Capítulo 5

Dados de Múltiplas Fontes da Web: coleta, integração e pré-processamento

Natércia A. Batista[1], Michele A. Brandão[1,2], Michele B. Pinheiro[1], Daniel H. Dalip[3] e Mirella M. Moro[1]

[1]Universidade Federal de Minas Gerais (UFMG)

[2]Instituto Federal de Minas Gerais (IFMG)

[3]Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG)

{natercia, micheleabrandao, mibrito, mirella}@dcc.ufmg.br

hasan@decom.cefetmg.br

Abstract

Web data are heterogeneous and unstructured, which defines challenges for data crawling, integration and preprocessing. Different studies are "data-oriented" (i.e. based on the available data) but their results are restricted to their specific data. In contrast, there are various problems prior to identifying what data is needed to solve them, and often multiple data sources are needed. In this context, crawling, integrating and preprocessing data appropriately enables to create datasets for solving such problems. Therefore, this short course addresses these three activities by discussing challenges and practical solutions.

Resumo

Atividades de coleta, integração e pré-processamento representam diferentes desafios para pessoas que necessitam de lidar com dados extraídos da Web por serem heterogêneos e não estruturados. Ademais, existem diferentes fontes de dados na Web que podem ser sites e aplicativos, mídias e redes sociais e até mesmo bancos (ou bases) de dados já construídos e disponibilizados. Considerar dados dessas diferentes fontes pode parecer irrelevante quando avaliados de forma isolada. Entretanto, quando combinados, conhecimentos novos, integrados e úteis podem ser descobertos. Tais dados podem ser aplicados na solução de problemas em diferentes campos, como sistemas inteligentes, ao permitir a ampliação dos dados utilizados como treinamento; marketing, ao possibilitar a identificação de público alvo; sistemas de recomendação, ao viabilizar a construção do

perfil de usuários, entre muitos outros. Nesse contexto, coletar, integrar e pré-processar dados adequadamente de múltiplas fontes permite a criação de conjuntos de dados enriquecidos que possibilitam a solução de problemas reais. Assim, este minicurso aborda essas três tarefas e apresenta seus principais desafios.

5.1. Introdução

A Web (do inglês *World Wide Web*, ou Rede Mundial de Computadores) é um sistema de documentos dos mais variados formatos que são interligados e acessados (ou executados) via Internet. Desde sua criação, a Web tem evoluído constantemente. Por exemplo, ela foi sendo gradualmente expandida para gerenciar os mais diversos tipos de documentos que vão muito além do original hipertexto (e.g., vídeo, som, imagem e afins) como também permitir que o próprio usuário crie e publique seu conteúdo. Com tantos *dados* disponíveis, a comunidade científica e a indústria de modo geral logo começaram a explorá-los das mais variadas formas e com propósitos imensuráveis.

De fato, existem diferentes fontes de dados da Web, incluindo sites e aplicativos, mídia e redes sociais e até mesmo bancos de dados completos. Os dados de tais múltiplas fontes são geralmente não estruturados, desnormalizados, inconsistentes, duplicados, incompletos e de qualidade variada [Farnadi et al., 2018, Geerts et al., 2018, Wang et al., 2018b]. Essas fontes incluem planilhas do Excel, arquivos *Comma Separated Values* (CSV), bancos de dados relacionais e não relacionais, armazém de dados e diferentes plataformas da Web (e.g., sites e aplicativos sociais).

Nesse vasto contexto, qualquer pesquisa orientada a dados Web requer o estabelecimento de uma relação entre eles para melhor combiná-los e analisá-los [Moro et al., 2009, Wang et al., 2017]. Ou seja, ao considerar múltiplas fontes de dados, pesquisadores e desenvolvedores adquirem uma visão maior sobre o contexto estudado, promovendo a descoberta de informações complementares, as quais permitem realizar inferências mais precisas, ou ainda, identificar padrões que só se tornam visíveis quando essas múltiplas fontes estão conectadas. Como exemplo prático e real, considere duas das maiores plataformas sociais online que são independentes uma da outra: Facebook (rede social de amizade) e GitHub (plataforma de desenvolvimento de software colaborativo). Pesquisas para compreender diferentes perfis de colaboração podem integrar dados de ambas a fim de, por exemplo: analisar como os relacionamentos pessoais influenciam o desenvolvimento de software; verificar se um desenvolvedor é popular no GitHub por também estar em outra mídia social; ou se um desenvolvedor é capaz de influenciar a comunidade desenvolvedora pela criação novos padrões de desenvolvimento, bem como disseminação desse novos padrões em outra rede social; entre muitas outras possibilidades interessantes.

Outro exemplo prático e real é usar dados de múltiplas fontes para apoiar a tomada de decisões, nesse caso considerando principalmente o aspecto financeiro [Geerts et al., 2018]. Em tal contexto, uma equipe de comércio eletrônico pode analisar o perfil de seus usuários em diferentes redes sociais para descobrir interesses e, em seguida, recomendar vendas combinadas. Para fazê-lo de uma maneira eficiente e eficaz, é necessário novamente coletar, integrar e pré-processar dados, que por sua vez são tarefas que representam maneiras de extrair *valor* (e então ganho financeiro) de tais dados.

Porém, para atingir os objetivos desses dois exemplos (desenvolvimento colabora-

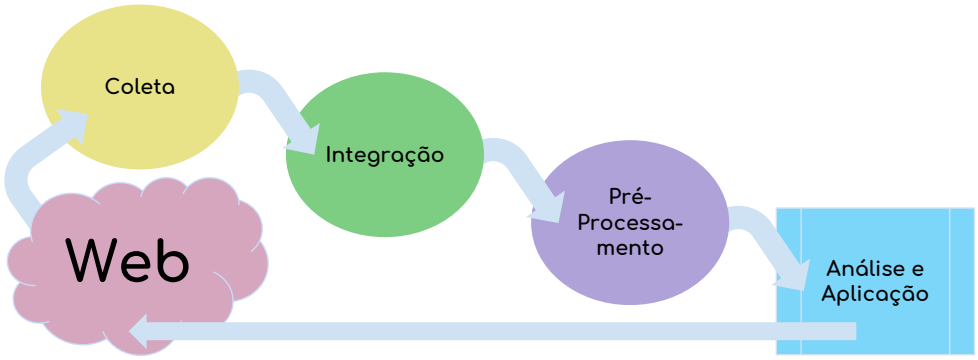

Figura 5.1: Processo de dados Web e tópicos abordados neste capítulo

tivo e marketing dirigido), é necessário obter os dados da Web. Tecnicamente, o primeiro problema é definir a estratégia de coleta (ou *crawling*), que pode ser classificada de acordo com o período e a forma como a semente (por onde a coleta inicia) é definida. A escolha de uma estratégia para integrar os dados de diferentes fontes também é importante para propiciar uma visão uniforme para usuários ou aplicativos, bem como armazenamento adequado para permitir consultas eficientes mais tarde. Finalmente, o pré-processamento de dados também pode ser necessário, o que ocorre antes ou depois da integração de dados, e envolve a resolução de dados ausentes e duplicados, normalização, etc.

Em resumo, como a Figura 5.1 ilustra, para desenvolver qualquer pesquisa ou aplicação com dados Web é necessário: *coletar* tais dados de fontes diversas (geralmente espalhadas pela própria Web), *integrar* tais dados, muitas vezes realizar *pré-processamentos* diferentes para então conseguir *analisar* e *aplicar* quaisquer que sejam as técnicas sendo pesquisadas ou desenvolvidas, que comumente têm seus resultados divulgados ou armazenados novamente na Web (fechando o ciclo). Nesse contexto, coletar, integrar e pré-processar dados de várias fontes (frequentemente heterogêneas) apresentam diferentes desafios, incluindo: coletar dados em tempo real, gerenciar ferramentas de coleta, decidir sobre questões de privacidade, padronizar dados diferentes, resolver dados duplicados, trabalhar com dados não uniformes, padrões úteis de mineração e qualidade, entre outros.

Apesar de conhecidas, tais tarefas são muitas vezes complexas e dependentes de contexto. Por exemplo, apenas o pré-processamento de dados requer cerca de 80% do tempo de cientistas de dados de acordo com a pesquisa publicada em [Tyagi et al., 2010]. Na prática, não apenas cientistas de dados, mas também programadores, pesquisadores, estudantes, empresas e usuários podem se beneficiar da solução desses desafios. Em especial, quando tais soluções retornam à comunidade através da publicação ou disponibilização de dados padronizados e completos na Web, o ganho é global.

Como exemplos, considere as seguintes publicações de tal retorno no âmbito da edição de 2017 deste evento (WebMedia). Araujo et al. [2017] utilizam de técnicas de coleta e pré-processamento de dados para predizer o sucesso de um álbum de música baseado em comentários de redes sociais online. Já Freitas et al. [2017] propõem uma estratégia de integração de dados baseada em ontologias e dados conectados (tecnologias a serem explicadas mais adiante neste capítulo) para calcular a probabilidade do risco de óbito maternos e infantil no Brasil. Ainda em integração, o conceito de ontologia é tão

versátil que Veiga et al. [2017] o utilizam para dados provenientes de redes de sensores e internet das coisas. Já utilizando um framework muito comum no contexto de dados Web, Maia and Oliveira [2017] compilam as pesquisas sobre o vírus Zika e analisam a reputação de seus pesquisadores no contexto da saúde mundial.

A motivação deste capítulo é: auxiliar pesquisadores e desenvolvedores que precisam de dados Web e diminuir o tempo de obtenção de tais dados através da divulgação de soluções existentes para vários dos desafios mencionados. Este capítulo aborda então as três questões de forma integrada (coleta, integração e pré-processamento) e apresenta soluções que podem ser aplicadas à pesquisa e ao desenvolvimento de aplicações comerciais. A organização deste capítulo segue a ordem supra-citada após a Seção 5.2 que resume fontes de dados Web; a Seção 5.3 discute coleta de dados Web; a Seção 5.4 resume algumas estratégias para integração de dados; a Seção 5.5 apresenta os principais problemas durante o pré-processamento de dados; e a Seção 5.6 detalha algumas aplicações reais de dados múltiplas fontes Web. Finalmente, a seção 5.7 apresenta considerações finais e ponteiros para outras fontes de informação sobre os assuntos aqui tratados.

5.2. Fontes de Dados Web

O processamento de dados da Web requer o prévio conhecimento das fontes disponíveis. Com tal entendimento, é possível realizar um planejamento da coleta e definir estratégias que podem melhorar sua eficiência e cobertura. Nesta seção, são apresentados alguns dos tipos de fontes de dados mais comumente encontrados na Web, sendo essas os dados abertos, dados conectados, APIs e páginas da Web. Sobre cada fonte, são também apresentados os desafios a serem considerados em seu processamento.

5.2.1. Dados Abertos

Os dados abertos, como o próprio nome expressa, são dados disponibilizados abertamente. Uma organização sem fins lucrativos *Open Knowledge Foundation* (OKF)[1] foi criada para incentivar e estipular um conjunto de diretivas para a publicação de dados abertos. Ela define, de forma geral, que os dados abertos devem ser disponibilizados de forma completa, em formato legível por máquina e sem restrições de utilização. Especificamente, dentre as diretivas apresentadas pela fundação citam-se:

- disponibilidade e acesso – definem que os dados abertos devem estar disponíveis de forma completa com um custo não maior do que o custo de reprodução, e de preferência disponíveis para download;
- reúso e distribuição – definem que os dados devem ser distribuídos com licença que permita sua reutilização incluindo a mistura com outras bases de dados, além de estar em um formato legível por máquinas; e
- participação universal – define que os dados devem viabilizar a participação universal, ou seja devem estar aptos a serem utilizados, reutilizados e distribuídos por qualquer pessoa sem discriminação a campos de atuação e grupos de pessoas, como restrições para fins não comerciais ou propósitos educacionais apenas.

Também houve um grande envolvimento de vários países na abertura de dados oficiais como forma de transparência do estado. Assim, surgiram grandes portais de pu-

[1]Open Knowledge Foundation: `https://okfn.org/`

blicação desses dados. A publicação desses dados na Web possui seus próprios desafios, como a indexação, catalogação e recuperação desses *datasets*. Geralmente, esses desafios são decorrentes da forma que os dados são publicados – como já mencionado, em formatos arquivos legíveis por máquina como XML (*Extensible Markup Language*), CSV (*Comma-separated Values*) e JSON (*JavaScript Object Notation*). Assim a organização dos dados é realizada via de regra através dos metadados que são informados sobre esses arquivos. Portanto, todas as operações de indexação, catalogação e recuperação de informação são realizadas em relação a seus metadados. Ao longo dos anos, surgiram diversas aplicações destinadas à publicação de catálogos de dados, que apesar de ainda não permitirem a recuperação de informações presentes dentro dos *datasets*, viabiliza a recuperação de informações sobre seus metadados. No universo de aplicações para dados de uso geral, destacam-se as ferramentas como CKAN[2] e Socrata[3]. São exemplos dessas grandes portais de dados públicos o portal Brasileiro[4], o Americano[5] e o Europeu[6].

Por outro lado, existem ainda bases de dados que possuem um tipo de dado específico, que são os dados espaciais. Para esse tipo de dados, existem aplicações específicas que permitem tanto a pesquisa sobre os metadados quanto a visualização dos dados espaciais, e até mesmo a sua visualização em conjunto por composição sobre formas de camadas. Essas aplicações recebem o nome de Infra-estruturas de Dados Espaciais (IDEs, ou em inglês SDI - *Spatial Data Infrastructure*). Existem diversas alternativas de implementações de IDEs, dentre elas estão o Geoserver[7], mais popularmente utilizada e o Mapserver[8]. Nesse caso os dados são disponibilizados sobre o um padrão serviços e formatos definidos pela *Open Geospatial Consortium* (OGC)[9].

5.2.2. Dados Conectados

Dados conectados estão no contexto de Web Semântica. Portanto, para compreender sua representação, é preciso primeiramente entender o propósito da Web Semântica, bem como a contextualização de pontos chave em sua de sua estruturação.

Web Semântica. A Web Semântica é uma extensão da Web tradicional que inclui informações sobre sentido e significado dos dados em suas páginas ou publicados abertamente. Sentido e significado dos dados são então introduzidos de forma que possam ser interpretados por aplicações, permitindo assim que executem tarefas mais complexas e de forma mais autônoma. Dessa forma, através desse tipo de informação, uma aplicação que coleta páginas da Web pode identificar sentido do termo "Bertha Lutz" como sendo o nome de uma pessoa, endereço (rua, avenida, bairro, etc), lugar (escola, hospital, etc), por exemplo. Partindo desse conhecimento, a aplicação de coleta de dados pode então executar ações mais precisas sobre esses dados, como armazená-los como dado espacial, descartar caso não seja o tipo de dado a qual a aplicação esteja interessada em coletar, ou ligar um

[2]CKAN: `https://ckan.org`
[3]Socrata: `https://socrata.com/`
[4]Portal Brasileiro de Dados Abertos: `http://dados.gov.br`
[5]data.gov: `https://www.data.gov`
[6]*European Data Portal*: `http://europeandataportal.eu`
[7]GeoServer: `http://geoserver.org`
[8]MapServer: `https://mapserver.org`
[9]OGC: `http://www.opengeospatial.org`

dado a outras informações encontradas anteriormente pelo processo de coleta, gerando assim um conjunto de dados mais complexo e com mais informações sobre seu contexto.

