011; BRONZI et al., 2013; ORTONA et al., 2016) para avaliar diferentes abordagens de extração de dados da Web.

Os resultados das abordagens *SSM* e *SWDE* foram obtidos a partir de Hao *et al.* (2011). Os resultados da abordagem *Trinity* foram obtidos a partir do protótipo fornecido pelos autores[5]. A abordagem *Trinity* possui dois parâmetros (min e max), que representam os tamanhos mínimo e máximo, respectivamente, de padrões compartilhados pelos quais a abordagem procura. Foi utilizada a configuração de parâmetros definida por Sleiman e Corchuelo (2014), ou seja, $min = 1$ e $max = 0,05 \times m$, em que m denota o tamanho em *tokens* da menor página-entidade do site. A abordagem **Orion** possui dois parâmetros: (i) δ - o número de páginas-entidade em que um valor da propriedade *value* deve ocorrer para ser considerado parte do *template*; e (ii) γ - o limiar de poda das consultas inferidas. Foram testados os seguintes valores 10%, 20%, .., 100% para os parâmetros δ e γ nos sites do tipo de entidade *livros* da base de dados $DATASET\_W$. O maior F1 foi obtido utilizando $\delta = 60\%$ e $\gamma = 30\%$. Essa configuração foi empregada em todos os demais experimentos.

O S-graph foi armazenado no Neo4j (Neo Technology, 2016) 3.0.6, que é um banco de dados orientado a grafos. Entretanto, outro banco de dados orientado a grafos que suporte Cypher pode ser utilizado. Outra linguagem de consulta de grafos também pode ser utilizada. Nesse caso, as consultas apresentadas nesta Tese precisam ser mapeadas para a nova linguagem. As literais “TEXT”, “TAG”, “TEMPLATE” e “DATA” foram substituídas por inteiros.

A subetapa de *seleção de consultas* da etapa de *Extração* descreve diferentes alternativas. Nos experimentos, foi adotada a alternativa manual, que é a mesma adotada pela abordagem *Trinity*. Nessa alternativa, um usuário especialista seleciona uma consulta Cypher (um grupo de captura no caso de *Trinity*) de cada site para cada atributo do gabarito. Utilizou-se um especialista simulado, uma vez que as consultas foram comparadas com o gabarito, que foi criado pelos autores das bases de dados.

4.4.2 Descrição dos experimentos

Esta subseção apresenta os experimentos que foram projetados para comparar as abordagens em termos de eficácia. Foram realizados dois experimentos: (i) Subseção 4.4.2.1 - compara a abordagem **Orion** com a abordagem *Trinity*; e (ii) Subseção 4.4.2.2 - compara a abordagem **Orion** com as abordagens *SSM* e *SWDE*.

[5]Disponível em: <http://www.tdg-seville.info/Download.ashx?id=341>. Último acesso em 10/10/2017.

4.4.2.1 ***Orion*** *versus Trinity*

O objetivo desse experimento é responder a seguinte questão: *qual abordagem é mais eficaz:* ***Orion*** *ou* Trinity*?* A eficácia foi medida utilizando a métrica F1. A Tabela 4.8 apresenta a revocação, a precisão e a F1 produzidas pelas abordagens em cada tipo de entidade das bases de dados. Em média, **Orion** e *Trinity* alcançaram um valor de F1 de 0,98 e 0,93, respectivamente. O ganho médio da abordagem **Orion** em termos de F1 foi de 5%. A escala da Web transforma esse percentual em um grande número de valores de atributos que pode ser corretamente extraído. O Teste T mostra que esse ganho é estatisticamente significante porque o valor de p bicaudal ($3,12 \times 10^{-7}$) é menor que o coeficiente de significância adotado ($\alpha = 0,05$).

Tabela 4.8 – Comparação entre **Orion** e *Trinity*.

Base de dados	**Tipo de Entidade**	**Revocação**		**Precisão**		**F1**		
		Trinity	**Orion**	**Trinity**	**Orion**	**Trinity**	**Orion**	**%**
DATASET_S	automóveis	0,96	0,99	0,96	1,00	0,96	0,99	3
	câmeras	0,96	0,91	0,98	1,00	0,97	0,95	-2
	empregos	0,96	0,92	0,97	0,98	0,96	0,94	-3
	filmes	0,97	0,97	0,99	1,00	0,97	0,98	1
	jogadores da NBA	0,95	1,00	0,95	1,00	0,95	1,00	5
	livros	0,90	0,97	0,92	1,00	0,90	0,98	**9**
	restaurantes	0,91	0,96	0,91	1,00	0,91	0,98	7
	universidades	0,93	0,97	0,93	1,00	0,93	0,98	5
DATASET_W	ações	0,92	1,00	0,95	1,00	0,92	1,00	8
	jogadores de futebol	0,92	0,96	0,91	0,97	0,91	0,97	6
	livros	0,86	0,98	0,86	0,99	0,86	0,99	**15**
	videogames	0,93	0,97	0,94	1,00	0,93	0,98	5
Média		0,93	0,97	0,94	0,99	0,93	0,98	5

* 148,326 páginas-entidade de 120 sites reais

Orion superou *Trinity* porque, ao contrário de *Trinity*, **Orion** não é afetado por: (i) atributos omissos (ou seja, atributos cujos valores são publicados apenas em algumas páginas-entidade do site) que estão publicados em elementos simples (ou seja, quando o elemento tem apenas um nodo textual); (ii) atributos publicados em tabelas horizontais (ou seja, o cabeçalho está na primeira linha da tabela); e (iii) páginas-ruído (ou seja, páginas no conjunto de entrada que seguem um *template* diferente da maioria das páginas-entidade do conjunto). *Trinity* erroneamente assume que cada valor de um atributo omisso é parte do valor do atributo imediatamente anterior no HTML da página. *Trinity* não é capaz de extrair corretamente valores de atributos que estão publicados nas colunas do meio de tabelas horizontais porque esses valores de atributo

possuem o mesmo prefixo e sufixo. Nesse caso, *Trinity* incorretamente considera valores de diferentes atributos como sendo valores de um mesmo atributo multivalorado. Uma página-ruído no conjunto de entrada faz com que *Trinity* crie uma árvore ternária incorreta e, consequentemente, gere uma expressão regular errônea, que afeta sua eficácia. Esse comportamento ocorre porque *Trinity* é baseado em padrões compartilhados por todas as páginas-entidade fornecidas como entrada. Entretanto, a tarefa de coletar as páginas-entidade de um site não é trivial e, eventualmente, algumas páginas-ruído acabam sendo coletadas indevidamente.

Orion superou *Trinity* em 83% dos tipos de entidade. **Orion** obteve o maior ganho de F1 no tipo de entidade *livros* (15% na base de dados $DATASET\_W$ e 9% na base de dados $DATASET\_S$). Os sites desse tipo de entidade contêm inúmeros atributos omissos. *Trinity* superou **Orion** nos tipos *câmeras* e *empregos*. Entretanto, a diferença não excedeu 3%. A Seção 4.8 explica os casos de falha da abordagem **Orion** que causaram essa diferença negativa.

É importante destacar que foram realizados experimentos exaustivos com **Orion** e *Trinity*, que incluíram 148.326 páginas-entidade de 120 sites reais de diferentes tipos de entidades. Em média, cada site possui 1.236 páginas-entidade. Os experimentos realizados pelos autores da abordagem *Trinity* incluíram, em média, 37,89 páginas-entidade por site e o número máximo de páginas-entidade de um site foi 252.

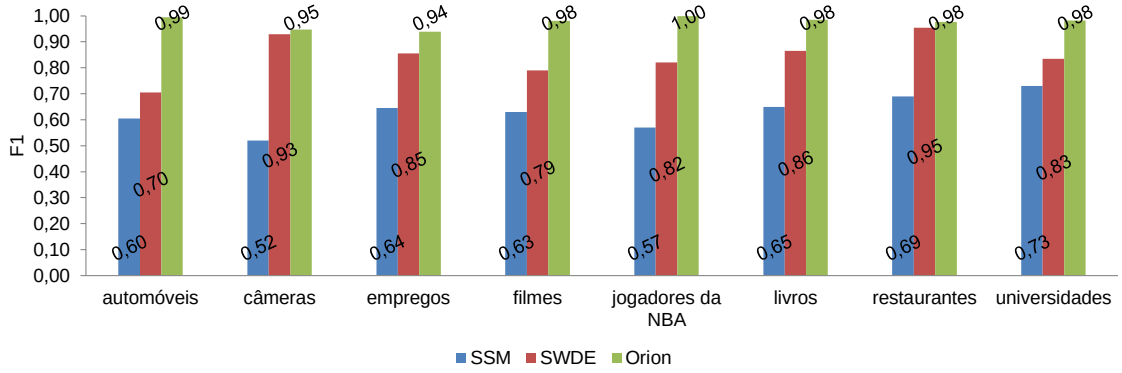
Os resultados mostram que **Orion** tem uma eficácia que é significativamente maior que *Trinity* porque, em média, **Orion** alcançou um valor de F1 maior. A razão para esse ganho é que **Orion**, ao contrário de *Trinity*, não é afetado por páginas-ruído, atributos publicados em tabelas horizontais, nem atributos omissos publicados em elementos simples. Portanto, **Orion** é capaz de extrair valores corretos em mais páginas-entidade e reduzir o número de valores extraídos incorretamente. Além disso, a maioria das subetapas da abordagem **Orion** são implementadas utilizando Cypher, que é uma linguagem declarativa. As linguagens declarativas são de mais alto nível que as linguagens imperativas (SCOTT, 2015).

*4.4.2.2 **Orion** versus SSM e SWDE*

O objetivo desse experimento é responder a seguinte questão: *qual abordagem é mais eficaz:* ***Orion****,* SSM *ou* SWDE*?* A eficácia foi medida utilizando a métrica F1. A Figura 4.4 apresenta os valores de F1 para cada tipo de entidade da base de dados $DATASET\_S$.

Orion superou *SSM* em todos os tipos de entidade. Em média, **Orion** e *SSM* alcançaram um F1 de 0,98 e 0,63, respectivamente. O ganho médio da abordagem **Orion** em termos de F1 foi de 56%, com um pico de 82% no tipo de entidade *câmeras*. Uma das principais

Figura 4.4 – Comparação entre *SSM*, *SWDE* e **Orion**.



razões para esse ganho de **Orion** sobre *SSM* é que *SSM* é baseado em um algoritmo de alinhamento parcial de árvore. Esse algoritmo: (i) é facilmente afetado por desalinhamentos entre as páginas-entidade, uma vez que um alinhamento preciso de árvore ainda é um problema não resolvido (HAO et al., 2011); e (ii) apenas considera informação das marcações HTML, e portanto não diferencia nodos textuais com semânticas distintas, mas caminhos DOM idênticos. Por outro lado, **Orion**: (i) não utiliza um algoritmo de alinhamento de árvore; e (ii) leva em consideração o conteúdo do nodo *template* mais próximo de cada nodo de dados, o que fornece alguma semântica ou contexto dos nodos de dados.

