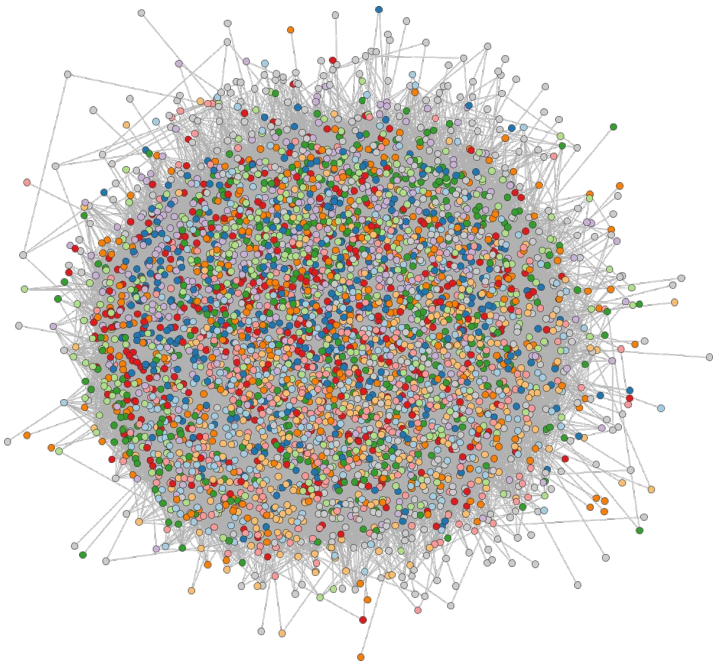


Sumariação de grafos semânticos de grande dimensão usando espaços de nomes

Ana Rita Santos Lopes da Costa
Ciência de Computadores
Departamento de Ciência de Computadores
2022

Orientador
José Paulo Leal, Professor Auxiliar, Faculdade de Ciências da Universidade do Porto

Supervisor
André Santos, Faculdade de Ciências da Universidade do Porto

Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto, _____/_____/_____

Declaração de Honra

Eu, Ana Rita Santos Lopes da Costa, inscrito(a) no Mestrado em Ciência de Computadores da Faculdade de Ciências da Universidade do Porto declaro, nos termos do disposto na alínea a) do artigo 14.º do Código Ético de Conduta Académica da U.Porto, que o conteúdo da presente dissertação reflete as perspetivas, o trabalho de investigação e as minhas interpretações no momento da sua entrega.

Ao entregar esta dissertação, declaro, ainda, que a mesma é resultado do meu próprio trabalho de investigação e contém contributos que não foram utilizados previamente noutros trabalhos apresentados a esta ou outra instituição.

Mais declaro que todas as referências a outros autores respeitam escrupulosamente as regras da atribuição, encontrando-se devidamente citadas no corpo do texto e identificadas na secção de referências bibliográficas. Não são divulgados na presente dissertação quaisquer conteúdos cuja reprodução esteja vedada por direitos de autor. Tenho consciência de que a prática de plágio e auto-plágio constitui um ilícito académico.

Ana Rita Santos Lopes da Costa

Pedroso, 30/09/2022

Agradecimentos

A realização desta dissertação contou com apoios e incentivos importantes. Gostaria de agradecer ao professor José Paulo Leal e ao supervisor André Santos por toda a orientação e grande disponibilidade manifestada no decorrer deste trabalho, e por me terem proporcionado a possibilidade da elaboração e da apresentação de um artigo numa conferência. Quero também agradecer aos meus pais e amigos por toda a compreensão, apoio e estímulo à minha motivação ao longo deste percurso.

Resumo

Os grafos semânticos, frequentemente expressos em Resource Description Framework (RDF), tendem a ter dimensões muito grandes, o que dificulta a sua utilização. Face aos problemas provenientes dessa dimensão, surge a necessidade de os sumariar. Estes sumários pretendem revelar as características importantes do grafo original e facilitar a sua compreensão.

Os grafos RDF utilizam espaços de nomes nos formatos de serialização. Apesar destes espaços de nomes não fazerem parte da semântica do RDF, têm significado intrínseco. A abordagem seguida para este trabalho explora o uso destes espaços de nomes e o seu significado de modo a sumariar grafos RDF de grandes dimensões. Os elementos RDF dos triplos são mapeados em diferentes grupos: os Internationalized Resource Identifiers (IRIs) são reduzidos aos seus espaços de nomes, os literais aos seus tipos de dados e os nós brancos a um grupo próprio, obtendo triplos reduzidos.

O sumário de um grafo será também um grafo RDF formado por esses triplos reduzidos reificados, de modo a afirmar a sua frequência. Para o grafo sumário, foi criada uma visualização que capta estatísticas e que permite conhecer rapidamente o grafo original.

O processo de sumariação foi aplicado para três grafos RDF com dimensões na ordem dos milhões de triplos, tendo sido obtidos bons resultados. Este trabalho contribui com um módulo Python no Python Package Index (PyPI) e a publicação de um artigo.

Abstract

Semantic graphs, often expressed in Resource Description Framework (RDF), tend to have large dimensions, which makes them difficult to use. In view of the problems arising from this dimension, there is a need to summarize them. These summaries are intended to reveal important charecteristics of the original graph and make it more understandable.

RDF graphs use namespaces in serialization formats. Although these namespaces are not part of the RDF semantics, they have intrinsic meaning. The approach followed for this work explores the use of these namespaces and their meaning in order to summarize large RDF graphs. The RDF elements of the triples are mapped into different groups: the Internationalized Resource Identifiers (IRIs) are reduced to their namespaces, the literals to their data types and the blank nodes to a group of their own, obtaining reduced triples.

The summary of a graph will also be a RDF graph formed by these reified reduced triples, in order to assert their frequency. For the summary graph, a visualization was created that captures statistics and allows to quickly know the original graph.

The summarization process was applied for three RDF graphs with millions of triples, and there were obtained good results. This work contributes with a Python module to Python Package Index (PyPI) and the publication of an article.

Conteúdo

Lista de Tabelas

Lista de Figuras

Lista de Blocos de Código

Acrónimos

IRI Internationalized Resource Identifier

JSON JavaScript Object Notation

RDF Resource Description Framework

RDFS RDF Schema

SPARQL SPARQL Protocol and RDF Query Language

UML Unified Modeling Language

URL Uniform Resource Locator

PyPI Python Package Index

Capítulo 1

Introdução

Os grafos são um tipo de dados composto por nós interligados por arcos. Estes permitem guardar informação sobre um determinado domínio através dos nós, que representam entidades e que são relacionadas através dos arcos. São utilizados numa variedade de assuntos e para resolver problemas da vida real. Por exemplo, as redes sociais têm associadas um grafo que representa todos os perfis de pessoas existentes, contendo informação pessoal sobre os utilizadores e as interações que se estabelecem entre eles.

O tema desta dissertação são os grafos semânticos, que representam conceitos como nós e relações entre estes como arcos. Estes grafos são frequentemente expressos em Resource Description Framework (RDF) em que são vistos como um conjunto de triplos *(sujeito, predicado, objeto)*: o *sujeito* e o *objeto* são, respetivamente, o nó de origem e o nó de destino, e o *predicado*, também designado por propriedade, o arco entre os nós [1]. Existem três elementos RDF nestes grafos: os Internationalized Resource Identifiers (IRIs), os nós brancos e os literais. Os nós brancos não têm associado nenhum IRI ou literal e não podem aparecer na posição de predicado. Os literais têm associado um tipo de dados e apenas aparecem na posição de objeto [2].

Uma abordagem comum na sumariação de grafos consiste em agrupar nós e arcos semelhantes. O desafio é identificar um número reduzido de grupos de nós e arcos num grafo RDF de modo a que o grafo sumário seja significativo e compreensível. Os grupos de nós e arcos devem conter parte do significado dos seus elementos, mas devem também ser pequenos o suficiente para serem facilmente compreendidos [3].

1.1 Motivação

Neste trabalho consideramos massivos os grafos com dimensões na ordem dos biliões de triplos (a título de exemplo, temos a DBpedia e a Wikidata [4]), e grandes os grafos de dimensões na ordem dos milhões de triplos. Os grafos semânticos com interesse para este trabalho são estes últimos. Dimensões nesta ordem de grandeza levantam vários problemas: dificuldade em perceber que tipo de informação o grafo contém, dificuldade em carregar o grafo em memória,

sendo a aplicação de algoritmos e *queries* demorada, dificuldade na utilização de ferramentas de visualização e aparecimento de ruído nos dados [3].

Os sumários pretendem revelar as características importantes dos grafos, podendo ser um sub grafo mais pequeno contendo apenas alguns nós e arcos ou um documento noutro formato que sumaria o conteúdo do grafo (tal como, estatísticas, caminhos e indicação dos tipos existentes no grafo), facilitando a sua compreensão. Têm como finalidade extrair informação concisa e significativa do grafo original [1].

Existem várias aplicações para os grafos sumários, como, por exemplo, ajudar a identificar os nós mais importantes, extrair um esquema dos dados e fazer *queries* mais rapidamente [1].

1.2 Abordagem

A abordagem seguida para esta dissertação passa por mapear os elementos RDF de cada triplo do grafo em diferentes grupos, produzindo triplos reduzidos. De seguida, é feita uma contagem dos triplos reduzidos que estão repetidos. Finaliza-se com a produção de uma visualização do grafo sumário.

Os elementos RDF mais frequentes de um triplo são os IRIs, normalmente um Uniform Resource Locator (URL). Em alguns formatos de serialização de grafos RDF, os IRIs são associados a um espaço de nomes, sendo que cada espaço de nomes é reduzido a uma etiqueta. Apesar de apenas existirem na serialização e não fazerem parte da semântica do RDF, os espaços de nomes têm significado intrínseco. Estes refletem uma semelhança de identificadores de recursos e vocabulários e são atribuídos pelos autores dos grafos em vez de serem gerados por um algoritmo [5]. Este trabalho explora o uso destes espaço de nomes e o seu significado para sumariar grafos RDF.

Os literais são reduzidos aos seus tipos de dados e os nós brancos são reduzidos a um grupo próprio.

No final do trabalho, o resumo deverá ser composto pelo grafo reduzido e por estatísticas dos nós e arcos.

1.3 Objetivos

Com este trabalho pretende-se perceber a informação que o grafo semântico de grandes dimensões guarda, tendo em conta que dado o seu grande volume de dados se torna difícil conhecê-lo. Assim, é necessário gerar um grafo sumário agrupando os nós e arcos de acordo com o seu tipo e ainda criar uma visualização do sumário que capte determinadas estatísticas e que permita revelar características importantes do grafo original.

Portanto, os objetivos do trabalho são os seguintes:

1. Produzir sumários para grafos semânticos de grandes dimensões, da ordem de grandeza dos milhões de triplos.
2. Produzir o grafo sumário em poucas horas, não excedendo um dia.
3. Obter informação sobre o grafo original de modo a conhecê-lo.
4. Reduzir o tamanho do grafo, agrupando os nós do mesmo tipo em super nós e os arcos em super arcos.
5. Visualizar com facilidade este grafo reduzido.

Um objetivo a longo prazo seria gerar sumários para grafos semânticos massivos. Mas no contexto desta dissertação, o processo de sumariação realizado foi testado para três grafos de tamanho grande. Conseguiu-se obter um grafo sumário e respetiva visualização para cada um deles. A duração máxima para obter um sumário foi de 1 hora e a mínima foi de 4 minutos. Com a visualização é possível observar a frequência dos espaços de nomes do grafos e como estes se relacionam uns com os outros.