Na tarefa de representar o contexto e o significado dos dados, a Web Semântica propõe dois conceitos importantes: Ontologia e Vocabulário. O conceito de ontologia vem da filosofia e implica no estudo da natureza da existência e propriedade do seres e das "coisas". Para a Computação, as ontologias são modelos de dados que representam a descrição de seres, objetos e coisas existentes no mundo. Dessa forma, as ontologias na Web Semântica têm a função de descrever a que um dado se refere. No caso do exemplo anterior, sobre o termo "Bertha Lutz", o lugar pode ser descrito por ontologias como rua, avenida, bairro, escola, hospital. Dentro dos conceitos de Web Semântica, também foram definidas linguagens para a descrição de ontologias, como a mais comumente utilizada OWL (*Web Ontology Language*). Através da OWL é possível descrever ontologias utilizando componentes como: Classes, Atributos, Indivíduos, Relações, Termos funcionais, Restrições, Regras, Axiomas e Eventos [Sikos, 2015].

Por outro lado, os vocabulários são coleções de termos utilizados para descrever uma área de interesse. Por exemplo, um dos vocabulários mais utilizados para descrever "Pessoas" é o chamado FOAF (*Friend of a Friend*)[10], e apresenta termos que representam classes como *OnlineAccount*, *PersonalProfileDocument*, propriedades como *homepage*, *knows*, *accountName*, dentre outras. Os vocabulários em geral podem ser referenciados durante a publicação dos dados através de *namespaces*, que por sua vez são representados através de atributos *"xmlns"*.

Em resumo, os conceitos de ontologia e vocabulário são semelhantes. Inclusive, de acordo com a W3C[11] não existe uma separação clara entre os dois. Porém, é usualmente referido a modelos de descrição de dados mais complexos como Ontologias.

Linked Data. Os dados conectados, conforme próprio nome, representam dados que estão relacionadas entre si. Estão no contexto de Web Semântica por explicitarem os sentido dos dados, assim como o significado das relações representadas. Assim, são capazes de representar de forma ampla ambos o sentido dos dados e o contexto de suas relações.

Nesse contexto, um dos formatos para a representação dos dados mais utilizados é o RDF (*Resource Description Framework*). Esse formato é capaz de representar a descrição de qualquer tipo de dado presente na Web. Para isso, o *framework* possui um vocabulário próprio que permite descrever informações sobre os dados como literais, classes, propriedades, *statements*, listas, conjuntos e sequências [Sikos, 2015]. O RDF é estruturado através de sentenças que seguem a forma sujeito-predicado-objeto (ou recurso-propriedade-valor), também conhecidas por *triplas RDF*. As triplas, dessa forma, são capazes de expressar as propriedades de um conteúdo presente na Web, bem como seu contexto. Expressar as propriedades de um recurso é mais simples, pois as triplas representam a sentença recurso-propriedade-valor. Assim, esse formato de tripla pode ser utilizado para caracterizar o recurso através de suas propriedades. Por outro lado, o RDF pode representar uma relação entre recursos presentes na Web, utilizando a triplas da forma sujeito-predicado-objeto, onde o objeto é outro recurso também disponível na

[10]FOAF: `http://xmlns.com/foaf/spec/`
[11]W3C: `https://www.w3.org/standards/semanticWeb/ontology`

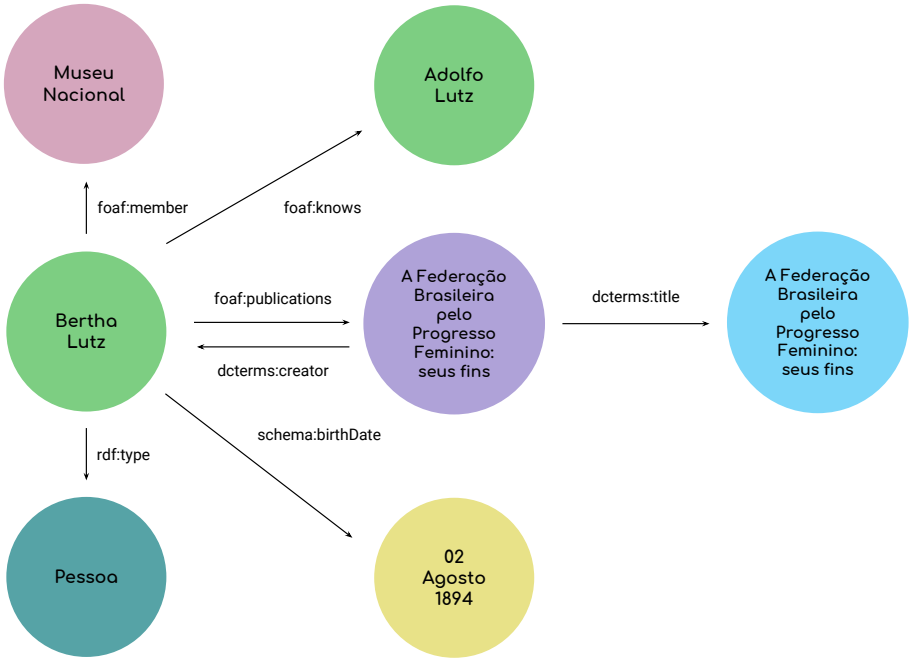

Figura 5.2: Representação gráfica do exemplo apresentado utilizando RDF em N-Triples

Web, e o predicado especifica o sentido da relação estabelecida entre ambos.

Voltando ao exemplo da Bertha Lutz, podemos criar um arquivo sobre o formato N-Triple que descreve a informação relacionada à bióloga brasileira, assim como outras informações relacionadas ao seu contexto. O Código 5.1 (a seguir) permite verificar as duas formas de utilização das triplas; ou seja, aplicada na descrição das propriedades de um recurso, e na descrição de sua conexão com outros recursos.

Código 5.1: Exemplo de RDF utilizando N-Triple

```
<https://www.wikidata.org/wiki/Q1264246> <rdf:type> <foaf:Person> .
<https://www.wikidata.org/wiki/Q1264246> <foaf:knows>
<https://www.wikidata.org/wiki/Q199652> .
<https://www.wikidata.org/wiki/Q1264246> <schema:birthDate> "02-08-1894"^^<xsd:date> .
<https://www.wikidata.org/wiki/Q1264246> <foaf:publications>
    <http://memoria.bn.br/DocReader/178691_05/36862> .
<https://www.wikidata.org/wiki/Q10301958> <dcterms:title>
    "A Federacao Brasileira pelo Progresso Feminino: seus fins" .
<https://www.wikidata.org/wiki/Q10301958> <dcterms:creator>
    <https://www.wikidata.org/wiki/Q1264246> .
```

O Código 5.1 utiliza uma forma de representar o RDF conhecido como N-Triples, mas existem outros formatos que seguem os conceitos propostos pelo RDF, como N-Quad e Turtle, que fazem parte da família das linguagens RDF chamada de *Turtle*, assim como o RDF/XML, RDFa e JSON-LD. A descrições dessas linguagens podem ser vistas na página da W3C sobre a utilização dos formatos para RDF[12].

Para compreender as relações estabelecidas pelas triplas RDF, o exemplo anterior é ilustrado na Figura 5.2. Assim, visualiza-se como as triplas estabelecem relações para: as propriedades de um recurso sobre a Bertha Lutz como sua a data de nascimento; e a

[12] https://www.w3.org/TR/rdf11-primer/#section-graph-syntax

relação entre recursos, como o predicado *foaf:publications*, que indica que Bertha Lutz publicou o artigo *A Federacao Brasileira pelo Progresso Feminino: seus fins*.

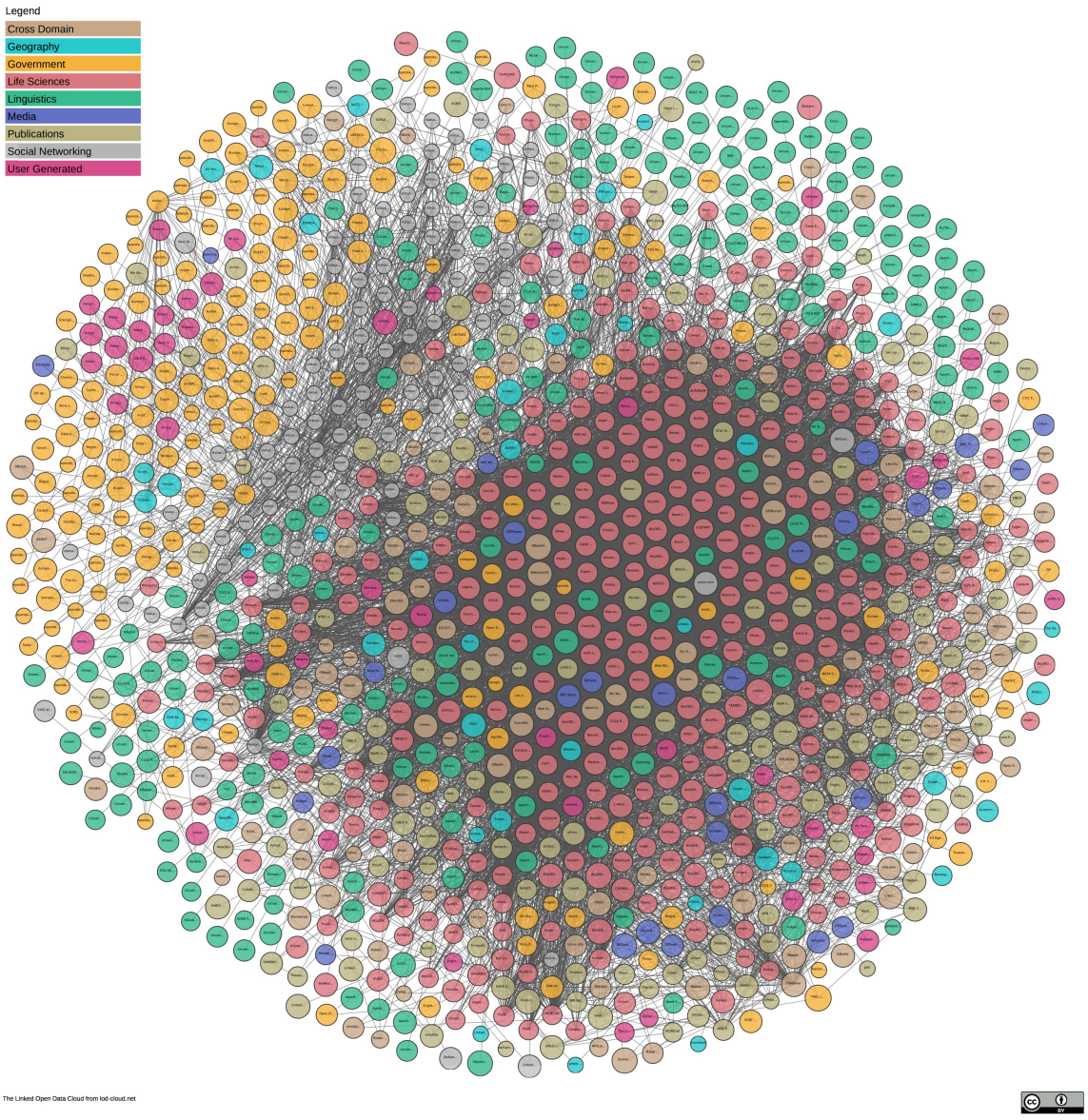

Figura 5.3: Distribuição das associações entre bases de dados conectados abertos. Fonte: *Linked Open Databases Cloud*

Os dados conectados recebem esse nome por representar e descrever as conexões entre recursos presentes na Web. Assim, é possível expressar as ligações entre conteúdos de contextos diferentes, como pessoas a objetos através de relações de pertencimento, interesses, etc. Nesse sentido existe um esforço também para a publicação desse tipo de dados de forma aberta. Uma vez que esses dados têm a capacidade de estarem interligados em diferentes contextos, a combinação dessas bases tem o potencial de gerar inúmeras aplicações complexas, uma vez que elas também carregam a semântica desses dados e relacionamentos. Nesse sentido, existe um esforço de padronização do processo de publicação desses tipos de dados sobre a forma de dados abertos. Os dados abertos

dessa natureza receberam o nome de *Linked Open Data* (LOD). Já existem diversas bases de dados LOD, como ilustrado na Figura 5.3[13], que começaram a ser publicadas em meados de 2007, com destaque para DBpedia, GeoNames e FreeBase.

5.2.3. Páginas da *Web*

Páginas da Web são a forma mais comum de dados que podem ser coletados. Tais páginas foram criadas para permitir a disseminação de informações na Web, com principal finalidade de exibir tais informações aos usuários obedecendo elementos visuais definidos pelos seus autores. Dessa forma, criou-se o *HyperText Markup Language* (HTML), o qual permite que as páginas sejam disseminadas mantendo a formatação estipulada pelos produtores de conteúdo. Os navegadores são então capazes de interpretar os arquivos HTML e realizar a diagramação do conteúdo da forma descrita por esses arquivos.

Devido ao propósito inicial da Web, suas páginas foram desenvolvidas de forma menos estruturada, pois apesar de os elementos visuais serem interpretados, não existe um modelo de dados comum entre páginas que garanta uma estrutura única entre as mesmas. Ou seja, ao desconsiderar os elementos de renderização, as páginas da Web não foram desenvolvidas visando a interpretação de seus dados por máquinas, e sim por humanos.

A coleta de dados dessas páginas, organizadas da forma tradicional, requer conhecimento de ferramentas de navegação e consulta em HTML, como o XPath (linguagem de consulta para documentos XML [Moro et al., 2009]) que permite a navegação na árvore da estrutura HTML. Além disso, apesar de não existir um modelo de dados bem definido para a Web como um todo, algumas vezes é possível encontrar um modelo de dados específico para alguns domínios. Isso ocorre quando páginas Web constituem uma forma de exibição de dados que já estão estruturados nas bases desses sites. Exemplos incluem perfis de usuários em redes sociais, páginas de venda de produtos, dados de diferentes publicações em uma biblioteca digital, entre outros.

Com o surgimento da Web Semântica, também houve um avanço nas convenções de utilização das *tags* HTML e em seus possíveis atributos a fim de compreender a descrição dos dados compartilhados. No caso das páginas da Web, a inserção de anotações semânticas permite que aplicações clássicas sejam melhoradas. Exemplos de tais aplicações são as máquinas de busca, que passaram a compreender melhor o conteúdo das páginas Web, aprimorando a indexação e a qualidade dos resultados das consultas. Dentre as principais alterações do HTML, destaca-se a inserção de metadados através de microformatos, RDFa, *microdata* para HTML5 e JSON-LD.

Os microformatos são uma das primeiras tentativas de inclusão da informação semântica nas páginas da Web e seguem ideias como, por exemplo, as do chamado *Plain Old Semantic HTML*[14]. Portanto, se baseiam na estratégia de reutilizar algumas partes dos atributos das *tags* já existentes para HTML (como *rel*, *class* e *rev*) para anotar o sentido e significado dos dados presentes em uma página. Existem também alguns padrões[15] para descrição de diversas entidades através dos microformatos, dentre os mais conhecidos pode-se citar o *hCalendar* para a descrição de eventos e o *hCard* para informações

[13]LOD Cloud: `https://lod-cloud.net/`
[14]Plain Old Semantic HTML: `http://microformats.org/wiki/posh`
[15]Microformats wiki: `http://microformats.org/wiki/Main_Page`

de contato de pessoas, companhias e organizações. Retomando novamente o exemplo da Bertha Lutz, pode-se publicar os dados sobre o evento da fundação da Liga para Emancipação Intelectual da Mulher, em que a mesma estava envolvida, assim como os dados sobre informações básicas da pesquisadora, como pode ser visto no Código 5.2.