Orion superou *SWDE* em todos os tipos de entidade. Em média, **Orion** e *SWDE* alcançaram uma F1 de 0,98 e 0,84, respectivamente. O ganho médio de **Orion** em termos de F1 foi de 17%, com um pico de 41% no tipo de entidade *automóveis*. Uma das principais razões para esse ganho de **Orion** sobre *SWDE* é que *SWDE* utiliza expressões XPath para agrupar os nodos textuais de diferentes páginas-entidade que extraem os valores do mesmo atributo. Essas expressões XPath iniciam na raiz da árvore DOM e terminam no nodo textual. Informações de contexto (prefixos, sufixos e rótulos) são utilizadas para complementar as expressões XPath. *SWDE* é mais suscetível que **Orion** às variações de *template* porque *SWDE* requer que o caminho da raiz até o nodo textual (que contém o valor de um determinado atributo) seja o mesmo em todas as páginas-entidade, enquanto que **Orion** requer que o caminho do nodo *template* ao nodo textual seja o mesmo.

Outra importante característica que afetou os resultados de *SSM* e *SWDE* foi a necessidade de um usuário anotar um conjunto de treinamento. Essa tarefa é tediosa, demorada e propensa a erros (HERNÁNDEZ et al., 2016). Ressalta-se que os resultados de *SSM* e *SWDE* foram obtidos a partir de Hao *et al.* (2011).

Os resultados mostram que **Orion** possui uma eficácia maior que *SSM* e *SWDE* porque, em média, **Orion** alcançou um valor de F1 maior. A principal razão para esse ganho é a forma

como **Orion** realiza a extração dos valores dos atributos (usando consultas Cypher).

4.5 Avaliação da etapa de Reforço

O objetivo desta seção é apresentar os experimentos realizados para avaliar a eficácia da etapa de *Reforço* da abordagem **Orion**. Essa avaliação verifica a robustez da abordagem **Orion**, uma vez que essa característica está atrelada a capacidade da abordagem proposta em suportar atributos com formato variante. A Subseção 4.5.1 apresenta a configuração dos experimentos. A Subseção 4.5.2 descreve os experimentos realizados e os resultados obtidos.

4.5.1 Configuração dos experimentos

Esta subseção está organizada da seguinte forma. Os *baselines* são apresentados na Subseção 4.5.1.1. A metodologia adotada é descrita na Subseção 4.5.1.2.

4.5.1.1 Baselines

A combinação das etapas de *Extração* e *Reforço* da abordagem **Orion** foi comparada com três *baselines*: (i) apenas a etapa de *Extração* da abordagem **Orion**; (ii) a abordagem *Trinity* (SLEIMAN; CORCHUELO, 2014); e (iii) a etapa de *Extração* da abordagem **Orion** combinada com uma etapa adicional que identifica as consultas complementares manualmente.

4.5.1.2 Metodologia

Os experimentos desta seção avaliam a extração de valores de atributos com formato variante realizada pela abordagem **Orion**. Para cada site, é fornecido um conjunto de páginas-entidade do site como entrada. Os valores que são retornados pela abordagem, como saída, são comparados com um gabarito que contém os valores dos atributos publicados nas páginas-entidade do conjunto. A avaliação é limitada aos atributos com formato variante.

Os verdadeiros positivos (VP), falsos negativos (FN) e falsos positivos (FP), que são utilizados para calcular as métricas de eficácia (revocação, precisão e F1), foram calculados da seguinte forma: (VP) - quantidade de valores extraídos que estão de acordo com o gabarito; (FN) - quantidade de valores que constam no gabarito e não foram extraídos; e (FP) - quantidade de valores extraídos que não constam no gabarito. Cada ocorrência de atributo multivalorado

(por exemplo, um livro com vários autores) teve o tratamento definido por Hao *et al.* (2011): (i) se pelo menos um valor foi extraído, foi contabilizado como um verdadeiro positivo; (ii) se nenhum dos valores foi extraído, foi contabilizado como um falso negativo. Foram escolhidas as bases de dados $DATASET\_S$ e $DATASET\_W$ uma vez que essas bases de dados foram utilizadas na literatura (HAO et al., 2011; BRONZI et al., 2013; ORTONA et al., 2016) para avaliar diferentes abordagens de extração de dados da Web.

A etapa de *Extração* da abordagem **Orion** e a abordagem *Trinity* foram executadas conforme a metodologia detalhada na Seção 4.4.1.2. A função q_{score}, usada pela etapa de *Reforço* da abordagem **Orion**, requer a definição das seguintes funções: (i) f_l - que é utilizada para calcular a Similaridade de Pivô; (ii) f_x - que é utilizada para calcular a Similaridade de Caminho; (iii) f_a - que é utilizada para calcular a Similaridade de Valor em Nível de Atributo; e (iv) f_e - que é utilizada para calcular a Similaridade de Valor em Nível de Entidade. A função Jaro-Winkler (WINKLER, 1999), que compara duas strings, foi utilizada para calcular f_l e f_x. O conjunto de funções de similaridade proposto por Cortez *et al.* (2010), que compara um bloco de texto com uma base de conhecimento, foi adaptado para calcular f_a. Esse conjunto de funções foi adaptado para substituir: (i) o bloco de texto por um valor extraído por uma consulta candidata; e (ii) a base de conhecimento pelos valores extraídos pela consulta selecionada. O conjunto de funções de distância proposto por Bronzi *et al.* (2013), que compara dois valores da mesma entidade extraídos em sites diferentes, foi adaptado para calcular f_e. Esse conjunto de funções foi adaptado para calcular a similaridade, em vez da distância. Essas funções são detalhadas no Apêndice A. A etapa de *Reforço* possui como parâmetros: (i) o peso de cada componente da função q_{score}; e (ii) o limiar de poda β. A configuração de parâmetros adotada é descrita na Subseção 4.5.2.2. A etapa de *Reforço* necessita que cada página esteja associada a um identificador que representa a entidade descrita pela página. Um atributo do gabarito foi escolhido como identificador para cada tipo de entidade. Foram escolhidos os mesmos identificadores utilizados por Bronzi *et al.* (2013). Por exemplo, *ISBN* (*International Standard Book Number* - Número de Identificação Internacional de Livros) foi o atributo utilizado como identificador no tipo de entidade *livros*. Essa tarefa pode ser feita de forma automática utilizando técnicas de expansão de conjuntos (por exemplo, Bronzi *et al.* (2013)).

4.5.2 Descrição dos experimentos

Esta subseção apresenta os experimentos realizados com o intuito de avaliar a eficácia da extração de valores de atributos com formato variante realizada pela abordagem **Orion**. Foram

realizados cinco experimentos: (i) Subseção 4.5.2.1 - analisa a ocorrência de atributos com formato variante nos sites; (ii) Subseção 4.5.2.2 - encontra a melhor configuração de parâmetros para a etapa de *Reforço* da abordagem **Orion**; (iii) Subseção 4.5.2.3 - verifica o impacto da inclusão da etapa de *Reforço* na abordagem **Orion** em termos de eficácia da extração de atributos com formato variante; (iv) Subseção 4.5.2.4 - compara a eficácia de **Orion** com *Trinity*; e (v) Subseção 4.5.2.5 - compara a eficácia da identificação de consultas complementares proposta na abordagem **Orion**, que é automática, com uma identificação manual.

4.5.2.1 Ocorrência de atributos com formato variante

O objetivo desse experimento é responder a seguinte questão: *o tratamento de atributos com formato variante é necessário?* Para responder essa questão, a ocorrência de atributos com formato variante foi analisada nos 120 sites reais que pertencem às bases de dados $DATASET\_S$ e $DATASET\_W$.

A Tabela 4.9 apresenta: (i) *Sites* - o número de sites (Total), o número (FV) e o percentual (%) de sites com pelo menos um atributo com formato variante; (ii) *Atributos* - o número de atributos (Total), o número (FV) e o percentual (%) de atributos com formato variante em pelo menos um site; e (iii) *Páginas-entidade* - o número de páginas-entidade com pelo menos um atributo com formato variante (Total), o número (FV) e o percentual (%) de páginas-entidade que tem pelo menos um valor de atributo com um caminho DOM (caminho da raiz até o nodo textual na árvore DOM) diferente do caminho DOM mais frequente para o atributo no site.

Aproximadamente 47% dos sites têm pelo menos um atributo com formato variante e 88% dos atributos possuem formato variante em pelo menos um site. Um total de 13.622 páginas-entidade publica um ou mais valores de atributos com um caminho DOM diferente do caminho DOM mais frequente para o atributo no site. A etapa de *Extração* da abordagem **Orion** não é capaz de extrair os valores dos atributos corretamente nessas 13.622 páginas-entidade porque seleciona apenas uma consulta Cypher para cada atributo. Uma única consulta Cypher não é capaz de extrair os valores de um atributo em todas as páginas-entidade de um site quando os valores desse atributo são publicados com marcações diferentes. Os demais experimentos dessa seção avaliam apenas os atributos com formato variante nos sites.

Tabela 4.9 – Ocorrência de atributos com formato variante.

Sites			**Atributos**			**Páginas-entidade**		
Total	FV	%	Total	FV	%	Total	FV	%
120	56	47	61	51	88	74.247	13.622	15

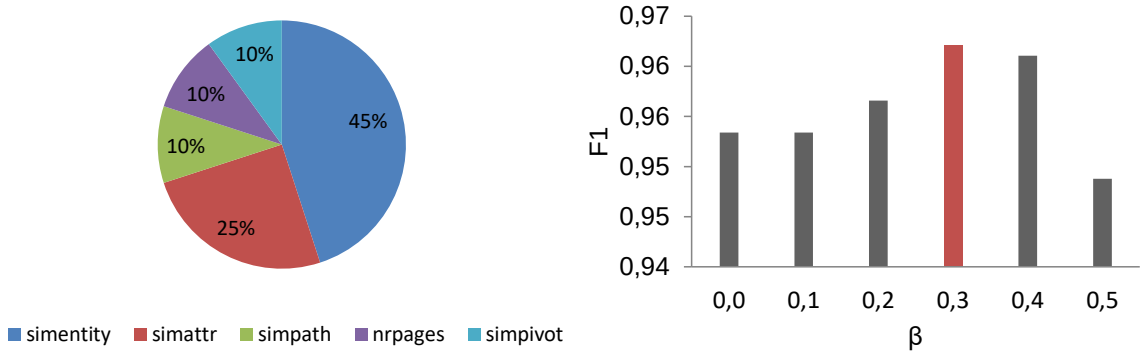
4.5.2.2 Melhor configuração da etapa de Reforço

O objetivo desse experimento é responder a seguinte questão: *qual é a melhor configuração de parâmetros da etapa de Reforço da abordagem* ***Orion****?* Essa questão inclui a definição dos pesos dos componentes da função q_{score} e do limiar de poda β.

