ENRIQUECIMENTO DE DADOS DE REFERÊNCIA PARA RECUPERAÇÃO DE INFORMAÇÃO GEOGRÁFICA UTILIZANDO LINKED DATA

TIAGO HENRIQUE VALADARES MENDES DE MOURA

ENRIQUECIMENTO DE DADOS DE REFERÊNCIA PARA RECUPERAÇÃO DE INFORMAÇÃO GEOGRÁFICA UTILIZANDO LINKED DATA

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Clodoveu Augusto Davis Junior

Belo Horizonte

Maio de 2015

© 2015, Tiago Henrique Valadares Mendes de Moura.
Todos os direitos reservados.

Moura, Tiago Henrique Valadares Mendes de

M929e Enriquecimento de dados de referência para recuperação de informação geográfica utilizando Linked Data / Tiago Henrique Valadares Mendes de Moura. — Belo Horizonte, 2015
xiv, 62 f. : il. ; 29cm

Dissertação (mestrado) — Universidade Federal de Minas Gerais
Orientador: Clodoveu Augusto Davis Junior

1. Computação - Teses. 2. Recuperação de Informação Geográfica - Teses. 3. Sistemas de informação geográfica – Teses. I. Orientador. II. Título.

CDU 519.6*73(043)

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Enriquecimento de dados de referência para recuperação de informação geográfica utilizando linked data

TIAGO HENRIQUE VALADARES MENDES DE MOURA

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. CLODOVEU AUGUSTO DAVIS JÚNIOR - Orientador
Departamento de Ciência da Computação - UFMG

PROF. CLÁUDIO DE SOUZA BAPTISTA
Centro de Engenharia Elétrica e Informática, Sistemas e Computação - UFCG

PROF. FREDERICO TORRES FONSECA
College of Information Sciences and Technology - Pennsylvania State University

PROF. RODRYGO LUIS TEODORO SANTOS
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 11 de junho de 2015.

Agradecimentos

Dois anos atrás estava tomando a decisão de iniciar o curso de mestrado, uma decisão que eu mesmo não sabia se era o que queria estar fazendo na época. Minha família teve papel fundamental nessa decisão sempre me apoiando a estudar e me aprimorar cada vez mais. Mesmo estando geograficamente distantes, meu pai Paulo, minha mãe Vanessa, minha avó "Lina", minha tia "Kau", meu tio "Zé"e meu grande amigo e irmão "Bê", foram pessoas muito presentes durante todo meu trabalho e certamente a tarefa teria sido mais complicada sem o apoio incondicional deles.

Não poderia deixar de agradecer a meu orientador, professor Clodoveu, que vem me orientando desde o final da graduação, sempre instigando e propondo caminhos para seguirmos com nossas pesquisas. E também meus amigos de república, Vinícius, Lucas e Leo, e ao Abner, que proporcionaram ótimos debates, sempre que algum desafio novo surgia ou algo novo era descoberto. Além dos meus amigos da 082 que me proporcionam bons momentos de lazer/diversão em meio a um período de muito estudo.

Foram anos em que me dividi entre estudar e trabalhar e é impossível não agradecer a compreensão das equipes em que trabalhei, seja na Montreal ou na OneForce. Uma pessoa em especial também foi bastante compreensiva e me apoiou bastante todo o tempo que esteve presente, minha namorada Arielle.

Foram dois anos de muito aprendizado e certamente uma decisão muito acertada para quem tinha tantas dúvidas no começo. Obrigado a todos que se envolveram nesse período!

"The acquisition of knowledge is always of use to the intellect, because it may thus drive out useless things and retain the good. For nothing can be loved or hated unless it is first known."

(Leonardo da Vinci)

Resumo

Gazetteers, ou dicionários toponímicos, são ferramentas comumente utilizadas para reconhecimento de nomes de lugares em documentos como páginas da *Web*, notícias e mensagens em redes sociais. Entretanto, criar e manter *gazetteers* ainda é uma tarefa complexa e que demanda bastante esforço. A utilização de fontes de dados que seguem o padrão *Linked Data* é uma forma de se obter informações para se criar um *gazetteer* mundialmente amplo e com detalhamento de informações em um nível intra-urbano. Neste trabalho é apresentado o Linked OntoGazetteer, um *gazetteer* ontológico que utiliza informações da *Web of Data* como fonte de dados, além de obter evidências extras como relacionamento de entidades geográficas com outras entidades (geográficas ou não), que enriquecem a base de conhecimento resultante, aumentando a capacidade de resolução de problemas típicos em Recuperação de Informação Geográfica (RIG) como a desambiguação de topônimos. Os objetivos deste trabalho vão desde a criação do *gazetteer* até a disponibilização de acesso a ele através de Web Services para que aplicações e iniciativas de pesquisa em RIG, processamento de texto, reconhecimento de nomes de entidades e outras possam utilizar. O *gazetteer* resultante deste trabalho possui mais de 13 milhões de lugares extraídos de quatro fontes de dados: GeoNames, Freebase, DBPedia e LinkedGeoData. Também é apresentada uma análise de como essas fontes de dados se sobrepõem, levando em consideração as entidades armazenadas em cada uma delas.

Abstract

Gazetteers are instrumental in recognizing place names in documents such as Web pages, news, and social media messages. However, creating and maintaining gazetteers is still a complex and demanding task. We propose using Linked Data sources to put together gazetteer data that can be both broad (e.g. planetary) and deep (e.g., down to urban detail). Linked data sources also allow enriching the resulting gazetteer with a set of geographic and semantic relationships involving place names, other geographic and non-geographic terms, thus expanding the possibilities for solving typical GIR problems such as disambiguation and filtering. This work shows the results of efforts to compose and maintain an ontological gazetteer, in which places and their names are connected to other places and to non-geographic entities through geographic and semantic relationships. The objective of this proposal is to create, organize and populate a large ontological gazetteer with information obtained from the Web of Data, to be exposed as a Web service to applications and research initiatives on geographic information retrieval, text processing, named entity recognition and others. The resulting gazetteer contains more than 13 million places, extracted from the four datasets used in this work: GeoNames, Freebase, DBPedia and LinkedGeoData. In addition, we present an analysis of how the datasets overlap one another.

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

A utilização da internet vem aumentando e se diversificando com o passar do tempo, novos sistemas e aplicações destinados aos mais diversos contextos são lançados diariamente. Frequentemente o usuário utiliza um conjunto de palavras-chave para indicar o que procura. Trabalhos anteriores [Backstrom et al., 2008; Sanderson & Kohler, 2004; Wang et al., 2005] mostram que parte significativa dessas consultas envolve termos como nomes de lugares ou expressões que denotam posicionamento. Por isso, é importante reconhecer a intenção do usuário que inclui termos geográficos em buscas, bem como determinar o escopo geográfico de documentos, em aplicações de Recuperação de Informação Geográfica (RIG).

Diversas aplicações de RIG comumente buscam fontes de dados externas, como *gazetteers*, para usarem como referência para atingirem seu objetivo. Dentre essas aplicações, a identificação do escopo geográfico de documentos tem recebido muita atenção recentemente. Nesse contexto, *gazetteers* são empregados tanto para *geoparsing* (identificação de nomes de lugares no texto) como para *geolocating* (a associação de um *footprint* com o documento). A simples realização de *geotagging* (associar nomes de lugares a documentos) pode ser insuficiente, pois nomes de lugares são frequentemente ambíguos com outros nomes de lugares e com nomes de pessoas ou objetos. *Gazetteers* também são utilizados para resolver ambiguidades, provendo evidências de relacionamentos existentes entre lugares mencionados no mesmo documento.

Entretanto, atualmente não existe um único *gazetteer* (publicamente disponível) que possua informação detalhada e ampla cobertura sobre os lugares da Terra, especialmente quando se trata de informações intra-urbanas. *Gazetteers* tradicionais mantêm informações sobre lugares de todo o planeta, embora exista uma grande concentração em países desenvolvidos [Ahlers, 2013], sem incluir muitas informações urbanas. Outras fontes, comumente associadas a redes sociais e iniciativas de informação geográfica

voluntária, como WikiMapia[1] ou OpenStreetMap[2], incluem detalhamento urbano, mas não são organizadas como *gazetteers*. Além disso, muitos *gazetteers* armazenam apenas relacionamentos geográficos entre os lugares, como em hierarquias de subdivisão territorial. Trabalhos anteriores exploram adicionar relacionamentos semânticos a um *gazetteer*, como o *OntoGazetteer*[3][Machado et al., 2011]), apresentando resultados interessantes, mas concluindo que manter e expandir tal *gazetteer* seria uma tarefa muito complexa e expendiosa.

Linked Data é um padrão que possui um conjunto de premissas simples e que buscam padronizar a forma de publicar dados na *Web* [Bizer et al., 2009a]. Esse conceito visa o compartilhamento de informações entre sistemas e aplicações através de *links* semanticamente ricos e da utilização de um formato bem estruturado para os dados, a fim de promover sua reutilização. As premissas que viabilizam esta ideia são [Berners-Lee, 2006]: (1) os dados precisam obedecer ao padrão RDF (*Resource Description Framework*); (2) os objetos precisam ser identificados por URIs (*Universal Resource Identifier*); (3) os dados devem estar acessíveis via protocolo HTTP; e (4) os objetos devem ser referenciados através de suas URIs. Esse padrão também vem se tornando cada vez mais popular, com crescente adoção por parte do meio acadêmico e por iniciativas como Geonames[4] e DBPedia[5].

A facilidade oferecida pela padronização na estruturação dos dados, somada aos relacionamentos semanticamente ricos armazenados e a quantidade e variedade de dados torna possível integrar informação de diversas fontes *online*, com o objetivo de viabilizar a execução de típicas tarefas de RIG através da combinação de informações semânticas e geográficas. Essa estratégia envolve o desafio de combinar fontes potencialmente grandes de dados com múltiplas (e também grandes) fontes de informação disponíveis na *Web of Data*, configurando assim o que chamamos de *Linked OntoGazetter*.

Nem sempre associar um nome ao lugar que ele representa é uma tarefa trivial. São Francisco, por exemplo, pode ser além de um santo católico, um bairro de Belo Horizonte. O contexto em que o nome aparece ajuda a resolver esse tipo de problema, pois oferece informações adicionais. Bases de dados enriquecidas com evidências externas são uma excelente fonte de informações para ajudar na contextualização, pois viabilizam a construção de estruturas de dados que relacionam diversas entidades semanticamente. O principal objetivo deste trabalho é buscar formas de enriquecer dados

[1] http://wikimapia.com
[2] http://www.openstreetmap.org
[3] http://aqui.io/ontogazetteer
[4] http://www.geonames.org
[5] http://www.dbpedia.org

de referência para apoiar tarefas típicas de RIG. Para demonstrar a aplicabilidade dos estudos e experimentos realizados, será apresentado um *gazetteer* que tira proveito dos padrões definidos pelo *Linked Data* para armazenar e disponibilizar um grande conjunto de dados sobre nomes de lugares, expandindo a definição tradicional de um *gazetteer* por manter informações sobre relacionamentos geográficos e semânticos entre lugares e diversas entidades (lugares ou não). O *gazetteer* está publicamente disponível por meio de uma API que externaliza funções para acesso aos dados do gazetteer, sendo que essas funções são definidas com base na expectativa de uso típico de problemas de RIG. Um dos resultados esperados ao final dos estudos é uma avaliação qualitativa sobre como as fontes disponíveis na *Web of Data* armazenam dados geográficos, não só isoladamente, mas também como os dados geográficos estão relacionados com outras entidades classificadas tanto como lugares quanto como não lugares. A cobertura e detalhamento de cada uma das fontes trabalhadas também é avaliada tanto quantitativamente, quanto qualitativamente.

As próximas seções deste trabalho detalham todas as etapas executadas neste projeto. O Capítulo 2 apresenta todo o estudo de trabalhos relacionados e referências que servem de base para essa pesquisa. Os Capítulos 3, 4 e 5 são responsáveis por detalhar as soluções propostas, sendo que o primeiro irá discutir as decisões de modelagem, o Capítulo 4 aborda a etapa de descoberta e pré-processamento de bases de dados que serão trabalhadas, enquanto o último tem o foco em discutir as etapas da criação do novo *gazetteer*, a partir do modelo de dados definido até as tecnologias envolvidas para torná-lo viável. As conclusões e trabalhos futuros são apresentados no Capítulo 6.

Capítulo 2

Trabalhos Relacionados

Toda notícia ou narrativa sempre terá em seu contexto um ou mais lugares relacionados, que comumente representam o local onde o fato que está sendo descrito ocorre. Até mesmo textos científicos possuem diversos indicadores de contexto geográfico, como instituições de ensino dos autores e local da conferência. Todos esses textos quando publicados na *Web* frequentemente estão na forma de documentos, normalmente semi ou não estruturados. Borges et al. [2007] mostram que documentos potencialmente contêm vários indicadores de seu escopo geográfico, porém evidências não ambíguas e de fácil reconhecimento não são comuns [Andogah et al., 2012]. Mesmo assim, o processo de identificar o contexto geográfico de um documento frequentemente utiliza dicionários de dados geográficos, *gazetteers*, para reconhecer e extrair topônimos existentes no texto [Souza et al., 2005; Goodchild & Hill, 2008] e, a partir de um conjunto de nomes de lugares identificados, inferir o contexto geográfico.

Em geral, a implementação desses dicionários organiza os lugares utilizando uma estrutura de dados simples, contendo apenas o nome, tipo e o *footprint* (representação da localização geográfica) do lugar [Hill, 2000]. Essa representação muitas vezes é insuficiente para realizar tarefas de RIG com sucesso, pois não aprofunda os relacionamentos existentes entre os lugares e possui uma representação geográfica muito simples, um ponto (coordenada geográfica), o que dificulta a execução de consultas geográficas como "está contido em".

Ambíguo é definido no dicionário como: *que contém ou pode conter múltiplos sentidos; com distintos significados; equívoco.* Em aplicações de RIG essa é uma característica constante relacionada aos topônimos encontrados em um documento, pois frequentemente um nome de lugar pode se referir a diversos lugares na realidade. A Figura 2.1 mostra uma nuvem de palavras que foi gerada a partir dos resultados que o GeoNames retorna quando ocorre uma busca pelo nome do lugar em questão. São

Francisco, no exemplo, pode ser a famosa cidade californiana ou uma pequena cidade no norte do estado de Minas Gerais, Brasil.

Figura 2.1: Nuvem de palavras representando a quantidade de retornos do GeoNames para cada busca feita.

Técnicas de identificação do escopo geográfico de documentos que utilizam bases de conhecimento e *gazetteers* [Han & Zhao, 2009; Hoffart et al., 2011; Cucerzan, 2007; Pouliquen et al., 2006] têm como principal desafio lidar com ambiguidade. Uma referência a um lugar pelo nome pode ser ambígua de duas formas: (1) com outros lugares (*ambiguidade geo/geo*) ou (2) com nomes de outras entidades (*ambiguidade geo/non-geo*) [Amitay et al., 2004]. A ambiguidade precisa ser resolvida antes da execução do algoritmo de determinação de escopo geográfico do documento. Idealmente, dados de referência, utilizados para identificação de menções à lugares no texto, provêm informação adicional que ajuda no processo de desambiguação. A Wikipédia[1] pode ser uma boa opção de fonte externa de dados de referência, tanto para a tarefa de reconhecimento de topônimos, quanto para desambiguação. Alencar et al. [2010] propõem uma estratégia para determinação do escopo geográfico que busca relacionar os tópicos da Wikipédia identificados no texto com os lugares, construindo assim um grafo que é utilizado para inferir o contexto geográfico do documento, que pode inclusive ser um lugar que não está explicitamente mencionado no documento.

Além de fonte de dados de referência para aplicações RIG, a Wikipédia também vem sendo utilizada como ponto de partida para construção de *gazetteers*. Smart et al. [2010] apresentam um arcabouço capaz de construir um *meta-gazetteer*, com dados de diversas fontes, incluindo Wikipédia, OpenStreetMap e GeoNames. Nesse modelo, os autores propõem a realização de integração entre as bases em tempo de consulta. Popescu et al. [2008] apresentam o Gazetiki, um gazetteer também construído a partir

[1] http://www.wikipedia.org

de diversas fontes de dados heterogêneas e disponíveis online. No Gazetiki o ponto de partida para a criação do *gazetteer* é o GeoNames e os dados obtidos das demais fontes são utilizados para complementar as informações existentes na base original, sendo que uma das fontes utilizadas é a Wikipédia. O Gazetiki também propõe a expansão da estrutura comumente utilizada por *gazetteers* tradicionais (nome do lugar, footprint, tipo de lugar) associando um nível de relevância a cada registro.

Evidências como termos e lugares relacionados também podem ser obtidos de *gazetteers* especialmente construídos para esse objetivo. Machado et al. [2011] apresentam o OntoGazetteer, um *gazetteer* enriquecido com nomes alternativos e nomes ambíguos de lugares, além de relacionamento entre eles. O *gazetteer* também inclui relacionamentos topológicos e relacionamentos geográficos, juntamente com relacionamentos semânticos. Utilizando os relacionamentos semânticos, o *gazetteer* constrói conexões entre lugares que pertencem a mesma categoria ontológica, por exemplo: capitais, cidades históricas ou todas cidades que são interligadas por uma rodovia. Como resultado, a coexistência de referências em um documento a lugares ou outras entidades relacionadas, assim como no *gazetteer*, ajuda a resolver ambiguidades.

A partir do estudo de todos esses trabalhos e outras referências, conclui-se que evidências externas são importantes para várias tarefas de RIG. Entretanto, manter um único *gazetteer* com toda a informação necessária, desde lugares conhecidos mundialmente até o detalhamento intra-urbano, incluindo detalhes como bairros e pontos de interesse [Bizer et al., 2009b], é uma tarefa muito difícil. Extrair tal informação de tópicos da Wikipédia é também trabalhoso, já que eles são organizados de maneira semiestruturada e provavelmente nunca estarão totalmente completos. Para solucionar todas essas dificuldades seria necessário um gazetteer que atendesse aos requisitos de possuir uma base mundialmente detalhada de nomes de lugares e outras referências geográficas, com fácil acesso e manutenibilidade. Atualmente, esses são requisitos que nenhum *gazetteer* sozinho consegue prover. Nesse cenário, o estudo da aplicação de Linked Data [Bizer et al., 2009a] sobre diversas fontes de dados, assim como no projeto *Linked Open Data*[2] (LOD), para enriquecer dados existentes se tornou uma alternativa bastante atraente.

O grupo que suporta o LOD é o principal difusor do Linked Data. Em sua página, o grupo mantém estatísticas dos conjuntos de dados e o histórico da evolução do crescimento da rede das fontes que fazem parte da *Web of Data*. As Figuras 2.2 e 2.3 mostram a diferença da quantidade e complexidade do grafo formado pelas fontes de dados (vértices) integrantes do LOD, no qual as arestas representam links existentes

[2]http://www.linkeddata.org/

entre as fontes. Apenas no último intervalo medido, entre os anos de 2011 e 2014, registrou-se um crescimento de 271% no número de fontes de dados que seguem os princípios do *Linked Data* e compõem a *Web of Data*, saindo de 294 fontes para 1091 [Schmachtenberg et al., 2014], o que comprova a popularização e adoção crescente do conceito.

