3
Trabalhos Relacionados

Algumas abordagens usadas para conectar fontes RDF são descritas neste capítulo. As abordagens foram agrupadas de acordo com a classificação apresentada no trabalho de Flores em [40]. A classificação define como os dados podem ser conectados em diferentes níveis:

1. **Nível de metadados**: Os dados são publicados com uma quantidade moderada de metadados baseado em texto, que inclui: título, agência de publicação, palavras-chave, descrição, categoria, *links*, etc. É possível processar metadados de dois conjuntos buscando qualquer sobreposição que exista entre eles, como por exemplo uma palavra-chave. Assim, qualquer conteúdo comum descoberto será utilizado para conectar dois ou mais conjuntos de dados. Dessa forma, a geração de descritores sobre os metadados é de grande ajuda no processo de conexão dos dados.

2. **Nível do conteúdo**: Quando a busca por palavras-chave não é suficiente para descobrir *links* entre fontes de dados, uma análise mais profunda da semântica dos dados é requerida.

   Neste nível, os valores de *strings* que compõem os dados são vistos como potenciais entidades de interesse. Uma entidade de interesse pode ser qualquer coisa que é concebível, tais como uma cidade, uma pessoa, ou um elemento químico.

   O objetivo das abordagens deste nível é executar a resolução de entidades, na qual uma variedade de técnicas de processamento de linguagem natural e verificação semântica são usadas para determinar se uma *string* pode ou não ser resolvida para uma entidade de interesse.

3. **Nível social**: Usar conceitos de Web semântica social [41] e redes sociais pode ajudar a fornecer soluções incrementais na integração de dados publicados na LOD. As mesmas agências que publicam os dados podem participar ativamente não somente para conectar seus dados, mas também para corrigir e enriquecer conteúdo já publicado.

   Este nível usa um modelo de produção que utiliza a inteligência e os conhecimentos coletivos e voluntários espalhados pela internet. Ele é

usado para resolver problemas, criar conteúdo e novas soluções. Este modelo recebe o nome de *crowdsourcing*.

As seções 3.1, 3.2 e 3.3, deste capítulo apresentam as abordagens agrupadas no nível de metadados, nível de conteúdo e nível social respectivamente.

3.1 Nível de Metadados

3.1.1 Geração de descritores VoiD com MapReduce

Uma abordagem para gerar descritores de forma automática, baseado no vocabulário VoiD e usando o paradigma MapReduce, é o trabalho feito por Böhm [37]. O trabalho de Böhm gera descritores usando o conceito de HTTP URI para conseguir agrupar as triplas em subconjuntos. Estatísticas são geradas baseados em termos contidos nas fontes de dados segundo as anotações do vocabulário VoiD (por exemplo, <*void:numberOfPredicates*> e <*void:numberOfPredicates*>). No trabalho de Böhm, define-se quatro tipos de conexões (*links*) entre os subconjuntos de dados da fonte, usando à anotação <*void:Linkest*> para descrevê-los:

1. Conexões que são identificadas nas triplas, onde o sujeito pertence a um conjunto de dados diferente do objeto.

2. Conexões que não são explicitamente declaradas. É adotada uma abordagem de *k-similaridade*, onde dois sujeitos são *k-similares* se e somente se **k** combinações de seus "predicados/objetos" são correspondências exatas.

3. Conexões que são identificadas por meio dos predicados que descrevem um recurso. Por exemplo, se o predicado <*owl:sameAs*> conecta dois recursos (A, B) e cada um deles tem uma descrição independente usando os predicados *livesIn* e *worksAt* (<A> <*livesIn*> <Berlin> , <B> <*worksAt*> <Lyon>). Então pode-se derivar novos predicados para os recursos (A,B).

4. Conexões conceituais. Dois recursos estão contidos no mesmo conjunto conceitual se e somente se eles são o mesmo, ou do mesmo tipo.

A abordagem de Böhm foi executada sobre o conjunto de dados *Billion Triple Challenge*(BTC) 2010[1] que contem muitos domínios e tópicos muito

[1]http://km.aifb.kit.edu/projects/btc-2010/

variados, mas propõe novos critérios para descobrir *links* entre diferentes conjuntos de dados.