Os pesos dos componentes da função q_{score} foram definidos em termos de sua capacidade individual para distinguir consultas complementares das demais consultas. Essa capacidade foi medida utilizando a Razão de Ganho de Informação (*Information Gain Ratio*) (QUINLAN, 1993). Para calcular essa razão, as consultas dos sites que possuíam atributos com formato variante foram manualmente classificadas como complementares ou não. A Razão de Ganho de Informação de cada componente da função q_{score} foi normalizada a fim de garantir que a soma de todas as razões fosse igual a 100%. A Figura 4.5(a) apresenta o peso adotado para cada componente da função q_{score} em todos os demais experimentos. A função de similaridade sim_{entity} teve o maior peso porque ela tem a maior capacidade individual para distinguir consultas complementares das demais consultas. O motivo para essa capacidade maior é que a função sim_{entity} é capaz de, por exemplo, distinguir uma consulta que extrai o time do jogador de futebol descrito em uma página-entidade de uma consulta que extrai um time que está em um menu de navegação. Essa distinção é possível porque a função sim_{entity} compara apenas valores extraídos em páginas que descrevem a mesma entidade.

O valor de β foi definido por meio da eficácia, que foi medida utilizando a métrica F1. Os seguintes valores de β foram testados: 0,0; 0,1; 0,2; 0,3; 0,4; e 0,5. A Figura 4.5(b) apresenta o valor de F1 para cada valor de β. O valor de β que produziu o maior valor de F1 foi 0,3 (destacado em vermelho). Esse valor de β foi utilizado em todos os demais experimentos.

Figura 4.5 – Configuração dos parâmetros da etapa de *Reforço*.



(a) Pesos dos componentes da função q_{score}

(b) F1 para cada valor de β

4.5.2.3 Impacto da etapa de Reforço

O objetivo desse experimento é responder a seguinte questão: *qual é o impacto da inclusão da etapa de Reforço em termos de eficácia da extração de valores de atributos com formato variante?* Para responder essa questão, a abordagem **Orion** foi testada com (**Orion+R**) e sem (**Orion-R**) a etapa de *Reforço*. A eficácia foi medida utilizando a métrica F1.

A Tabela 4.10 apresenta a revocação, a precisão e a F1 de **Orion+R** e **Orion-R**. Em média, **Orion+R** e **Orion-R** produziram um valor de F1 de 0,96 e 0,92, respectivamente. **Orion+R** (ou seja, a inclusão da etapa de *Reforço*) impulsionou o valor de F1 em 5%. O Teste T realizado mostra que esse ganho é estatisticamente significante porque o valor de p bicaudal calculado (2×10^{-06}) é menor que o coeficiente de significância adotado ($\alpha = 0,05$). **Orion+R** obteve o maior ganho em termos de F1 no tipo de entidade *empregos* (21%). Nesse tipo de entidade, 3.963 páginas-entidade publicaram os valores de atributos com caminhos DOM diferentes dos caminhos DOM utilizados para publicar os atributos na maioria das páginas-entidade do site.

Tabela 4.10 – Eficácia da abordagem **Orion** com e sem a etapa de *Reforço*.

Métrica	**Orion-R**	**Orion+R**	**%**
Revocação	0,87	0,95	9
Precisão	0,98	0,98	0
F1	0,92	0,96	5

Os resultados mostram que a etapa de *Reforço* da abordagem **Orion** é benéfica porque aumenta significativamente a eficácia da extração de valores de atributos com formato variante, e, consequentemente, a robustez da abordagem. A etapa de *Reforço* permitiu que **Orion** extraísse corretamente 13.454 valores de atributos a mais que foi possível sem essa etapa. A etapa de *Reforço* impulsionou a revocação em 9% sem afetar a precisão. A escala da Web transforma esse percentual em um grande número de atributos que podem ser corretamente extraídos.

*4.5.2.4 **Orion** versus Trinity*

O objetivo desse experimento é responder a seguinte questão: *qual abordagem é mais eficaz em extrair valores de atributos com formato variante: **Orion** ou* Trinity*?* A eficácia foi medida utilizando a métrica F1. Utilizou-se a abordagem **Orion** composta pelas etapas de *Extração* e *Reforço*, denominada **Orion+R**.

A Tabela 4.11 apresenta a revocação, a precisão e a F1 das abordagens. Em média, as abordagens **Orion+R** e *Trinity* produziram um valor de F1 de 0,96 e 0,84, respectivamente.

O ganho médio da abordagem **Orion+R**, em termos de F1, foi de 14%. O Teste T realizado mostra que esse ganho é estatisticamente significante porque o valor do p bicaudal calculado (1×10^{-04}) é menor que o coeficiente de significância adotado ($\alpha = 0,05$).

Tabela 4.11 – Comparação entre *Trinity* e **Orion**.

Métrica	**Trinity**	**Orion+R**	**%**
Revocação	0,84	0,95	12
Precisão	0,86	0,98	14
F1	0,84	0,96	14

Os resultados mostram que a abordagem **Orion+R** possui uma eficácia significativamente maior que a abordagem *Trinity* para extrair valores de atributos com formato variante. A principal razão para esse ganho é que **Orion**, ao contrário de *Trinity*, não é afetada por atributos publicados em tabelas horizontais nem por atributos omissos (quando publicados em elementos simples). Portanto, **Orion** é capaz de extrair corretamente valores de atributos em mais páginas-entidade e reduzir o número de valores que são extraídos incorretamente.

4.5.2.5 Eficácia da identificação automática de consultas complementares

O objetivo desse experimento é responder a seguinte questão: *a estratégia de identificação de consultas complementares da etapa de Reforço da abordagem **Orion** é eficaz?* Para responder essa questão, foram comparadas duas estratégias de identificação de consultas complementares (automática e manual) em termos de eficácia. A identificação automática é a realizada pela etapa de *Reforço* da abordagem **Orion**. A identificação manual requer que um especialista identifique as consultas complementares. Essa identificação foi realizada utilizando um especialista simulado, ou seja, as consultas foram comparadas com o gabarito, que foi criado pelos autores das bases de dados.

A Tabela 4.12 apresenta a revocação, a precisão e a F1 da abordagem **Orion** para cada estratégia de identificação de consultas complementares (manual e automática). Em média, a abordagem **Orion** produziu um valor de F1 de 0,96 com a estratégia automática e 0,97 com a estratégia manual. A variação percentual foi de apenas 1%. O Teste T realizado mostra que essa diferença é estatisticamente significante, pois o valor do p bicaudal calculado ($\approx 0,001$) é menor que o coeficiente de significância adotado ($\alpha = 0,05$).

Os resultados mostram que a estratégia de identificação automática é efetiva, mas ainda pode ser aperfeiçoada. É importante destacar que a identificação manual não obteve 100% de revocação e 100% de precisão porque a identificação manual, ao contrário da anotação manual

Tabela 4.12 – Comparação entre as estratégias de identificação de consultas complementares.

Métrica	**Manual**	**Automática**	**%**
Revocação	0,96	0,95	-1
Precisão	0,98	0,98	0
F1	0,97	0,96	-1

(gabarito), está sujeita à capacidade de generalização e discriminação das consultas Cypher inferidas na subetapa de *inferência de consultas* da etapa de *Extração*.

4.6 Avaliação da abordagem Orion integral

O objetivo desta seção é apresentar os experimentos realizados para avaliar a eficácia da abordagem **Orion** integralmente. A Subseção 4.6.1 apresenta a configuração dos experimentos. A Subseção 4.6.2 descreve os experimentos realizados e os resultados obtidos.

4.6.1 Configuração dos experimentos

Esta subseção está organizada da seguinte forma. Os *baselines* são apresentados na Subseção 4.6.1.1. A metodologia adotada é descrita na Subseção 4.6.1.2.

4.6.1.1 Baselines

A abordagem **Orion** foi comparada com a composição *INDESIT +Trinity*. Essa composição utiliza a abordagem *INDESIT* para realizar a descoberta de páginas-entidade e a abordagem *Trinity* para realizar a extração dos valores dos atributos. Essa composição foi escolhida porque: (i) a abordagem *INDESIT* foi o *baseline* que obteve a maior eficácia nos experimentos que avaliaram a tarefa de descoberta de páginas-entidade isoladamente (Seção 4.3); e (ii) a abordagem *Trinity* foi o *baseline* que obteve a maior eficácia nos experimentos que avaliaram a tarefa de extração de valores de atributos isoladamente (Seção 4.4).

4.6.1.2 Metodologia

Os experimentos apresentados nesta seção avaliam a abordagem **Orion** integralmente. A entrada é composta por uma página-entidade de cada um dos 10 sites da base de dados $DATASET\_D2$. A saída é composta pelos valores extraídos pela abordagem nas páginas-entidades descobertas, também pela abordagem, a partir da página de exemplo de cada site. Os valores retornados como saída foram comparados com um gabarito que contém os valores dos atributos publicados nas páginas-entidade dos 10 sites da base de dados utilizada.

Os verdadeiros positivos (VP), falsos negativos (FN) e falsos positivos (FP), que são utilizados para calcular as métricas de eficácia (revocação, precisão e F1), foram calculados da seguinte forma: (VP) - quantidade de valores extraídos que estão de acordo com o gabarito; (FN) - quantidade de valores que constam no gabarito e não foram extraídos; e (FP) - quantidade de valores extraídos que não constam no gabarito.

As abordagens **Orion**, *INDESIT* e *Trinity* foram executadas com os parâmetros definidos nas Seções 4.3, 4.4 e 4.5, que avaliaram as etapas de *Descoberta*, *Extração* e *Reforço* isoladamente. A Tabela 4.13 resume os parâmetros adotados.

Tabela 4.13 – Parâmetros adotados.

Abordagem	**Parâmetro**	**Valor**
INDESIT	T	0,9
	Notação	TAG+AN
Trinity	min	1
	max	0,05 * m
Orion	H	2
	δ	60%
	γ	30%
	β	0,3
	Peso da sim_{entity}	45%
	Peso da sim_{attr}	25%
	Peso da sim_{path}	10%
	Peso da nr_{pages}	10%
	Peso da sim_{pivot}	10%

4.6.2 Descrição dos experimentos

Esta subseção apresenta os experimentos realizados com o intuito de avaliar a eficácia da abordagem **Orion** integralmente. Foram realizados dois experimentos: (i) Subseção 4.6.2.1 - analisa a eficácia da abordagem **Orion**; e (ii) Subseção 4.6.2.2 - compara a abordagem **Orion** com a composição *INDESIT +Trinity*.

4.6.2.1 Eficácia

O objetivo desse experimento é responder a seguinte questão: *qual é a eficácia da abordagem **Orion**?* Para responder essa questão, foi avaliada a eficácia da abordagem **Orion** utilizando a métrica F1. Foi considerada a F1 média para cada atributo e para cada site.

A Tabela 4.14 apresenta o valor de F1 da abordagem **Orion** para cada atributo. A abordagem **Orion** produziu valores de F1 entre 0,89 e 0,96. O valor de F1 foi mais alto para o atributo *Equipe*, pois as páginas-entidade de pilotos inativos não publicam esse atributo. Logo, o fato da abordagem **Orion** não encontrar a página-entidade de um piloto inativo não afeta o valor de F1 para esse atributo.