Código 5.2: Exemplo de utilização dos padrões de microformato *hCalendar* e *hCard*.

```
<!-- hCalendar -->
<span class="vevent">
 <span class="summary">
  Fundacao da Liga para Emancipacao Intelectual da Mulher
 </span> on <span class="dtstart">1919</span>
 ocorrida no
 <span class="location">Brasil</span>
 organizada por <span class="organizer">Bertha Lutz</span>.
</span>

<!-- hCard -->
<div class="vcard">
 <span class="fn given-name">Bertha</span>
 <span class="fn family-name">Lutz</span>
 <div class="adr>
  <span class="type">Nascida em</span>
  <span class="locality">Sao Paulo</span>,
  <abbr class="region" title="Sao Paulo">SP</abbr>
  <span class="country-name">BRA</span>
 </div>
</div>
```

O RDFa é outra forma de anotar as informações semânticas em páginas HTML, especificamente voltado para o conceito de dados conectados. O RDFa propõe a implementação dos conceitos de triplas do RDF através de novos atributos (*vocab*, *typeof*, *property*, *resource*, e *prefix*) que são inseridos em *tags* HTML tradicionais. De forma geral, a página que contém o RDFa representa o sujeito das triplas, quando este não é destacado pelo atributo *resource*, como apresentado no Código 5.3 através da utilização do atributo na *tag div* mais externa. No caso do predicado, este é indicado pelo atributo *property*; e o objeto a que esse predicado se refere é o conteúdo existente dentro da *tag*, como foi utilizado na *tag h2* que contém o título do artigo da Bertha Lutz no Código 5.3. Os demais atributos indicam informações presentes no RDF, como vocabulário, tipos e prefixo, o qual é utilizado no RDF para referenciar um *namespace* ou vocabulário, exemplificado na *tag body* (Código 5.3).

Código 5.3: Exemplo de utilização do RDFa

```
<html>
  <head>
    ...
  </head>
  <body vocab="https://bib.schema.org/Thesis">
   <div
    resource="/bertha_lutz/publications/thesis"
    typeof="Thesis">
    <h2 property="https://schema.org/headline">
     A Nacionalidade da Mulher Casada perante
     o Direito Internacional Privado
    </h2>
    <h3
     property="https://schema.org/author"
     resource="#me">
      Bertha Lutz
    </h3>
    <div property="text">
```

```
      <!-- article content -->
     </div>
    </div>
   </body>
</html>
```

Outra forma de anotar a semântica dos dados em uma página da Web é por meio de *Microdata* para HTML5. Essa abordagem foi desenvolvida especificamente para HTML5, e portanto é compatível com a mesma (ao contrário dos Microformatos, por exemplo, que em alguns casos apresentam incompatibilidade no reuso de alguns atributos em elementos quando se utiliza HTML5 para codificar uma página). O *Microdata* utiliza o padrão chave-valor também através do atributos próprios introduzidos por esse formato, os quais incluem *itemscope*, *itemtype*, *itemprop*, *itemid*,*id*, *itemref*. Especificamente, o *itemprop* deve ser utilizado para indicar um novo escopo de objeto, como pode ser observado na primeira *tag section* Código 5.4. O atributo *itemprop* em geral deve ser acompanhado da informação semântica desse objeto através do atributo *itemtype*, o qual pode assumir valores que são uma referência a um vocabulário ou ontologia.

No Código 5.4, a primeira *tag section* é acompanhada da informação de que se trata de uma pessoa, descrita pelo vocabulário FOAF, identificado por esse atributo *itemtype*. O atributo *itemid* pode ser utilizado para indicar qual o identificador do objeto descrito no contexto do site ou aplicação Web. Portanto, pode ser utilizado por sites que exibem objetos que estão catalogados em um banco de dados, por exemplo. Esse atributo é utilizado no Código 5.4 como atributo da *tag section*, com valor 3309.

Por fim, os atributos de *id* e *itemref* são utilizados quando se deseja relacionar dois objetos em uma mesma página, quando esses objetos não estão aninhados, por exemplo. O *id* não necessariamente deve assumir o mesmo valor semântico do atributo *itemid*, apresentado anteriormente, mas deve ser consistente com sua referência utilizando *itemref*. Um exemplo de utilização dessas refernências pode ser observado no Código 5.4 quando é realizada uma descrição das publicações de Bertha Lutz. Os elementos que descrevem as informações básicas da feminista são apontados na primeira *section*; posteriormente, nas segunda *section*, essa referência é retomada para a descrição de suas publicações.

Código 5.4: Exemplo de utilização do *microdata*

```
<html>
 <head>
  <title>Bertha Lutz</title>
 </head>
 <body>
  <section itemscope itemtype="foaf:person"
   id="berthaLutz" itemid='3309'>
   <h1>
    <span itemprop="foaf:firstName">Bertha</span>
    <span itemprop="foaf:surname">Lutz</span>
   </h1>
  </section>
  <section itemref="berthaLutz" itemscope
   itemtype="foaf:publications">
   <div itemprop="schema:article">
    <a itemprop="rdf:resource"
     href="http://memoria.bn.br/docreader/178691_05/36862">
      A Federacao Brasileira pelo Progresso Feminino: seus fins
    </a>
    <span itemprop="dcterms:creator">Bertha Lutz</span>
    <span itemprop="dcterms:creator">Carmem de Carvalho</span>
    <span itemprop="dcterms:creator">Orminda Bastos</span>
```

```
    </div>
   </section>
  </body>
</html>
```

Dentre os formatos mencionados para a publicação de dados em páginas da Web com sua informação semântica, o JSON-LD é o que mais se diferencia. Esse formato utiliza a *tag* de *script* da linguagem HTML para disponibilizar todos os objetos referenciados na página utilizando JSON (*JavaScript Object Notation*). Para isso, a tag de *script* é acompanhada do atributo *type* que indica se tratar de um código no formato JSON-LD. Seguindo a lógica de anotação semântica, a chave *@context* e *@type* definem o vocabulário ou ontologia a qual o objeto se trata. Além dessas chaves, *@id* é outra chave definida pelo modelo e deve ser utilizada para anotar o identificador desse objeto no respectivo conjunto de dados. Além de tais propriedades, podem ser utilizadas quaisquer outras propriedades definidas pela ontologia. Um exemplo de sua utilização pode ser visto através do exemplo da Bertha Lutz no Código 5.5.

Código 5.5: Exemplo de utilização do JSON-LD

```
{
  "@context": "https://json-ld.org/contexts/person.jsonld",
  "@id": "https://pt.wikipedia.org/wiki/Bertha_Lutz",
  "name": "Bertha Lutz",
  "born": "1894-08-02",
  "father": "https://pt.wikipedia.org/wiki/Adolfo_Lutz"
}
```

5.2.4. APIs

Em meados de 2007, a Web passou por uma transformação na forma de organizar suas páginas. Essa transformação aconteceu pelo surgimento dos conceitos da Web 2.0 que mudam a forma com que os dados publicados na Internet são produzidos. Especificamente, a Web 2.0 trouxe uma nova visão sobre como melhor utilizar os recursos já disponíveis na Web para incluir os até então apenas *viewers* das páginas Web na produção do conteúdo que circula na rede. Assim também surgiu a ideia de *aplicações Web*, que agem como software tradicional para *desktop*, incluem as funções de criação, modificação, persistência dos dados e visualização conhecido pelo padrão CRUD (*Create*, *Retrieve*, *Update* e *Delete*). As aplicações Web se diferenciam das aplicações em *desktop* por permitirem acesso de qualquer navegador Web, estando disponíveis sem a necessidade da instalação de algum programa.

Por se tratar de um ambiente diferente do ambiente *desktop*, houve a necessidade do desenvolvimento de novos conceitos sobre a arquitetura de aplicações. Então, Fielding [2000] propõe a arquitetura em camadas, que em linhas gerais é amplamente utilizada atualmente nesse tipo de aplicação. Essa arquitetura divide as aplicações em componentes que estão distribuídos em três tipos principais de camadas: Aplicação, Serviços e Persistência. A Figura 5.4 ilutstra tal arquitetura. A camada de aplicação contém todas as aplicações que podem ser criadas com um mesma lógica do conjunto de dados presente na camada de Persistência. As APIs surgem então como componentes que integram a camada de Serviços (que é responsável por organizar toda a lógica das operações de CRUD) e controlam as permissões de execução desses tipos de operação em relação aos usuários que requisitam a operação, dentre outras atribuições de segurança.

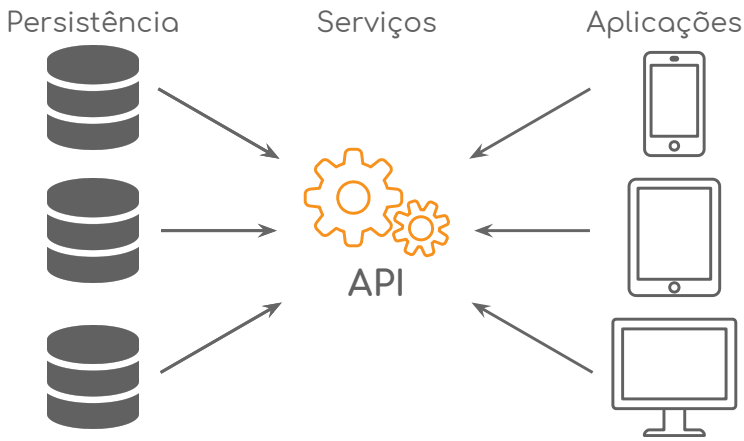

Figura 5.4: Arquitetura em camadas de aplicações Web

Para gerar aplicações secundárias que revertam em uma maior utilização dessas aplicações Web, as empresas comumente concedem o acesso a APIs que permitem a manipulação até certo ponto de seus dados através de requisições ao serviço. Essa estratégia permite que essas aplicações secundárias possam ser desenvolvidas sem que as empresas necessariamente tenham que despender recursos para isso. Através dessa concessão, essas APIs se tornam outra fonte de dados na Web.

Além disso, as APIs expandiram sua utilização com o conceito de arquitetura orientada a serviços (em inglês: *Service-Oriented Architecture* - SOA). Nesse sentido, o SOAP (*Simple Object Access Protocol*) foi o primeiro padrão de serviço a ser formalizado. Esse padrão estabelece que as requisições ao serviço devem ser realizadas através do protocolo HTTP (*Hypertext Transfer Protocol*) e RPC (Remote Procedure Call), e as mensagens devem ser trocadas utilizando o formato XML. Anos mais tarde, houve a proposta de um nova arquitetura de serviços chamada de REST (*Representational State Transfer*) [Fielding, 2000], ou RESTful. A grande maioria dos serviços implementados com o padrão REST utiliza também o formato JSON (*JavaScript Object Notation*) para as mensagens trocadas através do protocolo HTTP.

Note que o acesso a essas APIs é realizado com a *concessão* das suas empresas fornecedoras. Portanto ao contrário dos dados anteriores de acesso irrestrito, o acesso aos dados fornecidos pelas APIs requer um cadastro prévio. Esse cadastro em geral é realizado em nome das aplicações que desejam utilizar a API e não em nome dos desenvolvedores; e nele, os desenvolvedores garantem que essa aplicação seguirá as políticas de acesso e utilização dos dados impostas pelas empresas. Após o cadastro, as aplicações recebem uma chave (ou conjunto de chaves) de acesso que deve ser utilizada para autenticar as requisições realizadas à API. Através dessa autenticação, as empresas podem controlar quais aplicações requisitam suas informações e aplicar suas políticas de acesso.

As políticas de acesso variam entre APIs, mas em geral elas tentam garantir que o serviço não seja sobrecarregado por requisições, e que parte dos dados dos usuários só possam ser acessados se o próprio usuário concedeu essa permissão à aplicação. O primeiro caso de política de acesso é comum em praticamente todas as APIs, e é definido por medidas como número de requisições por intervalo de tempo (minutos, horas, dias,

por exemplo), ou por intervalo de tempo entre requisições (segundos, minutos, por exemplo). Em alguns casos, o descumprimento dessa política pode acarretar em penalizações às aplicações, em geral sobre a forma do interrompimento temporário da resposta às suas requisições. Dessa forma essa política deve ser observada pelas aplicações que consomem os dados das APIs, para que não impeça seu funcionamento.

O segundo tipo de política de acesso é relativa a aplicações que apresentem dados de usuários, em geral aplicações que implementam um protocolo de autenticação conhecido como OAuth, que atualmente possui duas versões. Esse protocolo de autenticação permite que os usuários concedam permissão às aplicações terceiras para acessarem seus dados. Em alguns casos, as empresas exigem que os desenvolvedores especifiquem quais dados as aplicações desejam acessar dos usuários, para que estes sejam informados ao conceder o acesso as seus dados.

Uma das vantagens dos dados presentes em APIs é que os mesmos apresentam um formato e um esquema claros e documentados para que seus usuários possam as utilizar. Por essas APIs seguirem um protocolo bem definido, o formato das suas mensagens em geral são sempre em XML e JSON. Por outro lado, o esquema dos dados fornecidos é característico de cada API, e em geral as empresas disponibilizam uma documentação sobre esses esquemas. Além do formato dos dados, no caso das APIs é importante entender o formato das requisições, tanto para realizar o processo de autenticação, quanto para compreender quais *endpoints* estão disponíveis para realizar as ações de coleta e manipulação dos dados pelas aplicações desenvolvidas. A documentação cobre todas essas informações, bem como as restrições de utilização desses *endpoints*, como as políticas de acesso, que também podem variar em uma mesma API.

5.3. Coleta de Dados Web

A partir dos tipos existentes de publicação de dados na Web (discutidos na seção anterior), existem diversas formas de coletá-los. Nos dois primeiros tipos de dados apresentados (Dados abertos e Dados conectados), ambos podem ser obtidos de forma completa pelos sites de dados abertos. Esses dados são disponibilizados de forma estruturada, sobre formatos legíveis por máquina como XML, CSV e JSON, para os dados abertos de forma geral; e em forma de RDF/XML, N-Triples, N-Quad no caso de dados abertos conectados (LOD). Normalmente é possível obter todos os dados sobre determinado assunto de uma fonte através de seu download, uma vez que esses sites já oferecem esse conteúdo organizado para os usuários.

De um outro modo, páginas da Web e APIs consideram dados fragmentados, ou espalhados, e requerem a utilização de uma aplicação para coletá-los e organizá-los em uma base estruturada pelo desenvolvedor. No caso das páginas da Web, os dados estão espalhados em diversos arquivos, que dependendo da intenção de sua utilização pelo usuário, podem estar espalhados inclusive sobre domínios diferentes, como o caso da indexação de documentos para máquinas de busca. No caso das APIs, apesar de o conteúdo estar bem formatado, o relacionamento entre as informações disponibilizadas é que pode não estar junto. Por exemplo, como as relações de amizade em redes sociais e os perfis dos usuários que fazem parte da relação de amizade. Além disso, o tamanho da base de dados que essas fontes cobrem é muito grande, e é possível que nem todos os dados des-

ses contextos sejam necessários para a análise desejada. Por esses motivos, nesses casos são utilizadas aplicações que realizam a coleta de forma organizada, com definição clara sobre quais informações serão coletadas, assim como, de que forma as mesmas serão armazenadas para que o usuário possa utilizá-las. Essa aplicações de coleta de dados são comumente chamados de coletores ou, em inglês, *crawlers*.

Coletores Web. Os coletores são comumento utilizados em diversas aplicações de áreas como recuperação de informação e redes complexas. Do ponto de vista da recuperação de informação, Baeza-Yates e Ribeiro-Neto [2011] enumeram seis tipos de de aplicações específicas desse contexto: busca na Web de forma geral, busca por tópicos específicos, arquivos de páginas da Web, caracterização da Web, análise de sites Web, e espelhos da Web. Do ponto de vista de redes complexas, os coletores podem ser utilizados em estudos sobre redes sociais on-line (*On-line Social Networks*, ou OSNs, em inglês) envolvendo a análise das relações entre seus usuários, por exemplo. Além disso, assim como a recuperação de informação, existem estudos de redes complexas que analisam o comportamento do grafo formado pelas páginas da Web.

Baeza-Yates e Ribeiro-Neto [2011] apresentam ainda três aspectos que os coletores exploram: atualização, qualidade e volume. O primeiro aspecto é tratado por coletores que estão constantemente funcionando e mantendo as bases atualizadas. O segundo presa por porções da Web que possuam melhor qualidade (que sua definição pode variar entre aplicações) nos dados disseminados, e que essa qualidade seja homogênea entre as páginas. Por fim, o volume trata da quantidade de páginas coletadas e pode sofrer redução em favor de se manter apenas dados atualizados, ou de melhor qualidade.