O trabalho de Böhm apresenta uma abordagem para geração automática de descritores guiado pelo vocabulário VoiD, gerando estatísticas representativas do conjunto de dados em questão. O Böhm utiliza MapReduce para acelerar o processo de análise da uma grande quantidade de dados.

A abordagem de Böhm faz uma análise *offline* dos dados. Em outras palavras, o Böhm não considera o acesso à fonte através de um SPARQL *endpoint*, somente via arquivos RDF *dump*. Neste trabalho não participa o usuário dos dados, que é a maior diferença com a nossa proposta. É claro que a abordagem de Böhm pode ser adaptada para qualquer conjunto de dados, mas gerar estatísticas de uma fonte de dados *online* é uma tarefa que não é considerada nesta abordagem.

3.2 Nível do Conteúdo

3.2.1 Descoberta de conexões em fontes de dados governamentais

O trabalho de Flores [18] utiliza uma abordagem centrada em entidades textuais. O problema específico de Flores é tratar de conectar conjuntos de dados na fonte `data.gov`, através de qualquer entidade que represente um estado ou território dos *U.S.*.

O processo de busca definido por Flores recebe três entradas:

1. Um conjunto de dados de `data.gov` no formato RDF.
2. Uma lista de palavras-chave para busca especificadas pelo usuário que contêm a palavra *state*.
3. Um conjunto de dados que representa os possíveis nomes dos estados dos *U.S.*. Os nomes estão conectados com uma entidade definida em uma fonte de dados conhecida (como por exemplo, DBpedia ou Geonames), através do predicado `owl:sameAs`.

A busca de entidades de Flores é composto por três processos: no primeiro processo, os predicados que contém a palavra *state* são obtidos da lista de palavras-chave, nomeando esses predicados como *state-related*. A segunda parte aplica uma resolução de entidades, onde cada valor do objeto correspondente ao predicado encontrado, tem que fazer *matching* com algum termo dos nomes de estado. O último processo nomeado "Processador de predicados" usa um conjunto de heurísticas para avaliar os predicados *state-related*

encontrados. As heurísticas definidas usam processamento de linguagem natural e distância entre *strings*, podendo encontrar ocorrências tanto dos valores objeto relacionados aos predicados *state-related*, como da mesma descrição textual do predicado *state-related*.

O trabalho de Flores define um conjunto de nomes dos estado dos *U.S.* como gabarito do processo de busca. Porém, para diferentes tópicos de interesse criar gabarito é uma tarefa demorada, pois quanto maior número de tópicos maior é a complexidade da tarefa para gerar os diferentes gabaritos.

Para cada predicado do tipo *state-related* uma consulta SPARQL é gerada para realizar o processo de *matching*. Em um conjunto de milhões de triplas, o processo de *matching* de Flores executado sequencialmente seria inviável. Abordagens de processamento em paralelo são assumidos pela nossa proposta, criada justamente para focar no problema execuções de consultas sequenciais.

Considerando que os tópicos de interesse variam em cada processo de busca, é importante o fato de restringir o domínio de busca e focar nas entidades que representam uma fonte de dados.

3.2.2 Reconciliação de entidades entre fontes de dados da LOD

A abordagem de Maali [16] é inspirada em *Google Refine*[2]. No trabalho de Maali utiliza-se o termo de **reconciliação**, para definir a comparação de dois objetos usando uma abordagem de similaridade. O trabalho de Maali compara abordagens de consulta com a linguagem SPARQL, técnicas de *ontology matching* e busca de texto sobre dados RDF.

As abordagens comparadas por Maali fornecem uma API bem definida para acessos remotos, assim como uma implementação disponível que pode ser aplicada em diferentes fontes de dados RDF. Foram comparadas quatro abordagens:

1. O SPARQL, a linguagem de consultas de RDF recomendada pela W3C [9].

2. O SPARQL com extensões adicionais de busca *full-text*.

3. O Silk *Server* que fornece uma interface RESTful [42] para conectar dois conjuntos de dados baseado no Silk *framework* [13].