1.4 Plano de Trabalhos

O trabalho desta dissertação foi dividido nas seguintes tarefas, sendo a imagem 1.1 o diagrama de Gantt associado:

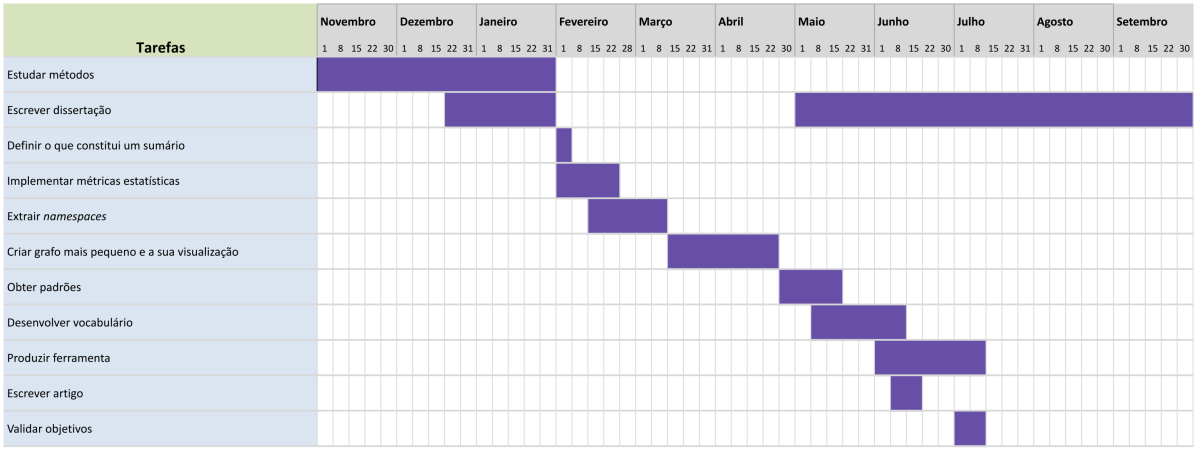


Figura 1.1: Plano de trabalhos

1. Estudar os tipos de métodos que existem para sumariar grafos em geral e, mais especificamente, grafos semânticos.
2. Definir o que constituirá o sumário que se quer produzir, com base no conhecimento anterior.

3. Implementar métricas estatísticas de grafos como, por exemplo, obter o número de propriedades únicas.
4. Extrair os espaços de nomes dos nós e arcos e utilizá-los para tipificar os sujeitos, propriedades e objetos.
5. Agrupar nós e arcos para criar um grafo com dimensões mais pequenas. Criar a visualização deste grafo.
6. Obter padrões a partir do grafo reduzido, de modo a identificar que espaços de nomes são utilizados em conjunto.
7. Desenvolver um vocabulário que produza os sumários conforme a sua definição.
8. Produzir uma ferramenta que implemente as métricas e o vocabulário, e aplicar a vários grafos semânticos de forma a obter resumos dos mesmos.
9. Escrever um artigo sobre o processo de sumariação criado e resultados obtidos.
10. Validar o cumprimento os objetivos.

1.5 Estrutura da dissertação

Esta dissertação está dividida em seis capítulos:

- **Capítulo 1: Introdução**: Neste capítulo é feita uma pequena introdução ao tema do trabalho, descrevendo a motivação para o realizar e os seus objetivos.
- **Capítulo 2: Conceitos Básicos**: São apresentados os conceitos básicos relativamente ao tema de trabalho.
- **Capítulo 3: Estado da Arte**: São descritos trabalhos integrados em vários tipos de métodos de sumariação de grafos.
- **Capítulo 4: Desenho e Desenvolvimento**: É descrito o desenvolvimento do trabalho prático realizado.
- **Capítulo 5: Validação**: São apresentados os resultados obtidos quando aplicado o processo de sumariação a vários grafos semânticos e a respetiva análise.
- **Capítulo 6: Conclusões**: Por último, são expostas as conclusões sobre o trabalho realizado e o trabalho futuro.

Capítulo 2

Conceitos Básicos

Neste capítulo são apresentados conceitos básicos sobre grafos semânticos de modo a ser possível compreender todo o trabalho realizado nesta dissertação.

Inicialmente é apresentada uma definição de grafos semânticos. Como este tipo de grafos pode estar associado a ontologias, a secção 2.1 contém duas subsecções: a subsecção 2.1.1 explica e dá um exemplo de ontologias; a subsecção 2.1.2 aborda alguns grafos semânticos. A parte principal da abordagem seguida tem a ver com a exploração do uso de espaços de nomes nestes grafos. Como tal, a secção 2.2 explica o que são espaços de nomes. Segue-se a secção 2.3, onde é descrito o que são Resource Description Framework (RDF) *statements*. A secção 2.4 mostra um exemplo de uma *query* SPARQL Protocol and RDF Query Language (SPARQL). De seguida, é explicado o que é uma árvore de prefixos e apresentado um exemplo (secção 2.5). O capítulo termina com a secção 2.6 em que é apresentado o formato DOT e a ferramenta Graphviz.

2.1 Grafos semânticos

Conforme referido anteriormente, os grafos semânticos são grafos que guardam informação sobre conceitos e as relações semânticas entre estes. Os conceitos ficam guardados nos nós e as suas relações nos arcos [6].

Estes grafos são expressos em RDF, isto é, o conhecimento sobre um dado domínio é representado por triplos *(sujeito, propriedade, objeto)*. Cada triplo indica que o *sujeito* tem uma determinada *propriedade* e que o valor dessa propriedade é o valor do *objeto* correspondente. Especificando melhor, face ao triplo *(s, p, o)*, conclui-se que o sujeito s tem uma propriedade p cujo valor é o objeto o [1]. O próprio RDF contém propriedades próprias para relacionar os sujeitos com os objetos como, por exemplo, a propriedade *rdf:type*, sendo também possível criar outras propriedades específicas para o domínio em questão.

Os nós do grafo RDF podem ser Internationalized Resource Identifiers (IRIs), literais ou nós brancos. Os literais apenas podem aparecer na posição de objeto do triplo e têm associado um

tipo de dados que indica qual a natureza do conteúdo desse literal. Podem ser, por exemplo, *strings*, inteiros ou datas. Os nós brancos, o elemento RDF menos frequente, são recursos anónimos em que não é atribuído nenhum IRI ou literal e apenas podem aparecer na posição de sujeito e objeto [2].

2.1.1 Ontologia

Os grafos semânticos são associados a um modelo de dados denominado ontologia. As definições de ontologia e de grafos de conhecimento por vezes sobrepõem-se, mas para este trabalho vamos considerar o seguinte: uma ontologia é uma especificação formal e explícita de uma formulação compartilhada que é caracterizada por uma alta expressividade semântica que, por sua vez, é necessária para aumentar a complexidade [7].

Uma ontologia define os tipos que existem no domínio em questão e as propriedades que podem ser usadas para os descrever. Tendo apenas a ontologia definida, não há informação sobre conteúdos específicos do domínio [8]. Por exemplo, uma ontologia que descreve livros será uma descrição formal dos tipos de entidades e relações existentes no domínio dos livros, sendo possível descrever que os livros são escritos por autores. No entanto, neste modelo não teremos informação de livros específicos. Se usarmos esta ontologia como *framework* e acrescentarmos nomes de livros específicos e os respetivos autores, já teremos um grafo semântico sobre livros.

A imagem 2.1 é um exemplo de um grafo semântico sobre livros descrito em RDF. Neste grafo está definido apenas um livro com informação do seu autor e do ano em que foi publicado.

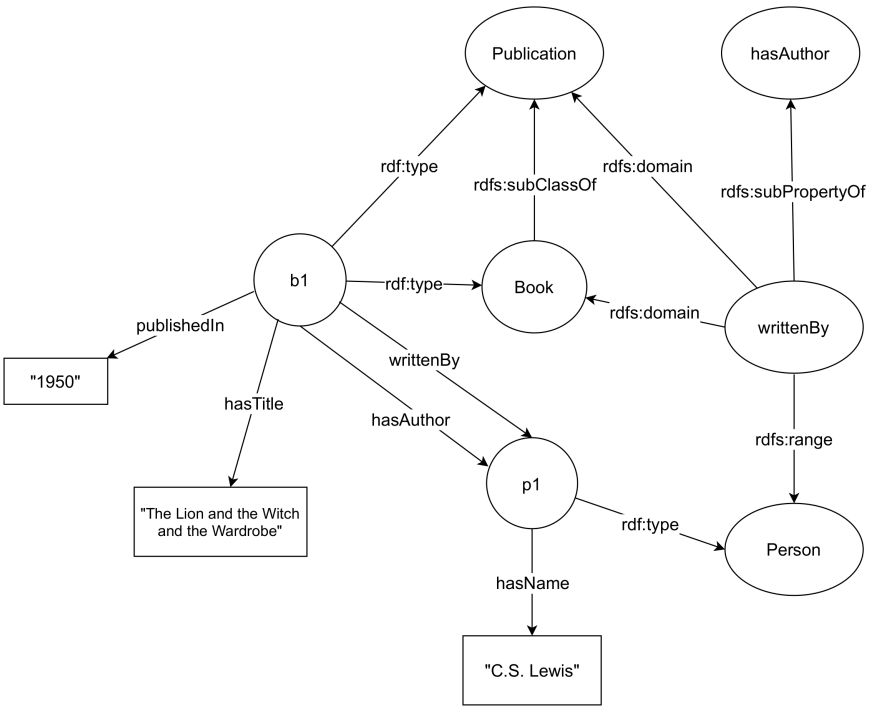


Figura 2.1: Grafo RDF de livros, adaptado de [2]

Para descrever ontologias, utiliza-se uma extensão do vocabulário RDF, o RDF Schema (RDFS). Esta permite criar restrições semânticas entre as classes e as propriedades usadas entre elas [1], bem como, acrescentar aos grafos RDF propriedades próprias para relacionar os nós, como, por exemplo, *rdfs:subClassOf*, *rdfs:subPropertyOf*, *rdfs:domain* e *rdfs:range* [2]:

- `c1 rdfs:subClassOf c2`: todas as instâncias da classe `c1` são instâncias da classe `c2`;
- `p1 rdfs:subPropertyOf p2`: todos os recursos relacionados pela propriedade `p1` também são relacionados pela propriedade `p2`;
- `p rdfs:domain c`: qualquer recurso que tem a propriedade `p` é uma instância da classe `c`;
- `p rdfs:range c`: os valores da propriedade `p` são instâncias da classe `c`.

Cada grafo RDF, como é o caso do grafo representado em 2.1, pode ser dividido em dois grafos mais pequenos: o grafo de ontologia que contém a hierarquia das classes e as suas restrições semânticas e o grafo de instâncias RDF que contém a informação guardada no grafo original [1]. Estes tipos de grafos contêm dois tipos de arcos, os arcos hierárquicos, que são aqueles que ligam dois nós através de uma propriedade hierárquica, como é o caso da propriedade *rdfs:subClassOf*, e os arcos semânticos, que são aqueles criados de acordo com o tema do grafo [9].