Desafios para Coleta. A coleta de dados possui ainda problemas específicos que devem observados como: tempo entre requisições, erro soft-404, identificação dos padrões de URL das páginas e extração dos dados do código fonte. O primeiro caso está presente tanto em páginas da Web quanto em APIs. No caso das páginas da Web, essa informação é disponibilizada pelos administradores dos sites em um arquivo chamado *robots.txt*. Esse arquivo formaliza as linhas gerais de boas práticas que os coletores devem seguir quando estão coletando suas páginas. Assim, nesses arquivos estão presentes as informações de tempo de requisição, bem como quais as seções do site o administrador permite que sejam coletadas, e quais gostaria que não fossem verificadas.

A segunda questão trata do erro soft-404. O erro 404 ocorre quando se tenta acessar uma página que não existe, e em geral essa resposta é dada diretamente pelo servidor. Existe uma pequena parcela de sites (29% dos links de páginas ausentes de acordo com [Baeza-Yates and Ribeiro-Neto, 2011]) que retornam uma página (com código de resposta 200 do servidor), porém essa página contém apenas a informação de que a página não foi encontrada. Esse problema aumenta a complexidade dos coletores e piora a eficiência do tempo de coleta, uma vez que somente é possível verificar se a página retornada é válida quando a mesma é aberta.

A identificação de padrões de URL permite que o coletor possa ser implementado utilizando a técnica de tentativa e erro. Em várias casos, os sites representam uma visualização dos dados presentes em bancos de dados. É comum utilizar uma padronização das URLs em função dos identificadores dos registros presentes nessas bases. Assim, sabendo

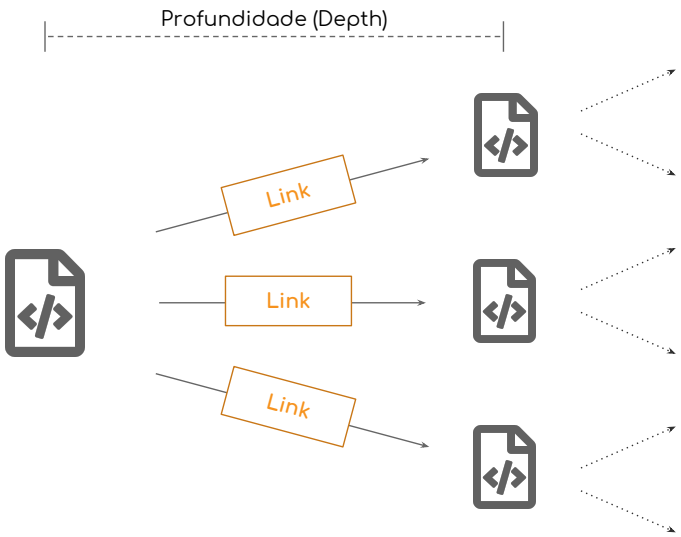

Figura 5.5: Caminhamento no grafo de páginas da Web

o número máximo de registros, é possível prever qual a estatura da chave desses registros (como registros sequenciais). Partindo dessa informação, o coletor fica responsável por verificar quais dessas chaves previstas levam a dados realmente existentes.

Finalmente a extração de código fonte é outra tarefa que depende do conhecimento de ferramentas de *parsing* ou caminhamento na árvore HTML das páginas Web. A inclusão da etapa de *parsing* durante a coleta reduz o volume de dados que serão armazenados, uma vez que todo o código HTML é removido.

Principais Técnicas de Coleta. Os coletores se baseiam em duas principais técnicas: caminhamento em grafo e amostragem probabilística. As técnicas baseadas em caminhamento em grafo realizam a coleta percorrendo o grafo de relações existente entre os objetos coletados. Esse tipo de coleta é muito útil quando não se tem informação sobre quantos objetos existem no conjunto de dados a ser coletado, assim como o endereço exato onde cada objeto se localiza ou qual identificador que pode ser utilizado para recuperá-lo. Sabe-se apenas que esses objetos estão conectados sobre algum tipo de relação.

Um exemplo de aplicação dessa técnica é a coleta de páginas da Web por máquinas de busca. Nesse caso, o tamanho do conjunto de dados é definido por todas as páginas da Web, e em geral não é possível saber antes da coleta o endereço Web de todas as suas páginas. Sabe-se que páginas da Web estão interligadas através de hiperlinks, que estabelecem uma relação de citação entre páginas. Assim é possível desenvolver um coletor que partindo de uma página inicial (comumente chamada de semente) seja capaz de encontrar novas páginas na Web através dos endereços citados por meio de hiperlinks. Dessa forma a coleta é realizada percorrendo o grafo formado pelas páginas da Web e seus hiperlinks, como o exemplo genérico ilustrado na Figura 5.5.

Existem três técnicas baseados em caminhamento em grafo para a coleta de dados: busca em largura (*Breadth-First Search*, BFS), *Snowball Sampling* e busca em profundidade (*Depth-First Search*, *DFS*). A BFS e a DFS são as técnicas mais simples de serem implementadas. Conforme o Algoritmo 1, a cada iteração, a BFS possui um conjunto de

Algoritmo 1 Algoritmo de busca em largura (BFS)

```
1: L conjunto de links da iteração i, inicialmente contém as sementes
2: LNext conjunto de links da iteração i+1, inicialmente vazia
3: AllLinks conjunto de todos os links obtidos
4: dMax altura máxima
5: d ← 0 altura atual
6: while d < dMax do
7:     for all l ∈ L do
8:         if l ∉ AllLinks then
9:             p ← collect(l)
10:            LNext.insert(p.links)
11:            AllLinks.insert(l)
12:    L ← LNext
13:    LNext ← ∅
14:    d ← d + 1
```

Algoritmo 2 Algoritmo de busca em profundidade (DFS)

```
1: S pilha de links, inicialmente contém apenas as sementes
2: AllLinks conjunto de todos os links obtidos
3: dMax altura máxima
4: d ← 0 profundidade atual
5: while d < dMax ∨ P = ∅ do
6:     l ← S.pop()
7:     if l ∉ AllLinks then
8:         p ← collect(l)
9:         S.push(p.links)
10:        AllLinks.insert(l)
11:    d ← d + 1
```

links de páginas que serão coletadas, sendo o conjunto inicial chamado de semente. A cada link, é verificado se a coleta já foi realizada para esse link; caso não tenha sido coletada, é então realizada sua coleta. Após a análise de cada página, obtem-se um conjunto de links que podem ser utilizados para acessar novas páginas (também chamadas de páginas vizinhas). Esse conjunto de links será utilizado em uma nova iteração do algoritmo.

O algoritmo de *Snowball Sampling* é bem similar ao BFS; a única diferença é que considera apenas uma amostra de k links presentes em cada página, limitando assim a largura máxima da árvore a k ramificações [Goodman, 1961]. No *Snowball* também é realizada a verificação se o novo conjunto de links já foi coletado.

Por fim, o algoritmo de busca em profundidade DFS, como o nome sugere, prioriza a exploração dos filhos das árvores antes de seus irmãos. O Algoritmo 2 apresenta o pseudo-código para DFS utilizando pilhas. Novamente, o algoritmo inicia com uma conjunto de links para páginas semente, que estão presentes na pilha S. Antes de coletar uma página, o algoritmo verifica se essa página já foi explorada. Os links da página coletada são então extraídos e inseridos no topo da pilha. Na próxima execução, o link a ser desempilhado é o primeiro filho da última página coletada. Dessa forma o algoritmo sempre

Figura 5.6: Principais etapas para integração de dados de múltiplas fontes utilizando ETL.

prioriza a coleta do primeiro filho de maior profundidade até que a profundidade máxima seja atingida. Após esse ponto, são desempilhados primeiramente os irmãos do primeiro filho de maior profundidade até que todos os ramos da árvore tenham sido explorados.

5.4. Integração dos Dados

Após a coleta de dados de múltiplas fontes, o próximo passo é *integrar*. Em resumo, a integração de dados consiste em combinar dados de diferentes fontes para obter informações valiosas. Tal tarefa tem sido foco de muitos estudos devido à ampla quantidade de dados heterogêneos disponíveis na Web [Doan et al., 2018, Freitas et al., 2017, Golshan et al., 2017]. A integração é importante para permitir que usuários tenham uma visão unificada de dados heterogêneos e consultem facilmente diferentes informações sobre os mesmos [Bouzeghoub et al., 2002]. Além disso, essa integração permite considerar várias definições/visualizações sobre um objeto. Por exemplo, Ma et al. [2017] usam dados de múltiplas fontes para identificar diversos efeitos colaterais de drogas; e Freitas et al. [2017] propõem um modelo baseado em ontologias e ligação de dados (*Linked Data*) para integrar conjuntos de dados de forma a permitir o cálculo da probabilidade do risco de óbito materno e infantil. Assim, os usuários podem descobrir melhor o conhecimento a partir de múltiplos dados e, então, essa integração pode fornecer suporte para a tomada de decisões, entre várias outras aplicações.

No entanto, existem diferentes desafios na integração de dados de múltiplas fontes, principalmente porque a maioria dos dados da Web são heterogêneos, não estruturados ou semi-estruturados. Além disso, os dados de várias fontes da Web possuem modelos distintos, diferentes representações de objetos do mundo real e nem sempre são confiáveis. Outro desafio relevante é manter o esquema de dados consistente após a integração, pois a cada alteração na fonte de dados é necessário verificar se as mudanças precisam ser propagadas pelo esquema, se novas consultas aos dados precisam ser elaboradas ou se o esquema precisa ser reescrito [Bouzeghoub et al., 2002, Laender et al., 2009].

Uma abordagem bastante utilizada e comum para integração de dados é a ETL: extração, transformação/limpeza e carregamento (do inglês *Extract*, *Transform* e *Load*) [Azeroual et al., 2018, Bansal, 2014], sumarizada na Figura 5.6. Considere diferentes conjuntos de dados, extraídos de fontes distintas, fornecidos como entrada para realização de um determinado estudo, os quais precisam ser integrados. Para tal, existem três etapas principais: (*i*) *extração* representa a aquisição de dados de múltiplas fontes; (*ii*) *transformação e limpeza* referem-se à padronização e limpeza dos dados, sendo nem sempre obrigatórias, mas boas práticas; e (*iii*) *carregamento* refere-se à inserção dos dados em

Tabela 5.1: Exemplos de abordagens para integração de dados e respectivas aplicações.

Abordagem	Aplicação
Sistema de mediação	Fornece uma visão única dos dados em formatos distintos para o usuário
Processamento de linguagem natural	Processa dados em formato de texto para permitir a padronização e posterior integração de tais dados
Abordagem Bayesiana	Lida bem com dados incompletos e inconsistentes de modo a garantir que a base resultante seja confiável

um sistema de organização incluindo, por exemplo, planilhas (apesar de não serem muito recomendadas para armazenamento e gerenciamento de grandes volumes de dados), armazém de dados, sistemas de gerenciamento de bancos de dados, etc.

É importante enfatizar que a ETL e a integração de dados são conceitos distintos. ETL são ferramentas de software, enquanto integração de dados é uma arquitetura[16]. Assim, a ETL pode ser utilizada como parte da etapa de integração de dados, mas não necessariamente representar toda a integração. Além da ETL, a tarefa de integração de dados também inclui modelagem de dados, criação de perfil dos dados, processamento de dados estruturados e não-estruturados, integração em tempo real, governança de dados, entre outras [Doan et al., 2018]. Ou seja, a área de integração de dados é bastante ampla e abrangente, e apenas as principais estratégias são abordadas aqui.

Na prática, pode-se coletar dados de cada fonte e armazená-los separadamente para posterior integração; ou armazenar todos os dados em um único local de forma integrada à medida que cada coleta de dados (por meio de rastreamento) é realizada. Há muitas vantagens e desvantagens das duas estratégias e diferentes formas de armazenamento. Especificamente sobre a primeira abordagem, pode ocorrer o armazenamento de diferentes volumes de dados em um único repositório chamado Lago de Dados (ou *Data Lake*). Tal armazenamento permite que os usuários façam diferentes consultas aos dados.

Entre muitas estratégias para integração de dados de múltiplas fontes, citamos: sistema de mediação [Bouzeghoub et al., 2002] que é uma abordagem para mapear diferentes fontes de dados em um esquema global; processamento de linguagem natural [Ma et al., 2017] que permite converter texto de diferentes fontes em códigos de identificadores exclusivos; e abordagem bayesiana [Wang et al., 2017, Zhao et al., 2012] que fornece uma maneira de integrar informações de confiabilidade em vários níveis.

Uma análise da definição e aplicação dessas abordagens revela que existem diferentes situações em que cada uma delas pode ser melhor utilizada para integrar dados, conforme mostra a Tabela 5.1. Ou seja, é necessário avaliar prós e cons para uma escolha mais acertada. Além disso, note que tais abordagens podem ser utilizadas de forma combinada. Por exemplo, o processamento de linguagem natural pode ser utilizado para identificar características relevantes no texto e possibilitar o armazenamento de tais dados de forma padronizada. Em seguida, a abordagem Bayesiana pode ser aplicada para inte-

[16]Diferença entre ETL e integração de dados: `https://www.passionned.com/is-data-integration-becoming-the-new-etl/`. Acessado em 20 de agosto de 2018.

Figura 5.7: Exemplo de integração de dados horizontal: as duas tabelas superiores representam clientes e os produtos comprados por eles, respectivamente. A tabela inferior é a combinação dessas duas tabelas. Observe que são perdidas informações individuais dos clientes e duplicatas são inseridas (exemplo adaptado de [Berthold et al., 2010]).

id	**nome**	**sobrenome**	**gênero**
c2	Joana	Oliveira	F
c5	Isis	Lima	F
c7	Rafael	Silva	M
...	...	...	...

cliente_id	**item_id**	**preço**
c2	i254	12,50
c5	i4245	1,99
c5	i32123	1,29
c5	i254	12,50
c5	i21435	5,99
c7	i254	12,50
...	...	...

item_id	**preço**	**nome**	**sobrenome**	**gênero**
i254	12,50	Joana	Oliveira	F
i4245	1,99	Isis	Lima	F
i32123	1,29	Isis	Lima	F
i254	12,50	Isis	Lima	F
i21435	5,99	Isis	Lima	F
i254	12,50	Rafael	Silva	M
...	...	...	...	...

grar os dados de forma a garantir a confiabilidade. Por fim, o sistema mediador pode ser utilizado para garantir uma visão única dos dados.

Outras estratégias usuais de integração de dados são específicas para armazenamento em bancos de dados relacionais [Berthold et al., 2010]. Um primeiro tipo de estratégia aborda o problema em uma perspectiva vertical, na qual as tabelas com essencialmente as mesmas informações são concatenadas. Por exemplo, unir uma tabela que representa fornecedores de equipamentos eletrônicos com uma que armazena fornecedores de baterias pode resultar em apenas uma tabela chamada *fornecedores* com as mesmas colunas das outras duas. Já um segundo esquema aborda o problema em uma perspectiva horizontal, na qual diferentes tipos de dados são combinados com o objetivo de enriquecer uma tabela existente, conforme mostra a Figura 5.7. Em geral, ambas as estratégias são simples, mas não isentas de questões relevantes. Por exemplo, lidar com duplicatas é necessário na estratégia vertical, enquanto o tratamento de excesso de representação e de explosão de dados é necessário na horizontal.

Esses dois últimos problemas estão presentes na Figura 5.7. Por exemplo, se a tabela resultante fosse utilizada para determinar o gênero dos compradores, o resultado seria um alto número de compradores do sexo feminino devido ao excesso de representação. Nesse caso, o ideal é não fazer consulta a tal tabela para buscar informações referentes aos compradores. Em relação à explosão de dados, também é um problema visível, pois as duplicatas que claramente não são necessárias para manter as informações essenciais.