Entretanto, a utilização das fontes de dados disponíveis no LOD provavelmente irá aumentar a ocorrência de ambiguidade, pois é comum existir uma sobreposição do conhecimento entre as bases de dados. O predicado OWL *sameAs*[3] tem como objetivo lidar com esse problema, mas somente uma pequena parte das fontes públicas faz uso relevante dele [Moura & Davis Jr, 2013]. Determinar as áreas de interseção entre as fontes de dados não é uma tarefa trivial, dada a heterogeneidade dos esquemas existentes na *Web of Data* [Freitas et al., 2012]. Soluções que realizam integração de dados no ambiente da *Web of Data* precisam levar em consideração os esquemas e as ontologias definidas por cada uma das bases. Shvaiko & Euzenat [2005] construiram um vasto apanhado de técnicas existentes na literatura, que realizam integração de dados baseada em esquemas e ontologias que podem ajudar a determinar as áreas de interseção entre as fontes do LOD. Prover uma solução adequada para esse problema é essencial para fornecer um serviço de qualidade para aplicações RIG, pois mantém uma menor quantidade de representações de uma mesma entidade, expande o conhecimento sobre objetos reais e auxilia na resolução de ambiguidades.

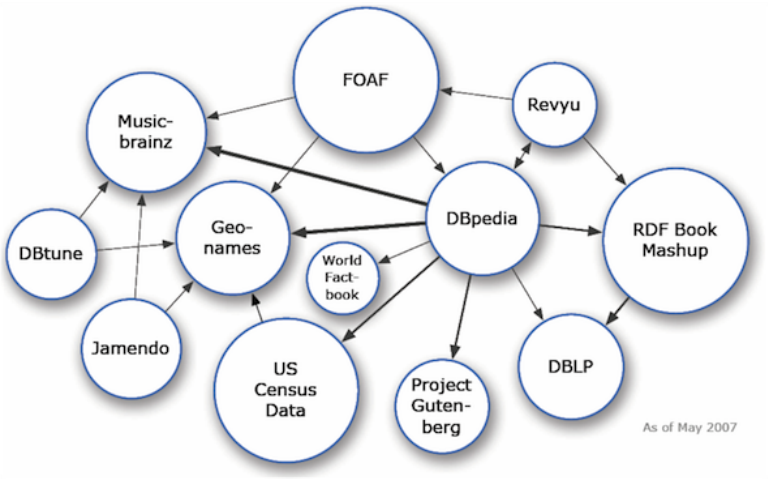


Figura 2.2: Fontes de dados que disponibilizavam informações utilizando Linked Data em maio de 2007

A presente dissertação busca, assim, explorar o potencial de *Linked Data* para construir um *gazetteer* de grande dimensão por meio de integração de diversas fontes

[3]http://www.w3.org/TR/owl-ref/#sameAs-def

de dados contendo nomes de lugares. Além disso, pretende enriquecer esse conteúdo utilizando relacionamentos geográficos e semânticos, inclusive com entidades que não são lugares. Ao contrário das iniciativas relatadas de construção de meta-*gazetteers*, o presente trabalho propõe uma estrutura de dados que permite a consulta eficiente aos dados de lugares, seus nomes e seus relacionamentos, sem tornar a tarefa de manutenção, uma tarefa dispendiosa.

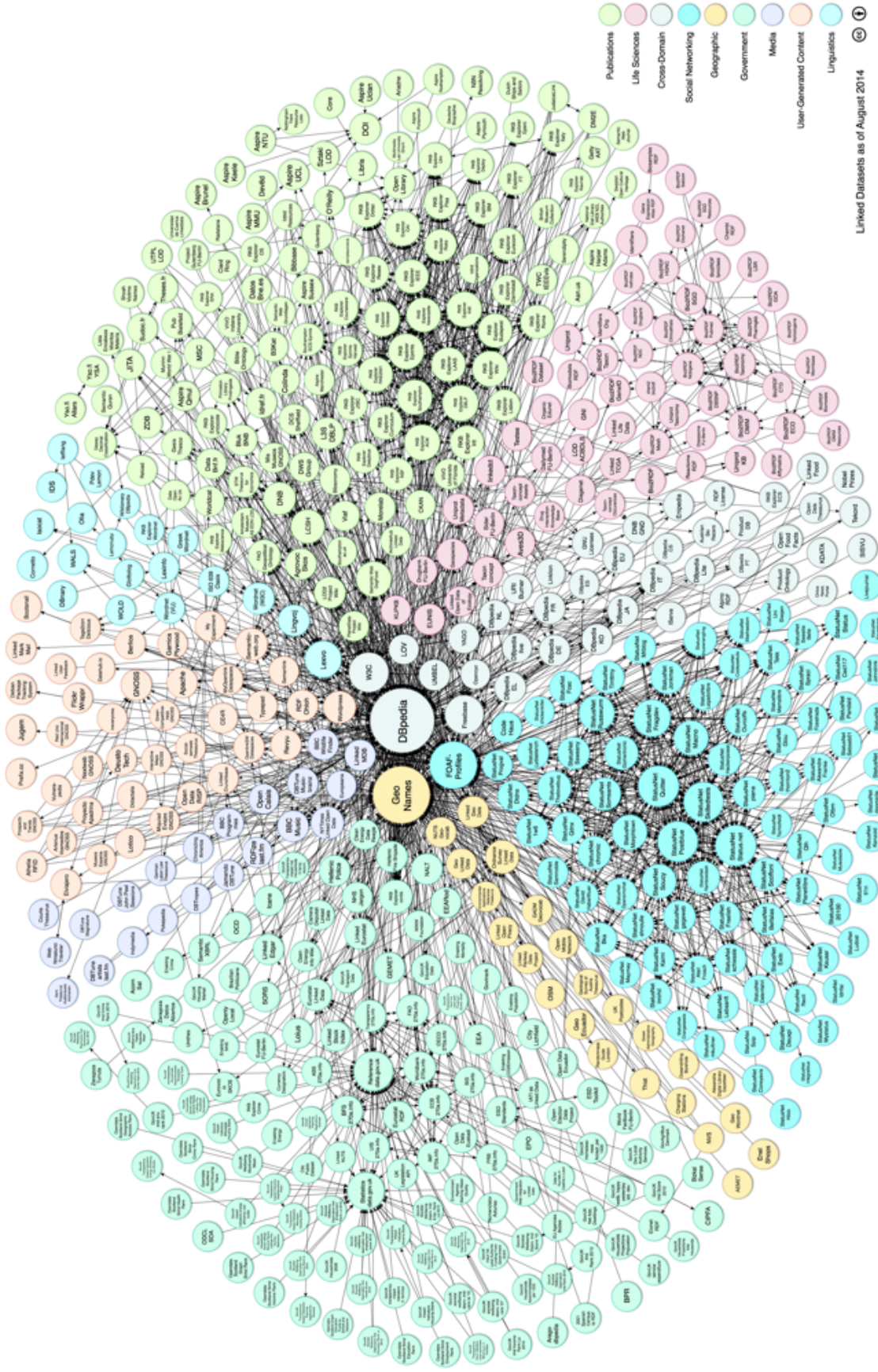


Figura 2.3: Fontes de dados que disponibilizavam informações utilizando Linked Data em agosto de 2014

Capítulo 3

Modelagem

Este capítulo discute como o OntoGazetteer é modelado, evidenciando os pontos fortes e fracos das decisões tomadas na sua construção e implementação. A partir dessa análise é apresentado um novo esquema para um *gazetteer*, o Linked OntoGazetteer, que busca minimizar os pontos fracos do OntoGazetteer. Por fim, é apresentado como os dados existentes na *Web of Data* podem ser mapeados dentro desse novo *gazetteer* e as tecnologias envolvidas na implementação dele.

3.1 Modelagem do OntoGazetteer

O OntoGazetteer proposto por Machado et al. [2011] possui um modelo de dados no qual a principal classe é chamada de `Place`. Instâncias dessa classe fazem referência a lugares reais do mundo e, como consequência, possuem diversos topônimos associados à instância. A forma como o OntoGazetteer mantém as diversas nomenclaturas do lugar é bastante particular: cada lugar possui um nome associado diretamente na forma de um atributo da classe `Place`. Esse atributo seria um nome principal (mais importante). Além desse atributo, existem duas outras classes, `Ambiguous name` e `Alternative place`, que mantêm, respectivamente, a informação de outros nomes comumente dados àquele lugar e uma lista de lugares homônimos (Figura 3.1).

Esse esquema traz alguns benefícios, como a manutenção em um único *gazetteer* de nomes ambíguos e lugares homônimos. Porém é difícil diferenciar, conceitualmente, o que é o nome de um lugar e o que são nomes ambíguos/alternativos, por exemplo: o Brasil, se desconsiderarmos representações em outras línguas, possui no mínimo dois nomes: Brasil e República Federativa do Brasil. Qual seria o principal, o oficial ou o mais popular? Sem contar que esse é um dado possivelmente volátil, principalmente em

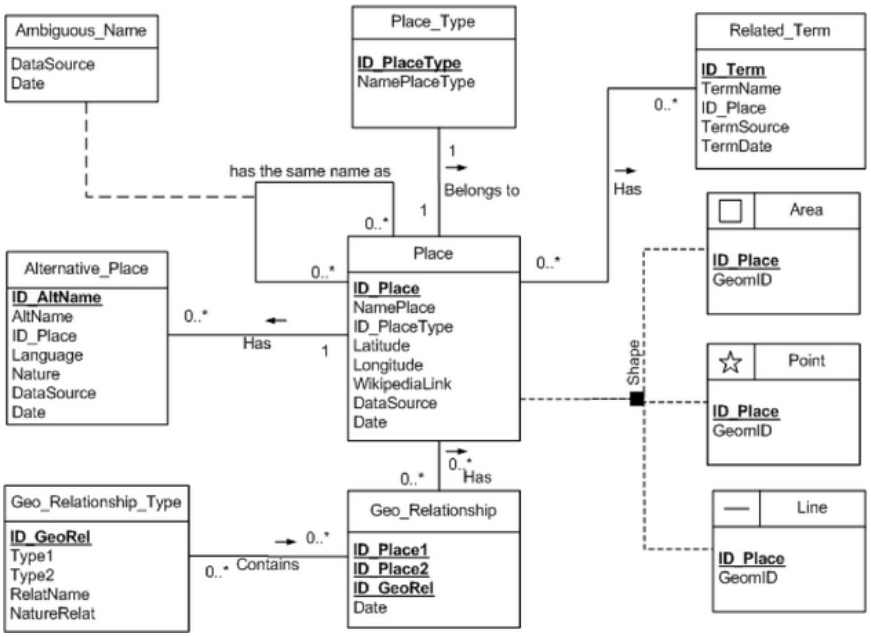


Figura 3.1: Esquema conceitual do OntoGazetteer

áreas de conflitos políticos e/ou sociais, onde territórios comumente sofrem alterações de nomes. Um exemplo são os países originados após o fim da União Soviética.

Muito embora esse modelo tenha sido implementado com sucesso utilizando um banco de dados relacional, ele traz uma dificuldade muito grande relacionada à busca de lugares por nome (provavelmente uma das consultas mais comuns realizadas sobre um *gazetteer*), pois envolve três entidades para conseguir chegar a um resultado (`Place`, `Alternative place` e `Ambiguous name`). Isso acaba tornando mais difícil a tarefa de identificação e resolução de ambiguidades. O OntoGazetteer também possui um volume pequeno de informações e uma estrutura de dados que dificulta a inserção de novos registros [Moura & Davis Jr, 2012], pois existem muitas entidades envolvidas. Um *gazetteer* completo precisa possuir cobertura de lugares em nível mundial, o que não é o caso do OntoGazetteer, que se restringe a dados do Brasil, mais especificamente Minas Gerais. Essa cobertura reduzida é possivelmente uma consequência da dificuldade inerente ao processo de expansão do OntoGazetteer. Além disso, criar um novo tipo de relacionamento entre lugares exige uma preparação prévia, o que nem sempre é possível quando se trabalha com grandes volumes de dados. Por esses motivos, e objetivando construir um *gazetteer* completo e de fácil manutenção, foi necessário revisar o esquema proposto para o OntoGazetteer.

3.2 Modelagem do Linked OntoGazetteer

A Figura 3.2 apresenta um esquema conceitual que busca manter as vantagens de se armazenar nomes alternativos e ambíguos de lugares em um mesmo *gazetteer*, sem adicionar complicações à consulta e ao processo de identificação e resolução de ambiguidades. Também é mantida representação geográfica dos lugares, podendo variar entre pontos, areas ou linhas. Nesse novo modelo, os nomes ficam em uma classe própria para representá-los: a classe `Name`, que por sua vez está relacionada com a classe `Place`. A cardinalidade do relacionamento das classes `Name` e `Place` é muitos-para-muitos, isso indica que um lugar pode ter vários topônimos, sejam eles de outras línguas, apelidos ou nomes populares. Além disso, um nome de lugar pode estar associado a diversos lugares diferentes, o que caracteriza ambiguidade do tipo geo/geo.

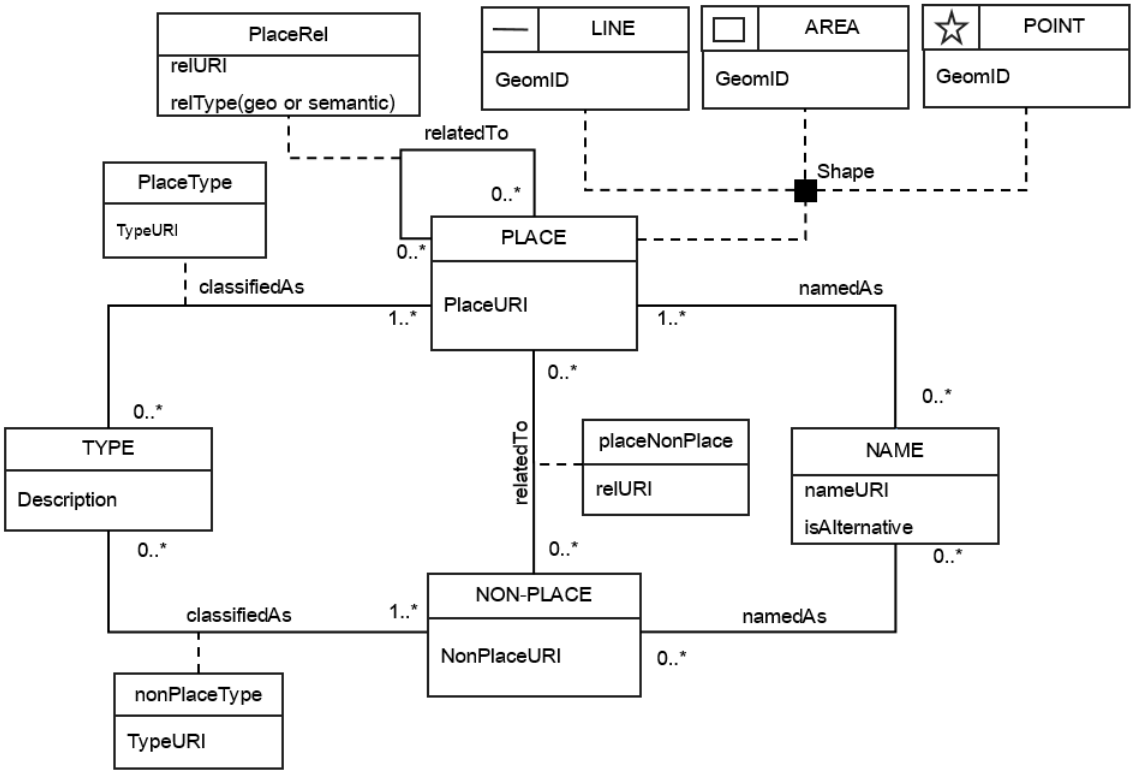


Figura 3.2: Esquema do Linked OntoGazetteer

Com essa nova representação, uma simples consulta sobre as instâncias de `Name` teria como resultado todo o conjunto de lugares ambíguos, tendo como referência o topônimo buscado. Apenas a simplicidade de se realizar tal consulta já é um benefício relevante, mas o modelo da Figura 3.2 traz ainda uma outra novidade: além de manter os nomes alternativos em um único *gazetteer*, também mantém o registro de entidades não classificadas como lugares. As instâncias da classe `Non-place` são responsáveis por

manter entidades não classificadas como lugares. O objetivo de armazenar entidades que não são lugares em um *gazetteer* é prover uma única fonte de informação capaz de suportar a identificação de ambos os tipos de ambiguidades (geo/geo e geo/non-geo).

O relacionamento da classe `Non-place` com a classe `Name` é idêntico ao existente entre `Place` e `Name`. Dessa forma, com simples consulta sobre as instâncias de `Name` não só os lugares ambíguos seriam identificados, mas como candidatos à ambiguidade do tipo geo/non-geo também.

Lugares e não-lugares possuem um relacionamento do tipo *muitos-para-muitos*, que busca representar a existência de alguma relação entre determinado lugar e algum elemento ou entidade que lhe seja característica. A Tabela 3.1 mostra alguns exemplos de relacionamento entre um lugar e um não lugar, e o motivo da existência desse relacionamento.

Lugar	**Não Lugar**	**Motivo**
Rio de Janeiro	Copa do Mundo FIFA 2014	Rio de Janeiro foi uma das sedes do evento
EUA	Barack Obama	Barack Obama é o atual presidente dos EUA
Belo Horizonte	Dilma Rousseff	Dilma nasceu em Belo Horizonte

Tabela 3.1: Relacionamentos entre Lugares e Não Lugares

Também existe o autorrelacionamento entre lugares, que busca representar relacionamentos de natureza geográfica ou semântica entre lugares. Tanto o relacionamento entre as classes `Place-Place` e `Place-Non-Place` possuem um atributo de relacionamento que armazena a especialização do relacionamento (`relURI`).

Todas as classes propostas, com exceção da classe `Name`, possuem um tipo. O objetivo da classe `Type` é definir uma categoria para lugares e não-lugares de uma maneira ampla. Exemplos de instância dessa classe são: Lugar, Pessoa e Evento. Assim como no relacionamento `relatedTo` a especialização da categoria é um atributo do relacionamento `classifiedAs` e identificado por `TypeURI`.

3.3 Mapeamento dos dados para um grafo

RDF é o padrão para estrutura de dados adotado pelo *Linked Data* como representação básica. Essa estrutura é composta por triplas, formadas por um sujeito, um predicado e um objeto ou literal[1]. A Figura 3.3 ilustra uma tripla simples.

[1] `http://www.w3.org/RDF/`

{sujeito, predicado, [objeto, literal]}

Figura 3.3: Formato de uma tripla RDF

As fontes de dados existentes no LOD são mantidas em armazéns de triplas de uso genérico, sem a definição de esquema particular. Triplas RDF podem ser facilmente convertidas para a estrutura de um grafo (vértices e arestas). A Figura 3.4 mostra uma forma de como isso poderia ser feito.

{sujeito**(vértice)**, predicado**(aresta)**, [objeto, literal]**(vértice)**}

Figura 3.4: Mapeamento de um tripla RDF para a estrutura de um grafo

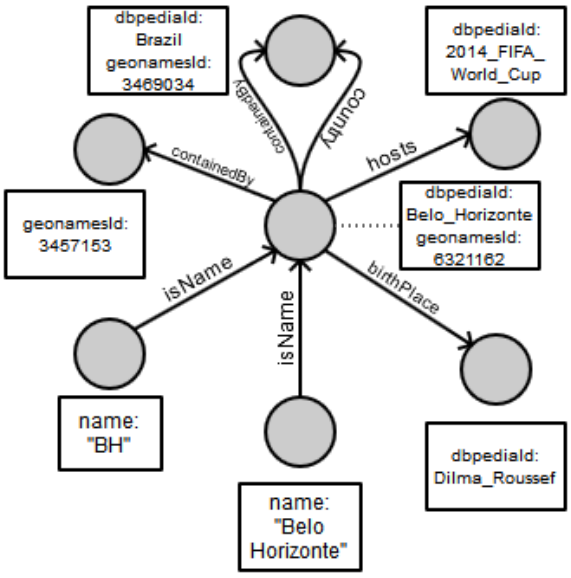


Figura 3.5: Representação em grafo de Belo Horizonte

A Figura 3.6 define formalmente o grafo G resultante. Ele pode ser deinfido como um *property graph*, ou seja, um grafo direcionado, com arestas rotuladas e attributos associados aos vértices e arestas.

$$G = (V, E, \lambda : E \to \Sigma, M : (V \cup E) \times R \to S)$$

Figura 3.6: Definição formal do grafo resultante do mapeamento dos dados. Σ: conjunto de rótulos – R: conjunto de chaves dos atributos – S: conjunto dos valores dos atributos.