Na figura 2.2 está descrito o grafo de ontologia sobre livros, isto é, uma construção hierárquica das classes e as suas relações no domínio de livros. A figura 2.3 contém o grafo de instâncias RDF, isto é, acrescenta informação relativa a livros concretos. Estas duas figuras escritas num ficheiro RDF constituem um grafo de conhecimento de livros.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/20000/01/rdf-schema#> .
@prefix book: <http://example.org/book/> .
@prefix person: <http://example.org/person/> .
@prefix pub: <http://example.org/pub/> .

book:Book rdfs:subClassOf pub:Publication .
person:writtenBy rdfs:domain book:Book ;
                 rdfs:domain pub:Publication ;
                 rdfs:range person:Person ;
                 rdfs:subPropertyOf person:hasAthor .
```

Figura 2.2: Grafo de ontologia sobre livros

```
book:b1 rdf:type book:Book ;
        book:hasTitle "The Lion, the Witch and the Wardrobe" ;
        person:writtenBy person:p1 ;
        person:hasAuthor person:p1 ;
        pub:publishedIn "1950" ;
        rdf:type pub:Publication .
person:p1 person:hasName "C.S. Lewis" ;
          rdf:type person:Person .
```

Figura 2.3: Grafo de instâncias RDF do grafo de livros

2.1.2 Exemplos de grafos RDF

Existem grafos semânticos massivos, isto é, com dimensões na ordem dos biliões de triplos. Por exemplo, a DBpedia, que é gerada extraindo automaticamente informação da Wikipedia e importa informação diretamente da Wikidata, e a Wikidata, que, por sua vez, serve como base de conhecimento para a Wikipedia [4].

Neste trabalho, testamos o processo de sumariação nos três seguintes grafos de grandes dimensões, isto é, na ordem dos milhões de triplos. A KBpedia é um grafo de conhecimento de código aberto que combina as principais bases de conhecimento público (Wikipedia, Wikidata, DBpedia, etc) numa estrutura integrada e computável [10]. A Linked Movie Database (LinkedMDB) é um grafo RDF que contém uma coleção de dados vinculados de filmes, atores, diretores e as relações entre estes [11]. A Wordnet é uma base de dados que guarda palavras no idioma inglês, as suas propriedades lexicais e as relações semânticas que existem entre elas [12].

2.2 Espaços de nomes

Em alguns formatos de serialização de grafos RDF, tais como, RDF-XML e Turtle, os IRIs que identificam nós e arcos podem ser associados a um espaço de nomes. Apesar de apenas existirem nesses formatos, têm significado intrínseco, ou seja, refletem um propósito comum ou uma origem comum dos IRIs. Geralmente, recursos semelhantes descritos num grafo semântico partilham o mesmo espaço de nomes [5].

São atribuídos nomes mais pequenos aos prefixos comuns dos IRIs de modo a identificá-los e a permitir não usar o IRI inteiro mas, sim, o prefixo ao referir os nós e os arcos. Por exemplo, no formato de serialização de RDF Turtle, a definição de um prefixo seria:
`@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.`
Tendo esta declaração, em vez de se usar a propriedade *type* com o IRI inteiro (`http://www.w3.org/1999/02/22-rdf-syntax-ns#type`), pode-se usar a forma abreviada `rdf:type`. Neste exemplo, `rdf` é a etiqueta associada ao espaço de nomes `http://www.w3.org/1999/02/22-rdf-syntax-ns#`. O mapeamento de espaços de nomes e etiquetas são internas a cada

formato de serialização e não têm impacto nas semânticas do grafo [5].

O *website prefix.cc*[1] contém um mapeamento de espaços de nomes e etiquetas e inclui uma pesquisa de espaços de nomes. O mapeamento contido neste *website* corresponde aquele que vários utilizadores costumam usar. Portanto, para cada etiqueta podem existir vários espaços de nomes associados. Para além de ser possível acrescentar espaços de nomes, também permite, de entre os que já existem, votar naquele que melhor corresponde ao prefixo associado ou no que é mais utilizado com determinada etiqueta [13]. Um problema encontrado com este *website* é que guarda vários espaços de nomes que são *substring* de outros.

2.3 RDF *statements*

No RDF pode ser necessário fazer afirmações sobre triplos. RDF*, uma extensão do RDF, está a ser desenvolvido para resolver esta questão [14]. Reificação RDF é outra abordagem que fornece forma de fazer RDF *statements* de modo a expressar algo numa linguagem usando a linguagem, sendo assim tratado pela linguagem [15]. Isto permite construir um triplo em que o sujeito é outro triplo. A figura 2.4 mostra como é possível fazer uma afirmação sobre o triplo `ex:Gardener ex:hasKilled ex:butler`, tornando possível outros triplos referirem o triplo reificado.

```
@prefix ex: <http://example.org/> .
ex:statementExample rdf:type rdf:Statement .
ex:statementExample rdf:subject ex:Gardener .
ex:statementExample rdf:predicate ex:hasKilled .
ex:statementExample rdf:object ex:Butler .
ex:statementExample ex:saidBy ex:Nurse .
```

Figura 2.4: Exemplo de um triplo reificado

2.4 SPARQL Protocol and RDF Query Language

Uma forma de utilizar grafos RDF é através de *queries* [1]. A linguagem que permite fazer *queries* a este tipo de grafos é a SPARQL. Através desta linguagem, é possível, por exemplo, retornar as propriedades associadas a uma determinada classe, consultar as instâncias que fazem parte de uma determinada classe, etc.

Relativamente ao grafo de exemplo apresentado em cima, se quisermos retornar o nome do livro descrito no grafo basta usar a *query* SPARQL 2.1.

Esta *query* terá como resultado apenas o título do livro que está descrito no grafo, como mostra a figura 2.5.

[1] http://prefix.cc

```
PREFIX book: <http://example.org/book>
SELECT ?title
WHERE {
        ?b book:hasTitle ?title .
}
```

Bloco de Código 2.1: Exemplo de uma *query* SPARQL

```
-------------------------------------------
| title                                   |
===========================================
| "The Lion, the Witch and the Wardrobe"  |
-------------------------------------------
```

Figura 2.5: Resultado da *query* anterior

2.5 Árvore de prefixos

Uma *trie* ou árvore de prefixos é uma estrutura de dados em árvore que guarda *strings*. Cada nó da árvore representa um caráter e cada nó folha representa uma palavra que é a concatenação dos caracteres guardados em caminhos desde a raiz até à folha. Permite encontrar *strings* de um conjunto de uma maneira eficiente, suportando operações de inserção e procura. Para procurar uma *string*, começa-se na raiz e desce-se na árvore seguindo os arcos conforme as letras da *string*. Se não houver nenhum arco para a próxima letra significa que a palavra não se encontra na árvore. Para inserir uma *string*, começa-se na raiz à procura da próxima letra da palavra. Se não for encontrada, cria-se um nó para essa letra e um arco ligando o nó anterior ao novo nó. Este processo é repetido até ao fim da palavra [16]. Cada nó da árvore guarda informação se o nó corresponde a fim de palavra.

A figura 2.6 ilustra uma árvore de prefixos que contém as seguintes palavras: triplos, *trie*, rdf.

2.6 DOT e Graphviz

Os grafos podem ser descritos utilizando a linguagem DOT. Esta linguagem é uma representação gráfica baseada em texto que pode ser transformada num diagrama usando ferramentas como o Graphviz [17].

A figura 2.7 representa o grafo de livros 2.1 na linguagem DOT.

Graphviz é um *software* de visualização de grafos de código aberto que permite criar nós e arcos com diferentes tamanhos e cores a partir da descrição em DOT do grafo. Fornece oito mecanismos de *layout* para desenhar o grafo [18].

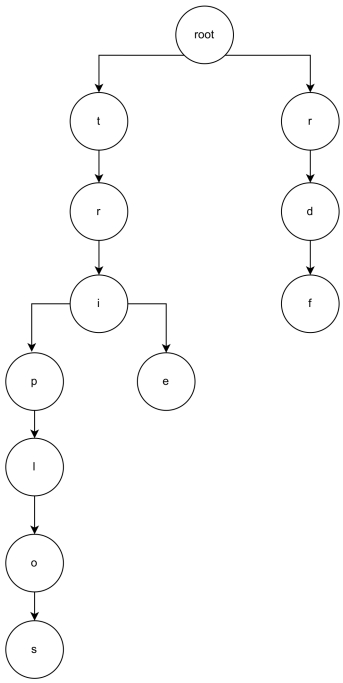


Figura 2.6: Exemplo de uma árvore de prefixos