Assim, o ideal é encontrar uma configuração do banco de dados para mantê-lo normalizado. A junção mostrada nessa figura é apenas um exemplo simples do que é possível realizar em um sistema real.

A mineração de dados relacionais (ou mineração em bancos de dados multirelacionais) consiste em encontrar informação em múltiplas tabelas [Berthold et al., 2010, Flach, 2001]. Nesse contexto, a integração de dados consiste em uma forma de transformar um problema difícil com múltiplas tabelas em um possível de lidar com métodos já existentes para uma única tabela. Entretanto, tais métodos, em geral, são bem especializados e consideram que existe uma única tabela bem formada com todos os dados relevantes. Assim, essa é uma simplificação, e a integração de dados requer muito mais tempo e esforço do que a própria análise dos dados.

5.5. Pré-Processamento dos Dados

Dados Web fornecem recursos inestimáveis para muitos pesquisadores e desenvolvedores. No entanto, esses dados podem vir com muitos problemas, especialmente quando coletados de várias fontes da Web, incluindo: valores ausentes, dados falsos (veracidade dos dados), dados duplicados, redução de dados e falta de padronização. Tais problemas geralmente são resolvidos na etapa de pré-processamento de dados, que requer cerca de 80% do tempo de cientistas de dados (como já foi apontado por muitos estudos como o de Tyagi et al. [2010]). Além disso, a representação dos dados coletados pode não ser ideal para o processamento dos algoritmos e análises sobre os mesmos. Dessa forma, é necessário converter os dados obtidos para representações adequadas.

As estratégias de pré-processamento de dados variam de acordo com o contexto e o tipo de dados. Para cada tipo de problema há algumas sugestões de solução; entretanto, a aplicação da solução aos dados deve ser analisada para cada caso. Por exemplo, no caso de valores ausentes é possível escolher completar os valores buscando de alguma outra fonte (quando disponível) ou eliminando os registros que não têm a informação completa. A seguir, discutimos cada um dos problemas apontados, apresentando sugestões de solução a partir de exemplos disponíveis na literatura.

5.5.1. Valores Ausentes

Os valores ausentes (*missing values*) ocorrem quando nenhum valor é armazenado para uma variável (ou um atributo) da base de dados. Os valores ausentes podem ocorrer em apenas alguns registros (quando existem registros que possuem o valor e registros que não possuem, por exemplo), ou podem ser dados importantes para a análise que não estão disponíveis para nenhum registro da base de dados. A questão, então, é se é possível extrair os dados de alguma outra fonte ou o banco de dados deve ser limpo para remover tais valores [Alves et al., 2016]. Em alguns casos, é possível buscar outra fonte de dados que tenha a informação específica disponível e que seja de fácil integração com a base atual. Avaliar a viabilidade da integração dos dados é importante, uma vez que fontes diferentes podem não apresentar um elo de ligação dos registros para que eles sejam relacionados. Como uma terceira opção, pode ainda ser encontrado um valor padrão (*default*) para substituir tais dados faltantes, quando necessário.

Um exemplo de *missing values* foi tratado por Alves et al. [2016] no trabalho re-

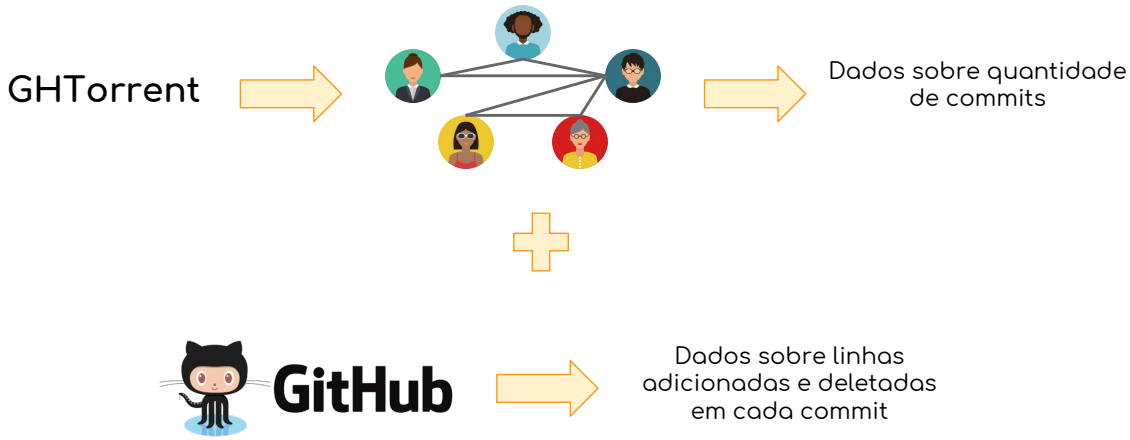

Figura 5.8: Correção de valores ausentes na rede de colaboração do GitHub [Alves et al., 2016]

lacionado ao GitHub. O trabalho apresenta uma métrica para a força de interação entre desenvolvedores do GitHub. Para isso, a partir de uma coleta pelo GHTorrent[17] foi construída a rede de colaboração. Entretanto, para uma métrica específica, observou-se que avaliar a quantidade de commits de um par de desenvolvedores não era suficiente e não estava claro o impacto desses commits. Para detalhar as alterações feitas efetivamente no código, foi necessário avaliar o número de linhas inseridas e excluídas em cada um desses commits. Desta forma, os dados de linhas adicionadas e retiradas foram coletados através da API do GitHub e agregados à base de dados, como resumido na Figura 5.8.

5.5.2. Veracidade dos Dados

O problema de veracidade dos dados se refere à avaliação e melhoria da precisão dos dados [Geerts et al., 2018]. Normalmente, a veracidade dos dados é comprometida pela presença de viés, anormalidades e ruído nos dados, que frequentemente estão presentes nos dados coletados na Web. Ao criar e manter bases de conhecimento, deve-se validar os dados e fornecer suas fontes a fim de garantir a exatidão e rastreabilidade das informações.

Como a veracidade é uma questão importante que afeta diretamente as informações e os conhecimentos extraídos dos dados, existem diferentes estratégias para melhorar ou garantir isso. Geerts et al. [2018] propõem o modelo DeFacto (*Deep Fact Validation*) que busca realizar a validação de fatos encontrando fontes confiáveis para os mesmos na Web. Para alcançar esse objetivo, o DeFacto fornece ao usuário trechos relevantes de páginas na Web, informações adicionais úteis e uma pontuação para a confiança da corretude do dado de entrada.

5.5.3. Remoção de Dados Duplicados

Dados duplicados aparecem em muitos contextos, especialmente ao coletar dados online. Por exemplo, nomes de autores duplicados ocorrem entre fontes de bibliotecas distintas ou até dentro da mesma fonte.

[17]GHTorrent: projeto que reúne um conjunto de dumps do GitHub (`http://ghtorrent.org/`)

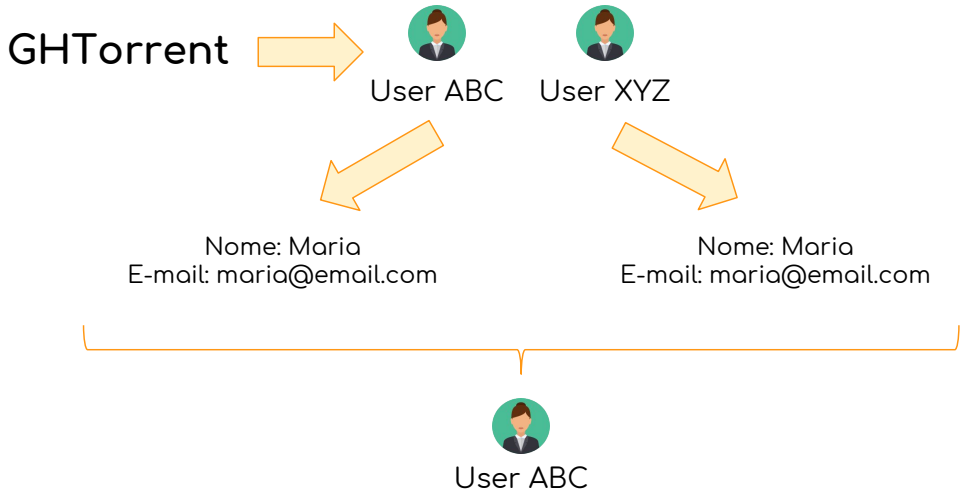

Figura 5.9: Deduplicação de nomes de usuários artificiais presentes no GitHub [Vasilescu et al., 2015]

A duplicidade de dados pode ocorrer de forma mais simples quando existem mais cópias dos mesmos dados em uma base. Esse é um caso mais fácil de ser resolvido, pois basta identificar valores idênticos e removê-los. Já o caso de registros que não são completamente idênticos é mais difícil, pois identificá-los requer entender se realmente aqueles registros dizem respeito à mesma informação. Por exemplo em nomes próprios com e sem abreviação ou omissão de algum dos sobrenomes: Mirella M. Moro, Mirella Moro e Mirella Moura Moro. Nesses casos, é possível usar recursos de contexto para identificar duplicatas, como nomes do usuário ou e-mail [Vasilescu et al., 2015] ou mesmo outras características importantes para o contexto da base [de Souza Silva et al., 2018].

Especificamente, o trabalho de Vasilescu et al. [2015] utiliza dados de desenvolvedores do GitHub coletados do GHTorrent. Há uma peculiaridade nesses dados em relação aos *usernames* dos usuários: quando por algum motivo específico o *username* não pode ser recuperado, o GHTorrent cria um usuário artificial com um *username* aleatório. Entretanto, usuários com *username* diferentes poderiam ser o mesmo usuário na realidade. Dessa forma, os autores realizaram uma deduplicação que impactou boa parte da base de dados utilizando informações como nome do usuário e e-mail de cadastro para identificar as duplicatas entre os usuários artificiais, como ilustra a Figura 5.9.

Enquanto isso, de Souza Silva et al. [2018] apresentam estratégias para melhorar o processo de remoção de duplicatas. A Figura 5.10 apresenta o processo para identificação de duplicadas em um banco de dados. Inicialmente, é feita a *indexação* dos registros a fim de realizar uma divisão dos mesmos em grupos iniciais. Em seguida, são realizadas *comparações* entre registros de mesmo grupo e, finalmente, é feita a *classificação* para identificar os dados em duplicidade. Os autores identificaram que na primeira etapa, de indexação, a escolha do atributo utilizado é de grande impacto para todo o processo. Então, propuseram uma melhor forma de escolher este atributo considerando aspectos como duplicidade, distinção, densidade e repetição.

Note que para ambos os exemplos apresentados, as estratégias de deduplicação

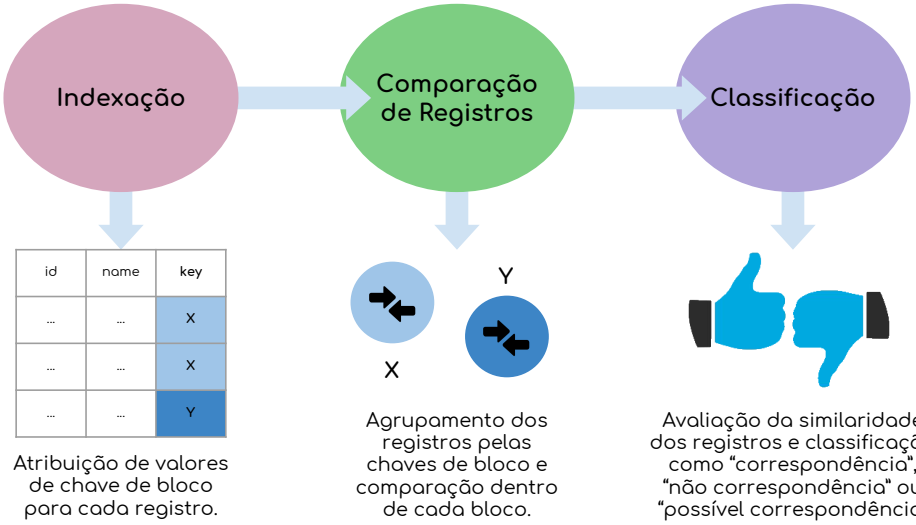

Figura 5.10: Processo de deduplicação de dados [de Souza Silva et al., 2018]

escolhidas estão diretamente relacionadas ao contexto em que os dados estão inseridos. Note que a identificação de duplicatas numa base de dados quase sempre depende de quais informações estão disponíveis para a correta classificação. Também é possível encontrar dados muito parecidos, mas que não são duplicatas, como por exemplo o nome "José P. Amaral", que pertence a pessoas distintas: "José Pedro Amaral"e "José Pereira Amaral". Por este motivo, a tarefa de deduplicação é delicada e requer um alto nível de precisão.

5.5.4. Redução de Dados

A redução de dados aborda o problema de minimizar a quantidade de dados que serão armazenados no conjunto de dados. Dependendo do volume de dados, armazená-los em um banco comum ou processá-los é um problema relacionado tanto a espaço quanto a tempo. As soluções geralmente utilizadas incluem: minimizar os dados ou armazená-los considerando outra forma de armazenamento [Liu and Ram, 2018], ou extrair amostras da base completa sem perder a generalidade das análises [Batista et al., 2017a].

Por exemplo, Batista et al. [2017a] utilizam ambas as estratégias visando a redução de dados armazenados. Especificamente, ao analisar toda a rede do GitHub, foram encontrados alguns desafios em relação ao volume de dados e algumas especificidades do contexto. Dessa forma, optou-se por dividir a rede de colaboração por linguagens de programação, selecionando as linguagens com maior quantidade de repositórios, conforme apresentado na Figura 5.11. Foram escolhidas as linguagens JavaScript, Java e Ruby. A partir das análises, os autores perceberam que os comportamentos dos desenvolvedores dentro da rede de cada linguagem é semelhante, o que permitiu que os resultados fossem obtidos dessa maneira. Ainda assim, o volume de dados para armazenamento era bastante expressivo. Dessa forma, foi escolhido armazenar os dados utilizando o MongoDB[18] ao invés do MySQL, utilizado anteriormente. A compressão de dados do MongoDB e a nova modelagem dos dados permitiram armazenar os mesmos dados com um espaço em disco menor e melhorar alguns filtros e consultas realizados na base.

[18]MongoDB: `https://www.mongodb.com/`

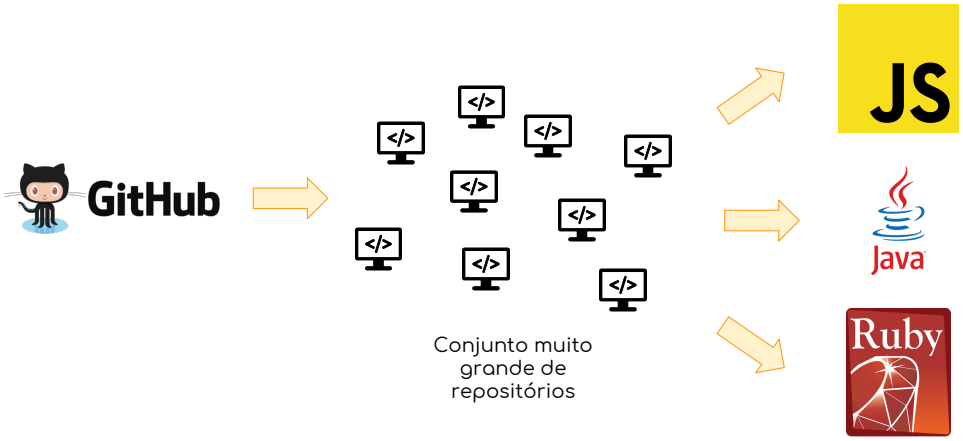

Figura 5.11: Escolha de repositórios por linguagem de programação

5.5.5. Ausência de Padronização

A padronização de dados é o processo de reestruturação de dados em um formato comum. Ter dados coletados de várias fontes da Web pode gerar um conjunto de dados que não é apenas heterogêneo, mas também em formato diferente. Então, iniciar a pesquisa real requer primeiro padronizar todos os dados.