Para exemplificar, a Tabela 3.2 apresenta uma representação bastante resumida da cidade de Belo Horizonte, no GeoNames e na DBPedia respectivamente, utilizando triplas RDF. A Figura 3.5 mostra como seria a representação resultante de Belo Horizonte no Linked OntoGazetteer, depois que os dados estruturados em triplas fossem

mapeados para o esquema mostrado na Figura 3.2, utilizando a proposta ilustrada na Figura 3.4. Para obedecer ao esquema do *gazetteer*, os predicados não foram mapeados como arestas diretamente, mas sim como rótulos identificados no esquema como: `TypeURI` e `relURI`. Porém, cada aresta possui uma propriedade que mantém a informação que define o relacionamento. Utilizando a Figura 3.5 como exemplo, essa propriedade estaria preenchida com valores como: *birthplace* e *hosts*.

gn:name	"Belo Horizonte"
gn:alternateName	"Belo Hte"
gn:parentFeature	3457153
gn:parentCountry	3469034
gn:parentADM1	3457153

(a)

foaf:name	"Belo Horizonte"@en
foaf:nick	"BH"@en
db:country	Brazil
db:isPartOf	Southeast_Region,_Brazil
db:isPartOf	Minas_Gerais

(b)

Tabela 3.2: (a) Predicados e objetos que compõem as triplas de Belo Horizonte no GeoNames (http://sws.geonames.org/6321162) e (b) na DBPedia (http://dbpedia.org/resource/Belo_Horizonte)

O fato das fontes de dados existentes no LOD utilizarem armazéns de triplas genéricos para manter os dados facilita muito o processo de adição de novos registros. Entretanto, torna as consultas mais complexas, devido à falta de informação sobre quais são os predicados válidos e o significado de cada um deles. Ontologias deveriam ser utilizadas para resolver esse problema. Mas o que de fato ocorre, quando múltiplas fontes do LOD são utilizadas em conjunto, é a necessidade de um processo de alinhamento das ontologias. Além disso, cada uma das fontes terá representações diferentes de um mesmo lugar, e nem sempre existirá uma indicação óbvia da correspondência entre elas. Esses são alguns dos desafios de se trabalhar simultaneamente com diferentes fontes de dados da *Web of Data*, no processo de coleta e preparação esses problemas serão novamente abordados e iremos mostrar como contornamos tais dificuldades.

Capítulo 4

Coleta e Preparação dos Dados

A Figura 4.1 representa a metodologia utilizada para coleta e preparação dos dados. No decorrer desta seção cada uma das etapas será detalhada.

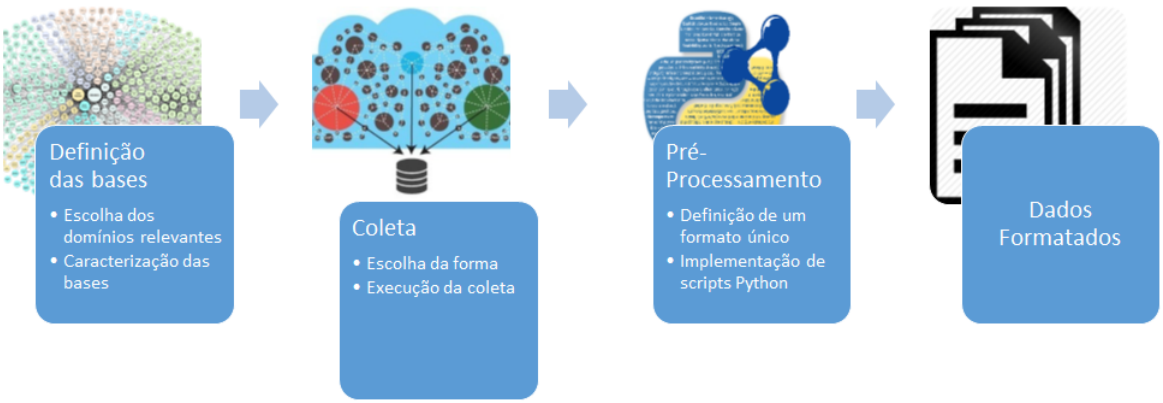


Figura 4.1: Metodologia do processo de Coleta e Preparação dos Dados

O primeiro passo para iniciar a coleta de dados da *Web of Data* é a definição de quais as fontes de dados serão estudadas. O pré-requisito mínimo é conter informações geográficas mas, para atingirmos nosso objetivo de construir um *gazetteer* com cobertura e detalhamento mundial, as fontes também deveriam possuir tais características. Além disso, como fonte de evidências extras para aplicações RIG, bases de dados que não sejam 100% geográficas são muito interessantes.

O grupo que suporta o LOD realiza um trabalho de acompanhamento desde 2010 sobre os conjuntos de dados que fazem parte da *Web of Data*. A análise resultante desse acompanhamento é publicada na *Web* e hoje está na sua quarta versão[1,2,3,4] e

[1]http://lod-cloud.net/state/2010-10-19/
[2]http://lod-cloud.net/state/2011-03-28/
[3]http://lod-cloud.net/state/
[4]http://linkeddatacatalog.dws.informatik.uni-mannheim.de/state/

recentemente um artigo também foi publicado [Schmachtenberg et al., 2014]. Nesse trabalho, um dos objetivos deles é classificar os conjuntos de dados em domínios baseados no tipo de dados que cada um mantém. A Tabela 4.1 apresenta os 8 domínios identificados e a quantidade de conjuntos de dados classificados como pertencentes a cada um deles.

Domínio	Conjuntos de Dados	%
Government	183	18.05%
Publications	96	9.47%
Life sciences	83	8.19%
User-generated content	48	4.73%
Cross-domain	41	4.04%
Media	22	2.17%
Geographic	21	2.07%
Social web	520	51.28%
Total	1014	100%

Tabela 4.1: Número de conjuntos de dados na Web of Data por Domínio em 30/08/2014

Os domínios relevantes para este trabalho são o geográfico (*Geographic*) e o de domínio variado (*Cross-domain*), pois dentro deles existem diversas entidades que representam lugares e outros objetos que frequentemente estão relacionados a um lugar, o que caracteriza elementos relevantes para o Linked OntoGazetteer. Dentro dessas duas categorias, que em agosto de 2014 representavam pouco mais de 6% dos conjuntos de dados, duas fontes possuem grande destaque (ver Figura 2.3): GeoNames e DBPedia. Além de serem dois grandes centralizadores de *links*, também possuem grande volume de dados e estão presentes no LOD desde sua primeira versão. Outras duas fontes, Freebase e LinkedGeoData, também possuem uma boa quantidade de *links* mas, além disso, fornecem dados que estão categorizados no mesmo domínio que DBPedia e GeoNames, respectivamente, e são mantidos de outra maneira. A Freebase [Bollacker et al., 2008] é uma comunidade suportada[5] pelo Google cujos dados alimentam o *Google's Knowledge Graph* [Singhal, 2012]. Os dados do LinkedGeoData são originalmente do OpenStreetMap, uma das mais populares ferramentas de contribuição geográfica voluntária atualmente, e certamente têm muito a acrescentar aos dados do GeoNames.

Nas próximas seções, as fontes utilizadas neste trabalho serão detalhadas quanto ao formato em que os dados estão disponíveis, tamanho dos conjuntos de dados e qual a natureza da informação existente em cada um deles.

[5]O suporte ao Freebase foi encerrado em 30 de Junho de 2015 e seus dados migrados para o Wikidata. Mais informações em: https://plus.google.com/109936836907132434202/posts/bu3z2wVqcQc

4.1 Geonames

O GeoNames é atualmente um dos *gazetteers* com maior cobertura no mundo e utilizado por grandes empresas como Microsoft (Bing Maps)[6] e *New York Times*[7]. Um dos motivos desse grande apelo comercial é o fato dele possuir muitas informações sobre países, cidades, divisões administrativas e elementos geográficos, como rios, montanhas e planícies, do mundo todo. Isso pode ser justificado pelo fato de suas principais fontes de dados serem basicamente agências cartográficas[8] como: *National Geospatial-Intelligence Agency's* e o Instituto Brasileiro de Geografia e Estatística (IBGE). A Figura 4.2 mostra a densidade de entidades existentes no GeoNames espalhadas pelo mundo. Quanto mais clara a região na imagem mais densa ela é. Essa figura é de autoria do próprio GeoNames e é datada de 2006[9].

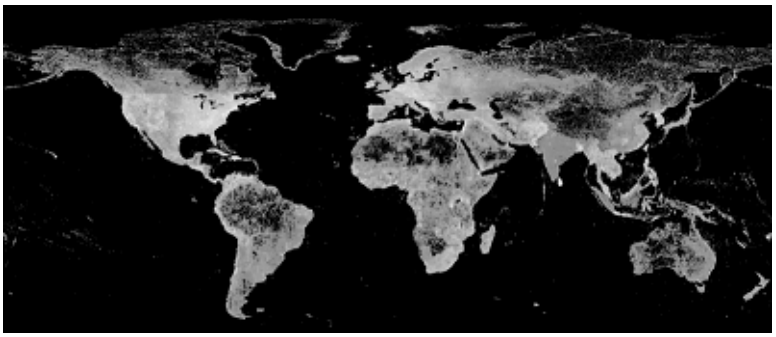


Figura 4.2: Densidade de entidades existentes no GeoNames. Fonte: `https://geonames.wordpress.com/2006/12/07/geonames-feature-density-map/`

A estrutura de dados básica utilizada pelo GeoNames é bastante similar à estrutura padrão de um *gazetteer* e apresenta os topônimos associados ao tipo e o *footprint* do lugar. Uma grande diferença é o armazenamento de nomes alternativos como apelidos e nomes regionais, além do nome do lugar em diversas línguas. A categorização dos lugares é feita associando a cada lugar uma *feature class* e um *feature code*, sendo que eles possuem um relacionamento do tipo um-para-muitos, ou seja, para cada *feature class* existe um conjunto de *feature codes* associados. Existem 9 *feature classes* que estão listadas e exemplificadas (exemplos retirados do próprio GeoNames[10]) na Tabela 4.2. Associados às classes, foram definidos cerca de 700 *feature codes*.

O GeoNames provê acesso aos dados através de um *Web Service*, mas limita o acesso diário a 30.000 entidades, além de existir também um limite de 2.000 requisições

[6] http://upload.maps.bing.com/Help/en-us/About.htm
[7] https://geonames.wordpress.com/2010/01/18/new-york-times-data-api-with-geonames/
[8] http://www.geonames.org/data-sources.html
[9] https://geonames.wordpress.com/2006/12/07/geonames-feature-density-map/
[10] http://www.geonames.org/export/codes.html

Sigla Feature Class	Exemplos
A	país, estado, região,...
H	fluxo de água, lago, ...
L	parques, áreas, ...
P	cidades, vilarejos,...
R	estrada, ferrovia
S	local, construção, fazenda
T	montanha, colina, rochedo,...
U	submarino
V	floresta,...

Tabela 4.2: Feature Class existentes no GeoNames

por hora. Com essas restrições, realizar a coleta de quase 9 milhões de entidades do GeoNames é impraticável. No entanto, também são disponibilizados duas cópias dos dados em formatos distintos: um em formatação tabular e outro utilizando uma notação RDF. Ambos são arquivos de texto, o primeiro tem atributos separados por tabulações, enquanto o segundo utiliza uma representação XML. A Figura 4.3 mostra como Belo Horizonte está representada em cada uma das cópias.

O formato orientado a tabela possui menor riqueza de informações e armazena apenas informações descritivas do lugar, como nome, latitude, longitude e população. O formato que utiliza o RDF como base também possui tais informações descritivas, mas vai além, e possui informações semanticamente mais ricas como *parentFeature* (análogo a "contido em") e *childrenFeatures* (análogo à "contém"). As duas formas foram utilizadas de maneiras complementares nesse trabalho.

Por ser um padrão mais verboso, a representação em RDF possui cerca de 10 vezes o tamanho da base tabular, sendo, respectivamente, 11GB e 1,1GB o tamanho de cada uma das fontes. No total o GeoNames mantém mais de 8,4 milhões de entidades, com mais de 125 milhões de triplas relacionando-as.

Depois de realizado o download dos arquivos, foi realizado um pré-processamento responsável por filtrar os predicados relevantes e formatar os dados utilizando um mesmo padrão utilizado em todas as outras fontes. O padrão escolhido foi o de tripla simples *RDF 1.1 Turtle* [Beckett et al., 2015] (essa foi uma etapa necessária em todas as fontes, com exceção do Freebase que já se encontrava em tal padrão). A Tabela 4.3 lista todos os predicados utilizados no trabalho e indica como cada um foi mapeado dentro do Linked OntoGazetteer. Cada predicado está categorizado como *Aresta* ou *Propriedade*, indicando como ele foi tratado.

A principal diferença entre o formato final e o original do GeoNames é como os dados estão organizados. Orginalmente, cada entidade existente no GeoNames é repre-

geonameid	name	asciiname	alternatenames	latitude	longitude	feature class	country code	timezone
6321162	Belo Horizonte	Belo Horizonte	BH,Belo Horizonte,Belo Hte,...	-19,92623	-43,93982	ADM2	BR	America/Sao_Paulo

(a)

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cc="http://creativecommons.org/ns#" xmlns:dcterms="http://purl.org/dc/terms/"
    xmlns:foaf="http://xmlns.com/foaf/0.1/" xmlns:gn="http://www.geonames.org/ontology#"
    xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:wgs84_pos="http://www.w3.org/2003/01/geo/wgs84_pos#">
  <gn:Feature rdf:about="http://sws.geonames.org/6321162/">
     <rdfs:isDefinedBy rdf:resource="http://sws.geonames.org/6321162/about.rdf" />
     <gn:name>Belo Horizonte</gn:name>
     <gn:alternateName xml:lang="pt">Belo Hte</gn:alternateName>
     <gn:featureClass rdf:resource="http://www.geonames.org/ontology#A" />
     <gn:featureCode rdf:resource="http://www.geonames.org/ontology#A.ADM2" />
     <gn:countryCode>BR</gn:countryCode>
     <wgs84_pos:lat>-19.92623</wgs84_pos:lat>
     <wgs84_pos:long>-43.93982</wgs84_pos:long>
     <gn:parentFeature rdf:resource="http://sws.geonames.org/3457153/" />
     <gn:parentCountry rdf:resource="http://sws.geonames.org/3469034/" />
     <gn:parentADM1 rdf:resource="http://sws.geonames.org/3457153/" />
     <gn:childrenFeatures rdf:resource="http://sws.geonames.org/6321162/contains.rdf" />
     <gn:locationMap rdf:resource="http://www.geonames.org/6321162/belo-horizonte.html" />
  </gn:Feature>
  <foaf:Document rdf:about="http://sws.geonames.org/6321162/about.rdf">
     <foaf:primaryTopic rdf:resource="http://sws.geonames.org/6321162/" />
     <cc:license rdf:resource="http://creativecommons.org/licenses/by/3.0/" />
     <cc:attributionURL rdf:resource="http://sws.geonames.org/6321162/" />
     <cc:attributionName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">GeoNames</cc:attributionName>
     <dcterms:created rdf:datatype="http://www.w3.org/2001/XMLSchema#date">2006-12-15</dcterms:created>
     <dcterms:modified rdf:datatype="http://www.w3.org/2001/XMLSchema#date">2014-07-18</dcterms:modified>
  </foaf:Document>
</rdf:RDF>
```

(b)

Figura 4.3: Estruturas de dados utilizadas pelo GeoNames para representar Belo Horizonte: (a) tabular e (b) RDF

sentada como um "sub-documento"dentro do arquivo principal, assim sendo, dentro de cada um desses "sub-documentos"o sujeito da tripla é a entidade representada, mas ele não é explicitamente definido. No arquivo resultante do pré-processamento, cada linha representa uma tripla simples completa e possui todos os elementos (sujeito, predicado e objeto). A Figura 4.4 mostra um trecho do arquivo final.