```
digraph {
    writtenBy -> Book [label = <rdfs:domain>]
    writtenBy -> hasAuthor [label = <rdfs:subPropertyOf>]
    writtenBy -> Person [label = <rdfs:range>]
    writtenBy -> Publication [label = <rdfs:domain>]
    Book -> Publication [label = <rdfs:subClassOf>]
    b1 -> p1 [label = <person:hasAuthor>]
    b1 -> Publication [label = <rdf:type>]
    b1 -> Book [label = <rdf:type>]
    b1 -> p1 [label = <person:writtenBy>]
    b1 -> Title [label = <hasTitle>]
    b1 -> Year [label = <publishedIn>]
    Title[label = <"The Lion, the Witch and the Wardrobe">]
    Year[label = <"1950">]
    p1 -> Person [label = <rdf:type>]
    p1 -> Name [label = <hasName>]
    Name[label = <"C. S. Lewis">]
}
```

Figura 2.7: Representação em DOT do grafo de livros

Capítulo 3

Estado da Arte

Como o motivo de estudo desta dissertação é a sumariação de grafos semânticos, neste capítulo são descritas as aplicações que existem para um sumário de um grafo, os tipos de sumários para grafos em geral e também para os grafos semânticos em particular.

3.1 Aplicações para um sumário

Existem várias aplicações para um sumário de um grafo de grandes dimensões:

- Aumentar a velocidade da aplicação de algoritmos e de *queries* aos grafos [3].
- Eliminar ruído [3];
- Indexar o grafo original [1];
- Estimar o tamanho do resultado das *queries* aplicadas ao grafo original [1];
- Ser capaz de saber previamente se o grafo contém uma certa informação [1];
- Visualizar grafos, que são muito grandes para carregar em memória [3], e o resultado de *queries* de forma mais eficiente [1];
- Extrair um esquema quando o grafo não contém previamente uma ontologia associada [1].

3.2 Sumariação de grafos

Existindo grafos de grandes dimensões, surgiu a necessidade de os sumariar de modo a obter ou um grafo mais pequeno ou um resumo do próprio grafo, indicando caminhos que existem, sub grafos, estatísticas, etc.

Liu et al. [3] apresentam tipos de métodos de sumariação de grafos. Têm em conta técnicas para grafos estáticos em que não é considera informação adicional à sua estrutura e técnicas para

grafos estáticos que contêm atributos nos nós e nos arcos. No que diz respeito aos grafos estáticos com atributos, existem os seguintes métodos: baseados em agregação, baseados em compressão de bits e baseados em influência. Quanto aos grafos estáticos sem atributos, acrescenta-se os métodos baseados em simplificação ou esparsificação. Bonifati et al. [19] abordam também os métodos estatísticos e os orientados para objetivos.

Os métodos baseados em compressão de bits servem para diminuir a quantidade de bits necessária para guardar um grafo. As técnicas baseadas em influência descobrem a descrição da propagação da influência formulando o problema de sumariação como um processo de otimização no qual alguma informação sobre a influência é mantida. Aqueles que são baseados em simplificação ou esparsificação removem os nós ou os arcos "menos importantes", tendo como resultado um grafo esparso [3]. Quanto aos métodos que se baseiam em estatísticas, estes têm em consideração medidas quantitativas e a contagem de ocorrências. Os métodos orientados para objetivos otimizam a quantidade de memória usada produzindo uma representação concisa que cabe em memória [19]. Destacam-se em seguida os métodos baseados em agregação.

3.2.1 Métodos baseados em agregação

Alguns destes métodos consistem em agrupar vários nós que partilhem propriedades num super nó, fazendo com que o grafo sumário contenha propriedades específicas. Outros aplicam *clustering* para mapear cada *cluster* densamente conectado num super nó, minimizando os arcos de *clusters* cruzados [3].

As técnicas que aplicam *clustering* ao grafo ajudam a identificar nós com propriedades semelhantes. Cada *cluster* é constituído por nós que são densamente conectados dentro do mesmo grupo e esparsamente conectados com os nós de outros grupos [19].

Bonifati et al. [19] apresentam duas categorias de sumariação baseada em *clustering*:

- *Clustering* estrutural, que tem em consideração a conectividade do grafo, isto é, as ligações entre os nós.
- *Attributed clustering*, que tem em consideração a topologia, isto é, a estrutura do grafo.

Relativamente ao *clustering* estrutural, há técnicas que implementam algoritmos baseados em partição e em calcular modularidade, densidade e medidas personalizadas aos grafos. Para além destas, existem aquelas que identificam os conjuntos de nós k-medianas que maximizam a probabilidade da conexão média entre cada nó e o centro do *cluster* correspondente [19].

Ainda na mesma categoria, Kuo et al. [20] sugerem um *clustering* orientado para as *queries* que, juntamente com métricas de equilíbrio dos *clusters* e de corte normalizado orientado à *query*, determinam os *clusters* através da relevância dos vários nós em relação à *query* considerada. A métrica de equilíbrio dos *clusters* é uma medida que avalia o quão próximo estão as pontuações dos *clusters* produzidos. A relevância destes deve ser semelhante de forma a que cada um

contenha os vértices com maior relevância ao vértice da *query*. Neste algoritmo, o grafo é dividido em *clusters* tendo cada um arcos *intra-cluster* fortes e arcos *inter-cluster* fracos. Estes arcos são ponderados pela relevância dos seus nós finais para a *query*. O resultado deste tipo de *clustering* deve ser obtido removendo os arcos que se encontram entre nós finais de grande importância (com o intuito de cada *cluster* conter vértices muitos relevantes e evitar a existência de *clusters* com apenas nós irrelevantes), e tendo em consideração que os arcos com pesos grandes devem ser cortados o mínimo possível.

3.3 Sumariação de grafos semânticos

No que diz respeito aos grafos semânticos, de acordo com Čebirić et al. [1], existem os seguintes métodos: estruturais, de *pattern-mining*, estatísticos, híbridos e aqueles que extraem esquemas.

3.3.1 Métodos estruturais

Estes métodos consideram a estrutura do grafo, isto é, os seus caminhos, padrões, nós frequentes, sub grafos, etc. Existem duas categorias de métodos estruturais. A categoria dos sumários de quocientes (*quotients summaries*), baseados em métodos de quocientes (*quotient methods*), permitem caracterizar alguns nós como "equivalentes"e sumariar o grafo atribuindo um representante a cada classe de equivalência dos nós do grafo original. Cada nó do grafo é representado por apenas um nó do sumário e cada nó só pode pertencer a uma classe de equivalência. Cada nó do sumário representa vários nós do grafo original e um arco entre dois nós do sumário representa as relações entre os vários nós do grafo de *input* que estão "dentro"dos nós do sumário. A categoria dos sumários não quocientes (*non-quotients summaries*) é baseada noutras medidas como, por exemplo, a centralidade (que é utilizada para identificar os nós mais importantes), de acordo com a definição que se dá à importância de um nó [1].

Para obter sumários aplicando métodos de quocientes, é utilizada, frequentemente, a noção de bissimulação [1] que permite definir conjuntos de nós do grafo que não podem ser distinguidos observando a sequência de predicados dos seus caminhos [21].

Tran et al. [22] propõem uma abordagem orientada à estrutura que explora os padrões da estrutura que são apresentados pelos dados capturados através de um índice de estrutura. Este índice, que utiliza a noção de bissimulação, é um grafo cujos vértices representam grupos de elementos que são semelhantes relativamente à estrutura. Pode ser utilizado para consultar e navegar nos dados Resource Description Framework (RDF) semi estruturados da web, dividir dados RDF e processar *queries* de nível de estrutura.

Utilizando métodos não quocientes, existem técnicas baseadas em sumariação de texto e recuperação de informação [1], cujo objetivo é extrair informação relevante ao utilizador a partir do conhecimento a priori do utilizador.

3.3.2 Métodos de *pattern-mining*

Estes extraem os padrões mais frequentes do grafo, isto é, o conjunto de instâncias do grafo que partilham um conjunto comum de tipos e de propriedades [1]. Nesta categoria de métodos, existem aqueles que extraem os padrões dos grafos e aqueles que sumariam o grafo baseando-se nas regras derivadas a partir de técnicas de *mining*.

Zneika et al. [23] apresentam uma abordagem que se baseia em determinar um conjunto de padrões aproximados do grafo de *input*. Têm como objetivo descobrir o conjunto de padrões mais pequeno que melhor descreve o grafo original e utilizá-los para construir o sumário, que será também um grafo. Esta abordagem começa por transformar o grafo RDF numa matriz binária, cujas linhas representam os sujeitos e as colunas os predicados. De seguida, são encontrados padrões nesta matriz de modo a criar um sumário do grafo de *input*. Estes padrões são definidos como sendo as propriedades (colunas da matriz) que ocorrem completamente ou parcialmente em vários sujeitos (linhas). Para identificar o conjunto de padrões mais pequeno que melhor descreve o grafo original, é utilizado uma versão modificada do algoritmo PaNDa+, que usa uma estratégia *greedy*, isto é, em cada passo é feita a escolha ótima localmente. Por último, o grafo sumário é construído através dos padrões.

Joshi et al. [24] sugerem tornar o grafo RDF mais pequeno removendo os triplos que podem ser inferidos através de regras lógicas e de inferência. Este método aplica o algoritmo *FP-Growth* de modo a identificar conjuntos de regras de associação. Os triplos que podem ser inferidos através de outras regras são determinados e removidos na criação do sumário.

3.3.3 Métodos estatísticos

Tal como para os grafos em geral, para os grafos semânticos também é possível aplicar medidas estatísticas para definir um sumário como, por exemplo, a contagem de ocorrências e a construção de histogramas de valor por classes e propriedades [1].

Wu et al. [25] propõem um algoritmo que identifica os recursos mais importantes de uma ontologia. Num primeiro passo, o grafo RDF é convertido num grafo de ontologia. De seguida, o algoritmo, aplicado à ontologia, identifica, de maneira iterativa, os conceitos e relações mais importantes. A importância de um conceito é dada através do número de relações que começam nesse conceito, as suas relações com os conceitos mais importantes e o peso dessas relações.

3.3.4 Métodos híbridos

Estes métodos consistem em juntar os métodos anteriormente descritos de modo a obter melhores resultados.

Alzogbi e Lausen [21] propõem um método de sumariação estrutural híbrido que consiste num passo de bissimulação seguido da aplicação de um algoritmo de *clustering* aglomerativo.

3.3.5 Extração de esquemas

Se o grafo não tiver associado uma ontologia, é possível extrair um esquema que servirá como sumário do grafo [1]. Esse esquema fornece informação sobre o conteúdo do grafo, ou seja, que tipos e propriedades é que existem, e ajuda na sua exploração [9].

Kellou-Menouer e Kedad [9] propõem um esquema como sendo um conjunto de definições de tipos e arcos, assente num algoritmo de *clustering* baseado em densidade. O sumário resultante será um grafo de tamanho reduzido contendo nos nós os tipos extraídos e nos arcos as relações entre estes. Esta abordagem, num primeiro momento, gera os tipos que descrevem o grafo e a descrição de cada um deles, denominada de perfis de tipo, tendo cada propriedade uma probabilidade associada, e em seguida gera os arcos semânticos entre os vários tipos como também os arcos hierárquicos analisando os perfis de tipo. Numa primeira fase, é aplicado um algoritmo de *clustering* baseado em densidade para agrupar entidades semelhantes (como se pode ver na figura 3.1: esta mostra, por exemplo, que os diferentes nós cujo tipo era "University"foram agrupados num só nó) e criar o perfil para cada classe. Estes perfis serão usados para encontrar arcos (semânticos e hierárquicos) entre os tipos e para gerar os tipos sobrepostos. Os perfis de tipo são definidos como sendo vetores de propriedades em que cada propriedade tem uma probabilidade associada. Esta probabilidade indica o quão importante é a sua propriedade.

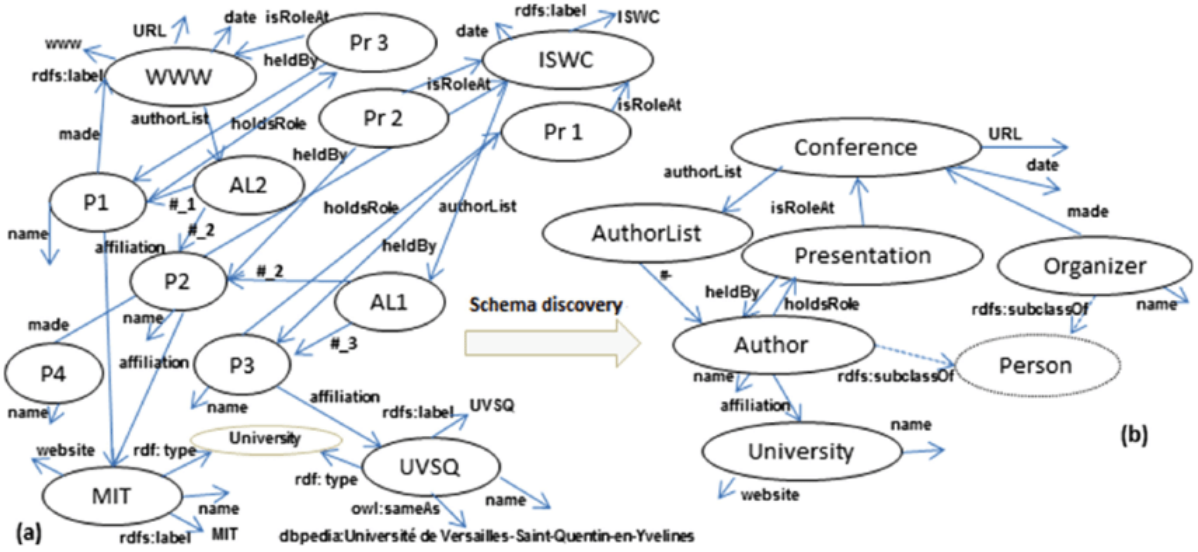


Figura 3.1: Exemplo de um grafo RDF e respetivo esquema, retirado de [9]

Como cada entidade pode ter vários tipos, é necessário gerar os tipos sobrepostos que o grafo contém. Para este fim, é obtido o conjunto de classes disjuntas e produzido as classes sobrepostas analisando os perfis de tipo. De seguida, é gerado tanto os arcos semânticos como os hierárquicos. Para estes últimos, em vez de ser aplicado um algoritmo de *clustering* hierárquico ao grafo inteiro, é produzido a hierarquia dos perfis de tipo, uma vez que esta forma é mais eficiente, dado que o número de perfis de tipo é menor que o tamanho do grafo. A hierarquia de perfis de tipo é formada agrupando tipos em pares de modo a encontrar os seus tipos genéricos, isto é, os tipos definidos através dos dois perfis de tipo mais semelhantes em cada nível da hierarquia.

Bouhamoum et al. [26] adaptam o trabalho anterior de modo a funcionar para grafos extremamente grandes, transformando o grafo numa representação concisa que contém todos os padrões que representam as combinações existentes das propriedades. Estes padrões são definidos como sendo uma lista das propriedades que aparecem em conjunto num ou mais nós. Para obter o esquema, é aplicado um algoritmo de *clustering* a estes padrões, sabendo que se obterá o mesmo resultado e de forma mais rápida em comparação a quando aplicado o algoritmo ao grafo original.

Capítulo 4

Desenho e Desenvolvimento

Neste capítulo é descrito o trabalho realizado. As próximas secções descrevem todo o processo de sumariação. A secção 4.1 detalha como os elementos Resource Description Framework (RDF) são mapeados em grupos. A secção 4.2 mostra como o grafo sumário é produzido. E, por fim, a secção 4.3 aborda o processo de visualização do grafo sumário.

A figura 4.1 contém um diagrama de atividades Unified Modeling Language (UML), definindo as principais fases do processo de sumariação realizado. Num primeiro momento, os elementos RDF de cada triplo são mapeados em grupos, produzindo triplos reduzidos. De seguida, os triplos reduzidos repetidos são contados e é produzido o grafo sumário. O processo termina com a criação de uma visualização do grafo resumo.

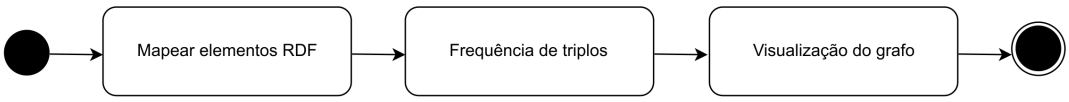


Figura 4.1: Diagrama de atividades UML

4.1 Mapear os elementos RDF em grupos

Os elementos RDF de um triplo podem ser Internationalized Resource Identifiers (IRIs), literais ou nós brancos. Cada um deles será tratado de forma diferente:

IRIs: são reduzidos ao seu espaço de nomes;

Literais: são reduzidos ao seu tipo de dados;

Nós brancos: são reduzidos a um grupo próprio.

A figura 4.1 mostra um triplo de um grafo RDF e a sua conversão para um triplo reduzido de espaços de nomes [5]. A coluna central contém o IRI original com o espaço de nomes sublinhado e a última coluna, a versão reduzida do mesmo IRI com o espaço de nomes substituído por uma etiqueta, que também está sublinhada e separada do sufixo através de ":".

Triplo	**IRI original**	**IRI reduzido**
Sujeito	`http://kbpedia.org/kko/rc/`Abbey	`kko`:Abbey
Predicado	`http://www.w3.org/2000/01/rdf-schema#`subClassOf	`rdfs`:subClassOf
Objeto	`http://dbpedia.org/ontology/`Abbey	`dbo`:Abbey

Tabela 4.1: Exemplo de um triplo e o seu triplo correspondente reduzido a espaços de nomes

Foi utilizada a biblioteca RDFLib da linguagem Python para ler e carregar o grafo RDF em memória e fazer *queries* SPARQL Protocol and RDF Query Language (SPARQL) ao mesmo. Uma vez feito o *parse* do grafo de forma a lê-lo, foi criada a *query* 4.1 para obter todos os triplos do grafo juntamente com o tipo de dados dos objetos. Esta *query* retorna cada um dos três elementos de um triplo (sujeito, propriedade e objeto) nas respetivas variáveis: `?s`, `?p` e `?o`. Na cláusula `Where` é feito um `bind` do tipo de dados de cada objeto para a variável de retorno `?dt`.