Por mais que os dados coletados sejam suficientes para as análises que serão realizadas, se os mesmos estiverem desorganizados ou sem um padrão, eles dificultam a análise ao invés de auxiliar. Desta forma, é importante avaliar a base de dados no sentido de encontrar a forma que esses dados devem ser organizados para gerar os resultados desejados. Em alguns casos, propor uma nova modelagem para os dados coletados é uma forma de organizá-los formalmente e padronizá-los como um todo [Batista et al., 2017a].

5.6. Aplicações Reais

Existem diversos estudos que combinam fontes de dados para diferentes propósitos. A maioria deles depende da rica informação disponível apenas pela coleta de dados de fontes distintas. Esta seção cobre uma pequena fração de tais estudos em diferentes domínios.

5.6.1. Estimativa da Qualidade em Documentos Colaborativos

Dalip et al. [2017] propuseram uma abordagem para estimativa da qualidade de conteúdo colaborativo – como enciclopédias colaborativas e fórum de perguntas e respostas. Assim, foi necessário identificar quais dimensões de qualidade são importantes nesta tarefa, por exemplo, organização, legibilidade, importância e maturidade do texto. Baseado em trabalhos anteriores sobre qualidade de informação e nas orientações de publicação fornecidos por repositórios colaborativos (como StackOverflow[19] e Wikipedia[20]) foi adaptada a lista de dimensões de qualidade apresentada por Tejay et al. [2006], proposta originalmente no domínio de dados estruturados.

[19]StackOverflow help: `http://meta.stackoverflow.com/help/how-to-answer`
[20]Wikipedia Assessing Articles: `https://en.wikipedia.org/wiki/Wikipedia:Assessing_articles`

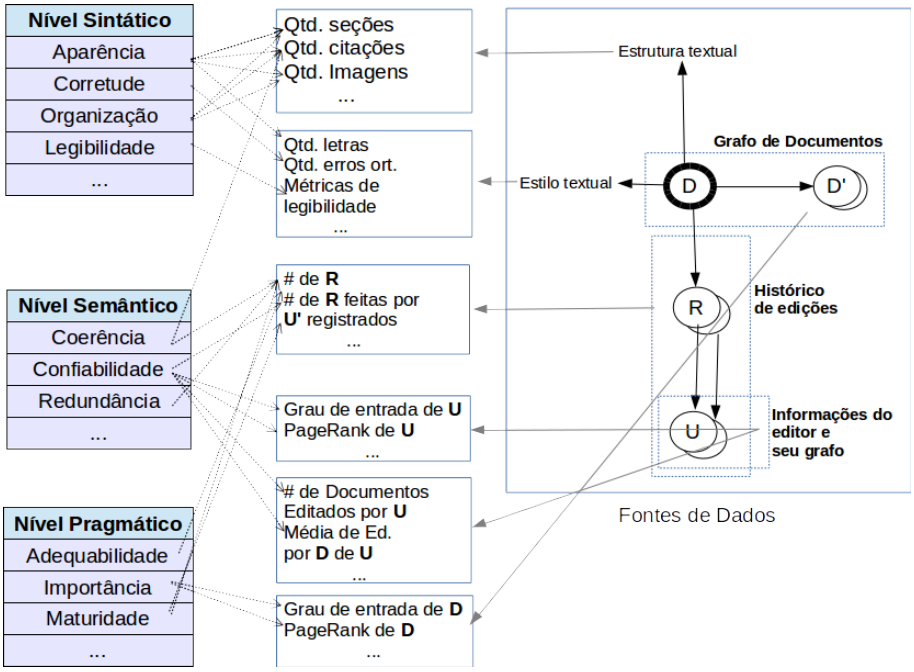

Figura 5.12: Uso de dados de múltiplas fontes para prever a qualidade de conteúdo em enciclopédias colaborativas (adaptado de Dalip et al. [2017])

Para isso, Tejay et al. [2006] propuseram um arcabouço conceitual visando agrupar as dimensões de qualidade em níveis semióticos: sintático, semântico e pragmático. De acordo com os autores, a semiótica pode ajudar a organizar as dimensões, pois ela estuda como um signo é criado, processado e usado. No presente contexto, signo é o próprio dado. Dimensões sintáticas são relacionadas em como o texto é apresentado. Dimensões semânticas relacionam o conteúdo do texto com o seu significado. Dimensões pragmáticas são relacionadas com a intenção do autor/leitor em um determinado contexto.

Posteriormente, foram definidos os indicadores de qualidade. Um indicador é um valor contendo uma medida estatística correlacionada com uma dimensão de qualidade. Por exemplo, o número de caracteres em um texto (indicador) pode ser correlacionado com a concisão do mesmo (dimensão de qualidade). A Figura 5.12 demonstra uma visão geral de como dimensões, indicadores e fontes estão relacionados no contexto de enciclopédias colaborativas. Neste domínio, Dalip et al. [2017] combinaram indicadores de qualidade do texto, do histórico de revisões e do grafo de links entre os artigos.

Para a Wikipédia, tais indicadores foram coletados de diversas fontes dessa enciclopédia colaborativa. Então, para cada fonte, precisou-se obter os dados de uma forma distinta. Inicialmente, extraiu-se a edição atual de todos os artigos da Wikipédia para a criação de uma amostra a partir deles. Para isso, foi feito o download do XML 'enwiki-20080101-pages-meta-current.xml.bz2' que estava disponível em `https://dumps.wikimedia.org/enwiki`[21]. A Wikipédia avalia manualmente seus artigos per meio

[21] A partir deste link `https://dumps.wikimedia.org/backup-index.html` pode-se obter os últimos *dumps* de diversas Wikis providas pela Wikimedia. Os dados extraídos deste estudo pode ser obtido em: `http://www.lbd.dcc.ufmg.br/lbd/collections/wiki-quality`

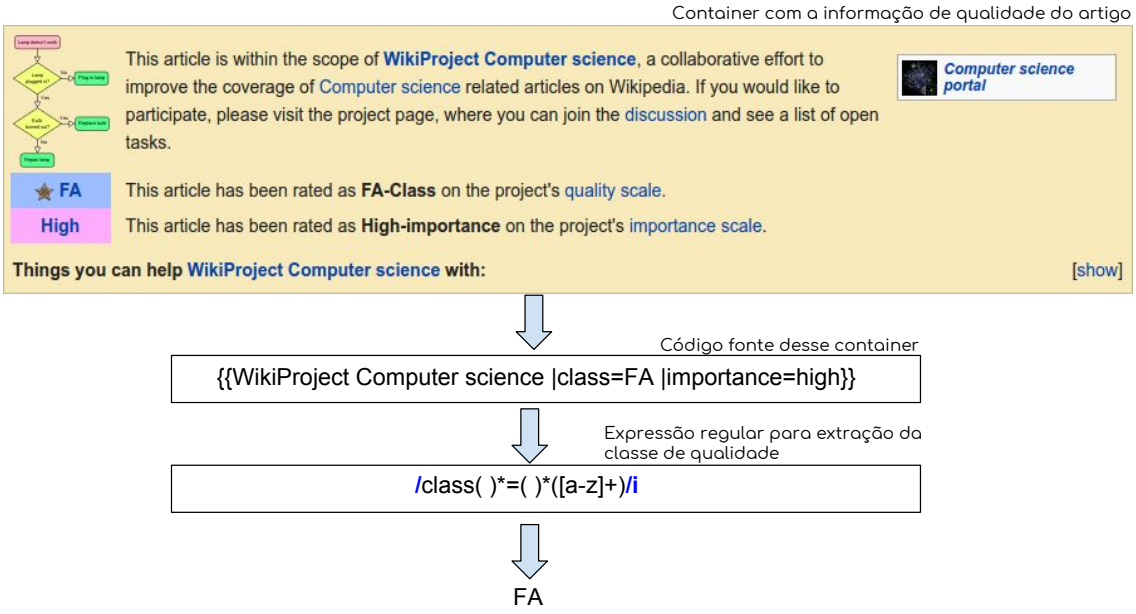

Figura 5.13: Exemplo de extração da classe de qualidade do artigo *Binary Search*.

de classes de qualidade. Assim, por meio deste link, foi coletado também a página de discussão de cada artigo com o objetivo de extrair a qualidade do mesmo. Um exemplo do funcionamento da extração dessa classe é ilustrado na Figura 5.13. A extração foi feita por meio de expressão regular para obter a sigla correspondente à classe de qualidade.

A partir desse mesmo link, extraiu-se também o grafo por meio do arquivo com sufixo 'pagelinks.sql.gz'. A partir desta página de *dumps*, também é possível extrair um XML contendo todas as edições de cada artigo da Wikipédia e, assim, pode-se extrair os indicadores de histórico de revisões (arquivo sufixo 'pages-meta-history.xml.bz2'). Porém, como o tamanho do XML é grande, optou-se por criar a amostra e usar a API da Wikipédia para coletar, para cada artigo dessa amostra, todas as suas edições[22].

Nessa API, é possível obter o XML completo de revisões por meio de requisições. Esse XML possui, para cada página, o texto de suas revisões e metadados (título, autor, data de revisão etc.). Como há um limite de quantidade de revisões por requisição, para cada página, deve-se extrair uma quantidade de revisões por vez. O Algoritmo 3 demonstra como foi feito a coleta por Dalip et al. [2017]. Nesse algoritmo, é necessário determinar a data l para que sejam coletadas as revisões até essa data e o conjunto de páginas P a serem coletadas. Assim, para cada página, são feitas requisições para extrair o conjunto de revisões (função *request_wiki*). Para cada revisão, é possível extrair seus dados para processamento dos indicadores (função $process\_features$) e obter seus metadados como, por exemplo, a data da revisão (função $get\_timestamp$)[23].

Após extrair todos os indicadores e a classe de qualidade correspondente de cada artigo, é possível utilizar a abordagem proposta por Dalip et al. [2017] para combinar tais fontes e, assim, estimar a qualidade de documentos colaborativos. A descrição completa

[22] `https://en.wikipedia.org/wiki/Special:Export`
[23] Exemplo da implementação: `https://github.com/lab-csx-ufmg/webmedia2018`

Algoritmo 3 Coleta de dados por meio da API da Wikipédia

Require: Conjunto P de páginas a serem coletadas (representadas pelo seu título)
Require: Data l representa a data limite de uma revisão a ser coletada. Ou seja, são coletadas todas as revisões de uma página até a data l.
Considere R o conjunto de revisões de uma página $p \in P$
Considere d_p a data da mais antiga revisão coletada para uma determinada pagina p
for all $p \in P$ **do**
    $has\_revision \leftarrow True$
    $d_p \leftarrow l$
    while $has\_revision = True$ **do**
        $R \leftarrow request\_wiki(p, d_p)$
        for all $r \in R$ **do**
            $process\_revision(r)$
            $d_p \leftarrow get\_timestamp(r)$
        if $|R| = 0$ **then**
            $has\_revision \leftarrow False$

da abordagem per se foge do escopo deste capítulo.

5.6.2. Estimativa da Força de Relacionamentos no GitHub

Social Coding é uma abordagem de desenvolvimento de software colaborativa para desenvolvedores, que incentiva a discussão e compartilhamento de ideias e conhecimento [Dabbish et al., 2012]. Essa metodologia tem alterado a forma de desenvolvimento de software, pois colaboradores geograficamente distantes podem acessar plataformas colaborativas remotamente. Alguns exemplos de sites que permitem o *Social Coding* são o Google Code[24] e o GitHub[25]. Os dados disponíveis nesses sites permitem definir redes sociais que conectam desenvolvedores a partir das suas atividades colaborativas em repositórios de software, formando uma grande rede implícita de codificação social.

Um tipo específico de aspecto social nessa rede é a força da interação entre desenvolvedores, ou a força do relacionamento no contexto de codificação social [Dabbish et al., 2012]. Nesse contexto, a força dos relacionamentos tem sido investigada no GitHub para analisar a produtividade de desenvolvedores em projetos [Casalnuovo et al, 2015], prever a colaboração entre desenvolvedores [Bartusiak et al., 2016] e investigar a aceitação de solicitações *pull requests* [Tsay et al., 2014].

Trabalhos que exploram extensivamente diferentes aspectos da força dos relacionamentos sociais entre desenvolvedores foram propostos em nosso grupo de pesquisa por Alves et al. [2016], Batista et al. [2017b] e Oliveira et al. [2018]. A Figura 5.14 apresenta uma visão esquemática do processo adotado nos trabalhos propostos, desde a fase inicial de coleta (GHTorrent e da API do GitHub), seguida da modelagem da rede e dos relacionamentos até, por fim, a fase das análises considerando propriedades semânticas e topológicas para analisar relacionamentos de codificação social.

Tais estudos foram realizados através da integração de dados coletados de base de

[24]Google Code: `code.google.com`
[25]GitHub: `github.com`

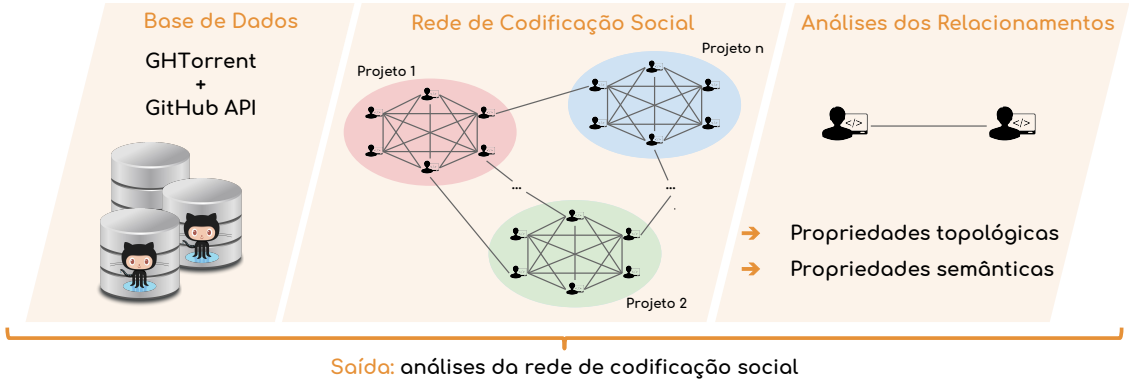

Figura 5.14: Visão geral do processo de coleta, integração e análise dos dados do GitHub

dados disponível na Web, chamada GHTorrent[26], e dados coletados por meio da API do GitHub. O GHTorrent fornece dados sobre repositórios, projetos, desenvolvedores, commits, pull requests, entre outros. Entretanto, Alves et al. [2016] e Batista et al. [2017b] precisam do número de linhas dos commits em uma das métricas para a força dos relacionamentos; e Oliveira et al. [2018] necessitam da posição de desenvolvedores em ranks gerados pelo GitHub. Tais dados foram obtidos por meio da API do GitHub ou por coleta diretamente ao Git Awards[27].

Para a coleta de informações adicionais, foram desenvolvidos coletores para recuperar dados de fontes diferentes, além dos dados do GHTorrent. A necessidade do número de linhas adicionadas e apagadas em cada commit teve objetivo de detalhar uma das métricas da rede que considera a quantidade de commits realizada por um par de desenvolvedores. A partir das análises, notou-se diferentes comportamentos entre desenvolvedores: alguns realizavam grandes commits em períodos maiores de tempo e outros realizavam vários commits durante o dia com pequenas alterações. Dessa forma, é importante saber o real impacto de cada commit ao repositório que ele se enquadra analisando o número de linhas de código que foram efetivamente inseridas ou excluídas pelo mesmo. Já as informações de rankings extraídas do Git Awards são importantes no sentido de utilizar um baseline para comparação da classificação de desenvolvedores na rede.