```
<http://sws.geonames.org/1/> <http://www.geonames.org/ontology#parentCountry> <http://sws.geonames.org/130758/> .
<http://sws.geonames.org/1/> <http://www.geonames.org/ontology#featureClass> <http://www.geonames.org/ontology#T> .
<http://sws.geonames.org/1/> <http://www.geonames.org/ontology#featureCode> <http://www.geonames.org/ontology#T.MT> .
<http://sws.geonames.org/1/> <http://www.w3.org/2003/01/geo/wgs84_pos#lat> "32.98333" .
<http://sws.geonames.org/1/> <http://www.w3.org/2003/01/geo/wgs84_pos#long> "49.13333" .
<http://sws.geonames.org/1/> <http://www.geonames.org/ontology#parentFeature> <http://sws.geonames.org/125605/> .
<http://sws.geonames.org/1/> <http://www.geonames.org/ontology#alternateName> "K\u016Bh-e Zardar"@fa .
<http://sws.geonames.org/1/> <http://www.geonames.org/ontology#name> "K\u016Bh-e Zardar" .
<http://sws.geonames.org/1/> <http://www.geonames.org/ontology#parentADM1> <http://sws.geonames.org/125605/> .
<http://sws.geonames.org/3/> <http://www.geonames.org/ontology#featureClass> <http://www.geonames.org/ontology#P> .
<http://sws.geonames.org/3/> <http://www.geonames.org/ontology#alternateName> "Zam\u012Bn S\u016Bkhteh"@fa .
<http://sws.geonames.org/3/> <http://www.geonames.org/ontology#featureCode> <http://www.geonames.org/ontology#P.PPL> .
```

Figura 4.4: Estrutura pré-processada do GeoNames

O arquivo resultante do pré-processamento possui 8,3GB e aproximadamente 79 milhões de linhas, que representam predicados importados para o Linked OntoGazet-

URI do Predicado	Linked OntoGazetteer	Tipo
gn:parentCountry	*Place –country–> Place*	Aresta
gn:parentFeature	*Place –containedBy–> Place*	Aresta
gn:alternateName	*Name –isName–> Place*	Aresta
gn:name	*Name –isName–> Place*	Aresta
gn:parentADM[12345]	*Place –containedBy–> Place*	Aresta
gn:wikipediaArticle	*Place.dbpediaId*	Propriedade
gn:featureClass	*Place.gnFeatureClass*	Propriedade
gn:featureCode	*Place.gnFeatureCode*	Propriedade
wgs84_pos:lat	*Place.gnPoint.y*	Propriedade
wgs84_pos:long	*Place.gnPoint.x*	Propriedade

Tabela 4.3: Predicados selecionados no GeoNames e respectivo mapeamento no Linked OntoGazetteer

teer. Todas as mais de 8,4 milhões de entidades foram mantidas até este ponto. Considerando que as triplas estão uniformemente distribuídas entre as entidades, temos uma média de 9,4 triplas para cada entidade. Esse número é ilustrativo para mostrar o volume de informação que está sendo trabalhada, já que na verdade as triplas não estão distribuídas de maneira uniforme.

4.2 DBPedia

Assim como GeoNames, a DBPedia faz parte do LOD desde sua versão inicial e atualmente é a base de dados mais referenciada da *Web of Data*. A DBPedia é suportada por uma comunidade que busca extrair informações da Wikipédia, principalmente informações estruturadas contidas nas *infoboxes* dos tópicos, e torná-las acessíveis utilizando os princípios do *Linked Data*. Por ser derivada da Wikipédia, a natureza das informações é variada, com uma cobertura mundial e um bom nível de detalhamento, dada a popularização da enciclopédia virtual nos dias de hoje.

Os dados disponibilizados pela DBPedia são cópias de todas as triplas existentes na base de conhecimento, sendo assim a formatação é bastante semelhante ao formato final definido. O conjunto de dados da DBPedia utilizado neste trabalho é referente à versão 3.9[11] publicada no final do ano de 2013. No final de 2014 uma nova versão[12] foi publicada, mas como o trabalho já estava em andamento optamos por não atualizar a base para evitar atrasos e manter resultados anteriores válidos. Apenas os tópicos que

[11] `http://downloads.dbpedia.org/3.9/`
[12] `http://downloads.dbpedia.org/2014/`

possuem entrada na Wikipédia em inglês estão sendo tratados, assim existem 3.241.079 entidades, das quais 639.462 são categorizadas como lugares, e 25.896.867 triplas.

Devido à característica colaborativa da Wikipédia, os predicados existentes variam muito. Para iniciar a análise de quais seriam os predicados relevantes, a busca por predicados foi restringida aos predicados associados a lugares. Para isso foi realizada uma consulta por todas as entidades cujo predicado *Tipo*[13] tivesse o valor *Lugar*[14]. Na ontologia da DBPedia, *Lugar* é um conceito de alto nível, descendente direto de *Coisa*[15]. Após esse filtro, os predicados relevantes foram selecionados. A Tabela 4.4 apresenta o resultado dessa etapa, e também mostra como cada predicado foi mapeado dentro do Linked OntoGazetteer.

Predicado da DBPedia	**Linked OntoGazetteer**	**Tipo**
foaf:name	*Name –isName–> Place*	Aresta
foaf:nick	*Name –isName–> Place*	Aresta
db:country	*Place –country–> Place*	Aresta
db:isPartOf	*Place –containedBy–> Place*	Aresta
db:location	*Place –containedBy–> Place*	Aresta
db:state	*Place –containedBy–> Place*	Aresta
db:region	*Place –containedBy–> Place*	Aresta
georss:point	*Place.dbpoint*	Propriedade

Tabela 4.4: Predicados selecionados no DBPedia e respectivo mapeamento no Linked OntoGazetteer

O gazetteer também inclui as entidades que não são categorizadas como lugar e seus relacionamentos com lugares pois, como apresentado por Alencar et al. [2010], esse tipo de informação pode ajudar aplicações RIG a realizarem tarefas como resolução de ambiguidades. Sendo assim, toda tripla (s, p, o) na qual $o \in Lugar$ também foi selecionada. A representação das entidades que não são lugares é análoga à representação de lugares: todas as arestas entre vértices da classe `Non-Place` e vértices da classe `Place` são armazenados como `Place`–`isRelated`–>`Non-Place` e a URI (p) do relacionamento original é mantida como rótulo da aresta.

Todas as triplas cujos predicados são *foaf:name* ou *foaf:nick* também foram mantidas, desde que elas se relacionem com entidades da classe `Place`, ou entidades da classe `Non-Place` que estejam relacionadas a um `Place`.

O resultado da etapa de pré-processamento dos dados da DBPedia foi dividido em 3 arquivos, que, juntos, ocupam pouco menos de 1,5GB: *allPlacesAttrDbpedia_en.nt*

[13] `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`
[14] `http://dbpedia.org/ontology/Place`
[15] `http://mappings.dbpedia.org/server/ontology/classes/Place`

contendo todas triplas que se enquadram nos predicados descritos na Tabela 4.4, *allNonPlacesDBPedia_en.nt* que guarda todas as entidades não classificadas como lugares que se relacionam a lugares e *allNonPlacesNamesDBPedia_en.nt* que armazena os nomes e nomes alternativos das entidades da classe *Non-Place*. A Tabela 4.5 sumariza as características de cada um deles.

Arquivo	**# de Entidades (E)**	**# de Triplas (T)**	T/E
allPlacesAttrDbpedia_en.nt	639.462	7.348.413	11,5
allNonPlacesDBPedia_en.nt	663.636	1.389.370	2,1
allNonPlacesNamesDBPedia_en.nt	594.026	707.400	1,2

Tabela 4.5: Resultado da etapa de coleta e preparação dos dados da DBPedia

A diferença entre a quantidade de entidades nos arquivos *allNonPlacesDBPedia_en.nt* e *allNonPlacesNamesDBPedia_en.nt*, é motivada pelo fato de nem todas as entidades não categorizadas como lugares possuírem triplas com os predicados *foaf:name* ou *foaf:nick*. A ausência de um nome associado inviabiliza a importação da entidade para o gazetteer, já que uma entidade sem nome jamais retornará em uma busca.

4.3 Freebase

A Freebase é um projeto de contribuição voluntária suportado pelo Google, ou seja, a plataforma é aberta para edição. Aplicações que suportam contribuição voluntária de usuário, normalmente são capazes de manter informações com maior quantidade de detalhes, pois comumente os autores estão diretamente relacionados aos tópicos. Ela também é uma base do LOD cujo domínio é diversificado (assim como a DBPedia), possuindo registros categorizados como músicas, livros, localidades e pessoas. Para este trabalho, as localidades (análogo a Lugar na DBPedia) são as principais entidades, e irão alimentar a classe `Place`. Todas as demais categorias que possuírem relacionamentos com localidades serão destinadas à classe `Non-Place`.

A Freebase disponibiliza os seus dados através de cópias atualizadas semanalmente [Google, 2013]. O download da cópia utilizada no trabalho foi feito em novembro de 2013, possui cerca de 360GB e aproximadamente 2,9 bilhões de triplas.

A Freebase também é fortemente influenciada pela Wikipédia mas, diferentemente da DBPedia, não possui foco nas informações estruturadas existentes. A ontologia por trás da plataforma suportada pelo Google também é muito peculiar, se comparada com a DBPedia, possuindo predicados semanticamente muito mais específicos. Para

iniciar a análise de quais seriam os predicados relevantes, a pesquisa foi restringida às localidades existentes na Freebase. Para isso foi realizada uma busca por todas as entidades cujo predicado *Tipo*[16] tivesse o valor *Localidade*[17]. Após esse filtro, os predicados relevantes foram selecionados e a Tabela 4.6 mostra o resultado dessa etapa, indicando como cada predicado foi mapeado dentro do Linked OntoGazetteer.

Predicado da Freebase	**Linked OntoGazetteer**	**Tipo**
fbase:ns:base.biblioness.bibs_location.country	*Place –country–> Place*	A
fbase:ns:common.topic.alias	*Name –isName–> Place*	A
fbase:ns:location.location.containedby	*Place –containedBy–> Place*	A
fbase:ns:location.location.contains	*Place –containedBy–> Place*	A
fbase:ns:location.location.nearby_airports	*Place –containedBy–> Place*	A
fbase:ns:type.object.name	*Name –isName–> Place*	A
fbase:key:user.metaweb.datasource.geonames	*Place.geonamesId*	P
fbase:key:wikipedia.en_title	*Place.dbpediaId*	P
fbase:ns:location.geocode.latitude	*Place.fbPoint.y*	P
fbase:ns:location.geocode.longitude	*Place.fbPoint.x*	P

Tabela 4.6: Predicados selecionados na Freebase e respectivo mapeamento no Linked OntoGazetteer: arestas (A) ou propriedades (P)

Por se tratar de uma base de dados também de domínio variado, o critério para a seleção de entidades que são instâncias da classe `Non-Place` foi o mesmo da DBPedia. O resultado da etapa de pré-processamento da Freebase também foi dividido em 3 arquivos, e as estatísticas de cada um deles estão sumarizadas na Tabela 4.7.

Arquivo	**# Entidades (E)**	**# Triplas (T)**	T/E
allPlacesAttrFreebase.nt	1.642.081	26.596.877	16,2
allNonPlacesFreebase.nt	5.791.701	7.320.213	1,3
allNonPlacesNamesFreebase.nt	3.902.886	9.828.708	2,5

Tabela 4.7: Resultado da etapa de coleta e preparação dos dados da Freebase

Assim como na DBPedia, nem todas entidades não categorizadas como lugares possuem predicados que identificam nomes (*fbase:ns:common.topic.alias*[18] ou *fbase:ns:type.object.name*[19]). Por esse motivo os arquivos *allNonPlacesFreebase.nt* e *allNonPlacesNamesFreebase.nt* possuem uma diferença de 1.888.815 entidades.

Na Freebase, a variedade de predicados que relacionam objetos não classificados como lugares aos que são é muito maior se comparada à DBPedia. São 1.963 predicados

[16] `http://rdf.freebase.com/ns/type.object.type`
[17] `http://rdf.freebase.com/ns/location.location`
[18] `http://rdf.freebase.com/ns/common.topic.alias`
[19] `http://rdf.freebase.com/ns/type.object.name`

diferentes, contra 266 da DBPedia. Desse universo de predicados existentes na base de conhecimento suportada pelo Google, apenas uma pequena parcela possui uma quantidade relevante de ocorrências, sendo que apenas 158 predicados possuem mais de 1.000 triplas. Por essa razão, apenas os predicados mais frequentes foram mantidos nas etapas subsequentes do trabalho. O filtro escolhido foi manter os predicados mais frequentes que, juntos, representem 90% das triplas que relacionam `Non-Place` e `Place`. A Tabela 4.8 mostra os 31 predicados resultantes dessa etapa.

4.4 LinkedGeoData

LinkedGeoData é uma base de informações geográficas que integra o projeto LOD. Os dados são coletados de uma das aplicações de contribuição geográfica voluntária mais populares do mundo, o OpenStreetMap (OSM). Diferentemente do GeoNames, que possui instituições cartográficas oficiais como o IBGE como fonte de dados, o projeto LinkedGeoData se baseia em contribuições de usuários. Isso faz com que o nível de detalhamento intra-urbano aumente muito se comparado ao GeoNames. Em contrapartida, gera um enviesamento da base de conhecimento para as regiões do mundo onde o OSM é mais popular.

Outra consequência de ser baseado em uma fonte de dados colaborativa é a variedade de representações utilizadas para representar lugares pelos usuários. Basicamente existem três estruturas de dados: *way*, *node* e *relation*. *Way* usa representações dos tipos `Linha` e `Polígono`. Suas instâncias comumente são utilizadas para definir estruturas de nível intra-urbano, como parques, lagos e construções. A Figura 4.5 mostra alguns exemplos de utilização dessa estrutura de dados.

(a) (b)

Figura 4.5: Exemplos de utilização do tipo de dados *Way* do OSM como (a) um polígono e (b) uma linha

Predicado	**# Triplas**	**% Triplas**
people.person.place_of_birth	954568	13,04%
music.release.region	950329	12,98%
people.person.nationality	886278	12,11%
book.book_edition.place_of_publication	580481	7,93%
common.image.appears_in_topic_gallery	391148	5,34%
education.education.institution	377465	5,16%
people.place_lived.location	359948	4,92%
film.film_regional_release_date.film_release_region	302879	4,14%
film.film.country	230222	3,15%
people.deceased_person.place_of_death	218833	2,99%
dataworld.gardening_hint.last_referenced_by	217966	2,98%
organization.organization.headquarters	198099	2,71%
dataworld.gardening_hint.replaced_by	141281	1,93%
common.webpage.topic	117389	1,60%
education.school.school_district	97677	1,33%
time.event.locations	81027	1,11%
medicine.manufactured_drug_form.available_in	51771	0,71%
music.artist.origin	51348	0,70%
tv.tv_program.country_of_origin	47820	0,65%
protected_sites.natural_or_cultural_site_listing.listed_site	42859	0,59%
book.written_work.subjects	39284	0,54%
time.time_zone.locations_in_this_time_zone	38722	0,53%
base.usnris.significance_level.listings	36398	0,50%
base.schemastaging.phone_sandbox.service_location	30100	0,41%
location.adjoining_relationship.adjoins	29463	0,40%
architecture.architectural_style.examples	26160	0,36%
business.employment_tenure.company	23202	0,32%
olympics.olympic_athlete_affiliation.country	22043	0,30%
location.geometry.delineates_location	20716	0,28%
transportation.road_starting_point.location	19347	0,26%
broadcast.broadcast.area_served	18653	0,25%
TOTAL	6.603.476	90,21%

Tabela 4.8: Predicados escolhidos da Freebase que relacionam entidades categorizadas como não lugares e lugares. Todos os predicados são prefixados por http://rdf.freebase.com/ns/

Um *node* é basicamente um ponto geográfico. A sua utilização é variada e comumente está relacionada a desde pontos de interesse como restaurantes, monumentos e pontos turísticos, até áreas maiores como bairros, cidades e divisões administrativas. A Figura 4.6 mostra a utilização do tipo *node* para representar o campus da Pampulha da UFMG.

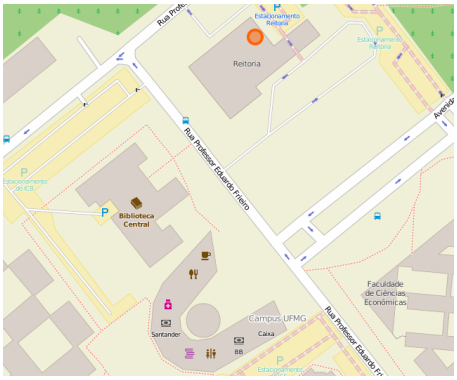
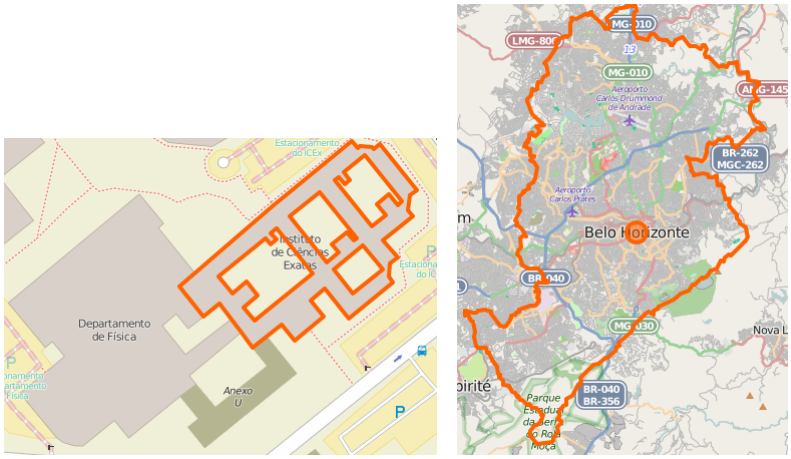


Figura 4.6: Exemplo de utilização do tipo de dados *Node* do OSM

A estrutura de dados *relation* é uma estrutura de dados complexa e resultado da associação de *ways* e *nodes*. Ela é utilizada para representar desde elementos intra-urbanos caracterizados por um conjunto de polígonos, até cidades e divisões administrativas que possuem seu perímetro representado como um *way*, mas também possuem um *node* como representação geográfica mais simples. A Figura 4.7 mostra duas formas em que *relation* é utilizada.



(a) (b)

Figura 4.7: Exemplos de utilização do tipo de dados *Relation* do OSM como (a) um conjunto de polígonos e (b) a associação de um *Way* com um *Node*

O grupo que suporta o LinkedGeoData disponibiliza cópias da base categorizadas por tipo e estrutura de dados. Neste trabalho, focamos na utilização de lugares que estão representados como *way* e/ou *node*, pois são as representações básicas para qualquer elemento no OSM e a estrutura de dados *relation* nada mais é que uma combinação das outras. Um lugar no LinkedGeoData é caracterizado pela existência da

tripla cujo predicado *Tipo*[20] possui valor *Lugar*[21] associado à uma entidade. A cópia dos dados do LinkedGeoData possui pouco menos de 16GB. Cerca de 600 mil lugares são representados como *ways* e 3 milhões de lugares são representados como *nodes*.

Diferentemente das outras fontes de dados utilizadas, o LinkedGeoData não utiliza URIs para referenciar objetos existentes dentro da base de conhecimento, ferindo assim um dos princípios do *Linked Data*. Um exemplo disso é o predicado *isIn*[22] (semanticamente representa a ideia de "estar contido em"), cujos objetos são literais multivalorados ou, mais precisamente, cadeias de caracteres separadas por vírgulas. A Figura 4.8 mostra um exemplo da utilização do predicado *isIn* e como seria a utilização correta, seguindo os princípios do *Linked Data*.

lgd:node246673089, lgdo:isIn, "Minas Gerais, Brasil"
(a)

lgd:node246673089, lgdo:isIn, lgd:node502334045
lgd:node246673089, lgdo:isIn, lgd:node424313734
(b)

Figura 4.8: (a) Forma como o predicado *isIn* é utilizado no LinkedGeoData e (b) como seria a forma correta se todos os princípios do *Linked Data* fossem seguidos

Essa decisão tomada pelo LinkedGeoData inviabiliza a utilização dos predicados *isIn* de maneira direta, sem antes ter de associar os nomes a objetos na base de conhecimento, o que é um processo sujeito a ambiguidades. Infelizmente, todos os predicados do LinkedGeoData que semanticamente representam relacionamentos do tipo "está contido em", e possivelmente seriam representados como arestas do tipo `Place` –`containedBy`–> `Place` no Linked OntoGazetteer, sofrem do mesmo problema que o predicado *isIn*. Mesmo assim, eles foram integrados, mas na forma de uma propriedade multivalorada que armazena todos os nomes de lugares relacionados (*lgdRelatedName*). A Tabela 4.9 apresenta todos os predicados selecionados do LinkedGeoData, não fazendo distinção entre *way* ou *node*, e como cada um dos predicados foi representado no Linked OntoGazetteer.

A variedade de predicados no LinkedGeoData é muito grande. São quase 6.000 predicados diferentes associados a lugares, sejam eles *way* ou *node*. Para este trabalho, foram selecionados muitos predicados, devido ao fato de haver muita sobreposição semântica entre eles. Um exemplo é a existência de 6 predicados mapeados para nomes que na prática não possuem diferenciação, como mostra a Tabela 4.9.

[20] `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`
[21] `http://linkedgeodata.org/ontology/Place`
[22] `http://linkedgeodata.org/ontology/isIn`

Predicado do LinkedGeoData	**Linked OntoGazetteer**	**Tipo**
lgdo:internationalName	*Name –isName–> Place*	Aresta
lgdo:alt_name_be	*Name –isName–> Place*	Aresta
lgdo:name_genitive	*Name –isName–> Place*	Aresta
lgdo:officialName	*Name –isName–> Place*	Aresta
rdfs#label	*Name –isName–> Place*	Aresta
skos:altLabel	*Name –isName–> Place*	Aresta
lgdo:geonames_id	*Place.geonamesId*	Propriedade
lgdo:wikipedia	*Place.dbpediaId*	Propriedade
geo#lat	*Place.lgdpoint.y*	Propriedade
geo#long	*Place.lgdpoint.x*	Propriedade
lgdo:addr_country	*Place.lgdRelatedName[]*	Propriedade
lgdo:addr_district	*Place.lgdRelatedName[]*	Propriedade
lgdo:addr_region	*Place.lgdRelatedName[]*	Propriedade
lgdo:addr_subdistrict	*Place.lgdRelatedName[]*	Propriedade
lgdo:addr:postcode	*Place.lgdRelatedName[]*	Propriedade
lgdo:cladr_code	*Place.lgdRelatedName[]*	Propriedade
lgdo:is_in_country_code	*Place.lgdRelatedName[]*	Propriedade
lgdo:is_in_country	*Place.lgdRelatedName[]*	Propriedade
lgdo:is_in_municipality	*Place.lgdRelatedName[]*	Propriedade
lgdo:is_in_province	*Place.lgdRelatedName[]*	Propriedade
lgdo:is_in_region	*Place.lgdRelatedName[]*	Propriedade
lgdo:is_in_state_code	*Place.lgdRelatedName[]*	Propriedade
lgdo:is_in_state	*Place.lgdRelatedName[]*	Propriedade
lgdo:is_in_village	*Place.lgdRelatedName[]*	Propriedade
lgdo:isIn	*Place.lgdRelatedName[]*	Propriedade
lgdo:postal_code	*Place.lgdRelatedName[]*	Propriedade

Tabela 4.9: Predicados selecionados no LinkedGeoData e respectivo mapeamento no Linked OntoGazetteer

O resultado da etapa de pré-processamento dos dados do LinkedGeoData foi mantido em arquivos diferentes, cujo critério de separação é a representação utilizada no OSM, *way* ou *node*. A Tabela 4.10 sumariza as informações sobre os arquivos finais desta etapa.

Arquivo	**# Entidades** (E)	**# Triplas** (T)	T/E
filteredLGDWayPlacesAttrs.nt	595.130	2.411.942	4,1
filteredLGDNodePlacesAttrs.nt	3.005.750	25.635.241	8,5

Tabela 4.10: Resultado da etapa de coleta e preparação dos dados da Freebase

Capítulo 5

Formação do Linked OntoGazetteer

5.1 Implementação

Os relacionamentos apresentados na seção 3.2 possuem cardinalidade muitos-para-muitos. Esse tipo de relacionamento tem certa complexidade para ser implementado por bancos de dados relacionais, pois (1) será criada uma tabela N:M que materialize o relacionamento entre duas entidades ou (2) o banco não estará normalizado. A primeira solução penaliza muito o tempo de consulta, pois adiciona junções na execução da consulta. Em contrapartida, a segunda solução simplifica a consulta, removendo a necessidade de executar junções, mas adiciona redundância de dados, o que certamente é um problema para a manutenção da integridade do banco. Nesse cenário, uma ferramenta que utilize uma estrutura de dados mais próxima do modelo apresentado é o ideal. A estrutura de um grafo é bastante adequada ao modelo e por isso foi decidido que seria utilizado um banco de dados orientado a grafos. Sendo assim, nesse banco de dados as classes `Place`, `Name`, `Non-place` e `Type` são mapeadas para vértices, enquanto todos os relacionamentos são mapeados para arestas rotuladas. Apesar de definir um esquema, a estrutura de grafo escolhida não torna a tarefa de criação de novos registros custosa. Isso porque o esquema definido é muito simples e não possui nenhuma restrição complexa de ser seguida.

O modelo foi implementado utilizando o Titan[1], um banco de dados NoSQL orientado a grafos. A instalação do Titan foi realizada sobre uma instância do Apache Cassandra[2], que também é um banco de dados NoSQL, mas do tipo *column family* [Lakshman & Malik, 2009]. No Titan, um vértice é representado por um conjunto de propriedades, e os vértices não precisam compartilhar uma estrutura predefinida. Ou

[1] http://thinkaurelius.github.io/titan/
[2] http://cassandra.apache.org/

seja, dois vértices em um mesmo grafo não precisam possuir nenhuma propriedade em comum. Isso permite que, em um só grafo, vértices sejam usados para representar instâncias de classes diferentes. Essa é uma característica importante para a implementação do modelo proposto, pois viabiliza a existência de vértices das diversas classes apresentadas. Entretanto, vértices de uma mesma classe tendem a possuir o mesmo conjunto de propriedades, para facilitar a construção de consultas e entendimento do esquema.

As propriedades existentes em um grafo não precisam ser definidas de maneira prévia, e podem ser criadas à medida que a necessidade surja, o que é uma funcionalidade bastante interessante para a manutenção e evolução do *gazetteer*. Uma propriedade também pode ter associada a ela algumas restrições. A Tabela 5.1 lista as restrições suportadas pelo Titan.

Restrição	**Descrição**	**Indexada**
Única	O valor dessa propriedade é único entre os vértices	Sim
Singular	Cada vértice só pode conter uma propriedade desse tipo	Sim
Lista	Propriedades multivaloradas	Não

Tabela 5.1: Restrições existentes para uma propriedade no Titan

A coluna *Indexada* da Tabela 5.1 indica que na criação de uma restrição (única ou singular), o Titan automaticamente gera um índice sobre aquela propriedade. O objetivo desse índice é viabilizar o acesso aos vértices utilizando propriedades como critérios de busca.

O Titan atualmente suporta poucas representações geográficas como atributos de vértices de maneira nativa, apenas pontos e retângulos. Neste primeiro momento, todas as representações foram resumidas a pontos e futuramente, quando o Titan suportar todas as representações propostas no esquema ou utilizando alguma outra estratégia as demais representações serão incorporadas. Esse não é um problema muito grave incialmente, pois todas as fontes de dados utilizadas neste trabalho, com exceção do LinkedGeoData, utilizam apenas a representação de pontos associados a lugares.

Arestas também podem conter propriedades, e internamente no Titan possuem uma definição muito semelhante à dos vértices quanto à estrutura de dados. A principal diferenciação entre vértices e arestas é o fato de arestas, além de possuirem propriedades que podem ter restrições, também podem indicar seu direcionamento, podendo ser uma aresta incidente, de saída ou ambos. A navegação entre vértices adjacentes por arestas

que possuem sua direção explicitada é feita de maneira mais eficiente do que utilizando arestas que não possuem indicação de direção.

Quando um vértice é criado, é gerada para ele uma propriedade *ID*. Essa é a única propriedade que existe obrigatoriamente em todos os vértices e arestas do grafo. Fazendo uma analogia aos bancos de dados relacionais, essa propriedade pode ser vista como uma chave substituta de uso geral e também é a maneira mais rápida para se acessar um registro dentro do Titan.

O Titan suporta a utilização de uma linguagem específica para caminhamento em grafos chamada Gremlin[3], o que facilita muito a elaboração de consultas sobre o grafo. Essa linguagem tem como objetivo prover uma maneira elegante de realizar buscas locais na estrutura armazenada pelo Titan. Além disso ela é um padrão que independe da implementação do banco de dados orientado a grafos. A Figura 5.2 mostra um exemplo de consulta Gremlin que calcula recomendações colaborativas para o usuário cujo login é `john.doe` (recomendações oriundas de "curtidas" de outras pessoas que "curtem" as mesmas coisas que ele) [Jannach et al., 2010], utilizando como base o grafo da Figura 5.1.

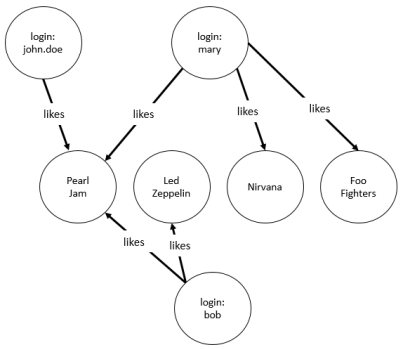


Figura 5.1: Grafo exemplo para a consulta da Figura 5.2