4. O Sindice[3] que é um índice semântico composto por serviços de busca e consulta sobre recursos na Web[12].

[2]http://code.google.com/p/google-refine/
[3]http://sindice.com/search

Os resultados obtidos na comparação feita por Maali favorecem o Silk *Server*, por esta ser uma ferramenta especialista em interligações, baseada em métricas de similaridade, mas que depende da complexidade dos conjuntos de dados referenciados.

Semelhante à proposta apresentada em nosso trabalho, o trabalho de Maali demonstra a possibilidade de reconciliar um conjunto de dados com fontes de LOD, como: DBPedia[4], Sindice e dados do Sider[5] carregados em banco de dados em triplas. O trabalho de Maali conecta dados com fontes populares, o que para alguns casos pode não ser a melhor solução. Para comparação das abordagens, somante propriedades do tipo texto são consideradas nas análises feitas por Maali. Essa restrição é mantida no estudo de caso realizado em nosso trabalho.

3.2.3 Processo de recomendação de fontes RDF

O quarto princípio de *Linked Data* [1] recomenda incluir *links* que apontam URIs de outros conjuntos na LOD. Dessa forma, informações adicionais podem ser obtidas navegando entre esses *links* [1]. Torna-se necessário então definir um processo de recomendação, que ajude o usuário a identificar fontes de dados candidatas, onde o mesmo possa encontrar informações relevantes e conseguir relacionar os tópicos de interesse. O trabalho de Nikolov [15] é baseado na busca de palavras-chave sobre um índice Web, o índice colecta documentos semânticos da LOD e da mesma Web. A seguir, uma breve descrição do processo proposto por Nikolov:

Pressuposto:

1. Pressupõe-se que existe um conjunto de dados a ser publicado D_p, contendo um conjunto de indivíduos I_p descritos com a ontologia O_p de tal modo que $D_p = \{O_p, I_p\}$. Além disso, cada indivíduo pertence ao menos a uma classe c_λ definida em O_p, assim, $I_p = \{i_j | c_\lambda(i_j), c_\lambda \in O_p\}$.

2. Pressupõe-se que na Web existe um conjunto de fontes de dados vinculadas $\{D_1, ..., D_n\}$, de modo que $D_j = \{O_j, I_j\}$. Por conseguinte, existe um subconjunto dessas fontes $\{D_1, ..., D_m\}$ que se sobrepõem com D_p, ou seja, que para $(\forall(j \leq m)\exists(I_j^O \subseteq I_j)(I_j^O = \{i_k | equiv(i_k, i_p), i_p \in I_p\}))$, onde *equiv* denota a relação de equivalência entre indivíduos.

[4]http://dbpedia.org
[5]http://www4.wiwiss.fu-berlin.de/sider/

3. Pressupõe-se que existe um serviço de busca por palavras-chave, que recebe como entrada um conjunto de palavras-chave $K = \{k_1, ..., k_i\}$. Como resposta, o serviço deve retornar um conjunto de indivíduos que podem pertencer a diferentes fontes de dados, assim tem se, $I^{srs} = I_1^{srs} \cup I_2^{srs} \cup .... \cup I_m^{srs}$, onde $I_j^{srs} \in I_j$. Para os indivíduos retornados $i_{jk} \in I_j^{srs}$, os respectivos tipos $\{c_{jk\lambda} | c_{jk\lambda}(i_{jk})\}$ são também disponíveis no resultado da busca.

Meta:

1. Identificar o subconjunto de fontes de dados $\{D_1, ..., D_m\}$ e classificá-las segundo o grau de sobreposição $\frac{|I_j^O|}{|I_p|}$.

2. Gerar um ranking para cada classe $c_\lambda \in O_p$, baseado no grau de sobreposição de indivíduos desta classe $|I_{j\lambda}^O|$, onde $I_{j\lambda}^O = \{i_k | equiv(i_k, i_p), i_p \in I_p, c_\lambda(i_p)\} \subseteq I_j^O$. Assim, conseguir conectar c_λ com as diferentes classes encontradas no subconjunto de fontes $\{D_1, ..., D_m\}$.