A base de dados, que está disponível online[28], foi capaz de reunir diversos dados sobre os repositórios, os usuários e, principalmente, as colaborações no GitHub considerando o tempo de contribuição entre usuários. A partir das coletas realizadas, foram propostas bases de dados com diferentes modelagens e versões [Batista et al., 2017b,a, Oliveira et al., 2018], sendo esta base atualizada incrementalmente.

A integração de dados foi um grande desafio no processo de construção da base. Por terem datas de referência diferentes, as bases possuíam uma série de divergências entre si. Projetos existentes no GHTorrent podiam não estar mais disponíveis para coleta dos números de linhas dos commits (os motivos principais são repositórios que foram excluídos ou transformados em repositórios privados, impossibilitando a coleta por

[26]GHTorrent: `http://ghtorrent.org/`
[27]Git Awards: `http://git-awards.com/`
[28]Dataset GitHub: `https://homepages.dcc.ufmg.br/~mirella/projs/apoena`

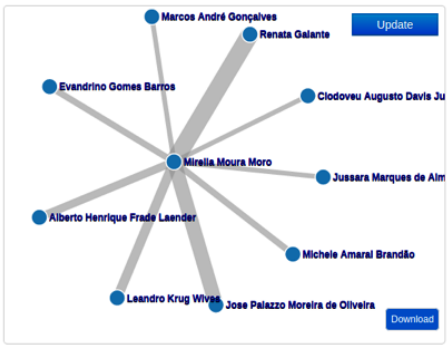

Figura 5.15: Rede centrada no pesquisador – adaptada de Brandão et al. [2018].

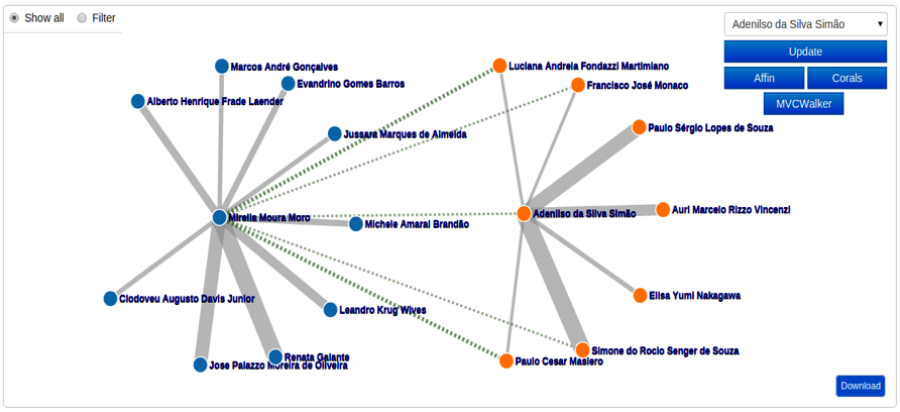

Figura 5.16: As linhas verdes representam colaborações recomendadas: quanto mais intenso, mais foi recomendado pelo algoritmo. As recomendações são geradas clicando em uma das opções com o nome dos algoritmos (figura extraída de Brandão et al. [2018]). A geração das recomendações e tal visualização só foi possível devido a integração de dados de fontes distintas.

meio de *crawlers*). No contexto dos rankings, alguns usuários que constavam no ranking, não necessariamente estavam também na base inicial, prejudicando a análise da sua classificação. Em sua maioria, os dados faltantes precisaram ser ignorados para que não impossibilitassem a extração de resultados e conhecimento a partir da base consolidada.

5.6.3. Caracterização e Visualização de Pesquisadores

No contexto de colaborações científicas, Brandão et al. [2018] combinam dados da DBLP[29] e dados coletados da biblioteca digital da ACM[30] para propor visualizações mais completas para pesquisadores, tais como rede centrada no pesquisador (Figura 5.15), recomendações de colaborações (Figura 5.16), rede de coautoria global e métricas de redes complexas. Dessa forma, foi proposta a ferramenta CNARe[31] que objetiva auxiliar pesquisadores a escolher colaboradores através de recomendações automáticas, visualizar recomendações, comparar os resultados de diferentes algoritmos de recomendação e ana-

[29]DBLP: `dblp.uni-trier.de`
[30]ACM digital library: `dl.acm.org`
[31]CNARe: `http://homepages.dcc.ufmg.br/~mirella/Tools/CNARe/`

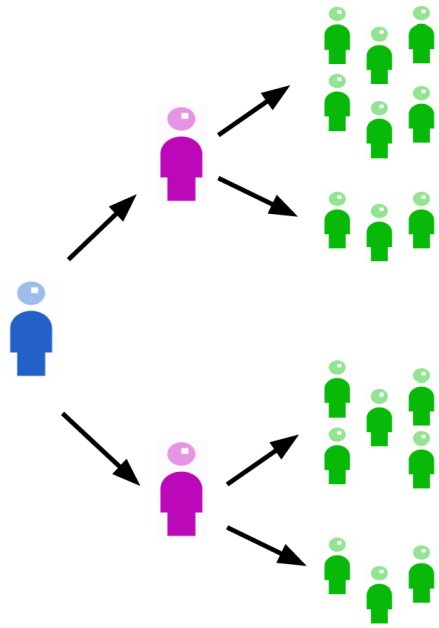

Figura 5.17: Exemplo de coleta utilizando a estratégia bola de neve.

lisar o impacto dos pesquisadores recomendados em sua rede atual.

O banco de dados inicial da ferramenta CNARe incluía apenas dados de publicações da área de Ciência da Computação, mas para montar as redes sociais e gerar recomendações é necessário dados sobre pesquisadores. Assim, informações disponíveis na página dos pesquisadores na biblioteca digital da ACM foram coletadas utilizando a estratégia de amostragem bola de neve (*snowball sampling*) [Goodman, 1961]. A Figura 5.17 mostra um exemplo de como novos dados de pesquisadores são coletados utilizando tal estratégia. A partir de um pesquisador semente conhecido (ou um conjunto de sementes), os coautores de tal pesquisador são coletados. Em seguida, os coautores desses coautores também são coletados até que a quantidade de dados disponível seja suficiente para análise. Em Brandão et al. [2018], o limite de coleta foi atingir os top 100 pesquisadores com maior quantidade de publicações, pois os algoritmos de recomendações implementados na CNARe não possuem bom desempenho para grandes volumes de dados.

A biblioteca da ACM foi escolhida por apresentar a área de cada publicação de acordo com o *ACM Classification System*. A página de cada pesquisador tem uma lista de publicações, na qual cada publicação tem o DOI (Digital Object Identifier System), a lista de coautores, a data e o local da publicação. A partir do DOI, a URL especificada é acessada para obter a área de pesquisa de cada publicação e informações sobre cada coautor (pesquisador): instituição, número total de publicações e nome do pesquisador (já que a lista de coautores fornece o nome no formato de citação). Após inserir os coautores em um banco de dados diferente do CNARe, uma nova consulta é executada para obter o coautor com o maior número de publicações cuja página ainda não foi visitada. Então, o processo de coleta começa novamente a partir da página deste pesquisador.

CNARe utiliza um banco de dados relacional, e o esquema do banco de dados

Tabela 5.2: Um exemplo de conjuntos de dados que representam os efeitos colaterais do Thyroxine, adaptado de Ma et al. [2017].

FAERS		**HealthBoards**	
ID Usuário	**Efeito colateral**	**ID Usuário**	**Efeito colateral**
110696642	Disfagia	2918	Enxaqueca
108294651	Disfagia	3171	Disfagia
108294651	Náusea	3171	Náusea
108294651	Mudança de humor	3171	Anemia
108325471	Disfagia	6871	Mudança de humor
108325471	Náusea	6871	Desidratação
108325471	Enxaqueca	27417	Desidratação

inicial foi alterado para receber os dados coletados da ACM. Antes de realizar a inserção dos dados no banco de dados do CNARe, foi necessário processá-los, que incluiu as atividades de verificar valores ausentes, remover dados duplicados e lidar com a ausência de padronização. A quantidade de valores ausentes era inferior a 10% dos dados coletados, o que permitiu que eles apenas fossem ignorados sem prejudicar o estudo. Além disso, alguns dados de publicações e autores foram coletados mais de uma vez, o que exigiu uma verificação antes de integrar os dados. Finalmente, em relação à falta de padronização, o principal problema foi com o nome dos autores. Por exemplo, o nome de um mesmo autor pode estar no formato Mirella M. Moro ou M. M. Moro. Também existem autores distintos com mesmo nome abreviado, por exemplo, C. J. Xia. Nesses casos, foi necessário verificar se dados como instituição e publicações correspondiam ao mesmo autor ou se eram diferentes para então fazer a integração.

5.6.4. Predição de Efeitos Colaterais de Medicamentos

Uma preocupação mundial na área da saúde são os efeitos que podem ser causados por medicamentos. Nesse contexto, Ma et al. [2017] propõem um modelo de grafos probabilísticos para prever efeitos colaterais de medicamentos. Para isso, foram consideradas três diferentes fontes de dados: (*i*) SIDER, base de dados contendo pares medicamento e efeitos colaterais; (*ii*) plataforma FAERS, que contém informação sobre cada efeito colateral enviado ao FDA (do inglês, *U.S. Food and Drug Administration*); e (*iii*) HealthBoards, uma plataforma que possui milhões de mensagens relacionadas a medicamentos e seus efeitos colaterais.

Especificamente, Ma et al. [2017] também mostram os benefícios de considerar múltiplas fontes de dados para obter os verdadeiros efeitos colaterais. A Tabela 5.2 exemplifica os efeitos colaterais da Thyroxine extraídos de dois conjuntos de dados distintos, FAERS e Healthboards. Cada linha na tabela representa um efeito colateral reportado por um usuário. É possível observar que disfagia, náusea e desidratação são efeitos colaterais verdadeiros, enquanto que os outros três são incorretos. Ao minerar os efeitos colaterais de apenas um conjunto de dados, por exemplo apenas do FAERS, dois efeitos colaterais podem ser obtidos: disfagia e náusea. Entretanto, ao utilizar FAERS e Healthboards, é

possível obter três efeitos colaterais corretos.

5.6.5. Predição de Informações de Usuários em Redes Sociais

Devido ao grande volume de dados disponíveis em redes sociais, diferentes estudos estão sendo realizados e abordam, por exemplo, marketing viral [Subramani and Rajagopalan, 2003], detecção de comunidades [Brandão and Moro, 2017, Kim and Hastak, 2018] e privacidade e segurança [Akcora et al., 2012, Yuan et al., 2010]. Dentre tais estudos, também há a predição de informações em redes sociais, que podem ser sobre usuários (nós em uma rede social que pode ser modelada como um grafo) e/ou seus relacionamentos (links ou arestas na rede social) [Brandão et al., 2013].

Nesse contexto, Farnadi et al. [2018] combinam múltiplas fontes de informações sociais usando redes neurais profundas com o objetivo de prever informações do perfil do usuário como idade, gênero e características da personalidade. Para isso, os autores usaram os dados do Facebook de 5.670 usuário e combinaram três diferentes fontes: (*i*) textual por meio das mensagens em seu *status*; (*ii*) visual pela foto do perfil; e (*iii*) dados relacionais através das páginas na quais o usuário indicou sua preferência. Por meio de experimentos, os autores demonstraram que combinando tais indicadores eles poderiam melhorar o desempenho para prever informações em seu perfil.

Em relação à predição de relacionamentos, o objetivo é inferir quais conexões são possíveis de inferir em um futuro próximo. Por exemplo, Lu et al. [2010] propõem um framework de aprendizagem supervisionada para predição de links que aprende de redes sociais dinâmicas (considera o aspecto temporal dos relacionamentos) na presença de redes auxiliares. Em outras palavras, os relacionamentos em uma rede A são preditos utilizando dados de redes auxiliares B, C e D. Para tal estudo, foram utilizadas redes sociais de coautoria construídas com as bases de dados do arXiv[32] com publicações de 1992 a 2003, CiteSeer[33] com publicações de 1995 a 2003, e SIAM (*Society of Industrial and Applied Mathematics*)[34] com publicações de 1999 a 2004.

Há também a predição de informações sobre os relacionamentos. Por exemplo, Wang et al. [2018a] propõem um framework para predizer o sinal do sentimento (positivo ou negativo) de relacionamentos entre usuários em uma rede heterogênea na ausência de dados sobre sentimentos. A realização desse estudo considerou dois conjuntos de dados reais: Weibo Tweets, uma das redes sociais online mais populares na China, e a base de conhecimentos Microsoft Satori.

5.7. Conclusões

Neste capítulo, abordamos três questões relacionadas à utilização de dados provenientes de diferentes fontes Web: coleta, integração e pré-processamento. A seguir, resumimos as principais metodologias para cada etapa, bem como discutimos potenciais variáveis que ficaram de fora desse estudo.

O capítulo iniciou descrevendo quatro principais fontes de dados Web: dados abertos, dados conectados, páginas Web e APIs. Considerando a primeira fase de pro-

[32]arXiv: `https://www.arxiv.org/`
[33]CiteSeer: `https://citeseerx.ist.psu.edu/`
[34]SIAM: `https://www.siam.org/`

cessamento de tais dados, foram abordados conceitos em relação ao processo de coleta de dados, discutindo principalmente os tipos de fontes de dados e as respectivas técnicas de coleta. Especificamente, apresenta-se um conjunto de classificações para os coletores, suas principais aplicações, os principais desafios e três técnicas simples de coleta baseados no caminhamento em grafo. Dentre essas técnicas de coleta estão *Breadth-First Search*, *Snowball Sample* e *Depth-First Search*.

Logo após, este capítulo discutiu como a integração de dados de múltiplas fontes é uma tarefa que permite a extração de informações de forma mais realista para diferentes finalidades. Sem tal integração, as chances de interpretar os dados de forma equivocada é muito maior, conforme mostrado na aplicação de efeitos colaterais de drogas. Por isso, conforme apresentado, diferentes estudos foram realizados e diferentes estratégias foram desenvolvidas para a integração de dados, por exemplo, sistema de mediação, processamento de linguagem natural e abordagem bayesiana. Também apresentou-se as diferenças entre ETL e integração de dados, bem como exemplos para duas estratégias de integração de dados para bancos de dados relacionais: a vertical e a horizontal.

Após a integração dos dados, diferentes tipos de problemas que podem ser identificados na base são tratados na etapa de pré-processamento. Nessa parte do capítulo, foram apresentados então alguns dos problemas mais comuns, incluindo: valores ausentes, veracidade dos dados, remoção de duplicatas, redução de dados e ausência de padronização. Há diversas formas de tratamento para cada um dos problemas, e as soluções ideais variam de acordo com o contexto e disponibilidade dos dados. Foram sumarizados exemplos de trabalhos que necessitaram de algum dos tipos de tratamento e a forma com que cada um lidou com o problema.

Para contextualizar as três etapas, também foram apresentadas diversas aplicações. Por exemplo, apresentou-se como foi feita a coleta de diversas fontes na Wikipédia por meio de dumps e de sua API. Demonstrou-se também que o uso diferentes fontes foi útil para melhorar a predição de efeitos colaterais de medicamentos e apresentar visualizações mais completas de pesquisadores. Além disso, foi demonstrado como identificar informações sociais (idade, gênero e características da personalidade) de perfis de usuários no Facebook e prever relações de amizade.

O tratamento dos dados até a geração dos resultados e análises dos mesmos são processos fundamentais para diversos tipos de pesquisa. A geração correta de conhecimento a partir de dados depende do processo completo de coleta, integração e pré-processamento a fim de compor informações robustas e mais próximas da realidade do assunto estudado. Em tal contexto, diversos dos trabalhos conduzidos pelo nosso grupo de pesquisa estão disponíveis na página do Projeto Apoena[35], com as bases de dados modeladas e tratadas também disponíveis. Os laboratórios que colaboram com essas pesquisas são o Lab CS+X[36] na UFMG e o Piim-Lab[37] no CEFET-MG.