Tabela 4.14 – F1 da abordagem **Orion** por atributo.

Atributo	**F1**
Equipe	0,96
Data de nascimento	0,92
Nome	0,91
Local de nascimento	0,89

A Tabela 4.15 apresenta o valor de F1 da abordagem **Orion** para cada site. A abordagem **Orion** produziu valores de F1 entre 0,79 e 1,00. A abordagem **Orion** alcançou um valor de F1 de 1,00 nos sites *FE*, *Indycar*, *Sky* e *Stockcar*. Esse resultado ocorreu porque: (i) as páginas relevantes desses sites possuem características de HTML e URL distintas das páginas irrelevantes; e (ii) esses sites não publicam páginas-entidade de pilotos inativos ou publicam na mesma árvore-entidade que as páginas-entidade dos pilotos ativos.

Os resultados mostram que a abordagem **Orion** é eficaz porque: (i) o valor médio de F1 foi superior a 0,85 para todos os atributos; e (ii) o valor médio de F1 foi igual ou superior a 0,85 em todos os sites (exceto no site *F1*). Esses resultados foram obtidos porque **Orion** combina características de HTML e URL para identificar as páginas-entidade nos sites e utiliza consultas Cypher para extrair os valores dos atributos presentes nessas páginas.

Tabela 4.15 – F1 da abordagem **Orion** por site.

Site	**F1**
FE	1,00
Indycar	1,00
Sky	1,00
Stockcar	1,00
Euro	0,99
Nascar	0,93
GP2	0,87
GPUPDATE	0,85
F3	0,85
F1	0,79

*4.6.2.2 **Orion** versus INDESIT+Trinity*

O objetivo desse experimento é responder a seguinte questão: *qual é o ganho de eficácia da abordagem **Orion**?* Para responder essa questão, a abordagem **Orion** foi comparada com a composição *INDESIT +Trinity* em termos de eficácia, que foi medida utilizando a métrica F1.

A Tabela 4.16 apresenta a revocação, a precisão e a F1 da abordagem **Orion** e da composição *INDESIT +Trinity*. Em média, **Orion** e *INDESIT +Trinity* produziram um valor de F1 de 0,92 e 0,64, respectivamente. O ganho médio da abordagem **Orion**, em termos de F1, foi de 43%. O Teste T mostra que esse ganho é estatisticamente significante porque o valor do p bicaudal ($3,50 \times 10^{-6}$) é menor que o coeficiente de significância adotado ($\alpha = 0,05$).

Os resultados mostram que a abordagem **Orion** é significativamente mais eficaz que a composição *INDESIT +Trinity*. O ganho de eficácia da abordagem **Orion** ocorreu por dois motivos principais: (i) a abordagem **Orion** foi mais eficaz em descobrir as páginas-entidade que a abordagem *INDESIT* porque considera tanto características de HTML quanto características de URL; e (ii) a abordagem **Orion** foi mais eficaz em extrair os valores dos atributos que a abordagem *Trinity* porque não é afetada por páginas-ruído, atributos publicados em tabelas horizontais nem atributos omissos publicados em elementos simples.

Tabela 4.16 – Comparação entre *INDESIT +Trinity* e **Orion** em termos de eficácia.

Métrica	**INDESIT+Trinity**	**Orion**	**%**
Revocação	0,60	0,89	46
Precisão	0,84	0,97	16
F1	0,64	0,92	43

4.7 Análise geral dos experimentos

A Tabela 4.17 sumariza os experimentos realizados, apresentando a questão e a resposta envolvida em cada experimento. A partir dessa descrição experimental, pode-se afirmar que a regularidade (na URL e no HTML) entre as páginas-entidade do mesmo site pode ser utilizada para a aquisição de valores de atributos de entidades do mundo real. Além disso, os experimentos comprovam que a regularidade de HTML e de URL entre as páginas-entidade do mesmo site pode ser utilizada para descobrir automaticamente as páginas-entidade nos sites, sem a necessidade de definição manual de limiares de similaridade. Esse cenário é promissor visto que a utilização de um limiar de similaridade fixo para todos os sites de uma base de dados não se mostrou efetivo. Os experimentos também mostram que podem ser utilizadas consultas Cypher em um banco de dados orientado a grafos para extrair os valores dos atributos publicados nas páginas-entidade. Essa forma de extração é intuitiva uma vez que Cypher é uma linguagem declarativa. As linguagens declarativas são mais alto nível que as linguagens imperativas porque as linguagens declarativas focam em "o que" o computador deve fazer enquanto que as linguagens imperativas focam em "como" o computador deve fazer (SCOTT, 2015).

Descobrir as páginas-entidade de um mesmo tipo em um site demonstrou-se ser uma tarefa complexa porque: (i) utilizar apenas características de HTML não é suficiente para distinguir as páginas-entidade das demais páginas em alguns sites; (ii) utilizar características visuais aumenta significativamente o tempo de processamento; e (iii) não há um limiar fixo de similaridade que permite distinguir as páginas-entidade das demais páginas em todos os sites de uma base de dados. Assim, foi proposta uma etapa da abordagem que descobre as páginas-entidade combinando características de URL com características de HTML. Nessa etapa, os limiares de similaridade de URL e de HTML são aprendidos automaticamente em cada site. Os termos de URL recebem diferentes pesos de acordo com sua capacidade de distinguir páginas relevantes de páginas irrelevantes. Os experimentos mostram que a abordagem proposta foi capaz de aumentar significativamente a eficácia da descoberta de páginas-entidade.

Tabela 4.17 – Resumo dos experimentos.

Questão	Resposta
Qual a melhor configuração de parâmetros da etapa de descoberta de páginas-entidade da abordagem **Orion** em cada base de dados?	H=4 na base de dados DATASET_M H=1 na base de dados DATASET_C H=2 na base de dados DATASET_D1
Qual a melhor configuração de parâmetros da abordagem *INDESIT* em cada base de dados?	Notação = Tag + AN nas três bases de dados T=0,6 na base de dados DATASET_M T=0,8 na base de dados DATASET_C T=0,9 na base de dados DATASET_D1
Qual a melhor configuração de parâmetros da abordagem *GPP* em cada base de dados?	K=3 na base de dados DATASET_M K=2 na base de dados DATASET_C K=3 na base de dados DATASET_D1
Qual abordagem é mais eficaz em descobrir páginas-entidade: **Orion** ou *INDESIT?*	A revocação de **Orion** foi 37% maior que a de *INDESIT* A precisão de **Orion** foi 51% maior que a de *INDESIT*
Qual abordagem é mais eficiente em descobrir páginas-entidade: **Orion** ou *INDESIT?*	Não houve diferença estatisticamente significante Não houve diferença estatisticamente significante
Qual abordagem é mais eficaz em descobrir páginas-entidade: **Orion** ou *GPP?*	A revocação de **Orion** foi 27% maior que a de *GPP* A precisão de **Orion** foi 66% maior que a de *GPP*
Qual abordagem é mais eficiente em descobrir páginas-entidade: **Orion** ou *GPP?*	O tempo de processamento de **Orion** foi 96% menor que o de *GPP* Não houve diferença estatisticamente significante
Qual abordagem é mais eficaz em extrair valores de atributos: **Orion** ou *Trinity?*	A F1 de **Orion** foi 5% maior que a de *Trinity*
Qual abordagem é mais eficaz em extrair valores de atributos: **Orion** ou *SSM?*	A F1 de **Orion** foi 56% maior que a de *SSM*
Qual abordagem é mais eficaz em extrair valores de atributos: **Orion** ou *SWDE?*	A F1 de **Orion** foi 17% maior que a de *SWDE*
O tratamento de atributos com formato variante é necessário?	Sim porque 47% dos sites possuem pelo menos um atributo com formato variante
Qual é a melhor configuração de parâmetros da etapa de *Reforço* da abordagem **Orion?**	$\beta = 0,3$ Peso do sim_{entity} = 45% Peso do sim_{attr} = 25% Peso do sim_{path} = 10% Peso do nr_{pages} = 10% Peso do sim_{pivot} = 10%
Qual o impacto da inclusão da etapa de *Reforço* em termos de eficácia da extração de valores de atributos com formato variante?	O valor de F1 de **Orion+R** foi 5% maior que o de **Orion-R**
Qual abordagem é mais eficaz em extrair valores de atributos com formato variante: **Orion+R** ou *Trinity*?	O valor de F1 de **Orion+R** foi 14% maior que o de *Trinity*
A estratégia de identificação de consultas complementares da etapa de *Reforço* da abordagem **Orion** é efetiva?	O valor de F1 utilizando a identificação de regras complementares da abordagem **Orion** foi apenas 1% menor que utilizando uma identificação manual
Qual a eficácia da abordagem **Orion** integral**?**	**Orion** produziu um valor médio de F1 superior a 0,85 em todos os sites (1 exceção)
Qual o ganho de eficácia da abordagem **Orion** integral?	O valor de F1 de **Orion** foi 43% maior que o da composição *INDESIT+Trinity*

Extrair os valores dos atributos publicados nas páginas-entidade também demonstrou ser uma tarefa complexa porque: (i) alguns atributos são publicados apenas em algumas páginas-entidade do site; (ii) atributos publicados nas colunas do meio de tabelas horizontais possuem o mesmo prefixo e sufixo; e (iii) a tarefa de descoberta de páginas-entidade não é precisa e, portanto, páginas-ruído também fazem parte do conjunto de entrada da etapa de extração. A abordagem proposta inclui uma etapa de extração de valores de atributos a partir de consultas Cypher em um banco de dados orientado a grafos. As páginas-entidade são inseridas em um banco de dados orientado a grafos e as consultas são inferidas automaticamente. Os experimentos mostram que essa abordagem foi capaz de aumentar significativamente a eficácia da extração de valores de atributos.

Verificou-se que a etapa de extração da abordagem **Orion** não é efetiva para extrair valores de atributos com formato variante uma vez que uma única consulta não é capaz de extrair valores do mesmo atributo publicados em caminhos DOM diferentes. A solução proposta foi adicionar uma etapa de reforço para tratar os atributos com formato variante. A etapa de reforço revisa as consultas inferidas na etapa de extração e identifica automaticamente consultas complementares. Os experimentos mostram que a etapa de reforço foi capaz de aumentar significativamente a eficácia da extração de valores de atributos com formato variante.

4.8 Análise dos casos de falha

Os casos de falha da abordagem **Orion** foram analisados a fim de melhor entender os resultados. Essa análise é útil para desenvolvedores de novas abordagens de aquisição de valores de atributos a partir de páginas-entidade baseadas em *template* porque mostra os desafios de manipular as particularidades desse tipo de página.

Esta seção está organizada da seguinte forma. A Subseção 4.8.1 descreve os casos de falha com relação à tarefa de descobrir as páginas-entidade. A Subseção 4.8.2 apresenta os casos de falha com relação à tarefa de extrair os valores dos atributos das páginas-entidade.