```
g.V('login', 'john.doe').out('likes').in('likes').out('likes').groupCount(m)
```

Figura 5.2: Poder de expressão da linguagem de Consulta Gremlin - consulta que calcula recomendações colaborativas

A Figura 5.2 mostra o quão expressiva é a linguagem, mas para efeito de comparação um pouco mais contextualizada a Figura 5.3 mostra a consulta SQL utilizada no OntoGazetteer para recuperar todos os lugares ambíguos dado um topônimo e a consulta utilizando o Gremlin que retorna o mesmo resultado no Linked OntoGazetteer.

[3]https://github.com/tinkerpop/gremlin/wiki

```
SELECT * FROM PLACE P JOIN ALTERNATIVE_PLACE AP                    (a)
ON P.ID_PLACE = AP.ID_PLACE
WHERE P.NAMEPLACE = 'Paris' OR AP.ALTNAME = 'Paris'

g.V('name', 'Paris').out('isName')                                 (b)
```

Figura 5.3: Comparação de uma consulta SQL sobre o OntoGazetteer (a) e uma consulta Gremlin sobre o Linked OntoGazetteer (b)

5.2 Integração de Fontes de Dados

Mesmo após toda a etapa de preparação dos dados, importá-los para o *gazetteer* diretamente fará com que as informações das quatro fontes utilizadas só compartilhem nomes, pois as entidades de cada uma das fontes ainda não foram avaliadas com o objetivo de identificar equivalência entre elas. Sendo assim, o resultado da importação nesse estágio do trabalho não seria satisfatório, já que as fontes de dados diferentes não seriam complementares em cobertura e detalhamento de informações. A Figura 5.4 mostra a diferença do grafo resultante nessa etapa com o desejado.

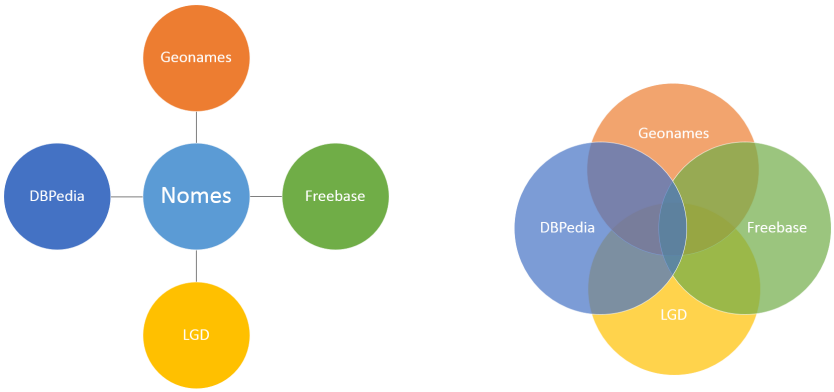


Figura 5.4: Grafo (a) com as fontes de dados se relacionando apenas através dos nomes e (b) com as fontes se relacionando diretamente

Vértices que representam lugares na primeira versão (Figura 5.4 a) e referem a um mesmo nome devem ser lugares homônimos, porém distintos, pois caso contrário ficaria caracterizada uma ocorrência de duplicidade de representação, que precisaria ser resolvida. O motivo pelo qual a duplicidade de representação precisa ser resolvida está relacionado ao fato de adicionar redundância de dados e aumentar o número de registros ambíguos, além de impedir que as fontes de dados funcionem de maneira complementar.

Com o objetivo de identificar as representações duplicadas, propomos um algoritmo que usa os relacionamentos e as propriedades para confirmar se dois vértices diferentes se referem, de fato, a um mesmo lugar, com um bom nível de certeza. Para avaliar a validade dessa estratégia assumimos que os predicados existentes nas fontes de dados que indicam equivalência entre elas estão corretos. Idealmente, o único predicado utilizado deveria ser o *sameAs*, mas devido à falta de padrão em sua utilização cada fonte acaba adotando uma maneira própria para realizar a indicação de equivalência. As URIs da DBPedia possuem uma particularidade: os identificadores dos objetos são derivados dos identificadores da Wikipédia. Assim, em alguns casos a identificação de equivalência foi realizada pelos tópicos na Wikipédia e não por uma referência direta à DBPedia.

O algoritmo considera candidato para integração todo par de lugares que compartilham um nome, ou seja, para cada vértice do tipo `Name` (N) que possui dois ou mais vizinhos do tipo `Place` é construída uma lista de lugares candidatos L_c. Para cada par de lugares $p_i, p_j \in L_c$ é avaliado se eles possuem algum *link* equivalente ao *sameAs*. Caso essa verificação seja positiva os nós p_i e p_j são integrados e o algoritmo continua.

Em caso negativo na comparação anterior o algoritmo precisa possuir critérios bons o suficiente para evitar falsos positivos (lugares apontados como duplicados mas que na verdade são apenas homônimos). Para isso o algoritmo faz as próximas comparações em três níveis. Primeiro, levando em consideração que a maioria dos lugares está associada a um único país, é avaliado se o país associado a p_i e p_j é o mesmo. Caso não seja, a dupla é considerada distinta e o algoritmo prossegue. Essa comparação adiciona evidências positivas à correspondência dos nomes aos lugares, deixando de lado apenas algumas entidades (continentes e países) que podem ser tratadas com métodos específicos. Nos testes iniciais verificamos que muitos erros nessa comparação aconteciam devido a variações nos nomes dos países de acordo com cada fonte de dados.

Em um segundo momento, é avaliada a lista dos lugares que contêm p_i e p_j, por exemplo lugares relacionados a p_i e p_j através do relacionamento *containedBy*. Se a interseção entre as listas é vazia, os lugares são considerados distintos. O caso de ser não vazia indica que p_i e p_j estão dentro de uma mesma região, o que diminui muito a chance deles serem distintos. Para finalizar, são comparadas as representações geográficas de p_i e p_j. Se a distância entre eles foi pequena (foi definido um limiar de 5 Km depois de analisar a distribuição das distâncias), os lugares são considerados equivalentes, e então unificados.

O algoritmo é simples e poderia levar em consideração muitos outros fatores, como relacionamentos semânticos e relacionamentos *geo/non-geo*. Entretanto, o obje-

tivo desse algoritmo é apenas reduzir o impacto da existência de duplicidade de representações, ficando fora do escopo desse trabalho a criação de um mecanismo sofisticado para resolver tal tarefa. A Figura 5.5 ilustra o pseudocódigo da solução.

```
for each n in N
    L_c = buildCandidateList(n)

    for(i=0; i < len(L_c)-1; i++)
        for(j=i+1; j < len(L_c)-1; j++)

            if(existsLinkBetweenPlaces(p[i], p[j]))
                merge(p[i], p[j])
                continue
            end

            if(isSameCountry(p[i], p[j]))
                containedByList_i = getContainedByPlaces(p[i])
                containedByList_j = getContainedByPlaces(p[j])

                if(not isDisjoint(containedByList_i,containedByList_j))
                    if(distance(p[i], p[j]) < 5000)
                        merge(p[i], p[j])
                    end
                end