Finalmente, é importante notar que este capítulo possui limitações que são intrínsecas a qualquer trabalho deste tipo. Especificamente, não foram discutidos aspectos de

[35]Projeto Apoena: `http://bit.ly/proj-apoena`
[36]Lab CSX: `http://www.labcsx.dcc.ufmg.br`
[37]Piim-Lab: `http://piim-lab.decom.cefetmg.br`

armazenamento dos dados ou tipos de sistemas de gerência para tais dados. Porém, é necessário clarificar que são muitas variáveis envolvidas em tais aspectos e que devem ser avaliadas caso a caso. Também poderíamos explorar melhor outros problemas inerentes aos grandes volumes de dados que trafegam na Web diariamente. Igualmente importante seria o processamento de strings, imagens e vídeos. Em resumo, pesquisadores e desenvolvedores que necessitam utilizar dados provenientes da Web têm outros desafios que não são abordados aqui e que precisam ser considerados para bom planejamento e execução de seus projetos.

Agradecimentos. As pesquisas que resultaram na escrita deste capítulo foram financiadas por CAPES, CNPq e FAPEMIG.

Referências

C. G. Akcora, B. Carminati, and E. Ferrari. Privacy in Social Networks: How Risky is Your Social Graph? In *IEEE International Conference on Data Engineering (ICDE)*, pages 9–19, Washington, DC, USA, 2012. doi: 10.1109/ICDE.2012.99.

G. B. Alves, M. A. Brandão, D. M. Santana, A. P. C. da Silva, and M. M. Moro. The Strength of Social Coding Collaboration on GitHub. In *Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 247–252, Salvador, Brasil, 2016.

C. V. Araujo, R. M. Neto, F. G. Nakamura, and E. F. Nakamura. Predicting Music Success Based on Users' Comments on Online Social Networks. In *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*, pages 149–156, Gramado, RS, Brazil, 2017. doi: 10.1145/3126858.3126885.

O. Azeroual, G. Saake, and E. Schallehn. Analyzing data quality issues in research information systems via data profiling. *International Journal of Information Management*, 41:50–56, 2018. doi: 10.1016/j.ijinfomgt.2018.02.007.

R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval: The Concepts and Technology Behind Search*. Addison-Wesley Publishing Company, USA, 2nd edition, 2011. ISBN 9780321416919.

S. K. Bansal. Towards a semantic extract-transform-load (etl) framework for big data integration. In *Proceedings of IEEE International Congress on Big Data (BigData Congress)*, pages 522–529, Anchorage, AK, USA, 2014.

R. Bartusiak, T. Kajdanowicz, A. Wierzbicki, L. Bukowski, O. Jarczyk, and K. Pawlak. Cooperation prediction in github developers network with restricted boltzmann machine. In *Asian Conference on Intelligent Information and Database Systems (ACIIDS)*, pages 96–107, Vietnam, 2016. doi: 10.1007/978-3-662-49390-8_9.

N. A. Batista, G. B. Alves, A. L. Gonzaga, and M. A. Brandão. GitSED: Um Conjunto de Dados com Informações Sociais Baseado no GitHub. In *Dataset Showcase Workshop, Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 224–233, Salvador, Brazil, 2017a.

N. A. Batista, M. A. Brandão, G. B. Alves, A. P. C. da Silva, and M. M. Moro. Collaboration strength metrics and analyses on GitHub. In *Proceedings of the International Conference on Web Intelligence*, pages 170–178, Leipzig, Germany, 2017b. doi: 10.1145/3106426.3106480.

M. R. Berthold, C. Borgelt, F. Höppner, and F. Klawonn. *Guide to intelligent data analysis: how to intelligently make sense of real data*. Springer Science & Business Media, 2010. doi: 10.1007/978-1-84882-260-3.

M. Bouzeghoub, B. F. Lóscio, Z. Kedad, and A. Soukane. Heterogeneous data source integration and evolution. In *Proceedings of International Conference on Database and Expert Systems Applications (DEXA)*, pages 751–757, Aix-en-Provence, France, 2002. doi: 10.1007/3-540-46146-9_74.

M. A. Brandão and M. M. Moro. Social professional networks. *Computer Communications*, 100(C):20–31, 2017. doi: 10.1016/j.comcom.2016.12.011.

M. A. Brandão, M. M. Moro, G. R. Lopes, and J. P. M. de Oliveira. Using link semantics to recommend collaborations in academic social networks. In *International Conference on World Wide Web (WWW), Companion Volume*, pages 833–840, Rio de Janeiro, Brazil, 2013. doi: 10.1145/2487788.2488058.

M. A. Brandão, M. A. Diniz, G. A. de Sousa, and M. M. Moro. Visualizing co-authorship social networks and collaboration recommendations with cnare. In N. Meghanathan, editor, *Graph Theoretic Approaches for Analyzing Large-Scale Social Networks*, pages 173–188. IGI Global, 2018. doi: 10.4018/978-1-5225-2814-2.ch011.

C. Casalnuovo et al. Developer onboarding in github: The role of prior social links and language experience. In *Proceedings of Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Sympo. Foundations of Software Engineering*, pages 817–828, Bergamo, Italy, 2015. doi: 10.1145/2786805.2786854.

L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb. Social coding in GitHub: transparency and collaboration in an open software repository. In *ACM Conference on Computer Supported Cooperative Work*, pages 1277–1286, Seattle, USA, 2012. doi: 10.1145/2145204.2145396.

D. H. Dalip, M. A. Gonçalves, M. Cristo, and P. Calado. A general multiview framework for assessing the quality of collaboratively created content on web 2.0. *Journal of the Association for Information Science and Technology*, 68(2):286–308, 2017. doi: 10.1002/asi.23650.

L. de Souza Silva, F. Murai, A. P. C. da Silva, and M. M. Moro. Automatic identification of best attributes for indexing in data deduplication. In *Proceedings of the 12th Alberto Mendelzon International Workshop on Foundations of Data Management*, Cali, Colombia, 2018.

A. Doan, P. Konda, A. Ardalan, J. R. Ballard, S. Das, Y. Govind, H. Li, P. Martinkus, S. Mudgal, E. Paulson, et al. Toward a system building agenda for data integration (and data science). *IEEE Data Eng. Bull.*, 41(2):35–46, 2018.

G. Farnadi, J. Tang, M. De Cock, and M.-F. Moens. User profiling through deep multimodal fusion. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 171–179, 2018. doi: 10.1145/3159652.3159691.

R. T. Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine, 2000.

P. A. Flach. Multi-relational data mining: a perspective. In *Portuguese Conference on Artificial Intelligence*, pages 3–4. Springer, 2001. doi: 10.1007/3-540-45329-6_2.

R. Freitas, C. Rocha, O. Braga, G. Lopes, O. Monteiro, and M. Oliveira. Using Linked Data in the Data Integration for Maternal and Infant Death Risk of the SUS in the GISSA Project. In *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*, pages 193–196, Gramado, RS, Brazil, 2017. doi: 10.1145/3126858.3131606.

F. Geerts, P. Missier, and N. Paton. Editorial: Special issue on improving the veracity and value of big data. *J. Data and Information Quality*, 9(3):13:1–13:2, 2018. doi: 10.1145/3174791.

B. Golshan, A. Halevy, G. Mihaila, and W.-C. Tan. Data integration: After the teenage years. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 101–106, Chicago, Illinois, USA, 2017. doi: 10.1145/3034786.3056124.

L. A. Goodman. Snowball sampling. *The annals of mathematical statistics*, pages 148–170, 1961.

J. Kim and M. Hastak. Social network analysis: Characteristics of online social networks after a disaster. *International Journal of Information Management*, 38(1):86–96, 2018. doi: 10.1016/j.ijinfomgt.2017.08.003.

A. H. F. Laender, M. M. Moro, C. Nascimento, and P. Martins. An x-ray on web-available XML schemas. *SIGMOD Record*, 38(1):37–42, 2009. doi: 10.1145/1558334.1558338.

J. Liu and S. Ram. Using big data and network analysis to understand wikipedia article quality. *Data & Knowledge Engineering*, 115:80–93, 2018. doi: 10.1016/j.datak.2018.02.004.

Z. Lu, B. Savas, W. Tang, and I. S. Dhillon. Supervised link prediction using multiple sources. In *Proceedings of IEEE 10th International Conference on Data Mining (ICDM)*, pages 923–928, 2010. doi: 10.1109/ICDM.2010.112.

F. Ma, C. Meng, H. Xiao, Q. Li, J. Gao, L. Su, and A. Zhang. Unsupervised discovery of drug side-effects from heterogeneous data sources. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 967–976, 2017. doi: 10.1145/3097983.3098129.

L. F. M. P. Maia and J. Oliveira. Investigation of research impacts on the zika virus: An approach focusing on social network analysis and altmetrics. In *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*, pages 413–416, Gramado, RS, Brazil, 2017. doi: 10.1145/3126858.3131593.

M. M. Moro, V. Braganholo, C. F. Dorneles, D. Duarte, R. de Matos Galante, and R. dos Santos Mello. XML: some papers in a haystack. *SIGMOD Record*, 38(2):29–34, 2009. doi: 10.1145/1815918.1815924.

G. P. Oliveira, N. A. Batista, M. A. Brandão, and M. M. Moro. Tie strength in github heterogeneous networks. In *Brazilian Symposium on Multimedia and the Web (WebMedia)*, 2018.

L. Sikos. *Mastering structured data on the Semantic Web: From HTML5 microdata to linked open data*. Apress, 2015.

M. R. Subramani and B. Rajagopalan. Knowledge-sharing and influence in online social networks via viral marketing. *Communications of the ACM*, 46(12):300–307, 2003. doi: 10.1145/953460.953514.

G. Tejay, G. Dhillon, and A. G. Chin. Data Quality Dimensions for Information Systems Security: A Theoretical Exposition. In *Security Management, Integrity, and Internal Control in Information Systems*, pages 21–39. Springer, 2006.

J. Tsay, L. Dabbish, and J. Herbsleb. Influence of social and technical factors for evaluating contribution in github. In *Procs. of the 36th International Conference on Software Engineering*, pages 356–366, Hyderabad, India, 2014. doi: 10.1145/2568225.2568315.

N. K. Tyagi, A. Solanki, and S. Tyagi. An algorithmic approach to data preprocessing in web usage mining. *International Journal of Information Technology and Knowledge Management*, 2(2):279–283, 2010.

B. Vasilescu, A. Serebrenik, and V. Filkov. A data set for social diversity studies of github teams. In *Proceedings of the 12th Working Conference on Mining Software Repositories*, pages 514–517, 2015. doi: 10.1109/MSR.2015.77.

E. F. Veiga, M. F. Arruda, J. A. B. Neto, and R. d. F. Bulcão Neto. An ontology-based representation service of context information for the internet of things. In *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*, pages 301–308, Gramado, RS, Brazil, 2017. doi: 10.1145/3126858.3126894.

H. Wang, F. Zhang, M. Hou, X. Xie, M. Guo, and Q. Liu. Shine: Signed heterogeneous information network embedding for sentiment link prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 592–600, Marina Del Rey, USA, 2018a. doi: 10.1145/3159652.3159666.

L. Wang, R. Pan, X. Wang, W. Fan, and J. Xuan. A bayesian reliability evaluation method with different types of data from multiple sources. *Reliability Engineering & System Safety*, 167:128–135, 2017. doi: 10.1016/j.ress.2017.05.039.

R. Wang, W. Ji, M. Liu, X. Wang, J. Weng, S. Deng, S. Gao, and C.-a. Yuan. Review on mining data from multiple data sources. *Pattern Recognition Letters*, 2018b. doi: 10.1016/j.patrec.2018.01.013.

M. Yuan, L. Chen, and P. S. Yu. Personalized Privacy Protection in Social Networks. In *Proceedings of Very Large Data Base Endowment*, pages 141–150, 2010. doi: 10.14778/1921071.1921080.

B. Zhao, B. I. P. Rubinstein, J. Gemmell, and J. Han. A bayesian approach to discovering truth from conflicting sources for data integration. *Proc. VLDB Endow.*, 5(6):550–561, 2012. ISSN 2150-8097. doi: 10.14778/2168651.2168656.

Biografia Resumida dos Autores

Natércia A. Batista. Natércia A. Batista. É aluna de mestrado em Ciência da Computação na Universidade Federal de Minas Gerais, Bacharel em Sistemas de Informação pela Universidade Federal de Minas Gerais (2017), e técnica em Informática Industrial pelo Centro Federal de Educação Tecnológica de Minas Gerais (2011). Atualmente trabalha no Laboratório de Computação Interdisciplinar CS+X e seus principais interesses são nas áreas de análise e gerenciamento de dados, e análise de redes sociais.

Michele A. Brandão É professora do Instituto Federal de Minas Gerais. É Doutora e Mestre em Ciência da Computação pela UFMG, Bacharel em Ciência da Computação pela Universidade Estadual de Santa Cruz (UESC, Bahia). Foi professora substituta na UFMG e PUC/Minas (Pontifícia Universidade Católica de Minas Gerais) e bolsista de Pós-Doutorado Júnior (PDJ-CNPq) no Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais (UFMG). Seus principais interesses de pesquisa estão nas áreas de mineração de dados, análise e gerenciamento de dados, sistemas de recomendação, predição de links e redes sociais. Seu projeto de pesquisa atual visa aplicar seu conhecimento para auxiliar no avanço da forense digital.

Michele B. Pinheiro Possui bacharelado (2013) e mestrado (2016) em Ciência da Computação pela Universidade Federal de Minas Gerais. Realizou pesquisas envolvendo *crowdsourcing*/*crowdsensing* ativo no contexto de dados geográficos, como contribuição voluntária geográfica. Atualmente trabalha em projetos interdisciplinares que envolvem o grupo de pesquisa CS+X (Departamento de Ciência da Computação - UFMG) e o grupo Indisciplinar (Escola de Arquitetura - UFMG), sobre a coleta de dados realizada por cidadãos e políticas públicas junto ao INCT de Tecnopolíticas (Indisciplinar).

Daniel H. Dalip É professor do Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG). É Doutor (UFMG/2015), Mestre (UFMG/2009) e Bacharel (Uni-BH/2006) em Ciência da Computação. Realiza pesquisas nas áreas de banco de dados e recuperação de informação. Tem experiência de docência nas disciplinas de Programação Web, Algoritmos, Recuperação de Informação, Pesquisa Operacional. Já lecionou na PUC-MG e Uni-BH. Durante seu mestrado e o doutorado, desenvolveu pesquisas sobre o uso de aprendizagem de máquina para avaliar automaticamente a qualidade em documentos colaborativos na Web.

Mirella M. Moro É professora associada do Departamento de Ciência da Computação (DCC) da Universidade Federal de Minas Gerais (UFMG). Possui doutorado em Ciência da Computação pela *University of California in Riverside* (2007), e graduação e mestrado em Ciência da Computação pela Universidade Federal do Rio Grande do Sul (UFRGS). Após o seu doutoramento, foi bolsista CNPq PDJ (PosDoc Junior) no Instituto de Informática da UFRGS. Foi membro do *Education Council* da ACM (*Association for Computing Machinery*), Diretora de Educação da SBC (Sociedade Brasileira de Computação, 2009-2015), editora-chefe da revista eletrônica SBC Horizontes (2008-2012), editora associada do JIDM (Journal of Information and Data Management, 2010-2012) e coordenadora da Comissão Especial de Bancos de Dados (CE-BD) da SBC (2015). Seus interesses de pesquisa estão na área de Banco de Dados, incluindo tópicos como processamento de consultas, redes sociais, recomendação, bibliometria e NoSQL.