4.8.1 Descoberta das páginas-entidade

Esta subseção apresenta os casos de falha da etapa de *Descoberta* da abordagem **Orion**. Os casos de falha são descritos, bem como os sites das bases de dados $DATASET\_M$, $DATASET\_C$ e $DATASET\_D1$ para os quais esses casos ocorreram, o comportamento

da abordagem nesses casos e as possíveis adaptações da abordagem para tratar cada caso.

O primeiro caso de falha ocorre quando as páginas relevantes estão distribuídas em diferentes árvores-entidade no site. A revocação é afetada porque a abordagem **Orion** encontra apenas páginas-entidade na mesma árvore-entidade que a página de exemplo. As páginas relevantes que estão em outras árvores-entidade não são recuperadas. Essa situação ocorreu nos sites: *F1*, *F3*, *GP2*, *Motor GP* e *Stanford EE*. Nos sites *F1*, *F3*, *GP2* e *Motor GP*, as páginas-entidade de pilotos ativos estão em uma árvore-entidade e as páginas-entidade de pilotos inativos estão em outra árvore-entidade. No site *Stanford EE*, as páginas-entidade sobre professores estão em uma árvore-entidade e as páginas-entidade sobre técnico-administrativos estão em outra árvore-entidade. Uma solução para esse problema é fornecer uma página de exemplo para cada árvore-entidade.

O segundo caso de falha ocorre quando as páginas relevantes são apontadas na página-índice por diferentes *link-paths*. A revocação é afetada porque a abordagem **Orion** recupera apenas as páginas-entidade apontadas na página-índice pelo mesmo *link-path* que a página de exemplo. Essa situação ocorreu nos sites: *Olympic- S* e *Recife*. Por exemplo, no site *Recife*, os links para todas as páginas relevantes estavam na mesma tabela HTML. Entretanto, alguns deles tinham a marcação <*p*> em seus *link-paths* e outros não. Uma solução para esse problema é usar uma estratégia mais flexível que considere páginas apontadas por outros *link-paths* desde que tenham URL e HTML similares à página de exemplo.

O terceiro caso de falha ocorre quando as páginas irrelevantes possuem URL e HTML similares às páginas relevantes. A precisão é afetada porque a abordagem **Orion** recupera páginas irrelevantes. Essa situação ocorreu nos sites: *Campo Grande*, *Champ*, *F1* e *Olympic-S*. Por exemplo, no site *F1*, as páginas-entidade de equipes e as páginas-entidade de pilotos possuem URL e HTML similares. Uma solução para esse problema é considerar outras características, como, por exemplo, características de conteúdo textual.

O quarto caso de falha ocorre quando as páginas relevantes possuem um HTML muito diferente. A precisão é afetada porque a abordagem **Orion** determina um limiar de similaridade de HTML baixo. Com isso, algumas páginas irrelevantes também satisfazem o limiar. Essa situação normalmente ocorre com páginas-entidade sem *template*. Por exemplo, páginas-entidade de professores que são criadas manualmente por cada professor. Esse tipo de página não é o foco desta Tese. Entretanto, o site *MIT EECS* tem um atributo opcional *bibliografia*, onde é possível adicionar conteúdo textual e marcações HTML, ou seja, o usuário pode mudar a estrutura HTML da página. A precisão não foi afetada nesse site porque a abordagem **Orion** utiliza também características de URL para distinguir as páginas relevantes das páginas irrelevantes.

Os casos de falha não foram tratados na Tese, pois os resultados da descoberta de páginas-entidade da abordagem **Orion** são satisfatórios e atendem aos objetivos propostos. Além disso, a abordagem **Orion** superou os dois *baselines* em termos de eficácia.

4.8.2 Extração dos valores dos atributos

Esta subseção apresenta os casos de falha relacionados à tarefa de extração de valores de atributos da abordagem **Orion**, ou seja, compreende as etapas de *Extração* e *Reforço*. Os casos de falha são descritos, bem como o comportamento da abordagem nesses casos e as possíveis adaptações da abordagem para tratar cada caso.

O primeiro caso de falha ocorre quando os valores de um atributo são publicados em nodos textuais que também contêm outras informações. Por exemplo, em alguns sites, o rótulo e o valor do atributo são publicados no mesmo nodo textual. Em outros sites, os valores de dois atributos são publicados no mesmo nodo textual. Nesse caso, a abordagem **Orion** extrai ruído com os valores do atributo porque não segmenta o conteúdo dos nodos textuais. Uma solução para esse problema é aplicar um pós-processamento para segmentar os valores extraídos e eliminar o ruído, como realizado por Ortona *et al.* (2016) ou Hertzog (2016).

O segundo caso de falha ocorre em sites cujos atributos são publicados em elementos mistos (elementos com dois ou mais nodos textuais). A abordagem **Orion** extrai valores incorretos quando os atributos são publicados em elementos mistos e uma das seguintes variações ocorre: (i) atributos omissos - atributos cujos valores são publicados apenas em algumas das páginas-entidade do site; (ii) atributos multivalorados - mais de um valor é publicado para o mesmo atributo da mesma entidade; e (iii) atributos com múltiplas ordens - os atributos são publicados em diferentes ordens nas páginas-entidade do site. Essas variações não afetaram a eficácia da abordagem **Orion** quando os valores de atributo estavam publicados em elementos simples (ou seja, elementos com apenas um nodo textual). Uma solução é aplicar um pré-processamento nos elementos mistos.

O terceiro caso de falha ocorre na extração de atributos com formato variante quando a etapa de *Reforço* não encontra as consultas complementares. Esse caso de falha ocorreu em três tipos de entidades: *câmeras*, *empregos* e *filmes*. No tipo de entidade *câmeras*, o atributo usado como identificador (*modelo*) não é único. Diferentes entidades de diferentes sites possuem o mesmo identificador e foram tratadas como se fossem a mesma entidade. No tipo de entidade *empregos*, os sites não compartilham entidades, ou seja, nenhuma oferta de emprego está publicada em mais de um site. No tipo de entidade *filmes*, diferentes sites publicam diferentes

valores para o mesmo atributo da mesma entidade. Por exemplo, o valor do atributo *gênero* para um determinado filme é *comédia* em um site e *romance* em outro site. A Similaridade de Valor em Nível de Entidade e, consequentemente, a identificação de consultas complementares não foram efetivas nesses casos. Uma solução é substituir a Similaridade de Valor em Nível de Entidade por outra função de similaridade quando os sites não compartilham entidades.

Os casos de falha não foram tratados na Tese, pois os resultados da extração de valores de atributos da abordagem **Orion** são satisfatórios e atendem aos objetivos propostos. Além disso, a abordagem **Orion** superou os *baselines* em termos de eficácia.

5 CONCLUSÃO

Neste item da Tese, é apresentada a conclusão. Assim, para efeito de fechamento deste trabalho os resultados obtidos e as contribuições da Tese são sumarizados e relacionados à produção científica resultante. Por fim, são discutidos alguns trabalhos futuros.

Esta Tese apresentou uma abordagem, denominada **Orion**, para aquisição de valores de atributos de entidades do mundo real por meio da descoberta e da extração de dados de páginas-entidade baseadas em *template*. **Orion** supera os *baselines* em termos de eficácia e é robusta porque inclui o tratamento de atributos com formato-variante. Dentre as contribuições da pesquisa realizada durante o doutorado, destacam-se:

1. Uma ampla revisão bibliográfica sobre os assuntos relacionados à Tese e comparativos entre os principais trabalhos relacionados ao tema nela abordado. Essa revisão incluiu trabalhos - (i) específicos de descoberta de páginas-entidade; (ii) de classificação e agrupamento que podem ser adaptados para a descoberta de páginas-entidade; (iii) específicos para extração de dados de páginas-entidade; (iv) de extração de dados em outros tipos de páginas; e (v) que realizam tanto a descoberta de páginas-entidade quanto a extração dos dados publicados nessas páginas. Os trabalhos estudados foram comparados entre si e também com a abordagem proposta por meio de um conjunto de critérios, que foram selecionados com o intuito de avaliar se os trabalhos podem ser efetivamente aplicados em cenários reais.

2. A especificação de uma etapa de descoberta de páginas-entidade, baseada em características de HTML e URL. Essa etapa é realizada de forma automática e independente de - (i) limiares manuais de similaridade entre páginas; (ii) renderização; (iii) de tipo de entidade; (iv) marcação HTML específica; (v) motor de busca; (vi) base de conhecimento; e (vii) contribuição colaborativa. A especificação da etapa de descoberta de páginas-entidade envolveu a definição de uma nova função de similaridade baseada em características de URL, um algoritmo para identificar páginas-índice; e um algoritmo para filtrar páginas com base na similaridade, que determina automaticamente os limiares de similaridade em cada site.

3. A avaliação da eficácia e da eficiência da etapa de descoberta de páginas-entidade. Essa avaliação incluiu a comparação dos resultados da etapa proposta com dois *baselines* utilizando bases de dados reais. Foram realizados experimentos para encontrar a melhor configuração de parâmetros dos *baselines* em cada base de dados. A etapa proposta foi mais eficaz e mais eficiente que um dos *baselines* e mais eficaz e tão eficiente quanto o outro *baseline*.

4. A análise dos casos de falha da etapa de descoberta de páginas-entidade. Essa análise é importante para o desenvolvimento de novas abordagens porque mostra as dificuldades em

descobrir as páginas relevantes em alguns sites, como é o caso daqueles em que as páginas relevantes estão distribuídas em diferentes árvores-entidade.

As contribuições citadas nos itens 2 a 4 estão publicadas nos seguintes artigos:

- MANICA, E.; GALANTE, R.; DORNELES, C. F. SSUP - a URL-based method to entity-page discovery. In: INTERNATIONAL CONFERENCE ON WEB ENGINEERING, 14., Toulouse, France, 2014. **Proceedings...** Cham, Switzerland: Springer, 2014. p. 254–271 (Qualis-CC-B1);
- MANICA, E.; DORNELES, C. F.; GALANTE, G. Combining URL and HTML features for entity discovery on the Web. **ACM Transactions on the Web**, Submetido para avaliação em fevereiro de 2017 (Qualis-CC-A2).

5. A especificação de uma etapa de extração de dados em páginas-entidade baseadas em *template*, que é inspirada pelos bancos de dados orientados a grafos e linguagens de consulta em grafo. Essa etapa representa as páginas-entidade como um grafo. A etapa de extração infere automaticamente um conjunto de consultas Cypher sobre o grafo, onde cada consulta extrai os valores de um atributo nas páginas-entidade de um site. Essa etapa possui semântica declarativa, ou seja, sua especificação é desacoplada de sua implementação. Além disso, a linguagem Cypher é mais robusta que a linguagem XPath uma vez que ela permite percorrer, consultar e atualizar o grafo, enquanto que a linguagem XPath é uma linguagem específica para percorrer um documento XML ou equivalente.