            end
        end
    end
end
```

Figura 5.5: Algoritmo para identificação de lugares duplicados

O algoritmo apresentado possui uma complexidade quadrática em função do tamanho de L_c. Além disso, a função `merge()` é computacionalmente cara, pois todas as arestas de um vértice precisam ser copiadas para o outro e só então o vértice desnecessário pode ser removido. Esse processo de cópia e remoção envolve muitas transações no banco de dados, e o Titan, assim como a maioria dos gerenciadores de bancos de dados NoSQL, não suportam integralmente a propriedade de consistência. Dessa forma, fica a cargo da aplicação fazer tais verificações, aumentando ainda mais o número de operações no banco de dados. Essas diversas operações sobre o Titan ocasionam uma penalidade na performance da aplicação, pois acabam gerando *locks* no Cassandra, impactando as verificações que ainda estão sendo feitas pelo algoritmo. A fim de evitar esse problema, o processo de integração dos lugares não unifica as entidades dupli-

cadas no momento em que identifica a duplicidade. O processo apenas salva em um arquivo os identificadores dos vértices que precisam ser unificados e após todos terem sido identificados realiza a unificação propriamente dita.

É importante destacar que mesmo o processo de identificação de duplicidade sendo computacionalmente custoso, ele é composto por várias tarefas individualmente independentes e portanto altamente paralelizável. Essa característica foi aproveitada na implementação do método e será detalhada na próxima seção.

5.3 Construção do Gazetteer

A importação dos dados para o Linked OntoGazetteer foi executada fonte por fonte. A Figura 5.6 ilustra o método utilizado para a importação dos dados. A cada vez que uma nova fonte é importada o algoritmo de identificação de duplicidade é executado, as duplicidades resolvidas e os resultados analisados. Essa análise tem como objetivo não só verificar o volume de entidades duplicadas encontradas, mas também entender como as fontes de dados estão se complementando quanto à cobertura e nível de detalhamento. A ordem em que as bases de dados foram importadas foi: DBPedia, GeoNames, Freebase e LinkedGeoData. Essa ordem não é algo que influencie o resultado final do trabalho.

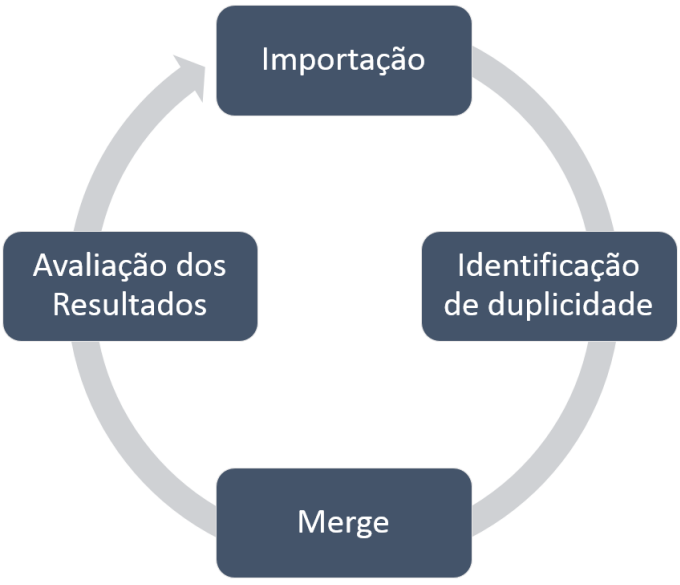


Figura 5.6: Metodologia utilizada para importação dos dados no Linked OntoGazetteer

Todo o processo de identificação e resolução de duplicidade foi executado em um

cluster com 8 máquinas virtuais no Microsoft Azure[4], uma plataforma para desenvolvimento nas nuvens. A topologia do cluster segue uma arquitetura cliente-servidor na qual o servidor foi configurado como uma instância A3 do Microsoft Azure, com 4 núcleos de processamento, 7GB de memória RAM e 2TB de armazenamento (divididos em dois HDs de 1TB cada). Os clientes são instâncias A1 com um único núcleo de processamento, 1,75GB de memória RAM e 30GB de disco. O servidor é responsável por prover acesso ao Titan e um serviço de troca de mensagens, que é usado para sincronizar trabalhos entre clientes. Os clientes são essencialmente executores de tarefas e não mantêm nenhuma informação, toda informação de entrada necessária e os resultados obtidos são transmitidos utilizando mensagens. Esse modelo de processamento é fortemente inspirado em processamento de fluxo de dados e os clientes podem ser organizados como cadeias de processamento.

5.3.1 Importação dos dados da DBPedia e do GeoNames

A primeira fonte de dados importada foi a DBPedia, mas como não existia nenhuma informação dentro do Linked OntoGazetteer previamente, nenhum processo de integração foi executado e passamos direto para a importação do GeoNames. Nessa rodada de importação foi executado um passo de validação do algoritmo de identificação de duplicidades, para garantir que a solução possui um bom nível de precisão. Para essa validação ser possível, consideramos como corretas as indicações de equivalência existentes entre DBPedia e GeoNames e retiramos essa informação do processo. Dessa forma, o algoritmo apenas montava a lista de candidatos e fazia validações dos 3 níveis propostos: países, contido em e distância entre os lugares.

Inicialmente foram encontrados 3.255.879 pares de lugares candidatos entre DBPedia e GeoNames. Os pares nessa lista não são únicos, pois dois lugares podem compartilhar mais de um mesmo nome. Dos 410.960 lugares em nosso conjunto de validação constituído pelos pares de lugar que possuem algum link de equivalência entre eles, apenas 10.892 não foram identificados no passo de validação. Desses lugares muitos estão localizados na Antártica, e por isso não estavam associados à países. Muitos outros estavam associados a “Coreia”, e a DBPedia parece possuir um aspecto curioso, no qual alguns lugares estão associados à Coreia unificada, sem indicar se hoje estão presentes na Coreia do Sul ou do Norte. Outras falhas estavam relacionadas a ilhas que não estão associadas a países e aos países propriamente ditos, como esperado, já que eles falham no teste de comparação de país.

[4] `http://azure.microsoft.com/`

Quando todas as evidências foram utilizadas o método conseguiu identificar e resolver 426.317 lugares duplicados. Desse resultado, 97% já estavam em nosso conjunto de validação e apenas 15.357 foram resultado apenas do nosso critério. Esse resultado preliminar indica que resolver as duplicidades restantes vai requerer mais esforço e métodos mais sofisticados. De toda forma, o resultado é satisfatório, o que viabiliza a utilização de nosso algoritmo para as demais bases importadas na sequência.

Após a execução da identificação e resolução das duplicidades entre GeoNames e DBPedia é importante avaliar como os dados deles se sobrepõem. O total de lugares unificados representa pouco mais que 5% dos registros existentes no GeoNames e cerca de 66,67% dos lugares da DBPedia. Com isso podemos inferir que a interseção entre elementos da DBPedia e GeoNames é relativamente pequena em termos da cobertura de lugares.

Como já foi caracterizado na Capítulo 4, o GeoNames possui uma cobertura mundialmente ampla e com nível de detalhamento baixo se comparado com a DBPedia, que possui muitos lugares intra-urbanos mas uma distribuição desigual no mundo.

Utilizando o Gremlin (linguagem para realizar navegações em grafos), fizemos uma análise sobre os lugares que não tiveram associação identificada entre DBPedia e GeoNames. Foram analisados todos os predicados que indicam o tipo das entidades originadas da DBPedia e que foram identificadas como lugares. O objetivo dessa análise é identificar quais foram os tipos com pior índice de duplicidades resolvidas pelo processo de integração. 71 tipos de lugares foram identificados com ao menos uma entidade da DBPedia não associada a uma outra no GeoNames. A Tabela 5.2 mostra as 10 categorias com maior número de falhas no processo de integração. Nessa avaliação, não foram considerados tipos que possuem menos de 100 entradas na DBPedia. É interessante destacar que 48 dos 71 tipos de lugares representam categorias de lugares intra-urbanos, ausentes no GeoNames. Em contrapartida, 86,3% das cidades existentes na DBPedia foram integradas com a sua correspondente no GeoNames. Além disso, a DBPedia, seguindo o conceito da Wikipédia, mantém registros de lugares históricos, como Prússia, Iugoslávia e Pérsia. Como esses lugares não são mais atuais, eles não existem no GeoNames, logo é impossível identificar um lugar correspondente.

Apenas 5% dos lugares do GeoNames foram associados com um par na DBPedia. Analisando os atributos *feature class* e *feature code*, foram identificados 506 *feature codes*, com exemplares de todas as 9 *feature classes*, com ao menos um lugar, cuja integração ocorreu de maneira bem sucedida.

Avaliando as *feature classes* envolvidas na integração, pode-se observar que o valor que tem maior índice de sucesso no processo de integração é a classe A (administrativa), descrita pelo GeoNames como "*country, state, region,...*", e em segundo lugar

Tipo de Lugar	F	T	F/T
http://dbpedia.org/ontology/PopulatedPlace	145.856	427.068	34,15%
http://dbpedia.org/ontology/Settlement	126.684	395.655	32,02%
http://dbpedia.org/ontology/ArchitecturalStructure	110.956	132.479	83,75%
http://dbpedia.org/ontology/Building	62.192	67.287	92,43%
http://dbpedia.org/ontology/Village	58.293	119.860	48,63%
http://dbpedia.org/ontology/Infrastructure	47.474	63.436	74,84%
http://dbpedia.org/ontology/NaturalPlace	31.422	50.719	61,95%
http://dbpedia.org/ontology/RouteOfTransportation	24.716	25.369	97,43%
http://dbpedia.org/ontology/BodyOfWater	23.905	35.072	68,16%
http://dbpedia.org/ontology/Stream	20.233	25.262	80,09%

Tabela 5.2: 10 tipos de lugar da DBPedia com maior número de falhas no processo de integração. $\#F$ - Número total de falhas, $\#T$ - Número total de lugares

a classe P (lugares povoados), definidos como "*city, village,...*". No nível do *feature code*, a taxa média de integração não realizadas, pois o algoritmo não identificou os lugares como o mesmo, é $91,32\%$, com um desvio padrão de $15,08\%$. Esse é um resultado esperado, pois o GeoNames contém mais de 10 vezes o número de lugares que a DBPedia. Entretanto, *feature codes* das classes A e P foram integrados com uma maior taxa de sucesso. A Tabela 5.3 mostra os 10 *feature codes* com maior índice de sucesso no processo de integração.

Feature Code	**F. Class**	F	T	F/T
Independent political entity	A	4	193	2,07%
First-order administrative division	A	107	3.900	2,74%
Dependent political entity	A	3	35	8,57%
Capital of a political entity	P	28	241	11,62%
Seat of a first-order administrative division	P	440	3.478	12,65%
Historical political entity	A	2	11	18,18%
Parish	A	27	132	20,45%
Seat of a second-order administrative division	P	3.617	12.562	28,79%
Nunataks	T	101	305	33,11%
Section of plain	T	3	9	33,33%

Tabela 5.3: 10 *feature codes* do GeoNames com menor índice de falha no processo de integração. $\#F$ - Número total de falhas, $\#T$ - Número total de lugares

A classe T (elementos topográficos) é definida pelo GeoNames como "*mountain, hill, rock, ...*". Como já foi detalhado anteriormente, o GeoNames possui muita informação sobre rios, lagos e montanhas. *Feature codes* e *feature classes* que descrevem esse tipo de lugar tiveram as maiores taxas de falhas no processo de integração, sugerindo que existe uma falha na cobertura da DBPedia sobre esse tipo de lugar, ou

elementos dessa categoria não estão corretamente classificados como lugares dentro da ontologia da DBPedia. A Tabela 5.4 mostra os 10 *feature codes* com maior índice de falha no processo de integração. Paradoxalmente, o *feature code* que representa lugares povoados, que é parte da classe P, é classificado como a pior taxa de sucesso, indicando que um número muito grande de lugares conhecidos no GeoNames ainda não possuem um correspondente na DBPedia.

Feature Code	**Feature Class**	F	T	F/T
Populated place	P	2.716.999	2.962.818	91.70%
Stream	H	762.478	776.892	98.14%
Mountain	T	312.902	326.757	95.76%
School	S	249.911	263.751	94.75%
Lake	H	244.978	251.184	97.53%
Church	S	238.611	239.810	99.50%
Hill	T	176.368	181.225	97.32%
Farm	S	173.507	174.358	99.51%
Cemetery	S	136.529	136.956	99.69%
Populated locality	P	129.335	132.602	97,54%

Tabela 5.4: 10 *feature codes* do GeoNames com maior índice de falha no processo de integração. $\#F$ - Número total de falhas, $\#T$ - Número total de lugares

5.3.2 Importação dos dados da Freebase

Depois de executar o ciclo de importação do GeoNames deu-se início ao ciclo da integração dos dados da Freebase. Nesse caso, dos 1.642.081 lugares identificados na base de conhecimento, um total de 924.853 $(56,32\%)$ não tiveram sucesso no processo de integração e portanto foram incorporados como lugares novos. Os demais obtiveram sucesso na integração e foram categorizados em algum dos seguintes cenários de duplicidade de representação: (1) duplicidade apenas com a DBPedia, (2) duplicidade apenas com GeoNames e (3) duplicidade com ambas as fontes previamente importadas.

Como já era de se esperar (já que a Wikipédia exerce forte influência sobre as duas outras fontes), a Freebase possui maior percentual de sobreposição apenas com a DBPedia (caso 1): foram 337.498 integrações bem sucedidas, pouco mais que 20,5% do total de lugares. Esse valor poderia ser maior se outras 376.787 entidades contidas na Freebase e que possuem um link para a Wikipédia possuíssem também uma entrada na DBPedia. Para o GeoNames, apenas 61 duplicidades foram resolvidas de maneira exclusiva (2), o que demonstra que a Freebase não utiliza o GeoNames como uma fonte relevante de dados geográficos. Levando em consideração as duas fontes previamente

importadas, foram identificadas 2.882 duplicidades, sendo que em 4 delas os registros da DBPedia e GeoNames não existiam no Linked OntoGazetteer. Sobre esse último cenário, foram registrados 268 casos onde existiam registros para a DBPedia e para o GeoNames, mas o passo de identificação e resolução da duplicidade executado anteriormente não foi capaz de integrar as representações, e o registro da Freebase funcionou como uma "ponte", integrando as representações existentes. A Figura 5.7 ilustra esse fenômeno.

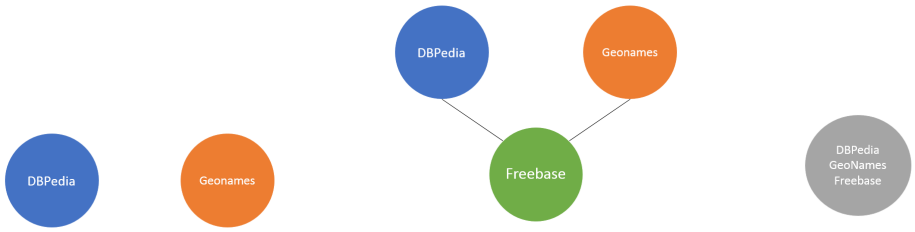


Figura 5.7: Registro da Freebase funciona como ponte (b), integrando duas entidades previamente distintas (a) em uma única representação (c)

Mesmo sendo interessante esse fenômeno de "ponte"que existiu na importação dos dados da Freebase, em uma inspeção manual foram encontradas algumas exceções que levantam suspeitas sobre a qualidade dos *links* que indicam equivalência entre as bases de conhecimento. Um exemplo está ilustrado na Figura 5.8. Nela, o registro da DBPedia para Hong Kong[5] possui um *link* de equivalência para o GeoNames[6] e também para o Freebase[7]. Até esse ponto, não existe erro algum, porém o registro da Freebase que representa Hong Kong[7] está incorretamente relacionado com a entidade no GeoNames que representa a Ilha Hong Kong[8]. Novamente, a DBPedia possui os *links* corretos tanto para o GeoNames quanto para a Freebase, mas as duas últimas nesse caso não possuem indicação de equivalência.

O Freebase possui uma hierarquia de tipos muito longa e específica, o que dificulta a realização de uma análise semelhante à realizada entre DBPedia e GeoNames e os tipos de lugares que tiveram mais sucesso no processo de integração. Além disso, o número de conexões feitas exclusivamente com o GeoNames não é grande o suficiente para ser considerado estatisticamente relevante para tirar-se conclusões, mesmo assim é interessante ver que 92% dessas poucas integrações são com cidades[9]. A Tabela

[5] `http://dbpedia.org/page/Hong_Kong`
[6] `http://www.geonames.org/1819729`
[7] `http://www.freebase.com/m/03h64`
[8] `http://www.geonames.org/1819727`
[9] `http://rdf.freebase.com/ns/location.citytown`

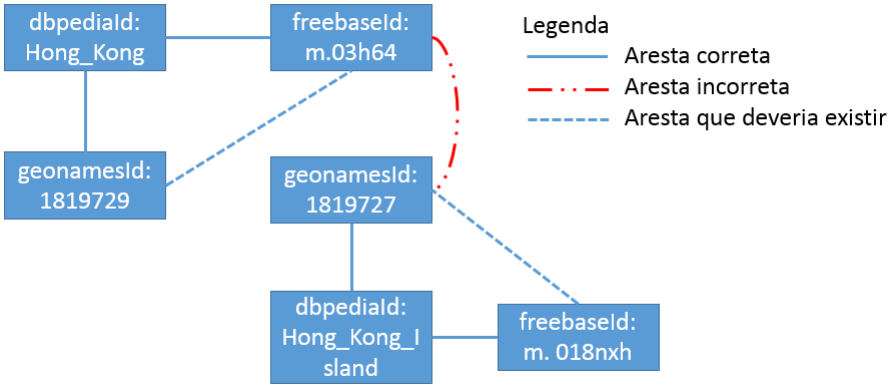


Figura 5.8: Exemplo de exceção ocorrida no fenômeno retratado na Figura 5.7

5.5 mostra alguns tipos da Freebase selecionados manualmente para ilustrar como os lugares que foram integrados com a DBPedia estão categorizados dentro da base de conhecimento. A maior parte dos lugares está categorizada como cidade, sendo que tipos que representam lugares intra-urbanos ou elementos geográficos também possuem uma quantidade de registros significante. As categorias listadas na Tabela 5.5 não são únicas, ou seja, um mesmo lugar pode estar associado a múltiplos tipos.

Tipo	**% Entidades**	**# Entidades**
fbase:ns:location.statistical_region	40,22%	135.730
fbase:ns:location.citytown	35,23%	118.895
fbase:ns:architecture.structure	10,02%	33.809
fbase:ns:geography.geographical_feature	9,34%	31.530
fbase:ns:architecture.building	6,13%	20.692
fbase:ns:location.administrative_division	5,63%	19.005
fbase:ns:geography.body_of_water	4,95%	16.721
fbase:ns:base.aareas.schema.administrative_area	4,14%	13.981
fbase:ns:metropolitan_transit.transit_stop	3,42%	11.546
fbase:ns:geography.river	3,13%	10.564
fbase:ns:transportation.road	2,27%	7.666
fbase:ns:aviation.airport	2,17%	7.332
fbase:ns:geography.mountain	1,89%	6.392
fbase:ns:travel.tourist_attraction	1,88%	6.357
fbase:ns:geography.lake	1,39%	4.679
fbase:ns:architecture.venue	1,26%	4.247
fbase:ns:geography.island	1,21%	4.100

Tabela 5.5: Tipos de lugares da Freebase e quantidade de registros integrados exclusivamente com a DBPedia

Lugares classificados como *fbase:ns:location.statistical_region*[10] frequentemente estão relacionados a lugares ou regiões que possuem informações estatísticas, como as resultantes de um Censo, associadas. Isso faz com que essa categoria frequentemente esteja associada a lugares também classificados como cidades, o que justifica o fato desses dois tipos possuírem números de ocorrência muito semelhantes. Todos os lugares que foram identificados como coexistentes nas 3 bases de dados estão classificados como *fbase:ns:location.statistical_region*, sendo que 96% são cidades.

5.3.3 Importação dos dados do LinkedGeoData

As cópias disponíveis para download do LinkedGeoData não apresentam as informações que representam relacionamentos internos na estrutura de *links*, como mostrado na Figura 4.8. Para lidar com esse problema utilizamos os dados previamente importados para transformar o literal que representa a hierarquia topológica em arestas para vértices já existentes no Linked OntoGazetteer. Esse literal é composto por uma sequência de topônimos separados por vírgula $(t_0, t_1, ..., t_n)$, e implicitamente ele expressa a relação de *estar contido em* entre os topônimos adjacentes, ou seja, t_0 *está contido em* t_1, e assim sucessivamente. A Figura 5.9 ilustra como essa estrutura é mantida. Partindo dessa premissa, foi elaborada uma consulta Gremlin ilustrada na Figura 5.10, que busca refletir esse relacionamento para navegar no grafo e identificar quais vértices os nomes de lugares estão representando.

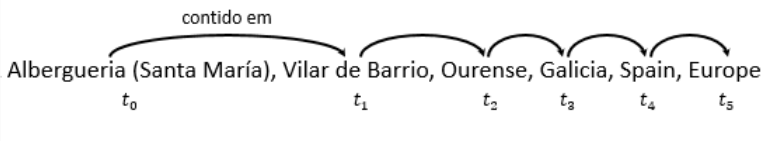


Figura 5.9: Exemplo da estrutura dos literais dos predicados *isIn* do Linked GeoData