```
select ?s ?p ?o ?dt
where {
    ?s ?p ?o .
    bind (datatype(?o) as ?dt)
}
order by ?s ?p ?o ?dt
```

Bloco de Código 4.1: *Query* SPARQL de consulta ao grafo

Para cada triplo resultante da *query* 4.1 foi verificado se os sujeitos e objetos são nós brancos. Na biblioteca RDFLib, os nós brancos são identificados como sendo `term.BNode`. Com a função `type()` do próprio Python, que recebe o sujeito/objeto como argumento, é verificado se se trata de um nó branco (`term.BNode`). Se for, essa informação é guardada numa variável para depois o sujeito/objeto ser reduzido a um grupo próprio denominado `bnode` na fase da produção do grafo sumário.

A redução dos literais ao seu tipo de dados é opcional: os literais só serão reduzidos se à função de sumariação for passada a *string* `"normalize literals"`. Se for para converter os literais, com a *query* 4.1 temos informação do tipo de dados do objeto literal. Assim, na fase de produção do sumário, serão escritos os tipos de dados em vez dos literais. Caso contrário, o grafo sumário irá conter o conteúdo dos literais na íntegra.

Com vista a encontrar os espaços de nomes correspondentes a cada IRI, realizamos a pesquisa desta forma:

1. procurando nas declarações de prefixos do ficheiro do grafo que queremos sumariar;

2. procurando na base de dados de prefixos do *website* Prefix.cc[1];

3. procurando através de correspondência de padrões.

Para o primeiro caso, através da biblioteca RDFLib é possível extrair as declarações de prefixos que o ficheiro do grafo possa conter juntamente com uma série de prefixos da própria biblioteca. A tabela 4.2 mostra os espaços de nomes contidos na biblioteca e as respetivas etiquetas.

Etiquetas	**Espaços de nomes**
brick	https://brickschema.org/schema/Brick#
csvw	http://www.w3.org/ns/csvw#
dc	http://purl.org/dc/elements/1.1/
dcat	http://www.w3.org/ns/dcat#
dcmitype	http://purl.org/dc/dcmitype/
dcterms	http://purl.org/dc/terms/
dcterms	http://purl.org/dc/terms/
doap	http://usefulinc.com/ns/doap#
foaf	http://xmlns.com/foaf/0.1/
odrl	http://www.w3.org/ns/odrl/2/
org	http://www.w3.org/ns/org#
owl	http://www.w3.org/2002/07/owl#
prof	http://www.w3.org/ns/dx/prof/
prov	http://www.w3.org/ns/prov#
qb	http://purl.org/linked-data/cube#
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	http://www.w3.org/2000/01/rdf-schema#
schema	https://schema.org/
skos	http://www.w3.org/2004/02/skos/core#
sosa	http://www.w3.org/ns/sosa/
ssn	http://www.w3.org/ns/ssn/
time	http://www.w3.org/2006/time#
vann	http://purl.org/vocab/vann/
void	http://rdfs.org/ns/void#
xsd	http://www.w3.org/2001/XMLSchema#

Tabela 4.2: Etiquetas e respetivos espaços de nomes extraídos da biblioteca RDFLib

Foi criado um dicionário que guarda as etiquetas nas chaves e os respetivos espaços de nomes nos valores, de acordo com as declarações extraídas através da biblioteca RDFLib. Todos os dicionários que foram criados com o mapeamento de etiquetas e espaços de nomes serão convertidos num ficheiro JavaScript Object Notation (JSON).

[1] http://prefix.cc

Os IRIs que não forem encontrados neste primeiro ficheiro JSON que é criado, são procurados no ficheiro JSON extraído do *website* Prefix.cc, que contém o mapeamento de espaços de nomes e etiquetas para quase 3000 prefixos. Para este ficheiro foi feita uma limpeza manual. Por exemplo, existiam vários espaços de nomes que continham a seguinte parte em comum: `http://data.linkedmdb.org/`, sendo que apenas os carateres finais eram diferentes. Estes vários espaços de nomes foram removidos e foi apenas escrito a parte do IRI comum no mapeamento com uma etiqueta correspondente.

Todos os espaços de nomes reunidos até ao momento são inseridos numa árvore de prefixos, permitindo que a pesquisa pelos IRIs do grafo seja feita de forma eficiente. Os dois ficheiros JSON, que contêm o mapeamento de etiquetas e espaços de nomes, são combinados num só dicionário que será invertido, de modo a ter como chave os espaços de nomes e como valor a respetiva etiqueta, tornando mais direta a pesquisa por espaços de nomes.

Geralmente, os espaços de nomes terminam com um "#"ou uma "/". Para todos os diferentes IRIs que foram retornados pela *query* 4.1, os caracteres que estiverem após o último "#"ou a última "/"são removidos. Estes IRIs mais pequenos são procurados na árvore de prefixos que guarda os espaços de nomes reunidos até ao momento. Quando for encontrada uma correspondência entre um IRI truncado e um espaço de nomes da árvore de prefixos, este último é procurado no dicionário que contém o mapeamento de espaços de nomes e etiquetas que já foram obtidos, de modo a extrair a etiqueta correspondente. Os IRIs truncados que não foram encontrados na árvore de prefixos, são contados tantas vezes quantas aparecem no grafo inteiro. Cada um desses IRIs truncados que apareça várias vezes é escrito como valor para um novo ficheiro JSON, em que a chave correspondente será derivada desse mesmo IRI.

Tendo agora três ficheiros JSON com o mapeamento necessário, estes são combinados num só dicionário que é invertido: as chaves serão os espaços de nomes e os valores as etiquetas. Todos os espaços de nomes reunidos são inseridos numa nova árvore de prefixos.

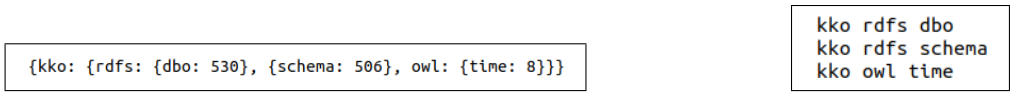
Cada IRI do grafo é procurado nesta nova árvore. Quando encontrado uma correspondência entre o IRI e um espaço de nomes guardado na árvore, esse espaço de nomes é procurado no dicionário, retornando a etiqueta associada.

Existe a possibilidade de serem acrescentados manualmente espaços de nomes ao dicionário que contém o mapeamento, se não forem encontrados de nenhuma das três maneiras anteriores.

4.2 Criação do grafo sumário

As etiquetas dos espaços de nomes, bem como o tipo de dados dos literais e o nó `bnode` para reduzir os nós brancos, à medida que vão sendo encontrados, são escritos para um dicionário da seguinte forma: `{sujeito: {predicado: {objeto: contagem}}}`. Para além de serem guardados neste dicionário os triplos reduzidos, é também guardado o número de triplos do grafo original que são convertidos ao mesmo triplo reduzido.

A figura 4.2 mostra como os triplos guardados no dicionário ficam escritos num ficheiro.



(a) Exemplo do dicionário (b) Exemplo de novos triplos de espaços de nomes

Figura 4.2: Exemplo do dicionário e dos triplos que lá estão guardados

Esta imagem ilustra que o triplo reduzido `kko rdfs dbo` corresponde a 530 triplos do grafo original. O triplo reduzido `kko rdfs schema` corresponde a 506 triplos e o triplo `kko owl time` corresponde a 8 triplos originais.

Com a informação da contagem guardada no dicionário, são criadas RDF *statements* para afirmar qual a frequência de cada triplo reduzido no grafo inicial. O grafo sumário será um grafo RDF definido pelos triplos reduzidos reificados para afirmar a sua frequência.

A figura 4.3 ilustra a ontologia associada aos triplos reduzidos. Aqui está descrito o predicado `ngs:num_occurrences` que é utilizado para indicar quantos triplos originais foram convertidos ao mesmo triplo reduzido. Todos os recursos que têm `ngs:num_occurrences` como propriedade são instâncias da classe `ngs:reified_triple` e os valores desta propriedade são todos inteiros.