6. A avaliação da eficácia da etapa de extração. Essa avaliação incluiu a comparação dos resultados da etapa proposta com três *baselines* por meio de bases de dados reais de diferentes tipos de entidade, incluindo mais de 145 mil páginas-entidade. A etapa proposta foi mais eficaz do que os três *baselines*, obtendo um ganho em F1 entre 5% e 56%.

7. A análise dos casos de falha da etapa de extração. Essa análise foi importante para o desenvolvimento de uma etapa de reforço que melhorou a eficácia dos resultados.

As contribuições citadas nos itens 5 a 7 estão publicadas no seguinte artigo:

- MANICA, E.; DORNELES, C. F.; GALANTE, R. Orion: a cypher-based Web data extractor. In: INTERNATIONAL CONFERENCE ON DATABASE AND EXPERT SYSTEMS APPLICATIONS, 28., Lyon, France, 2017. **Proceedings...** Cham, Switzerland: Springer, 2017 (Qualis–CC–B1).

8. A especificação de uma etapa de reforço a fim de incluir o tratamento de atributos com formato variante. Foi identificado que a etapa de extração proposta (que é baseada em consultas Cypher) assim como as abordagens de extração de dados baseadas em XPath não são efetivas

para extrair valores de atributos com formato variante. Essa lacuna de pesquisa foi resolvida por meio de uma etapa de reforço. Essa etapa reanalisa as consultas Cypher geradas pela etapa de extração a fim de identificar aquelas que devem ser combinadas para extrair os valores de um atributo em todas as páginas-entidade do site. Essa identificação é baseada em uma nova função de ranqueamento que leva em consideração diferentes características das consultas Cypher. A etapa de reforço foi projetada para ser acoplada em uma abordagem que realiza a extração de dados baseada em consultas Cypher. No entanto, ela também foi adaptada para ser integrada a uma abordagem de extração de dados baseado em XPath.

9. A avaliação da eficácia da extração após a inclusão da etapa de reforço. Foi realizado um experimento para analisar a ocorrência de atributos com formato variante em sites reais. Esse experimento mostrou que aproximadamente 47% dos sites possuem pelo menos um atributo com formato variante. Foram realizados experimentos para encontrar a melhor configuração de parâmetros para a etapa de reforço. Foi realizado um experimento para verificar o ganho da inclusão da etapa de reforço na abordagem **Orion**, que mostrou que a etapa de reforço impulsionou a revocação em 9% sem afetar a precisão. Foi realizado um experimento que comparou **Orion** incluindo a etapa de reforço com um *baseline* que também suporta atributos com formato variante. O ganho médio em F1 da abordagem **Orion** foi de 14%.

10. A análise dos casos de falha da etapa de reforço. Essa análise é importante para o desenvolvimento de novas abordagens porque mostra as dificuldades em extrair atributos com formato variante em sites que descrevem determinados tipos de entidade, como nos tipos de entidade em que os sites não compartilham entidades.

As contribuições citadas nos itens 8 a 10 estão publicadas nos seguintes artigos:

- MANICA, E.; DORNELES, C. F.; GALANTE, R. A framework for entity-page discovery and attribute extraction. In: BRAZILIAN SYMPOSIUM ON DATABASES, 30., Petrópolis, Brazil, 2015. **Proceedings...** [S.l.], 2015;
- MANICA, E.; DORNELES, C. F.; GALANTE, R. R-extractor: a method for data extraction from template-based entity-pages. In: ANNUAL COMPUTER SOFTWARE AND APPLICATIONS CONFERENCE, 41., Torino, Italy, 2017. **Proceedings...** Washington, EUA: IEEE Computer Society, 2017 (Qualis–CC–A2).

11. A avaliação da eficácia da abordagem **Orion** para aquisição de valores de atributos. Nesse experimento, a abordagem **Orion** foi avaliada de forma integral e comparada com um *baseline*. Os resultados mostraram que a abordagem **Orion** é significativamente mais eficaz do que o *baseline*. O ganho médio da abordagem **Orion**, em termos de F1, foi de 43%.

Além dos artigos citados anteriormente, outros trabalhos foram desenvolvidos explorando assuntos indiretamente relacionados a esta Tese:

- MANICA, E.; DORNELES, C. F.; GALANTE, G. Handling temporal information in websearch engines. **SIGMOD Record**, v. 41, n. 3, p. 15–23, 2012 (Qualis-CC-A1);
- COLPO, M. P.; MANICA, E.; GALANTE, R. Um método para a identificação e a busca de páginas-objeto. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, 28., Recife, Brasil, 2013. **Anais...** [S.l.], 2013.
- CORREA, A. B. et al. Uso de expressões temporais em busca na Web: Uma análise através das sugestões de consulta. In: ESCOLA REGIONAL DE BANCO DE DADOS, São Francisco do Sul, Brasil, 2014. **Anais...** [S.l.], 2014;
- CORREA, Y. ; MENEGAIS, D. A. F. N. ; MANICA, E.; SILVA, F. P. ; SCHNAID, F. ; BIASUZ, M. C. V. Un objeto de aprendizaje multimodal sobre la perspectiva de Gagné. In: Objetos de aprendizaje multimodales: proyectos y aplicaciones, 1ª ed., Barcelona: UOC, 2014, v. 1, p. 41-54;
- CORREA, Y.; MENEGAIS, D. A. F. N.; MANICA, E.; SILVA, F. P.; SCHNAID, F.; BIASUZ, M. C. V. Um objeto de aprendizagem multimodal sob a ótica de Gagné. In: Objetos de aprendizagem multimodais: projetos e aplicações. 1ª ed., Barcelona: UOC, 2014, v. 1, p. 41-54.

Por fim, a experiência adquirida durante o doutorado permitiu contribuir diretamente com o desenvolvimento dos seguintes trabalhos de conclusão de curso e dissertação:

- CERATTI, V. Consultas temporais em banco de dados de grafos. 2014. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Universidade Federal do Rio Grande do Sul. Orientadora: Profª Renata Galante. Co-orientador: Edimar Manica;
- COLPO, M. OPIS: Um método para identificação e busca de páginas-objeto. 2014. Dissertação (Mestrado em Computação) - Universidade Federal do Rio Grande do Sul. Orientador: Renata Galante. Colaborador: Edimar Manica;
- SERAFINI, R. Extração de dados usando redundância de conteúdo. 2015. Trabalho de conclusão de Curso (Graduação em Ciência da Computação) - Universidade Federal do Rio Grande do Sul. Orientadora: Profª Renata Galante. Co-orientador: Edimar Manica;
- HERTZOG, I. L. Um novo método para identificação e correção de ruído em extração de dados. 2016. Trabalho de conclusão de Curso (Graduação em Ciência da Computação) - Universidade Federal do Rio Grande do Sul. Orientadora: Profª Renata Galante. Co-orientador: Edimar Manica;

- CATARINA, R. S.. 2017. TEWS - Um serviço Web para extração de dados de páginas-entidade baseadas em template. Trabalho de Conclusão de Curso (Especialização em Engenharia de Software) - Universidade Federal do Rio Grande do Sul. Orientadora: Profª Renata Galante. Co-orientador: Edimar Manica.

Ao longo do desenvolvimento deste trabalho, alguns aspectos da abordagem proposta foram identificados como passíveis de melhorias ou extensões. Assim, destaca-se que é possível melhorar a eficácia dos resultados a partir de um pós-processamento dos valores extraídos. Ortona *et al.* (2016) e Hertzog (2016), por exemplo, segmentam os valores extraídos a fim de eliminar o ruído. Outra melhoria que pode aumentar a eficácia da abordagem **Orion** é substituir a Similaridade de Valor em Nível de Entidade na etapa de reforço por outra função de similaridade quando os sites não compartilham entidades, uma vez que a solução atual depende de que os diferentes sites compartilhem um número mínimo de entidades.

A abordagem **Orion** descobre apenas páginas-entidade da Web de superfície. No entanto, a etapa de extração de **Orion** é capaz de extrair valores de atributos de páginas-entidade tanto da Web de superfície quanto da Web oculta. Com isso, é possível incorporar uma etapa de descoberta de páginas-entidade da Web oculta a fim de aumentar a abrangência da abordagem **Orion**. Furche *et al.* (2014) e He *et al.* (2013) descrevem formas para coletar páginas-entidade em sites da Web oculta, que podem ser incorporadas à abordagem **Orion**.

Outra tarefa que pode ser realizada é a ampliação dos experimentos. Uma possibilidade é a realização de um experimento que compare as diferentes alternativas de seleção de consultas apresentadas na Subseção 3.3.6, uma vez que esta Tese inclui apenas uma comparação teórica das alternativas. Outra possibilidade é realizar um experimento para avaliar a eficiência das etapas de extração e reforço. Os experimentos sobre essas duas etapas avaliaram apenas a eficácia. Apenas a etapa de descoberta foi avaliada em termos de eficácia e eficiência.

Por fim, a abordagem **Orion** pode ser utilizada para construir uma base de conhecimento sobre entidades. Apesar do tamanho das bases de conhecimento existentes, elas não estão completas (DONG et al., 2014). Além disso, as bases de conhecimento existentes possuem apenas as entidades mais populares. Um experimento de Dalvi, Machanavajjhala e Pang (2012) mostrou que entidades menos populares podem ter grande valor para os usuários da Web. Uma vez criada a base de conhecimento, é possível desenvolver uma ferramenta que forneça consultas diretas sobre entidades do mundo real. Consulta direta é um recurso de busca em que conteúdo relevante é fornecido diretamente na página de resultado (CHILTON; TEEVAN, 2011). A ferramenta pode incluir as entidades que não possuem o recurso de consulta direta disponível nos motores de busca - por exemplo, uma consulta para obter informações sobre um determi-

nado vereador de uma cidade brasileira. Um estudo, realizado por Chilton e Teevan (2011), mostrou o valor das consultas diretas para os usuários da Web por meio de experimentos. Ao disponibilizar uma ferramenta de consultas diretas sobre entidades do mundo real é necessário incorporar um tratamento adequado para a informação temporal, uma vez que os valores de inúmeros atributos evoluem com o tempo. Nesse contexto, Jin *et al.* (2008) apresentam um método para selecionar a expressão temporal que descreve melhor as informações publicadas em uma página. Essa técnica pode ser adaptada para identificar o tempo associado às informações descritas nas páginas-entidade que são extraídas para popular a base de conhecimento.

As contribuições apresentadas no item 11 e os resultados de alguns dos trabalhos futuros serão submetidos a um periódico internacional no início de 2018. Além disso, será desenvolvido um projeto de cooperação com o IFRS (Instituto Federal do Rio Grande do Sul) para dar continuidade à pesquisa apresentada nesta Tese.

REFERÊNCIAS

AGRAWAL, R.; SRIKANT, R. Fast algorithms for mining association rules in large databases. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, 20., Santiago de Chile, Chile, 1994. **Proceedings...** San Francisco, USA: Morgan Kaufmann, 1994. p. 487–499.

AHN, L. von; DABBISH, L. Labeling images with a computer game. In: CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS, Vienna, Austria, 2004. **Proceedings...** New York, USA: ACM, 2004. p. 319–326.