Processo:

1. Selecionar um subconjunto de indivíduos de D_p pertencentes a uma classe c_p. O subconjunto deve ser suficientemente grande para gerar um ranking confiável das fontes.

2. Consultar o serviço de busca, usando os valores das propriedades de tipo texto (*rdf:label*, *foaf:name*, *dc:title*, etc) de cada indivíduo do subconjunto selecionado (as propriedades de tipo texto forma parte do conjunto de propriedades de tipo de dados - *Data properties*).

3. Salvar o resultado de cada busca em um repositório em comum. O conjunto de indivíduos i_{jk} retornados são agrupados segundo sua fonte de origem D_j.

4. Ordenar as fontes de dados segundo o número de indivíduos retornados pelo serviço de busca, ou seja $|\{i_{jk} | i_{jk} \in D_j\}|$.

O processo de Nikolov também usa técnicas de alinhamento de ontologias, que consegue melhorar os resultados obtidos. O processo de Nikolov, anteriormente definido, possui as seguintes limitações:

– O processo de análise e busca de dados que se pretende publicar: Normalmente os dados em RDF contêm muitas descrições e identificá-los

é complicado. Além disso uma busca não automatizada seria muito demorada. Quanto se trata de grande quantidades de dados, analisar dados RDF somente com consultas SPARQL não resolveria completamente o problema, devido à complexidade da linguagem [43].

– A quantidade de dados que se pretende processar: Esta é uma característica que não deveria limitar o usuário no processo de busca.

– A ausência de uma ferramenta que materialize e sustente o processo definido: A comunidade de usuários precisa de aplicativos que ajudem e incentivem a publicação e interligação de dados.

Estas limitações fazem parte das motivações para a realização deste trabalho. O trabalho apresentado neste documento adapta este processo, e tem com objetivo mitigar estas dificuldades.

3.3 Nível Social

3.3.1 Conexão incremental de fontes de dados governamentais

TWC LOGD desenvolvido por Ding [7] é um portal para um ecossistema de *Linked Open Government Data* (LOGD), que tem transformado conjuntos de dados de `data.gov` em RDF. O ferramenta do Ding é descrito como o portal tem sido projetado para servir como um recurso na comunidade global de LOGD, sendo desenvolvido nos *U.S.*. O portal contribui em todas as etapas do processo de publicação na *Government Linked Data* (GLD) [8].

O processo de conexão de dados de Ding é aplicado em diferentes granularidades dos dados. Na etapa de geração, os dados brutos são transformados usando o padrão RDF baseado em diferentes mapeamentos. Os mapeamentos de dados definidos em Excel *Spreadsheet*, tabelas de banco de dados relacionais e outros diferentes formatos recebem conceitos correspondentes. Por exemplo, uma coluna de uma tabela corresponde com o nome de uma propriedade particular.

No nível de metadados, os conjuntos gerados são descritos pelo vocabulário VoiD. Anotações como `void:subsets` são usadas para conectar e agrupar subconjuntos da mesma fonte. Anotações de tipo *Proof Markup Language* (PML) [44] são usadas para rastrear o processo de transformação, gerando informações de proveniência do processo.

O TWC LOGD de Ding usa heurísticas para fazer sugestões de *ontology mappings* entre diferentes conjuntos de dados. No processo de transformação é analisado semanticamente cada registro, onde entidades de interesses tais

como uma pessoa, organizações e localizações, podem ser identificadas. Assim mapeamentos através do predicado `owl:sameAs` são criados para relacionar diferentes entidades.

O TWC LOGD aproveita a plataforma Web de semântica social [41], possibilitando a interação entre fornecedores e consumidores de dados. Cada recurso URI é dereferenciado para um documento RDF/XML, exportado para uma *wiki page* na qual os usuários podem agregar mais definições semânticas.

O trabalho de Ding define uma infraestrutura que reúne tecnologias da Web Semântica, sobre cada umas das etapas do processo de publicação [8]. Com isto consegue aproximar tanto a responsáveis da publicação e quanto os consumidores dos dados em um mesmo fluxo de operações. Este é o mesmo objetivo adotado no desenvolvimento de nosso trabalho.