```
g.V('name', 'Galicia').out('isName').as('from')
.out('containedBy').as('to').in('isName').has('name', 'Spain')
.select(['from','to']).dedup()
```

Figura 5.10: Consulta Gremlin que navega no grafo e identifica os vértices que representam Galícia e Espanha, sendo que o estado está contido no país.

Apenas os predicados exatamente iguais a `http://linkedgeodata.org/ontology/isIn` foram selecionados para serem tratados. Um total de 718.981 entidades possuem tal predicado. Desse total existiam apenas 44.156 literais únicos, que

[10] `http://rdf.freebase.com/ns/location.statistical_region`

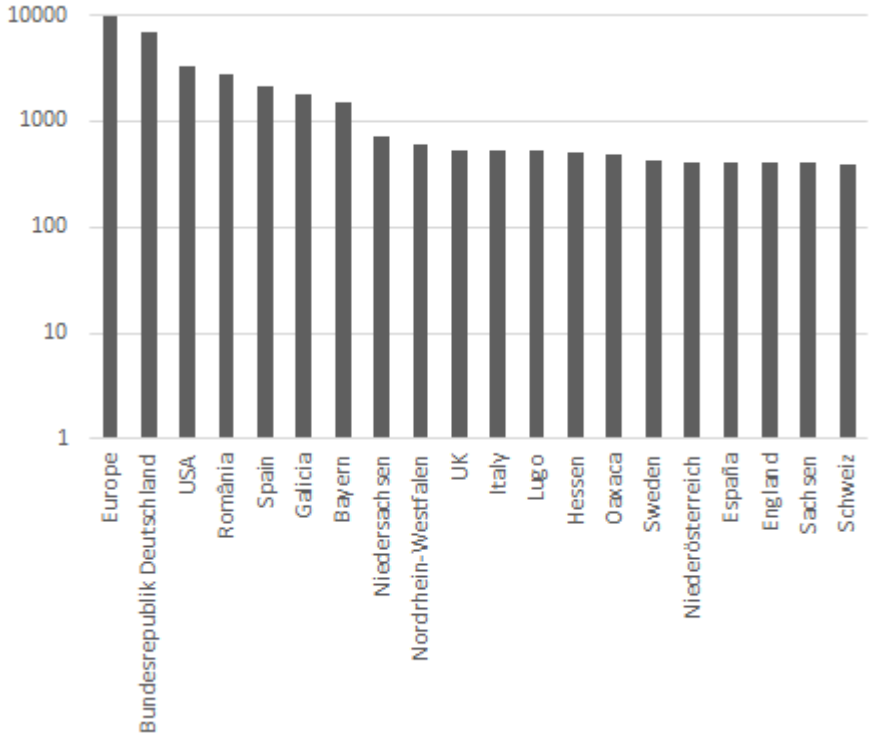


Figura 5.11: 20 topônimos com maior número de associações a vértices bem sucedidas

são compostos por 155.244 nomes de lugares. Desses topônimos, 70.695 foram associados a um único vértice no Linked OntoGazetteer, o restante não foi bem sucedido, pois ou ficaram ambíguos ou a estrutura de *contido em* não foi localizada no *gazetteer*. As Figuras 5.11 e 5.12 mostram os 20 topônimos que obtiveram maior número de associações bem sucedidas e com falha respectivamente.

Muitas das falhas se deram pelo fato da estrutura de *estar contido em* não ter sido encontrada e os nomes de lugares individualmente serem ambíguos, por exemplo: Europe[11,12]. Entretanto, a maior parte das falhas foi motivada pela utilização de nomes alternativos pouco usuais, abreviações, erros de ortografia e vários outros problemas comuns com dados oriundos de contribuição voluntária.

Durante a etapa de importação dos dados, esses novos relacionamentos entre entidades do LinkedGeoData e lugares do Linked OntoGazetteer ainda não tinham sido descobertos, o que impactou diretamente no número de duplicidades identificadas e resolvidas. Dos 3.600.880 de lugares divididos entre *nodes* e *ways* do LinkedGeoData apenas 17.755 foram integrados a vértices já existentes no *gazetteer*. O motivo desse valor pequeno é o fato de ser exclusivamente dependente dos *links* que indicam equi-

[11] `http://www.geonames.org/6944144` — Hotel na Itália
[12] `http://www.geonames.org/6255148` — Continente

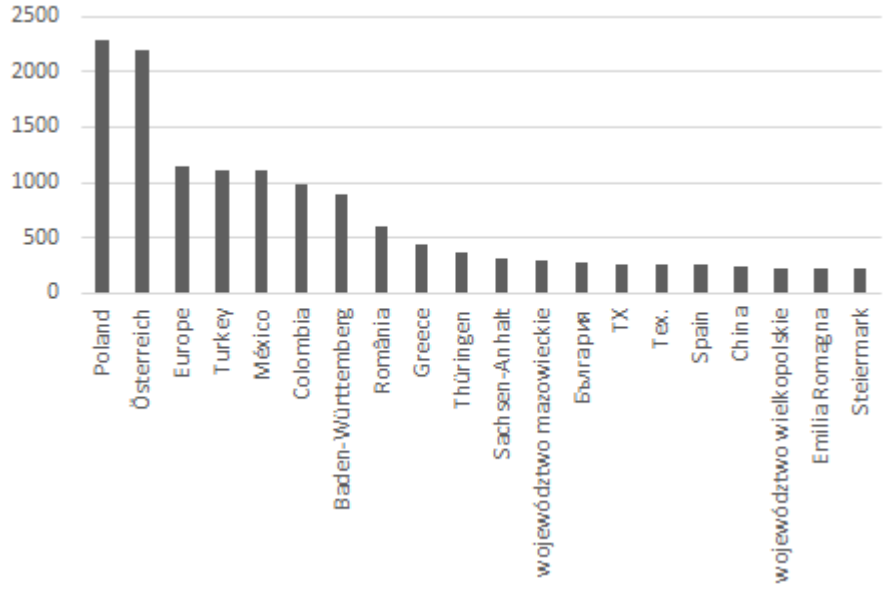


Figura 5.12: 20 topônimos com maior número de associações a vértices mal sucedidas

valência entre as fontes de informação. Além disso esses *links* estão em formatos muito pouco padronizados, o que dificulta o trabalho de reconhecimento da URI.

5.4 Arquitetura Final

Após todos os dados importados, o grafo resultante possui mais que 34 milhões de vértices. Desses vértices 17.371.811 são instâncias da classe `Name`. Os vértices que representam lugares são divididos em 1.064.518 originados da DBPedia, 8.513.978 do GeoNames, 1.063.974 da Freebase e 3.586.661. Esses valores incluem lugares que foram identificados como coexistente em mais de uma fonte de dados. Também foram incorporadas pouco mais que 4 milhões de entidades que não são lugares ao *gazetteer*.

Para a implementação final do Linked OntoGazetteer a topologia do *cluster* utilizado no Azure foi modificada. As máquinas clientes foram desligadas e a máquina que era o servidor teve sua capacidade de processamento escalada verticalmente, passando de uma instância do tipo A3 para uma instância do tipo A6 com 4 processadores e 28GB de memória RAM. O motivo dessa modificação foram os novos serviços implementados relacionados ao Linked OntoGazetteer e que serão providos aos usuários que precisem das informações do *gazetteer*.

Todo o desenvolvimento e criação do Linked OntoGazetteer foi feito utilizando ferramentas abertas e/ou disponíveis na comunidade. As linguagens de programação utilizadas foram Java (1.7) e Python (3.4.0). A primeira para a construção da aplicação

web e uma biblioteca para suporte a operações no Titan (0.4.4) e a segunda para criação de scripts para realizar o pré-processamento e limpeza dos dados obtidos. Também foi utilizado o Apache Maven (3.2.5), ferramenta que oferece suporte ao desenvolvimento na gestão de dependências e compilação das aplicações, e o Eclipse (4.4.1) como ambiente de desenvolvimento.

A Figura 5.13 mostra uma visão geral da arquitetura final do Linked OntoGazetteer e todos os componentes criados neste trabalho. Cada componente será melhor detalhado na sequência.

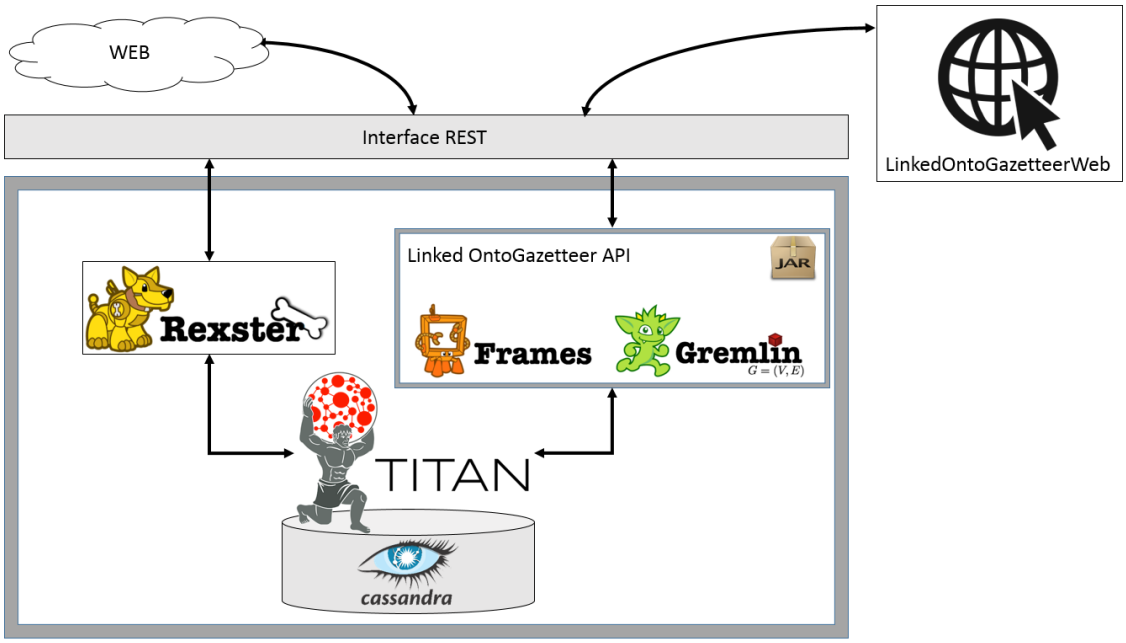


Figura 5.13: Visão geral da arquitetura e tecnologias utilizadas na construção do Linked OntoGazetteer

Como já foi apresentado anteriormente, o banco de dados escolhido foi o Titan instalado em uma única instância, utilizando o Apache Cassandra (2.0.2) como *backend* de armazenamento. O Titan provê suporte a armazenamento de dados geográficos simples, como pontos e retângulos, mas não possui nativamente mecanismos para executar consultas geográficas sobre índices. Por esse motivo um mecanismo de indexação externo foi utilizado. O escolhido foi o Elasticsearch, que provê suporte a consultas geográficas e consultas por aproximação de chaves.

O Titan é implementado seguindo uma API (Application Programming Interface) genérica para grafos chamada Blueprints. Isso viabiliza a utilização de diversas outras ferramentas, como por exemplo o Gremlin. Além da linguagem de consultas, outras duas ferramentas que também seguem os padrões definidos pelo Blueprints foram utilizadas neste trabalho: Rexster e Frames. O Rexster é um servidor que externaliza acesso

a grafos através de protocolos *web*, *Representational State Transfer* (REST) e um outro proprietário (RexPro), e também possui uma interface *web* administrativa muito útil para avalição de scripts Gremlin, cujo nome é *The Dog House*. O Frames, por sua vez, é um arcabouço Java que abstrai as representações em grafos (vértices e arestas) para objetos de domínio. Um exemplo de objeto de domínio em nossa solução são os Lugares. Ele foi muito utilizado na construção da biblioteca criada para manipulação dos dados do Linked OntoGazetteer, que é base do componente Linked OntoGazetteer API, um conjunto de operações de consulta disponíveis para utilização via interface REST, detalhado na próxima seção.

O conjunto de operações externalizadas pelo Rexster viabilizou a implementação do Linked OntoGazetteer Web[13], uma aplicação *web* que permite submeter consultas de topônimos sobre o *gazetteer* e visualizar as entidades que estão associadas a eles. A Figura 5.14 mostra parte da interface para visualização de detalhes de um resultado da busca. Na interface é possível visualizar todos os nomes associados ao lugar selecionado (no exemplo Belo Horizonte[14]), todas as informações das fontes de dados originais e a representação geográfica do lugar.

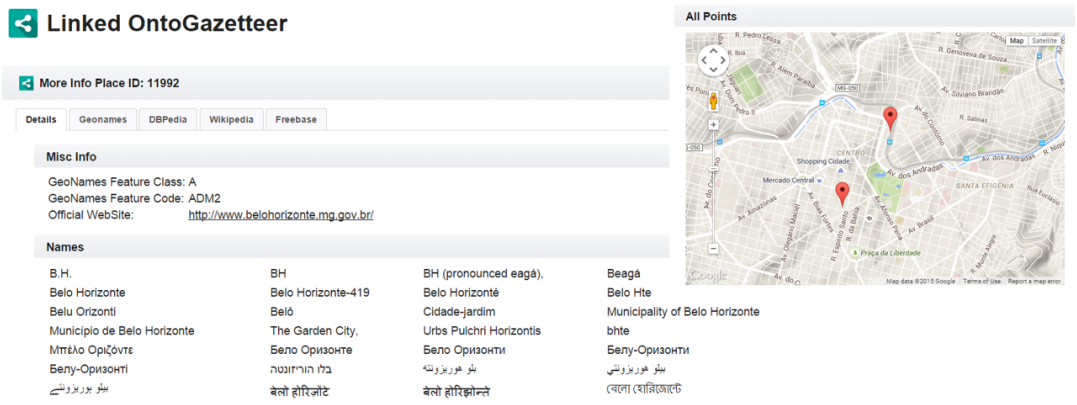


Figura 5.14: Exemplo da interface do Linked OntoGazetteer Web

5.5 Linked OntoGazetteer API

Gazetteers são ferramentas frequentemente utilizadas por diversas aplicações que trabalham com dados geográficos. Por isso uma API que suporte as operações comumente utilizadas é peça fundamental para utilização de um *gazetteer*.

[13] `http://www.aqui.io/log/`

[14] `http://sandwich.lbd.dcc.ufmg.br:8080/linkedOntoGazetteerWeb/vertex.xhtml?id=11992`

O Rexster traz grande flexibilidade para o acesso ao Titan, externalizando pontos para submissão de consultas Gremlin. Entretanto, a linguagem Gremlin não é tão popular quanto outras linguagens de consulta, como por exemplo SQL, e exigir que usuários que precisem obter informações do Linked OntoGazetteer dominem a sintaxe do Gremlin dificulta sua utilização em larga escala. Para isso definimos um conjunto de métodos que constituem a API do Linked OntoGazetteer e serão apresentados a seguir. Esse conjunto de métodos busca suprir necessidades recorrentes relacionadas a um *gazetteer*, principalmente em aplicações de RIG.

A API, assim como o Rexster, externaliza o acesso aos métodos utilizando uma interface REST composta apenas por métodos que suportam requisições *HTTP GET*. O formato das requisições e respostas é sempre JSON (JavaScript Object Notation). Dessa forma, fornecemos um meio totalmente independente de linguagem de programação para acesso aos dados do Linked OntoGazetteer. Todos os *endpoints* dos serviços devem ser prefixados pelo servidor onde a API está disponibilizada. Atualmente ela está implantada em: `http://sandwich.lbd.dcc.ufmg.br:8080/linkedOntoGazetteerWeb`.

- `retrievePlacesByName(String name)`
  - **Descrição**: Busca por todos os lugares que estão associados ao nome informado. Equivalente à consulta Gremlin:
    ```
    g.V('name', name).out('isName').as('x').in('type')
    .has('entityType', 'PLACE').select(['x'])
    ```
  - **Parâmetros**:
    `name` - Nome que os lugares devem estar associado
  - **Endpoint**: `/api/place/name/{name}`
  - **Retorno**: Lista de lugares com todos atributos existentes em vértices desse tipo.
- `retrieveAllEntitiesByName(String name)`
  - **Descrição**: Busca todas as entidades ambíguas independente da classe à qual ela pertença. Equivalente à consulta Gremlin:
    ```
    g.V('name', name).out('isName')
    ```
  - **Parâmetros**:
    `name` - Nome que deve estar associado as entidades
  - **Endpoint**: `/api/entity/name/{name}`

  - **Retorno**: Lista com o identificador de todas as entidades ambíguas.

- `isPlace(Long id)`
  - **Descrição**: Verifica se a entidade identificada por `id` é um lugar. Equivalente à consulta Gremlin:
    `g.v(id).in('type').has('entityType', 'PLACE').hasNext()`
  - **Parâmetros**:
    `id` - Identificador da entidade
  - **Endpoint**: `/api/isPlace/{id}`
  - **Retorno**: *True* se a entidade cujo identificar é `id` for um lugar. *False* caso contrário.

- `retrieveNamesByPlaceId(Long placeId)`
  - **Descrição**: Busca todos os nomes dado um lugar. Equivalente à consulta Gremlin:
    `g.v(placeId).in('isName')`
  - **Parâmetros**:
    `placeId` - Identificador único dos lugares. ID do vértice gerado pelo Titan.
  - **Endpoint**: `/api/name/place/{placeId}`
  - **Retorno**: Lista de nomes associados ao lugar identificado por `placeId`.

- `retrieveRelatedEntities(Long placeId)`
  - **Descrição**: Busca todas as entidades não classificadas como lugar que estão relacionadas ao lugar identificado por `placeId`. Equivalente à consulta Gremlin:
    `g.v(placeId).in('related')`
  - **Parâmetros**:
    `placeId` - Identificador único dos lugares. ID do vértice gerado pelo Titan.
  - **Endpoint**: `/api/entity/relatedPlace/{placeId}`
  - **Retorno**: Lista de entidades da classe `Non-Place` relacionadas ao lugar identificado por `placeId`.

- `retrieveRelatedPlacesByEntityName(String entityName)`

  - **Descrição**: Busca todos os lugares relacionados à entidade associada ao nome `entityName`. Equivalente à consulta Gremlin:
    ```
    g.V('name', entityName).out('isName').as('n')
    .out('related').as('p').in('type')
    .has('entityType', 'PLACE').select(['n','p']).dedup()
    ```
  - **Parâmetros**:

    `entityName` - Nome que deve estar associado a uma entidade.
  - **Endpoint**: `/api/place/entity/name/{name}`
  - **Retorno**: Lista de lugares relacionados às entidades associadas ao nome `entityName`. Os elementos da lista têm o formato $[n_{id}, p_{id}]$ onde n_{id} é o identificador da entidade e p_{id} do lugar que estão relacionados.

- `retrievePlacesInRectangle(Point a, Point b, String reference)`
  - **Descrição**: Busca todos os lugares que estão contidos dentro do retângulo definido pelos pontos a e b utilizando como referência a propriedade definida pelo terceiro parâmetro. Equivalente à consulta Gremlin:
    ```
    g.query().has(reference,WITHIN,
    Geoshape.box(a.x, a.y, b.x, b.y)).vertices())
    ```
  - **Parâmetros**:

    `a` - Ponto superior esquerdo do rentângulo.

    `b` - Ponto inferior direito do retângulo.

    `reference` - Nome da propriedade que será utilizada como referência na busca. Esse parâmetro é necessário, pois os lugares possuem até quatro representações geográficas, uma para cada fonte de dados importada.
  - **Endpoint**: `/api/place/inRectangle/{reference}`
  - **Retorno**: Lista de lugares com todos os atributos existentes em vértices desse tipo.

- `retrievePath(Long fromPlaceId, Long toPlaceId, int maxSize)`
  - **Descrição**: Busca caminhos no grafo saindo do lugar identificado por `fromPlaceId` até `toPlaceId` utilizando arestas do tipo *containedBy*. Se o caminho máximo não for informado será utilizado o padrão de 5, sendo que quanto maior o caminho mais tempo será gasto no cálculo. Equivalente à consulta Gremlin:

```
g.v(fromPlaceId).out('containedBy')
.loop(1){it.object.id != toPlaceId && it.loops < maxSize}.path
```

  - **Parâmetros**:

    `fromPlaceId` - Identificador único do lugar que é o início do caminho.

    `toPlaceId` - Identificador único do lugar que é o fim do caminho.

    `maxSize` - Tamanho máximo do caminho. Se não informado, o valor padrão é 5.

  - **Endpoint**: `/api/place/path/{fromPlaceId}/{toPlaceId}`

  - **Retorno**: Lista com os identificadores dos vértices que forma o caminho. São retornados todos os caminhos possíveis encontrados e que respeitem o tamanho informado.

- `isContainedBy(String pNameA, String pNameB)`

  - **Descrição**: Verifica se existe um lugar associado ao topônimo `pNameA` que está contido por outro lugar associado ao topônimo `pNameB`. Equivalente à consulta Gremlin:
    ```
    g.V('name', pNameA).out('isName').as('from')
    .out('containedBy').as('to').in('isName').
    has('name', pNameB).select(['from','to']).dedup()
    ```

  - **Parâmetros**:

    `pNameA` - Nome de um lugar.

    `pNameB` - Nome de um lugar.

  - **Endpoint**: `/api/place/name/containedBy/{pNameA}/{pNameB}`

  - **Retorno**: Par com os identificadores dos lugares associados ao topônimos que foram encontrados com a relação contido em. Retorna todos os pares de lugar, caso os topônimos sejam ambíguos.

- `isContainedBy(Long placeIdA, Long placeIdB)`

  - **Descrição**: Funciona da mesma forma que o `isContainedBy` anterior, mas utilizando os identificadores dos lugares como ponto de partida. Por isso é bem mais eficiente. Equivalente à consulta Gremlin:
    ```
    g.v(placeIdA).out('containedBy').filter{it.id ==
    placeIdB}.hasNext()
    ```

  - **Parâmetros**:

    `placeIdA` - Identificador de um lugar.

    `placeIdB` - Identificador de um lugar.

  - **Endpoint**: `/api/place/id/containedBy/{placeIdA}/{placeIdB}`

  - **Retorno**: *True* caso o lugar identificado por `placeIdB` contenha o luar identificado por `placeIdA`. *False* caso contrário.

- `retrievePlaceAdjacentListByName(String name)`

  - **Descrição**:Busca todos os lugares que estão associados ao nome passado como parâmetro e os lugares pertencentes à lista de adjacência dele. Equivalente à consulta Gremlin:
    ```
    g.V('name', name).out('isName').
    groupBy{it}{it.both('containedBy')}
    {it.findAll{it.in('type').has('entityType', 'PLACE')}}.cap
    ```

  - **Parâmetros**:

    `name` - Nome associado aos lugares.

  - **Endpoint**: `/api/place/name/adjacentList/{name}`

  - **Retorno**: Lista de adjacência dos lugares que estão associados ao nome passado como parâmetro.

- `retrievePlaceAdjacentList(Long placeId)`

  - **Descrição**:Busca todos os lugares que são vizinhos do lugar identificado por `placeId`. Esse método é bastante caro, pois existe uma verificação se os membros da lista de adjacência são lugares. Equivalente à consulta Gremlin:
    ```
    g.v(placeId).both.filterit.in('type')
    .has('entityType', 'PLACE').hasNext().gather
    ```

  - **Parâmetros**:

    `placeId` - Identificador do lugar.

  - **Endpoint**: `/api/place/id/adjacentList/{placeId}`

  - **Retorno**: Lista de adjacência dos lugares que estão associados ao nome passado como parâmetro.

5.5.1 Exemplo de Utilização

Fazer uso da API implementada é bastante simples, pois as linguagens de programação mais modernas e populares, como Python e Java, possuem bibliotecas que reduzem grande parte do esforço para consumir serviços REST. Nesta seção, iremos demonstrar como é simples construir um cliente REST em Python que seja capaz de consumir os recursos disponibilizados pela API.

Para implementar os exemplos que serão mostrados a seguir foi utilizada a biblioteca Python `Requests`[15]. O primeiro exemplo da Figura 5.15, mostra como é simples fazer um request e imprimir o resultado em texto plano, como mostrado na Figura 5.16.