ANDERSON, T.; FINN, J. **The New Statistical Analysis of Data**. New York, USA: Springer-Verlag, 1996.

ANGLES, R.; GUTIERREZ, C. Survey of graph database models. **ACM Computing Surveys**, v. 40, n. 1, p. 1–39, 2008.

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Modern Information Retrieval: The Concepts and Technology Behind Search**. 2nd. ed. Harlow, England: Pearson Education Ltd., 2011.

BLANCO, L.; CRESCENZI, V.; MERIALDO, P. Efficiently locating collections of web pages to wrap. In: INTERNATIONAL CONFERENCE ON WEB INFORMATION SYSTEMS AND TECHNOLOGIES, 1., Miami, USA, 2005. **Proceedings...** [S.l.]: INSTICC Press, 2005. p. 247–254.

BLANCO, L. et al. Supporting the automatic construction of entity aware search engines. In: INTERNATIONAL WORKSHOP ON WEB INFORMATION AND DATA MANAGEMENT, 10., Napa Valley, California, USA, 2008. **Proceedings...** New York, USA: ACM, 2008. p. 149–156.

BLANCO, L.; DALVI, N. N.; MACHANAVAJJHALA, A. Highly efficient algorithms for structural clustering of large websites. In: INTERNATIONAL CONFERENCE ON WORLD WIDE WEB, 20., Hyderabad, India, 2011. **Proceedings...** New York, USA: ACM, 2011. p. 437–446.

BRONZI, M. et al. Extraction and integration of partially overlapping web sources. **Proceedings of the VLDB Endowment**, v. 6, n. 10, p. 805–816, 2013.

CARLSON, A.; SCHAFER, C. Bootstrapping information extraction from semi-structured web pages. In: EUROPEAN CONFERENCE ON MACHINE LEARNING AND KNOWLEDGE DISCOVERY IN DATABASES, Antwerp, Belgium, 2008. **Proceedings...** Berlin, Heidelberg: Springer-Verlag, 2008. p. 195–210.

CATTELL, R. Scalable sql and nosql data stores. **SIGMOD Record**, v. 39, n. 4, p. 12–27, 2011.

CHANG, C.-H. et al. A survey of web information extraction systems. **IEEE Transactions on Knowledge and Data Engineering**, v. 18, n. 10, p. 1411–1428, 2006.

CHEN, L. et al. Aggregating semantic annotators. **Proceedings of the VLDB Endowment**, v. 6, n. 13, p. 1486–1497, 2013.

CHILTON, L. B.; TEEVAN, J. Addressing people's information needs directly in a web search result page. In: INTERNATIONAL CONFERENCE ON WORLD WIDE WEB, 20., Hyderabad, India, 2011. **Proceedings...** New York, USA: ACM, 2011. p. 27–36.

CHUANG, S.-L.; CHANG, K. C.-C.; ZHAI, C. Context-aware wrapping: Synchronized data extraction. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, 33., Vienna, Austria, 2007. **Proceedings...** New York, USA: ACM, 2007. p. 699–710.

COHEN, W. W.; RAVIKUMAR, P.; FIENBERG, S. E. A comparison of string distance metrics for name matching tasks. In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, WORKSHOP ON INFORMATION INTEGRATION ON THE WEB, 18., Acapulco, Mexico, 2003. **Proceedings...** San Francisco, USA: Morgan Kaufmann, 2003. p. 73–78.

CORTEZ, E. et al. Ondux: on-demand unsupervised learning for information extraction. In: INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, Indianapolis, USA, 2010. **Proceedings...** New York, USA: ACM, 2010. p. 807–818.

CRESCENZI, V.; MECCA, G.; MERIALDO, P. Roadrunner: Towards automatic data extraction from large web sites. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, 27., Roma, Italy, 2001. **Proceedings...** San Francisco, USA: Morgan Kaufmann, 2001. p. 109–118.

CRESCENZI, V.; MERIALDO, P.; QIU, D. A framework for learning web wrappers from the crowd. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE, 22., Rio de Janeiro, Brazil, 2013. **Proceedings...** New York, USA: ACM, 2013. p. 261–272.

CRESTAN, E.; PANTEL, P. Web-scale table census and classification. In: INTERNATIONAL CONFERENCE ON WEB SEARCH AND WEB DATA MINING, 4., Hong Kong, China, 2011. **Proceedings...** New York, USA: ACM, 2011. p. 545–554.

DALVI, N.; KUMAR, R.; SOLIMAN, M. Automatic wrappers for large scale web extraction. **Proceedings of the VLDB Endowment**, v. 4, n. 4, p. 219–230, 2011.

DALVI, N.; MACHANAVAJJHALA, A.; PANG, B. An analysis of structured data on the web. **Proceedings of the VLDB Endowment**, v. 5, n. 7, p. 680–691, 2012.

DESHPANDE, O. et al. Building, maintaining, and using knowledge bases: A report from the trenches. In: INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, New York, USA, 2013. **Proceedings...** New York, USA: ACM, 2013. p. 1209–1220.

DONG, X. et al. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 20., New York, USA, 2014. **Proceedings...** New York, USA: ACM, 2014. p. 601–610.

FUMAROLA, F. et al. Hylien: A hybrid approach to general list extraction on the web. In: INTERNATIONAL CONFERENCE ON WORLD WIDE WEB, 20., Hyderabad, India, 2011. **Proceedings...** New York, USA: ACM, 2011. p. 35–36.

FURCHE, T. et al. Oxpath: A language for scalable, memory-efficient data extraction from web applications. **Proceedings of the VLDB Endowment**, v. 4, n. 11, p. 1016–1027, 2011.

FURCHE, T. et al. DIADEM: thousands of websites to a single database. **Proceedings of the VLDB Endowment**, v. 7, n. 14, p. 1845–1856, 2014.

GENTILE, A. L. et al. Unsupervised wrapper induction using linked data. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE CAPTURE, 7., Banff, Canada, 2013. **Proceedings...** New York, USA: ACM, 2013. p. 41–48.

GIBSON, D.; PUNERA, K.; TOMKINS, A. The volume and evolution of web page templates. In: INTERNATIONAL CONFERENCE ON WORLD WIDE WEB, SPECIAL INTEREST TRACKS AND POSTERS, 14., Chiba, Japan, 2005. **Proceedings...** New York, USA: ACM, 2005. p. 830–839.

GOLLAPALLI, S. D. et al. Improving researcher homepage classification with unlabeled data. **ACM Transactions on the Web**, v. 9, n. 4, p. 17:1–17:32, 2015.

GRÜNWALD, P. D. **The Minimum Description Length Principle**. Cambridge, USA: The MIT Press, 2007.

GULHANE, P. et al. Exploiting content redundancy for web information extraction. **Proceedings of the VLDB Endowment**, v. 3, n. 1, p. 578–587, 2010.

GUPTA, R.; SARAWAGI, S. Joint training for open-domain extraction on the web: Exploiting overlap when supervision is limited. In: INTERNATIONAL CONFERENCE ON WEB SEARCH AND WEB DATA MINING, 4., Hong Kong, China, 2011. **Proceedings...** New York, USA: ACM, 2011. p. 217–226.

HAO, Q. et al. From one tree to a forest: A unified solution for structured web data extraction. In: INTERNATIONAL CONFERENCE ON RESEARCH AND DEVELOPMENT IN INFORMATION RETRIEVAL, 34., Beijing, China, 2011. **Proceedings...** New York, USA: ACM, 2011. p. 775–784.

HE, Y. et al. Crawling deep web entity pages. In: INTERNATIONAL CONFERENCE ON WEB SEARCH AND DATA MINING, 6., Rome, Italy, 2013. **Proceedings...** New York, USA: ACM, 2013. p. 355–364.

HERNÁNDEZ, I. et al. CALA: classifying links automatically based on their URL. **Journal of Systems and Software**, v. 115, p. 130–143, 2016.

HERTZOG, I. L. **ICNDE : um novo método para identificação e correção de ruído para extração de dados**. 2016. Monografia (Bacharel em Ciência da Computação), UFRGS (Universidade Federal do Rio Grande do Sul), Porto Alegre, Brasil.

HOLZSCHUHER, F.; PEINL, R. Performance of graph query languages: Comparison of cypher, gremlin and native access in neo4j. In: INTERNATIONAL CONFERENCE ON EXTENDING DATABASE TECHNOLOGY / INTERNATIONAL CONFERENCE ON DATABASE THEORY, WORKSHOP, Genoa, Italy, 2013. **Proceedings...** [S.l.]: ACM, 2013. p. 195–204.

JIN, P. et al. Tise: A temporal search engine for web contents. In: INTERNATIONAL SYMPOSIUM ON INTELLIGENT INFORMATION TECHNOLOGY APPLICATION, 2., Shanghai, China, 2008. **Proceedings...** Washington, USA: IEEE Computer Society, 2008. p. 220–224.

KAUR, K.; RANI, R. Modeling and querying data in nosql databases. In: INTERNATIONAL CONFERENCE ON BIG DATA, Santa Clara, USA, 2013. **Proceedings...** Washington, USA: IEEE Computer Society, 2013. p. 1–7.

KITTUR, A.; CHI, E. H.; SUH, B. Crowdsourcing user studies with mechanical turk. In: CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS, Florence, Italy, 2008. **Proceedings...** New York, USA: ACM, 2008. p. 453–456.

LAFFERTY, J. D.; MCCALLUM, A.; PEREIRA, F. C. N. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 18., Williamstown, USA, 2001. **Proceedings...** San Francisco, USA: Morgan Kaufmann, 2001. p. 282–289.

LEE, L. Measures of distributional similarity. In: ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS, 37., Maryland, USA, 1999. **Proceedings...** Stroudsburg, USA: Association for Computational Linguistics, 1999. p. 25–32.

MADHAVAN, J. et al. Google's deep web crawl. **Proceedings of the VLDB Endowment**, v. 1, n. 2, p. 1241–1252, 2008.

MILLER, G. A. Wordnet: A lexical database for english. **Communications of the ACM**, v. 38, n. 11, p. 39–41, 1995.

MUROLO, A.; NORRIE, M. C. Revisiting web data extraction using in-browser structural analysis and visual cues in modern web designs. In: INTERNATIONAL CONFERENCE ON WEB ENGINEERING, 16., Lugano, Switzerland, 2016. **Proceedings...** Cham, Switzerland: Springer, 2016. p. 114–131.

Neo Technology. **The Neo4j Manual v2.3.3**. 2016. Available: <http://neo4j.com/docs/stable/index.html>. Accessed: 02/feb./2017.

NORDHEIMER, D. et al. Collaborative majority vote: Improving result quality in crowdsourcing marketplaces. In: LI, WEI AND HUHNS, MICHAEL N. AND TSAI, WEI-TEK AND WU, WENJUN. **Crowdsourcing: Cloud-Based Software Development**. Berlin, Heidelberg: Springer-Verlag, 2015.