```
@prefix ngs: <https://ww.dcc.fc.up.pt/~up201605706/ngs#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

ngs:ReifiedTriple rdfs:subClassOf rdf:Statement ;
                  rdfs:comment "RDF statement about the reduced triple".

ngs:num_occurrences rdfs:domain ngs:ReifiedTriple ;
                    rdfs:range xsd:integer ;
                    rdfs:comment "Indicates how many originals triples
                  are converted into the same reduced triple".
```

Figura 4.3: Ontologia associada ao grafo sumário

A figura 4.4 contém uma RDF *statement* que indica que o triplo reduzido `ngs:kko ngs:rdf ngs:dbo` corresponde a 530 triplos do grafo original.

4.3 Visualização do grafo sumário

Após a produção do grafo sumário, seguiu-se a criação da sua visualização. O ficheiro que contém o grafo resumo foi convertido para um ficheiro DOT de modo a utilizar a ferramenta Graphviz

```
ngs:t1 rdf:type rdf:Statement
ngs:t1 rdf:subject ngs:kko
ngs:t1 rdf:predicate ngs:rdfs
ngs:t1 rdf:object ngs:dbo
ngs:t1 ngs:num_occurrences 530
```

Figura 4.4: Exemplo de uma RDF *statement* sobre um triplo reduzido do grafo sumário

para desenhar o grafo. Foram experimentados os vários mecanismos de *layout* do Graphviz no desenho do grafo. No entanto, desta forma não se estava a conseguir obter a visualização com as características desejadas.

Assim, para produzir a visualização do grafo sumário foi utilizado um *software* desenvolvido pelo supervisor da dissertação. As visualizações criadas por este *software* têm as seguintes características:

- Tamanho dos nós: os nós que aparecem mais frequentemente no grafo sumário são representados num tamanho maior;
- Cor dos nós: os grupos que correspondem a tipos de dados de objetos literais são representados numa cor diferente;
- Espessura dos arcos: os triplos com um maior número de ocorrências no grafo sumário são representados através de arcos mais espessos;
- Cor dos arcos: os arcos são desenhados com diferentes cores, cada uma correspondente a um grupo diferente.

Com estes recursos visuais é possível observar a frequência dos espaços de nomes usados nos nós e arcos e a conexão entre estes, tornando possível também detetar grafos desconexos e componentes fortemente conectadas.

O ficheiro com as RDF *statements* sobre os triplos reduzidos gerado pelo processo de sumariação é lido, de modo a convertê-lo para um ficheiro DOT. Cada sujeito, predicado, objeto e respetiva contagem descritos nessas RDF *statements* é guardado em diferentes dicionários, dependendo se correspondem a nós, a arcos ou à frequência do triplo reduzido. Cada um destes dicionários é percorrido de modo a escrever os nós e os arcos no formato DOT, onde lhe são atribuídas cores e tamanhos conforme o descrito acima. No desenho do grafo, os nós, de uma forma geral, contêm os seus nomes dentro deles, enquanto que os nomes dos arcos são escritos numa tabela de acordo com as cores associadas, tabela esta que constitui a legenda do grafo. Aos nós cujos nomes são demasiado grandes para serem escritos dentro do próprio nó, são atribuídos números e adicionados à legenda.

A frequência dos nós e dos arcos é obtida através das RDF *statements*. A frequência dos arcos é obtida diretamente através das mesmas, uma vez que estas indicam, para cada triplo reduzido, a quantos triplos do grafo original correspondem. A frequência dos nós é derivada das referidas RDF *statements* da seguinte forma: o número de ocorrências descrito em cada RDF *statements* que contém esse nó é somado independentemente da sua posição no triplo, isto é, o nó tanto é contado enquanto sujeito como enquanto objeto.

Ao criar a visualização dos grafos sumários foi adicionado um tamanho mínimo para os nós de modo a ser possível desenhá-los. Isto significa que as imagens produzidas têm diferentes grupos representados por nós pequenos, que apesar de igual tamanho, podem não aparecer o mesmo número de vezes no grafo original.

Capítulo 5

Validação

Neste capítulo são apresentadas *queries* SPARQL Protocol and RDF Query Language (SPARQL) que se desenvolveram de forma a explorar os grafos que iam ser utilizados para testar o processo de sumariação. De seguida, são expostos e discutidos os resultados obtidos quando aplicado o método de sumariação a determinados grafos Resource Description Framework (RDF).

Dado que a biblioteca utilizada para processar os grafos RDF (RDFLib) precisa de carregar os grafos inteiros para memória, não foi possível testar o método para grafos massivos, como é o caso da DBpedia com mais de 1 bilião de triplos. Portanto, escolheram-se os seguintes grafos de tamanho grande, com dimensões menores que os massivos:

- KBpedia de 1 175 147 triplos;
- Wordnet de 2 637 168 triplos;
- Linked Movie Database de 3 579 531 triplos.

Um fator que esteve na base da escolha destes grafos foi o facto de estes serem de grandes dimensões, embora não considerados massivos, uma vez que as suas dimensões são da ordem de grandeza dos milhões de triplos. Escolheu-se a KBpedia por incluir o mapeamento para as bases de conhecimento principais, ou seja, este grafo contém cobertura quase completa para a Wikidata e a Wikipedia [10]. Optou-se pela Wordnet por ser um grafo que, ao guardar palavras no idioma inglês e as suas relações semânticas, é muito utilizado em sistemas de processamento de linguagem natural [12]. A Linked Movie Database foi escolhida por ser um grafo que contém informação de filmes, que constitui algo comum e apreciado pela maioria das pessoas, e por ligar os seus recursos aos correspondentes da DBpedia, um grafo massivo, e a *websites* de filmes muito conhecidos (como é o caso do IMDb.com e o RottenTomatoes.com) [11].

A secção 5.1 apresenta as *queries* feitas de modo a explorar os grafos RDF descritos acima com os respetivos resultados. Cada um dos grafos será apresentado numa secção diferente, na qual se expõe e explica a respetiva visualização. Apresentam-se também os resultados obtidos

numa tabela na última secção. Normalizar literais corresponde à sua conversão nos seus tipos de dados.

5.1 *Queries* SPARQL de exploração dos grafos RDF

Num primeiro momento, com o intuito de explorar os grafos RDF escolhidos, foram realizadas algumas *queries* SPARQL.

A *query* 5.1 retorna o número total de triplos dos grafos.

```
select (count(*) as ?num)
where {
  ?s ?p ?o
}
```

Bloco de Código 5.1: *Query* SPARQL que retorna o número total de triplos

A *query* 5.2 retorna o número total de objetos literais que existem, podendo haver repetidos.

```
select (count(?o) as ?num)
where {
   ?s ?p ?o .
   filter (isLiteral(?o))
}
```

Bloco de Código 5.2: *Query* SPARQL que retorna o número total de literais

A *query* 5.3 retorna o número total de recursos únicos independentemente da posição do triplo onde aparecem.

```
select (count(distinct ?rec) as ?num)
where {
    { ?rec ?p ?o . }
    union
    { ?p ?rec ?o . }
    union
    { ?o ?p ?rec . }
}
```

Bloco de Código 5.3: *Query* SPARQL que retorna o número total de recursos únicos

A tabela 5.1 contém os resultados obtidos para cada uma destas *queries*: número total de triplos, de literais e de recursos únicos.

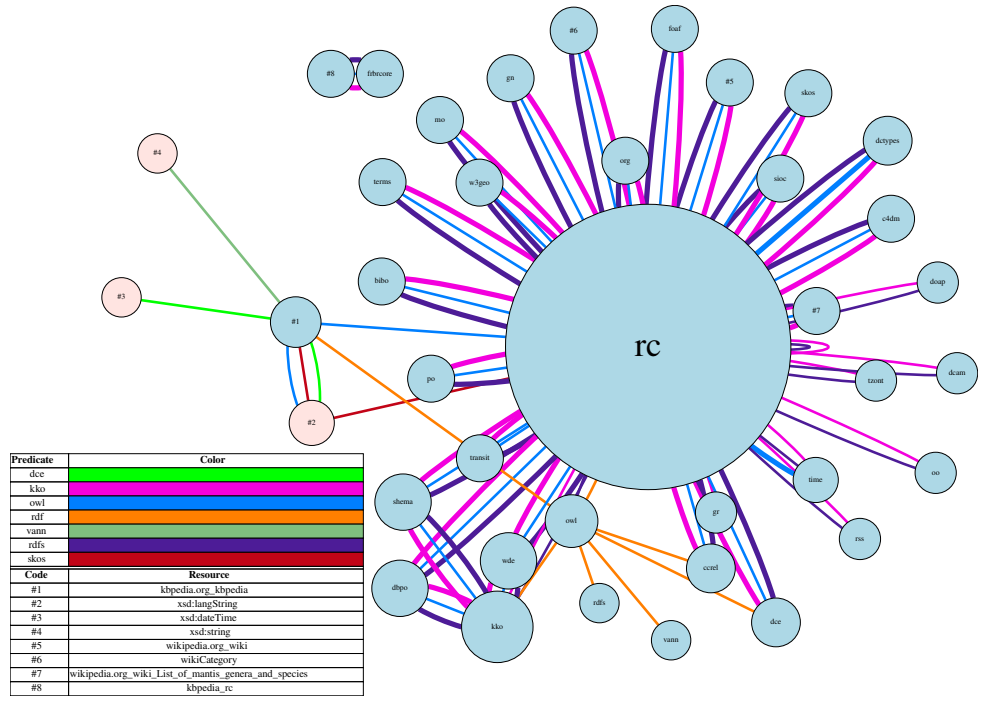
Grafo	#triplos	#literais	#recursos únicos
KBpedia	1 175 147	471 236	587 616
Wordnet	2 637 168	541 777	1 292 039
LinkedMDB	3 579 531	1 912 513	1 405 307

Tabela 5.1: Estatísticas dos grafos

5.2 Grafo sumário da KBpedia

O processo de sumariação, ao converter os literais nos seus tipos de dados para o grafo KBpedia, conseguiu obter 164 triplos reduzidos a partir dos 1 175 147 originais. Ao escrever o conteúdo dos literais na íntegra no grafo sumário, obtiveram-se 442 295 triplos reduzidos.

A figura 5.1 mostra a visualização criada para o grafo sumário do grafo KBpedia quando se converteram os literais nos seus tipos de dados.



Predicate	Color
dce	
kko	
owl	
rdf	
vann	
rdfs	
skos	

Code	Resource
#1	kbpedia.org_kbpedia
#2	xsd:langString
#3	xsd:dateTime
#4	xsd:string
#5	wikipedia.org_wiki
#6	wikiCategory
#7	wikipedia.org_wiki_List_of_mantis_genera_and_species
#8	kbpedia_rc

Figura 5.1: Visualização do grafo sumário da KBpedia de 164 triplos

- O nó **rc**, que representa o espaço de nomes `http://kbpedia.org/kko/rc/`, é o nó mais central e o que é mencionado mais frequentemente;
- O segundo maior nó (grupo **kko**) corresponde ao espaço de nomes `http://kbpedia.org/ontologies/kko#` que especifica elementos da própria ontologia;
- O espaço de nomes utilizado pela KBpedia para identificar o próprio grafo é o correspondente ao grupo **#1** (espaço de nomes `http://kbpedia.org/kbpedia/`) que se conecta ao nó **rc** e a três grupos de literais: **xsd:dateTime**, **xsd:string** e **xsd:langString**;

- A maior parte dos predicados do sumário vem de três grupos diferentes: **rdfs** (espaço de nomes `http://www.w3.org/2000/01/rdf-schema#`), **kko** e **owl** (espaço de nomes `http://www.w3.org/2002/07/owl#`);
- O grupo **#8**, correspondente ao espaço de nomes `http://kbpedia.org/kbpedia/rc/`, conecta-se apenas ao grupo **frbrcore**, que representa o espaço de nomes `http://purl.org/vocab/frbr/core#`. Os triplos reduzidos que contêm estes dois nós estão desconexos do resto do grafo sumário;
- O grupo de literal predominante da KBpedia é o **xsd:langString**. Este tipo de *strings* têm indicação do idioma a que pertencem. De modo a saber quantos objetos literais deste tipo o grafo contém, foi desenvolvida a *query* 5.4. Observando os resultados obtidos apresentados na tabela 5.2, conclui-se que este tipo de literais corresponde a 40% do grafo;
- Existem vários triplos na KBpedia cujo sujeito e objeto são reduzidos ao mesmo grupo. No grafo sumário estão representadas essas ligações (lacetes): o nó **rc** liga-se a ele próprio através do grupo **rdfs** e do grupo **kko**.

A *query* 5.4 retorna o número total de literais agrupados pelo tipo de dados, isto é, indica para cada tipo de dados quantos literais dessa natureza existem.