```
import requests, json
baseUrl = "http://mgeo00.cloudapp.net:8880/linkedOntoGazetteerWeb/"
apiService = "api/place/name/"
placeName = "BH"

url = baseUrl + apiService + placeName

r = requests.get(url)

#Print response body text
print r.text
```

Figura 5.15: Exemplo de implementação de um cliente da Linked OntoGazetteer API

```
$ python sampleAPI1.py
{"version":"2.4.0","queryTime":40.7014,"results":[{"name":null,"_id":12
7898068,"_type":"vertex","dbpediaId":"BH_Radio_1","geonamesId":null,"fr
eebaseId":null,"dbpPoint":null,"gnPoint":null,"frbPoint":null,"gnFeatur
eClass":null,"gnFeatureCode":null,"official_website":null,"nonplace":tr
ue},{"name":null,"_id":125634200,"_type":"vertex","dbpediaId":"Berita_H
arian","geonamesId":null,"freebaseId":null,"dbpPoint":null,"gnPoint":nu
ll,"frbPoint":null,"gnFeatureClass":null,"gnFeatureCode":null,"official
_website":null,"nonplace":true},{"name":null,"_id":35696232,"_type":"ve
rtex","dbpediaId":null,"geonamesId":"3470127","freebaseId":null,"dbpPoi
nt":null,"gnPoint":"point[-19.92083,-43.93778]","frbPoint":null,"gnFeat
ureClass":"P","gnFeatureCode":"PPLA","official_website":null,"nonplace"
:null},{"name":null,"_id":126914852,"_type":"vertex","dbpediaId":"BH_Gl
obal","geonamesId":null,"freebaseId":null,"dbpPoint":null,"gnPoint":nul
```

Figura 5.16: Parte da saída do exemplo da Figura 5.15

O segundo exemplo, ilustrado na Figura 5.17, mostra um caso de uso, que além de realizar a requisição, também manipula a resposta. O resultado desse código pode

[15] `http://docs.python-requests.org/`

ser visto na Figura 5.18 e exibe os *ids* das fontes de dados DBPedia, GeoNames e Freebase.

```
import requests, json
baseUrl = "http://mgeo00.cloudapp.net:8880/linkedOntoGazetteerWeb/"

apiService = "api/place/name/"
placeName = "BH"

url = baseUrl + apiService + placeName
r = requests.get(url)

#Print the request status --- 200 is OK
print "REQUEST STATUS:" + str(r.status_code)

results = r.json()['results']

for result in results:
        print "DBPedia ID: " + str(result['dbpediaId']) +
        "\t\tGeoNames ID: " + str(result['geonamesId']) +
        "\t\tFreebase ID: " + str(result['freebaseId'])
```

Figura 5.17: Exemplo de implementação de um cliente da Linked OntoGazetteer API que manipula a resposta do *WebService*

```
$ python sampleAPI2.py
REQUEST STATUS:200
DBPedia ID: BH_Radio_1          GeoNames ID: None               Freebase ID: None
DBPedia ID: Berita_Harian               GeoNames ID: None               Freebase ID: None
DBPedia ID: None                GeoNames ID: 3470127            Freebase ID: None
DBPedia ID: BH_Global           GeoNames ID: None               Freebase ID: None
DBPedia ID: Belo_Horizonte              GeoNames ID: 6321162            Freebase ID: m.013q2
DBPedia ID: BH_Telecom          GeoNames ID: None               Freebase ID: None
DBPedia ID: BH_Air              GeoNames ID: None               Freebase ID: None
```

Figura 5.18: Saída do exemplo da Figura 5.17

Capítulo 6

Conclusões e Trabalhos Futuros

As principais contribuições deste trabalho foram a construção de um *gazetteer* capaz de manter relacionamentos semanticamente ricos entre lugares e diversas outras entidades, que podem ou não ser lugares, um método para integração de fontes de dados integrantes do LOD e uma API que externaliza acesso aos dados do Linked OntoGazetteer através de uma interface REST.

Ao fim do processo de implementação o *gazetteer* foi populado com 13.074.366 lugares, que certamente ainda guarda registros duplicados. Mais de 140 milhões de atributos e relacionamentos foram criados em associação a lugares. Coexistindo com os lugares no banco de dados existem 4.477.739 entidades não classificadas como lugares e aproximadamente 6 milhões de arestas relacionando tais entidades com lugares. Juntamente com o *gazetteer* foi criada uma API disponibilizada por uma interface *web* com 12 pontos de entrada, que busca suportar aplicações que precisem acessar as informações e relacionamentos mantidos em nossa base de conhecimento.

Todos os dados de referência utilizados para popular o Linked OntoGazetteer foram obtidos de fontes online que publicam os dados de acordo com *Linked Data*. Um dos resultados deste trabalho foi uma análise sobre quanto as fontes de informação utilizadas (GeoNames, DBPedia, Freebase e LinkedGeoData) se sobrepõem e qual tipo de informação é mais redundante. O GeoNames é a fonte que mais provê informações sobre lugares, mas aproximadamente 95% das entidades que existem nele não possuem pares nas demais fontes abordadas. A maior fonte de lugares coexistentes nas bases de dados utilizadas são cidades e divisões administrativas, mesmo em fontes com razoável quantidade de informações intra-urbanas. Os resultados dessa análise mostram que as fontes de informação na *Web of Data* ainda sofrem muito para se integrar, e mesmo não usando técnicas sofisticadas para realizar identificação e fusão de entidades foi alcançado um resultado satisfatório. Certamente, melhorar a técnica utilizada para fazer

o casamento de registros duplicados é um trabalho futuro importante para melhorar a qualidade dos dados existentes no *gazetteer*, eliminando ambiguidade desnecessária.

No processo de limpeza e preparação dos dados ficou evidente o impacto de contribuições voluntárias na qualidade dos dados. O GeoNames, por possuir um ambiente fechado a contribuições, possui um esquema melhor definido e dados com uma qualidade muito boa. Isso não quer dizer que seja perfeito. Ele guarda registros duplicados (geralmente de lugares pouco populares), relacionamentos equivocados (no GeoNames Cuba é o país onde a América se encontra) e informações esparsas no nível urbano. A sequência das bases trabalhadas adiciona cada vez mais o fator "contribuição voluntária". A DBPedia possui alguns problemas, principalmente no que diz respeito à variabilidade de predicados e o real significado deles. Na Freebase o problema da variabilidade de predicados é ainda maior. Com o objetivo de possuir predicados semanticamente muito ricos, o Freebase opta por predicados específicos demais e de difícil diferenciação. Os dados do LinkedGeoData são originalmente do OpenStreetMap, uma das maiores iniciativas de contribuição voluntária geográfica do mundo, e apresentam sérios problemas, tanto na obediência dos padrões do Linked Data, como na qualidade dos predicados e dos dados propriamente ditos. Isso não quer dizer que iniciativas de contribuição voluntária estejam evoluindo na direção oposta ao Linked Data, mas indica que precisam ser melhor elaboradas, para darem liberdade ao usuário que quer contribuir e ao mesmo tempo fornecer suas informações de maneira minimamente estruturada.

A representação geográfica utilizada pelas fontes disponíveis na *Web of Data* também é muito simples, até mesmo para o GeoNames e LinkedGeoData, que são fontes especializadas em dados geográficos, já que a grande parte dos lugares só possui um par de coordenadas associado. Também existe a restrição por parte do Titan quanto ao armazenamento e indexação de dados geográficos, e para se adequar totalmente ao esquema proposto no OntoGazetteer original uma possível abordagem seria mesclar a utilização do Titan com outros bancos de dados que suportam de maneira nativa dados geoespaciais, como PostgreSQL e sua extensão PostGIS. Assim, pode-se conceber um esquema físico híbrido, em que o banco de dados em grafo seja utilizado para navegar pelos nomes, lugares e relacionamentos, e um banco de dados objeto-relacional mantenha representações espaciais mais complexas (linha, polígono, imagens geolocalizadas) de parte dos lugares presentes no gazetteer.

Uma possibilidade para aumentar a quantidade de informações intra-urbanas dentro dos *gazetteer* pode ser a utilização de dados publicados por prefeituras, como informações de rotas de transporte público, atrações turísticas e instalações públicas (escolas, hospitais e delegacias). Essa é ainda uma iniciativa recente, e particular-

mente no Brasil, bastante embrionária. As bases disponíveis ainda possuem poucas referências geográficas diretas, os dados não estão disponíveis de forma padronizada e a adoção, mesmo que crescente, ainda se limita a cidades de países desenvolvidos, basicamente. Seria interessante que dados geográficos urbanos viessem a ser publicados, em larga escala, por cada município ou região metropolitana, utilizando o paradigma de infraestruturas de dados espaciais (IDE). Mesmo assim, um estudo mais aprofundado sobre esses dados e como incorporá-los a um *gazetteer* pode ser um trabalho que potencialmente traria bastante crescimento no volume de dados intra-urbanos.

Uma grande dificuldade encontrada na utilização dos dados da *Web of Data* foi a inexistência de padronização nos formatos, formas de acesso e ontologias. Todos os casamentos de esquema foram feitos na base de inspeções manuais e muitas vezes com ajuda de *scripts* que eliminavam elementos irrelevantes devido à pequena popularidade. Muitas dessas dificuldades são recorrentes em aplicações de recuperação de informação e jamais deixarão de existir. A falta de padronização, principalmente das ontologias, coloca em cheque a adoção do Linked Data e sua utilização como foi concebida. Primordialmente o intuito era a reutilização de dados, mas o cenário atual caminha para cada vez mais replicação e dados disponíveis sem a mínima relação com outras fontes. Isso não reduz a importância desse padrão na atualidade, no que diz respeito à popularização e expansão da *web* semântica. Um trabalho futuro interessante seria a criação de agentes que percorressem a *Web of Data* validando as fontes de dados e seus relacionamentos, com o intuito de qualificar e apontar melhorias. Algo parecido começa a ser feito pelo grupo que suporta o LOD, mas suas análises são normalmente amplas e pouco objetivas. Técnicas de agrupamento e classificação certamente reduziriam o impacto da variabilidade de predicados e ajudariam na identificação de dados mal estruturados.

Todo o trabalho foi feito utilizando cópias de dados e processamento off-line. Sem dúvidas, um importante passo na evolução deste trabalho é avaliar se é viável tornar o processo de importação do Linked OntoGazetteer online, ou seja, a incorporação de novos elementos e atualizações acontecendo de maneira simultânea à utilização da ferramenta.

Com o surgimento de aplicações que utilizem o Linked OntoGazetteer como fonte de evidências para resolução de problemas de RIG, será possível avaliar os impactos de armazenar tal volume de informação e os problemas que isso possa trazer em relação a qualidade e eficiência dos métodos implementados. Também será possível evoluir a API, pois com o *feedback* dos usuários será viável avaliar o catálogo de serviços disponíveis. Uma forma de obter esse *feedback* dos usuários pode ser a implementação de registros de acesso dos usuários. Durante a execução do trabalho existiu uma preocupação quanto

à performance da solução, dado o volume de informações trabalhadas, mas esse não foi o foco principal e uma avaliação do desempenho dos métodos expostos na API seria um interessante trabalho futuro.

Durante a construção deste trabalho duas etapas resultaram em publicações com contribuições para a comunidade científica. O estudo e caracterização da utilização de Linked Data no contexto geoespacial foi publicado no XIV Simpósio Brasileiro de GeoInformática (GeoInfo) de 2013, e o trabalho foi intitulado "Linked Geospatial Data: desafios e oportunidades de pesquisa" [Moura & Davis Jr, 2013]. A segunda publicação foi feita em 2014 no *8th ACM SIGSPATIAL Workshop on Geographic Information Retrieval*, com o título "Integration of linked data sources for gazetteer expansion" [Moura & Davis Jr, 2014] e trata em detalhes do processo de integração de dados proposto para o Linked OntoGazetteer.

Referências Bibliográficas

Ahlers, D. (2013). Assessment of the accuracy of geonames gazetteer data. Em *Proceedings of the 7th Workshop on Geographic Information Retrieval*, GIR '13, pp. 74--81, New York, NY, USA. ACM.

Alencar, R. O.; Davis Jr., C. A. & Gonçalves, M. A. (2010). Geographical classification of documents using evidence from wikipedia. Em *Proceedings of the 6th Workshop on Geographic Information Retrieval*, GIR '10, pp. 12:1--12:8, New York, NY, USA. ACM.

Amitay, E.; Har'El, N.; Sivan, R. & Soffer, A. (2004). Web-a-where: Geotagging web content. Em *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pp. 273--280, New York, NY, USA. ACM.

Andogah, G.; Bouma, G. & Nerbonne, J. (2012). Every document has a geographical scope. *Data & Knowledge Engineering*, 81:1--20.

Backstrom, L.; Kleinberg, J.; Kumar, R. & Novak, J. (2008). Spatial variation in search engine queries. Em *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pp. 357--366, New York, NY, USA. ACM.

Beckett, D.; Berners-Lee, T.; Prud'hommeaux, E. & Carothers, G. (2015). Rdf 1.1 turtle, http://www.w3.org/tr/2014/rec-turtle-20140225. 24 de Fevereiro de 2015.

Berners-Lee, T. (2006). Linked data - design issues, http://www.w3.org/designissues/linkeddata.html. Acessado em 5 de Agosto de 2013.

Bizer, C.; Heath, T. & Berners-Lee, T. (2009a). Linked data-the story so far. *International journal on semantic web and information systems*, 5(3):1--22.

Bizer, C.; Lehmann, J.; Kobilarov, G.; Auer, S.; Becker, C.; Cyganiak, R. & Hellmann, S. (2009b). Dbpedia-a crystallization point for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154--165.

Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T. & Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. Em *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pp. 1247--1250. ACM.

Borges, K. A. V.; Laender, A. H. F.; Medeiros, C. B. & Davis Jr., C. A. (2007). Discovering geographic locations in web pages using urban addresses. Em *Proceedings of the 4th ACM Workshop on Geographical Information Retrieval*, GIR '07, pp. 31--36, New York, NY, USA. ACM.

Cucerzan, S. (2007). Large-scale named entity disambiguation based on wikipedia data. Em *EMNLP-CoNLL*, volume 7, pp. 708--716.

Freitas, A.; Curry, E.; Oliveira, J. G. & O'Riain, S. (2012). Querying heterogeneous datasets on the linked data web: Challenges, approaches, and trends. *Internet Computing, IEEE*, 16(1):24--33.

Goodchild, M. F. & Hill, L. L. (2008). Introduction to digital gazetteer research. *International Journal of Geographical Information Science*, 22(10):1039--1044.

Google (2013). Freebase data dumps, https://developers.google.com/freebase/data. 18 de Novembro 2013.

Han, X. & Zhao, J. (2009). Named entity disambiguation by leveraging wikipedia semantic knowledge. Em *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 215--224. ACM.

Hill, L. L. (2000). Core elements of digital gazetteers: placenames, categories, and footprints. Em *Research and advanced technology for digital libraries*, pp. 280--290. Springer.

Hoffart, J.; Yosef, M. A.; Bordino, I.; Fürstenau, H.; Pinkal, M.; Spaniol, M.; Taneva, B.; Thater, S. & Weikum, G. (2011). Robust disambiguation of named entities in text. Em *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 782--792. Association for Computational Linguistics.

Jannach, D.; Zanker, M.; Felfernig, A. & Friedrich, G. (2010). *Recommender systems: an introduction*. Cambridge University Press.

Lakshman, A. & Malik, P. (2009). Cassandra: structured storage system on a p2p network. Em *Proceedings of the 28th ACM symposium on Principles of distributed computing*, pp. 5--5. ACM.

Machado, I. M. R.; Alencar, R. O.; Oliveira Campos Jr, R. & Davis Jr, C. A. (2011). An ontological gazetteer and its application for place name disambiguation in text. *Journal of the Brazilian Computer Society*, 17(4):267--279.

Moura, T. H. V. M. & Davis Jr, C. A. (2012). Expansão do conteúdo de um gazetteer: nomes hidrográficos. Em *Proceedings of the XIII Brazilian Symposium on Geoinformatics*, pp. 78--83, Campos do Jordao, Sao Paulo, Brasil.

Moura, T. H. V. M. & Davis Jr, C. A. (2013). Linked geospatial data: challenges and research opportunities (in portuguese). Em *Proceedings of the XIV Brazilian Symposium on Geoinformatics*, pp. 13--18, Campos do Jordao, Sao Paulo, Brasil.

Moura, T. H. V. M. & Davis Jr, C. A. (2014). Integration of linked data sources for gazetteer expansion. Em *Proceedings of the 8th Workshop on Geographic Information Retrieval*, p. 5. ACM.

Popescu, A.; Grefenstette, G. & Moëllic, P. A. (2008). Gazetiki: automatic creation of a geographical gazetteer. Em *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*, pp. 85--93. ACM.

Pouliquen, B.; Kimler, M.; Steinberger, R.; Ignat, C.; Oellinger, T.; Blackler, K.; Fuart, F.; Zaghouani, W.; Widiger, A.; Forslund, A.-C. et al. (2006). Geocoding multilingual texts: Recognition, disambiguation and visualisation. *arXiv preprint cs/0609065*.

Sanderson, M. & Kohler, J. (2004). Analyzing geographic queries. Em *SIGIR Workshop on Geographic Information Retrieval*, volume 2.

Schmachtenberg, M.; Bizer, C. & Paulheim, H. (2014). Adoption of the linked data best practices in different topical domains. Em *The Semantic Web–ISWC 2014*, pp. 245--260. Springer.

Shvaiko, P. & Euzenat, J. (2005). A survey of schema-based matching approaches. Em *Journal on Data Semantics IV*, pp. 146--171. Springer.

Singhal, A. (2012). Introducing the knowledge graph: things, not strings. *Official Google Blog, May*.

Smart, P. D.; Jones, C. B. & Twaroch, F. A. (2010). Multi-source toponym data integration and mediation for a meta-gazetteer service. Em *Geographic Information Science*, pp. 234--248. Springer.

Souza, L. A.; Davis Jr, C. A.; Borges, K. A.; Delboni, T. M. & Laender, A. H. (2005). The role of gazetteers in geographic knowledge discovery on the Web. Em *Web Congress, 2005. LA-WEB 2005. Third Latin American*, pp. 9--pp. IEEE.

Wang, C.; Xie, X.; Wang, L.; Lu, Y. & Ma, W.-Y. (2005). Detecting geographic locations from Web resources. Em *Proceedings of the 2005 Workshop on Geographic Information Retrieval*, GIR '05, pp. 17--24, New York, NY, USA. ACM.