ORTONA, S. et al. Joint repairs for web wrappers. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, 32., Helsinki, Finland, 2016. **Proceedings...** Washington, USA: IEEE Computer Society, 2016. p. 1146–1157.

QIU, D. et al. Dexter: Large-scale discovery and extraction of product specifications on the web. **Proceedings of the VLDB Endowment**, v. 8, n. 13, p. 2194–2205, 2015.

QIU, D.; LUCE, L. Extraction and integration of web sources with humans and domain knowledge. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE, 23., Seoul, Republic of Korea, 2014. **Proceedings...** New York, USA: ACM, 2014. p. 1295–1298.

QUINLAN, J. R. **C4.5: Programs for Machine Learning**. San Francisco, USA: Morgan Kaufmann, 1993.

RABINER, L. R. A tutorial on hidden markov models and selected applications in speech recognition. **Proceedings of the IEEE**, v. 77, n. 2, p. 257–286, 1989.

REIS, D. C. et al. Automatic web news extraction using tree edit distance. In: INTERNATIONAL CONFERENCE ON WORLD WIDE WEB, 13., New York, USA, 2004. **Proceedings...** New York, USA: ACM, 2004. p. 502–511.

SCOTT, M. **Programming Language Pragmatics**. 3rd. ed. Burlington, USA: Morgan Kaufmann, 2015.

SLEIMAN, H. A.; CORCHUELO, R. Trinity: On using trinary trees for unsupervised web data extraction. **IEEE Transactions on Knowledge and Data Engineering**, v. 26, n. 6, p. 1544–1556, 2014.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining**. Boston, USA: Addison-Wesley Longman, 2005.

VIDAL, M. L. A. et al. Structure-driven crawler generation by example. In: INTERNATIONAL CONFERENCE ON RESEARCH AND DEVELOPMENT IN INFORMATION RETRIEVAL, 29., Seattle, USA, 2006. **Proceedings...** New York, USA: ACM, 2006. p. 292–299.

WENINGER, T.; JOHNSTON, T. J.; HAN, J. The parallel path framework for entity discovery on the web. **ACM Transactions on the Web**, v. 7, n. 3, p. 16:1–16:29, 2013.

WINKLER, W. E. **The State of Record Linkage and Current Research Problems**. [S.l.], 1999. Statistical Research Division, U.S. Census Bureau (Technical report).

YOSHINAGA, N.; TORISAWA, K. Open-domain attribute-value acquisition from semi-structured texts. In: INTERNATIONAL SEMANTIC WEB CONFERENCE, ONTOLEX WORKSHOP, Busan, Korea, 2007. **Proceedings...** Berlin, Heidelberg: Springer, 2007. p. 55–66.

YU, H.; HAN, J.; CHANG, K. C.-C. PEBL: Web page classification without negative examples. **IEEE Transactions on Knowledge and Data Engineering**, v. 16, n. 1, p. 70–81, 2004.

ZHAI, Y.; LIU, B. Structured data extraction from the web based on partial tree alignment. **IEEE Transactions on Knowledge and Data Engineering**, v. 18, n. 12, p. 1614–1628, 2006.

APÊNDICE A —

A etapa de *Reforço* da abordagem **Orion** requer a definição das funções de similaridade f_l, f_x, f_a e f_e. Este apêndice descreve as funções de similaridade utilizadas nos experimentos.

Antes de explicar as funções de similaridade, é essencial definir algumas variáveis comuns que são usadas nelas: (i) Q_c é uma consulta candidata; (ii) Q_s é uma consulta selecionada; e (iii) $A = \{a_1, ..., a_n\}$ é o conjunto das consultas alinhadas com a consulta selecionada Q_s.

A.1 Funções f_l e f_x

Esta seção descreve as seguintes funções: (i) f_l - que é utilizada pela função de Similaridade de Pivô; e (ii) f_x - que é utilizada pela função de Similaridade de Caminho. As funções f_l e f_x são definidas como segue:

$$f_l(s_1, s_2) = f_x(s_1, s_2) = jw(s_1, s_2)$$

em que $jw(s_1, s_2)$ é o escore de Jaro-Wilker entre as strings s_1 e s_2.

O escore de Jaro-Wilker (WINKLER, 1999) é definido como segue:

$$jw(s_1, s_2) = j(s_1, s_2) + (l \times p \times (1 - j(s_1, s_2)))$$

em que l é o tamanho do prefixo comum no início da string até um máximo de 4 caracteres; p é um fator de escala constante que define o quanto o escore é ajustado para cima por ter prefixos comuns (foi utilizado o valor padrão $p = 0,1$); e $j(s_1, s_2)$ é o escore de Jaro entre duas strings.

O escore de Jaro (WINKLER, 1999) entre duas strings é definido como segue:

$$j(s_1, s_2) = \begin{cases} 0, & \text{se } m = 0 \\ \frac{m}{3\times|s_1|} + \frac{m}{3\times|s_2|} + \frac{m-t}{3\times m}, & \text{caso contrário} \end{cases}$$

em que m é o número de caracteres correspondentes e t é a metade do número de transposições.

A.2 Função f_a

Esta seção descreve a função fa, que é utilizada pela função de Similaridade de Valor em Nível de Atributo. A função fa é definida como segue:

$$fa(v,\Omega) = \begin{cases} fa_p(v,\Omega), & \text{se } type(\Omega)\ IN\ (\text{ISBN}, \text{Data}, \text{Fone}) \\ fa_n(v,\Omega), & \text{se } type(\Omega) = \text{Numérico} \\ fa_s(v,\Omega), & \text{caso contrário} \end{cases}$$

em que v é um valor extraído por Q_c; Ω é o conjunto dos valores extraídos por Q_s; $type(\Omega)$ é o tipo de dado mais frequente[1] entre os valores que pertencem ao conjunto Ω; $fa_p(v,\Omega)$ é uma função de similaridade para valores que seguem um determinado padrão (por exemplo, ISBN, data e telefone); $fa_n(v,\Omega)$ é uma função de similaridade para valores numéricos; e $fa_s(v,\Omega)$ é uma função de similaridade para valores string.

A função fa_p é definida como segue:

$$fa_p(v,\Omega) = \begin{cases} 1, & \text{se } type(v) = type(\Omega) \\ 0, & \text{caso contrário} \end{cases}$$

em que v é um valor extraído por Q_c; Ω é o conjunto dos valores extraídos por Q_s; $type(\Omega)$ é o tipo de dado mais frequente entre os valores que pertencem ao conjunto Ω; e $type(v)$ é o tipo de dado de v.

A função fa_n é definida como segue:

$$fa_n(v,\Omega) = e^{-\frac{1}{2}(\frac{v-\mu}{\sigma})^2}$$

em que v é um valor extraído por Q_c; Ω é o conjunto dos valores extraídos por Q_s; σ e μ são a média e o desvio padrão, respectivamente, dos valores que pertencem ao conjunto Ω.

A função fa_s é definida como segue:

$$fa_s(v,\Omega) = \frac{\sum_{t \in T(\Omega) \cap T(v)} fitness(t,\Omega)}{|T(v)|}$$

em que v é um valor extraído por Q_c; Ω é o conjunto dos valores extraídos por Q_s; $T(\Omega)$ é o conjunto de todos os termos encontrados nos valores que pertencem ao conjunto Ω; $T(v)$ é o conjunto dos termos encontrados no valor v; e a função $fitness(t,\Omega)$ avalia o quão representativo um termo t é entre os valores que pertencem ao conjunto Ω.

A função $fitness$ é definida como segue:

$$fitness(t,\Omega) = \frac{f(t,\Omega)}{N(t)} \times \frac{f(t,\Omega)}{f_{max}(\Omega)}$$

[1] O tipo de dado de um valor é determinado por meio de expressões regulares.

em que t é um termo; Ω é o conjunto dos valores extraídos por Q_s; $f(t, \Omega)$ é o número de valores do conjunto Ω que contém o termo t; $f_{max}(\Omega)$ é a maior frequência de qualquer termo entre os valores do conjunto Ω; e $N(t)$ é o número total de ocorrências do termo t em todas as extrações das consultas selecionadas para o site (há uma consulta selecionada para cada atributo).

A.3 Função f_e

Esta seção descreve a função fe, que é utilizada pela função de Similaridade de Valor em Nível de Entidade. A função fe é definida como segue:

$$fe(v_c^{e_i}, v_a^{e_i}) = \begin{cases} fe_p(v_c^{e_i}, v_a^{e_i}), & \text{se } type(A)\ IN\ (\text{ISBN}, \text{Data}, \text{Fone}) \\ fe_n(v_c^{e_i}, v_a^{e_i}), & \text{se } type(A) = \text{Numérico} \\ fe_s(v_c^{e_i}, v_a^{e_i}), & \text{caso contrário} \end{cases}$$

em que $v_c^{e_i}$ e $v_a^{e_i}$ são os valores extraídos (na página que descreve a entidade e_i) pela consulta candidata Q_c e pela consulta alinhada a_i, respectivamente; $A = \{a_1, ..., a_n\}$ é o conjunto das consultas alinhadas; type(A) é o tipo de dado mais frequente entre os valores extraídos pelas consultas alinhadas; $fe_p(v_c^{e_i}, v_a^{e_i})$ é uma função de similaridade para valores que seguem um padrão específico (por exemplo, ISBN, data e telefone); $fe_n(v_c^{e_i}, v_a^{e_i})$ é uma função de similaridade para valores numéricos; e $fe_s(v_c^{e_i}, v_a^{e_i})$ é uma função de similaridade para valores string.

A função fe_p é definida como segue:

$$fe_p(v_c^{e_i}, v_a^{e_i}) = \begin{cases} 1, & \text{se } norm(v_c^{e_i}) = norm(v_a^{e_i}) \\ 0, & \text{caso contrário} \end{cases}$$

em que $v_c^{e_i}$ e $v_a^{e_i}$ são os valores extraídos (na página que descreve a entidade e_i) por uma consulta candidata e uma consulta alinhada, respectivamente; $norm(v_c^{e_i})$ e $norm(v_a^{e_i})$ são valores normalizados de $v_c^{e_i}$ e $v_a^{e_i}$, respectivamente.

A função fe_n é definida como segue:

$$fe_n(v_c^{e_i}, v_a^{e_i}) = \begin{cases} 0, & \text{se } |v_c^{e_i} - v_a^{e_i}| > \rho \\ 1, & \text{caso contrário} \end{cases}$$

em que $v_c^{e_i}$ e $v_a^{e_i}$ são os valores extraídos (na página que descreve a entidade e_i) por uma consulta candidata e uma consulta alinhada, respectivamente; e ρ é um limiar que representa a diferença

máxima tolerada. O limiar ρ é definido como $\rho = \delta - \theta$, em que θ é 0,02 (conforme definido por Bronzi *et al.* (2013)), e δ é o menor valor entre a média dos valores absolutos extraídos pela consulta candidata e a média dos valores absolutos extraídos pela consulta alinhada.

A função fe_s é uma variante da função de Jensen-Shannon, denominada DirichletJS (COHEN; RAVIKUMAR; FIENBERG, 2003).