```
select ?dt (count(?o) as ?num)
where {
   ?s ?p ?o .
   bind (datatype(?o) as ?dt)
   filter (isLiteral(?o))
}
group by ?dt
```

Bloco de Código 5.4: *Query* SPARQL que agrupa o número de literais por tipo

Com esta última *query*, obtiveram-se os resultados apresentados na tabela 5.2 para cada um dos grafos utilizados.

Grafo	xsd:langString	xsd:string	xsd:dateTime	xsd:int	xsd:double
KBpedia	471 233	2	1	——	———
Wordnet	423 986	117 791	———	——	———
LinkedMDB	———	1 516 700	———	395 567	246

Tabela 5.2: Número total de literais por tipo

5.3 Grafo sumário da Wordnet

Em relação ao grafo Wordnet, foram produzidos apenas 16 triplos reduzidos dos 2 637 168 iniciais convertendo os literais nos seus tipos de dados. Quanto a não normalizar os literais, obtiveram-se

266 666 triplos reduzidos.

A figura 5.2 mostra a visualização criada para o grafo sumário do grafo Wordnet quando se converteram os literais nos seus tipos de dados.

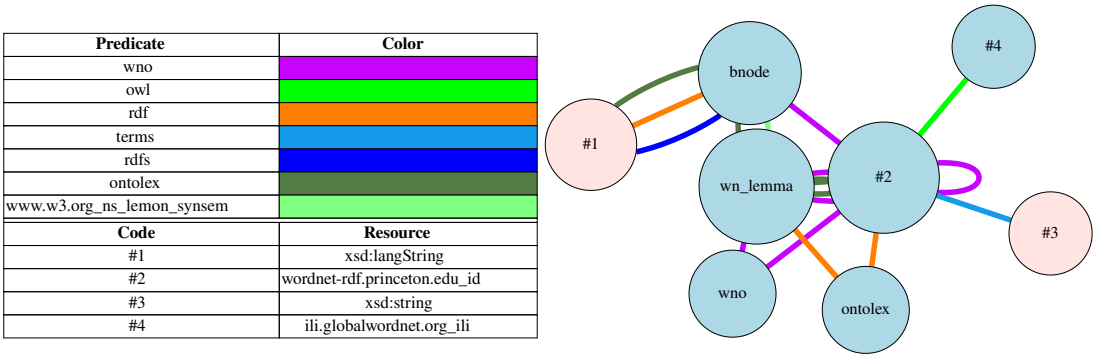


Figura 5.2: Visualização do grafo sumário da Wordnet de 16 triplos

- Os nós mais frequentes correspondem ao grupo **wn_lemma**, que representa o espaço de nomes `http://wordnet-rdf.princeton.edu/rdf/lemma/`, e ao grupo **#2**, que corresponde ao espaço de nomes `http://wordnet-rdf.princeton.edu/id/`;
- Existem dois grupos de literais: **xsd:string** e **xsd:langString**. Com os resultados da *query* 5.4 apresentados na tabela 5.2, verifica-se que estes dois tipos de literais constituem 20,5% do grafo;
- O segundo nó mais frequente é o grupo **bnode**, que identifica os nós brancos. De modo a saber quantos nós brancos o grafo contém, foi feita a *query* 5.5. Tendo um total de 847 972 nós brancos, observa-se que estes constituem 32% do grafo;
- Os nós brancos ligam-se a nós que guardam conceitos e pequenas definições de conceitos (grupo **xsd:langString** que está representado pelo grupo **#1**) através de três grupos diferentes: **ontolex** (espaço de nomes `http://www.w3.org/ns/lemon/ontolex#`), **rdf** (espaço de nomes `lhttp://www.w3.org/1999/02/22-rdf-syntax-ns#`), **rdfs**;
- A Wordnet contém vários sujeitos e objetos que se reduzem ao mesmo grupo: o nó **wn_lemma** conecta-se a ele próprio através do grupo **ontolex** e do grupo **wno** (espaço de nomes `http://wordnet-rdf.princeton.edu/ontology#`); e o nó **#2** liga-se a ele próprio pelo grupo **wno**.

A query 5.5 retorna o número total de nós brancos existentes no grafo.

5.4 Grafo sumário da LinkedMDB

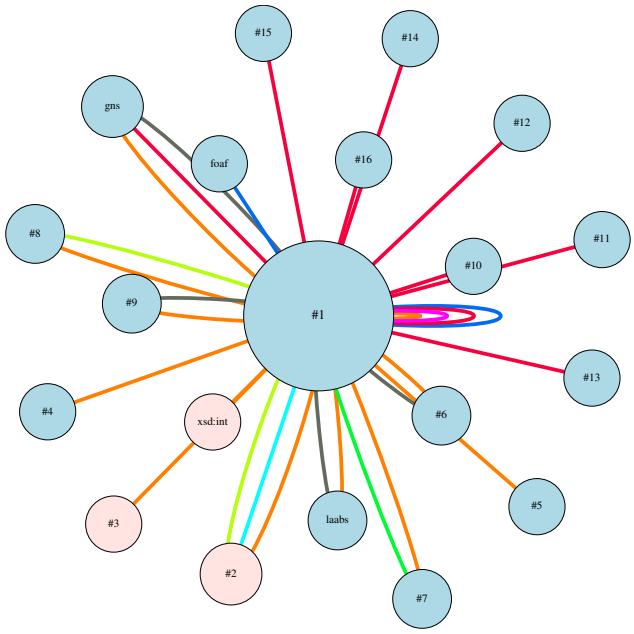
Relativamente ao grafo LinkedMDB, ao normalizar os literais, o método obteve 41 triplos reduzidos dos 3 579 531 originais. Quanto a não converter os literais, foram obtidos 967 941 triplos

```
select (count(?r) as ?n)
where {
   { ?r ?p ?o }
   union
   { ?s ?p ?r }
   filter(isBlank(?r))
}
```

Bloco de Código 5.5: *Query* SPARQL que retorna o número total de nós brancos

reduzidos.

A figura 5.3 mostra a visualização criada para o grafo sumário do grafo LinkedMDB quando se converteu os literais nos seus tipos de dados.



Predicate	Color
dbpedia2	
skos	
linkedmdb	
rdf	
terms	
owl	
rdfs	
foaf	

Code	Resource
#1	linkedmdb
#2	xsd:string
#3	xsd:double
#4	www.lingvoj.org_lingvo
#5	www4.wiwiss.fu-berlin.de_bookmashup_books
#6	mpii.de_yago_resource
#7	www4.wiwiss.fu-berlin.de_flickrwrappr_photos
#8	www4.wiwiss.fu-berlin.de_bookmashup_persons
#9	zitgist.com_music_artist
#10	www.freebase.com_view_guid
#11	www.imdb.com_name
#12	imdb.com_name
#13	www.imdb.com_title
#14	www.freebase.com_view_topic_en
#15	www.rottentomatoes.com
#16	imdb.com_title

Figura 5.3: Visualização do grafo sumário da LinkedMDB de 41 triplos

- O nó mais central e mais frequente corresponde ao grupo **#1**, que representa ao espaço de nomes `http://data.linkedmdb.org/`;
- A maior parte dos predicados do sumário vem de dois grupos diferentes: o grupo **linkedmdb** e o grupo **foaf** (espaço de nomes `http://xmlns.com/foaf/0.1/`);
- O nó **#1**, através do grupo de predicado **foaf**, liga-se a vários nós que representam espaços de nomes de algumas base de dados de filmes. Ilustra que a LinkedMDb tem ligação ás seguintes base de dados: IMDb.com, RottenTomatoes.com e Freebase;
- O grupo **linkedmdb** especifica elementos da própria LinkedMDB;

- O grupo **laabs**, correspondendo ao espaço de nomes `http://dbpedia.org/resource/`, representa recursos da DBpedia;
- A LinkedMDB faz ligação entre os seus recursos e os correspondentes na DBpedia através dos seguintes triplos reduzidos: **linkedmdb linkedmdb laabs** e **linkedmdb owl laabs**;
- Existem vários sujeitos e objetos que foram reduzidos ao mesmo grupo: o nó **#1** conecta-se a ele próprio através dos grupos **rdf**, **foaf**, **linkedmdb** e **skos** (espaço de nomes `http://www.w3.org/2004/02/skos/core#`;
- Os literais utilizados na LinkedMDB são o **xsd:string**, o **xsd:int** e o **xsd:double**;
- Realizada a *query* 5.4 e uma vez obtidos os resultados expostos na tabela 5.2, conclui-se que o grupo de literal **xsd:string** é o mais frequente, constituindo 42% do grafo; o grupo **xsd:int** corresponde apenas a 11% do grafo e o grupo **xsd:double** é o grupo de literais menos frequente, com apenas 0.007% do grafo.

5.5 Desempenho

A tabela 5.3 apresenta os resultados obtidos para cada um dos grafos RDF: o número de triplos reduzidos que constituem o grafo sumário, o tempo aproximado que o programa demorou a correr até produzir um grafo menor, a percentagem de redução que foi obtida da seguinte maneira: $\frac{\#triplos - \#triplos\_reduzidos}{\#triplos} * 100$ e o número de triplos processados por segundo: $\frac{\#triplos}{tempo}$.

Tabela 5.3: Tabela de resultados

Grafo	#Triplos	Normalizar literais?	#Triplos Reduzidos	%Redução de triplos	Duração	#Triplos/s
Wordnet	2 637 168	Sim	16	100	0h06	7 325,47
		Não	266 666	89,89	0h20	2 197,64
LinkedMDB	3 579 531	Sim	41	100	0h9	6 628,76
		Não	967 941	72,96	0h42	1 420,45
KBpedia	1 175 147	Sim	164	99,99	0h04	4 896,45
		Não	442 295	62,36	01h00	326,43

Observa-se na tabela de resultados que quando não se normalizam os literais, a criação do grafo sumário demora sempre mais tempo em comparação com quando se agrupam os literais nos seus tipos de dados. Como a maior parte dos literais destes grafos são do tipo *string*, o tamanho dessas *strings* tem influência na duração da execução do método de sumariação, visto que, o conteúdo desses literais vai ser escrito na sua íntegra.

O processo de sumariação, relativamente à normalização dos objetos literais, reduziu 7 325,47 triplos por segundo do grafo Wordnet, tendo em 6 minutos obtido 16 triplos reduzidos dos 2 637 168 triplos iniciais, o que corresponde a uma taxa de redução de 100%. Para o caso de não

reduzir os literais, processou-se 2 197,64 triplos por segundo, tendo-se obtido em 20 minutos 266 666 triplos reduzidos, equivalendo a uma redução de 89,89%.

O grafo Wordnet tem, aproximadamente, o dobro (um pouco mais) do tamanho do grafo KBpedia. No entanto, a criação do grafo sumário para a Wordnet sem normalizar os literais demora muito menos tempo do que a criação do grafo sumário para a KBpedia sem normalizar os literais. Isto pode ser explicado pelo facto de os literais da KBpedia, maioritariamente *strings*, terem um conteúdo muito maior que os literais *strings* da Wordnet, precisando de muito mais tempo para serem todos processados e escritos no grafo sumário.

No que diz respeito ao grafo LinkedMDB, quando se normalizou os objetos literais, houve 6 628,76 triplos a serem reduzidos por segundo; em 9 minutos, os 3 579 531 triplos iniciais converteram-se em 41 triplos reduzidos, alcançando uma redução de 100%. Quando os literais foram deixados com o seu conteúdo original no sumário, foram reduzidos 1 420,45 triplos por segundo, obtendo-se 967 941 triplos reduzidos em 42 minutos, o que equivale a uma redução de 72,96%.

A LinkedMDB é o maior dos três grafos, tendo, aproximadamente, o triplo (um pouco mais) do tamanho da KBpedia. Apesar do seu maior tamanho e de conter o maior número de literais, a criação do respetivo sumário sem normalizar os literais não é o mais demorado. A KBpedia, tal como referido anteriormente, guarda *strings* grandes, o que faz com que este seja o grafo com duração maior, quando não se reduz os literais.

Relativamente ao grafo KBpedia, ao normalizar os literais, reduziu-se 4 896,45 triplos por segundo, obtendo 164 triplos reduzidos dos 1 175 147 iniciais em 4 minutos, atingindo assim uma redução de 99,99%. Em relação a não converter os literais nos seus tipos de dados, esta técnica reduziu 326,43 triplos por segundo, conseguindo 442 295 triplos reduzidos em 60 minutos, o que corresponde a uma taxa de redução de 62,36%.

A Kbpedia tem mais de 400 mil literais distintos, sendo quase todos do tipo *string*. A maioria destas *strings* contêm um grande número de caracteres. Como tal, ao converter os triplos originais em triplos reduzidos, foram apenas processados, aproximadamente, 327 triplos por segundo, um número muito inferior aos restantes nesta coluna da tabela de resultados.

5.6 Reflexão final

O grafo sumário da Wordnet é o mais pequeno dos três sumários produzidos. Isto pode ser explicado pelo facto da Wordnet ter poucas ligações para outras bases de dados. A KBpedia contém vários recursos da Wikipedia e a LinkedMDB conecta-se a algumas bases de dados de filmes, ligando-se também a recursos da DBpedia.

A Wordnet guarda conceitos e definições curtas desses conceitos, fazendo com que as *strings* do grafo tenham poucos caracteres, o que permite que o sumário, não reduzindo os literais, seja obtido rapidamente. As strings guardadas na LinkedMDB são relativas a filmes e a aspetos que

se associam aos mesmos, tendo, portanto, um conteúdo mais curto e passível de escrever mais rapidamente na íntegra no sumário face à KBpedia, que guarda *strings* vindas da Wikipedia.

O método de sumariação não é perfeito. A KBpedia contém vários Internationalized Resource Identifiers (IRIs) que têm como *substring* o seguinte: `http://wikipedia.org/wiki/`, indicando que representam algo no espaço de nomes da Wikipedia. No entanto, este método não conseguiu reduzir todos esses IRIs ao espaço de nomes `http://wikipedia.org/wiki/`, acabando por os reduzir a três espaços de nomes diferentes que estão representados pelos grupos: **#5** (**wikipedia.org_wiki**), **#6** (**wikiCategory**) e **#7** (**wikipedia.org_wiki_List_of_mantis_genera_and_species**). Estes três grupos correspondem, respetivamente, aos seguintes espaços de nomes: `http://wikipedia.org/wiki/Category:`, `http://wikipedia.org/wiki/` e `http://wikipedia.org/wiki/List_of_mantis_genera_and_species#`.

Capítulo 6

Conclusões

Os grafos semânticos de grandes dimensões, com mais de um milhão de triplos, são difíceis de entender e visualizar. Uma solução para resolver esta questão consiste em resumir os grafos criando um grafo mais pequeno. A maioria dos métodos de sumariação existentes agrupam os nós em super nós e os arcos em super arcos de acordo com um dado critério, reduzindo assim o tamanho do grafo.

Os objetivos do trabalho realizado passaram pela produção de sumários para grafos semânticos de grandes dimensões que fossem obtidos em poucas horas, não excedendo um dia. Este grafo sumário tem a finalidade de dar a conhecer o grafo do grafo Resource Description Framework (RDF) original, que não seria possível de outra forma, dado o seu grande volume de dados. Pretendeu-se reduzir o tamanho do grafo inicial através da junção de nós do mesmo tipo em super nós e dos arcos em super arcos. O objetivo final consistiu, assim, em criar uma visualização para o grafo sumário que fosse facilmente obtida e que permitisse tirar conclusões sobre que grupos é que se relacionam entre si e observar a frequência desses mesmos grupos.

Nesta abordagem, os elementos RDF que constituem os triplos foram mapeados em diferentes grupos de acordo com o seu tipo: os Internationalized Resource Identifiers (IRIs) foram reduzidos aos seus espaços de nomes, os literais ao seu tipo de dados (opcionalmente) e os nós brancos a um grupo particular. O resultado deste mapeamento é um conjunto de triplos dos elementos mapeados, em que a maioria dos triplos estão repetidos várias vezes. O grafo sumário é um grafo RDF em que cada triplo é reificado de modo a afirmar quantos triplos do grafo original foram convertidos para esse mesmo triplo reduzido. Por fim, o grafo sumário foi convertido para a linguagem de descrição de grafos DOT de forma ser possível a criação da sua visualização. Esta visualização foi produzida com determinados recursos visuais que facilitam observar a frequência dos nós e dos arcos e tirar conclusões mais rapidamente.

O trabalho aqui proposto explora o uso de espaços de nomes para obter super nós e super arcos para grafos RDF, contribuindo com uma técnica de sumariação para este tipo de grafos de grandes dimensões e um módulo no Python Package Index (PyPI), o repositório de software oficial de terceiros para Python, que implemente essa técnica. O módulo é denominado **rdf_summarizer**

e está disponível neste endereço: https://pypi.org/project/rdf-summarizer/. No contexto desta dissertação foi escrito e publicado um artigo, *Large Semantic Graph Summarization using Namespaces* [5], que detalha o processo de sumariação criado e os resultados obtidos até à data da submissão. O artigo está disponível em: https://drops.dagstuhl.de/opus/volltexte/2022/16758/.

O processo de sumariação foi aplicado a três grafos RDF de grandes dimensões: grafo de conhecimento KBpedia (com mais de 1 milhão de triplos), Wordnet (com mais de 2 milhões de triplos) e Linked Movie Database (com mais de 3 milhões de triplos). Para cada um destes grafos, convertendo os objetos literais nos respetivos tipos de dados, foi possível obter um grafo sumário pequeno e em pouco tempo e foi criada a respetiva visualização.

Com a análise dos resultados obtidos, conclui-se que normalizar os literais melhora o número de triplos reduzidos, bem como o tempo de duração da execução do método de sumariação. Efetivamente, verificou-se que a quantidade de literais existentes e o seu tamanho, quando são do tipo *string*, têm impacto na quantidade de triplos que são processados por segundo e no tempo que demora o grafo sumário a ser criado. A análise de cada uma das visualizações do grafo sumário permitiu tirar conclusões específicas e relevantes sobre cada grafo original.

6.1 Trabalho Futuro

O trabalho futuro consiste em tornar o processo de sumariação mais eficiente de modo a ser possível aplicá-lo a grafos semânticos massivos, com mais de 1 bilião de triplos, como é o caso da DBpedia e a Wikidata. Nesta dissertação é utilizada a biblioteca RDFLib do Python, em que é necessário carregar o grafo inteiro para memória de forma a poder-se trabalhar com ele. Esta maneira de ler o grafo funciona para os grafos de grandes dimensões que foram usados. No entanto, não é uma boa prática quando se utiliza grafos massivos, de dimensões maiores. Uma melhoria a implementar passa por ler o grafo original triplo a triplo, sendo feita a conversão em grupos à medida que se lê cada triplo.

Outro melhoramento a implementar é tornar interativa a fase da criação dos ficheiros JavaScript Object Notation (JSON) que guardam o mapeamento de espaços de nomes e etiquetas. Ou seja, à medida que se encontram IRIs truncados que não correspondem a nenhum espaço de nomes do mapeamento, que se obteve de três maneiras diferentes, o utilizador poderá decidir se quer adicionar algum desses IRIs a esse mapeamento. Assim, poderá ser melhorada a quantidade de triplos reduzidos.

Algo a melhorar na visualização dos grafos sumários é torná-la também interativa: o utilizador pode filtrar a imagem de modo a ver apenas determinadas coisas, nomeadamente, ver apenas os nós e arcos mais frequentes, ver apenas os nós que representam tipos de literais e suas ligações, ver apenas os espaços de nomes, etc.

Bibliografia

[1] Šejla Čebirić, François Goasdoué, Haridimos Kondylakis, Dimitris Kotzinos, Ioana Manolescu, Georgia Troullinou, and Mussab Zneika. Summarizing semantic graphs: a survey. *The VLDB journal*, 28(3):295–327, 2019.

[2] Haridimos Kondylakis, Dimitris Kotzinos, and Ioana Manolescu. Rdf graph summarization: principles, techniques and applications. In *EDBT*, pages 433–436, 2019.

[3] Yike Liu, Tara Safavi, Abhilash Dighe, and Danai Koutra. Graph summarization methods and applications: A survey. *ACM computing surveys (CSUR)*, 51(3):1–34, 2018.

[4] Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web*, 9(1):77–129, 2018.

[5] Ana Rita Santos Lopes da Costa, André Santos, and José Paulo Leal. Large semantic graph summarization using namespaces. In *11th Symposium on Languages, Applications and Technologies (SLATE 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[6] David Vandegrift. Semantic graphs: What they are and why you should care, 2016.

[7] Christina Feilmayr and Wolfram Wöß. An analysis of ontologies and their success factors for application to business. *Data & Knowledge Engineering*, 101:1–23, 2016.

[8] Bess Schrader. What's the difference between an ontology and a knowledge graph?, 2020.

[9] Kenza Kellou-Menouer and Zoubida Kedad. Schema discovery in rdf data sources. In *International Conference on Conceptual Modeling*, pages 481–495. Springer, 2015.

[10] Mike Bergman and Frédérick Giasson. kbpedia, 2022.

[11] Oktie Hassanzadeh and Mariano P Consens. Linked movie data base. In *LDOW*, 2009.

[12] Berk Ekmekci and Blake Howald. Waffle: A graph for wordnet applied to freeform linguistic exploration. In *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*, pages 147–157, 2020.

[13] Richard Cyganiak. prefix.cc, 2022.

[14] Dörthe Arndt, Jeen Broekstr, Bob DuCharme, Ora Lassila, Peter F. Patel-Schneider, Eric Prud'hommeaux, Jr. Ted Thibodeau, and Bryan Thompson. RDF-star and SPARQL-star, 2022.

[15] Tim Berners-Lee. Reifying rdf (properly), and n3, 2005.

[16] Ferenc Bodon and Lajos Rónyai. Trie: an alternative data structure for data mining algorithms. *Mathematical and Computer Modelling*, 38(7-9):739–751, 2003.

[17] Emden Gansner, Eleftherios Koutsofios, and Stephen North. Drawing graphs with dot, 2006.

[18] Amelia Carolina Sparavigna and Roberto Marazzato. Graph visualization software for networks of characters in plays. *International Journal of Sciences February*, 2014.

[19] Angela Bonifati, Stefania Dumbrava, and Haridimos Kondylakis. Graph summarization. *arXiv preprint arXiv:2004.14794*, 2020.

[20] Li-Yen Kuo, Chung-Kuang Chou, and Ming-Syan Chen. Query-oriented graph clustering. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 749–761. Springer, 2017.

[21] Anas Alzogbi and Georg Lausen. Similar structures inside rdf-graphs. In *LDOW*, 2013.

[22] Thanh Tran, Günter Ladwig, and Sebastian Rudolph. Managing structured and semistructured rdf data using structure indexes. *IEEE Transactions on Knowledge and Data Engineering*, 25(9):2076–2089, 2012.

[23] Mussab Zneika, Claudio Lucchese, Dan Vodislav, and Dimitris Kotzinos. Rdf graph summarization based on approximate patterns. In *International Workshop on Information Search, Integration, and Personalization*, pages 69–87. Springer, 2015.

[24] Amit Krishna Joshi, Pascal Hitzler, and Guozhu Dong. Towards logical linked data compression. In *Proceedings of the Joint Workshop on Large and Heterogeneous Data and Quantitative Formalization in the Semantic Web, LHD+ SemQuant2012, at the 11th International Semantic Web Conference, ISWC2012*, 2012.

[25] Gang Wu, Juanzi Li, Ling Feng, and Kehong Wang. Identifying potentially important concepts and relations in an ontology. In *International Semantic Web Conference*, pages 33–49. Springer, 2008.

[26] Redouane Bouhamoum, Kenza Kellou-Menouer, Stephane Lopes, and Zoubida Kedad. Scaling up schema discovery for rdf datasets. In *2018 IEEE 34th International Conference on Data Engineering Workshops (ICDEW)*, pages 84–89. IEEE, 2018.