Towards RDF Anonymization

Irvin Dongo

UNIVERSITE DE PAU ET DES PAYS DE L'ADOUR

ECOLE DOCTORALE DES SCIENCES EXACTES ET LEURS APPLICATIONS

Towards RDF Anonymization

Prepared by: **Irvin Franco Benito DONGO ESCALANTE**

A thesis submitted in fulfillment for the degree of Doctor of Philosophy in Computer Science

Examination Committee:

Prof. Alban GABILLON, Université de la Polynésie Française, France (Reviewer)

Dr. Said TAZI, Université de Toulouse, UT1 - LAAS, France (Reviewer)

Prof. Béchara AL-BOUNA, Université Antonine, Baabda, Lebanon (Examiner)

Prof. José AGUILAR, Universidad de los Andes, Venezuela (Examiner)

Dr. Sebastién LABORIE, Université de Pau et Pays de l'Adour, France (Examiner)

Prof. Richard CHBEIR, Université de Pau et Pays de l'Adour, France (Advisor)

Prof. Yudith CARDINALE, Universidad Simón Bolivar, Venezuela (Co-Advisor)

Dr. Firas AL-KHALIL, University College Cork, Ireland (Co-Advisor)

20 December 2017

To God, for being the guide of my way, my family especially my aunt Lupe and uncle Wilber, all the teachers who guided me during this stage, and my friends with whom I enjoyed this period.

Acronyms

HTML *HyperText Markup Language*

HTTP *Hypertext Transfer Protocol*

IRI *Internationalized Resource Identifier*

LOD *Linking Open Data*

OWL *Web Ontology Language*

RDF *Resource Description Framework*

RDF(S) *RDF Schema*

SW *Semantic Web*

URI *Uniform Resource Identifier*

URL *Uniform Resource Locator*

URN *Uniform Resource Name*

XML *eXtensible Markup Language*

W3C *World Wide Web Consortium*

WWW *World Wide Web*

XSD *XML Schema Definition Language*

DTD *Document Type Definition*

LD *Linked Data*

DNS *Domain Name System*

Acknowledgments

First of all, I would like to express my gratitude to Prof. Alban GABILLON and Dr. Said TAZI for having the time to read and report my thesis. Also, I would like to thank to Prof. Béchara AL-BOUNA, Prof. José AGUILAR, and Dr. Sebastién LABORIE for being members of my jury.

I would also like to express my sincere gratitude and humble acknowledgement to my advisors for having contributed to the preparation and completion of this research. I am deeply thankful to Prof. Richard CHBEIR for accepting me as his student and for all support, patience, guidance, knowledge, and motivation during these three years. Equally, I would like to thank my co-advisor Pr. Yudith CARDINALE and Dr. Firas AL-KHALIL for all supervision, exigency, help and generous guidance during the completion of my PhD and for their time spent in reviewing and improving my work.

I would like to show my gratitude to my family, for their support and encouragement, specially to my aunt/mother Lupe, uncle/father Wilber and sister Karla.

I am thankful for my friends from France and Peru for their support and advices during all this period: Regina Ticona, José Negrete, Marita Vargas, Antonio Flores, Analía Boggia, Denisse Muñante, Aritz Echenique, Aizea Lojo, Joseba Nevado, Marta Toribio, Frédéric Bonnet, Fawzy Kattar, Renato Guzmán, and all the members of the LIUPPA lab.

I would also like to thank all the professors and administrative staff of the IUT de Bayonne for the pleasant atmosphere that I have lived during all these years.

I also want to thanks to all the institutions/organizations that supported me in different ways as FINCYT/INNOVATE (Peru) and LIUPPA (France).

Lastly and most importantly, my deepest gratitude and love goes to God, who filled me with strength, patience, perseverance, and wisdom to finish this work.

Abstract

With the advance of the Semantic Web and the Open Linked Data initiatives, a huge quantity of RDF data is available on the Web. The goal is to make this data readable for humans and machines, adopting special formats and connecting them by using International Resource Identifiers (IRIs), which are abstractions of real resources of the world. As more data is published and shared, sensitive information could be also provided. In consequence, the protection of entities of interest (e.g., people, companies) is a real challenge, requiring adequate techniques to avoid the disclosure/production of sensitive information on the Web.

Three main aspects are considered to ensure entity protection: (i) Preserving *information*, by identifying and treating the data that can disclose entities (e.g., identifiers, quasi-identifiers); (ii) Identifying the *utility* of the data to be published (e.g., statistics, testing, research) to adopt an adequate; and (iii) Modeling *background knowledge* that can be used for adversaries (e.g., number of relationships, a specific relationship, information of a node) to discover sensitive information.

Anonymization is one technique for entity protection that has been successfully applied in practice [RGCGP15]. However, studies regarding anonymization in the context of RDF documents, are really limited, showing practical anonymization approaches for simple scenarios as the use of generalization and suppression operations based on hierarchies. Moreover, the complexity of the RDF structure requires a high interaction of the expert user to identify and select the RDF's elements to be protected (main entities), and the ones related to them (identifiers, quasi-identifiers, sensitive information, and unsensitive information).

Additionally, the similarity among entities to discover similar data in other datasets, is compromised by disjoint similarities (e.g., the similarity between `float` and `double` is 0 for literal nodes). In literal nodes, datatypes play an important role, since it has been proven in the literature that the presence of datatypes, constraints, and annotations improves the similarity among XML documents (up to 14%). RDF adopts the datatypes from XML Schema, which are defined by the W3C.

Thus, in this context, the contributions are summarized as follows:

• An analysis of datatypes in the context of RDF matching/integration documents, its limitations and adequate applicability for the Semantic Web;

• An extended version of the W3C datatype hierarchy, where a parent-child relationship expresses subsumption (parent subsumes children);

• A new similarity measure for datatypes to take into account several aspects related to the new hierarchical relations among compared datatypes such as: distance and depth among datatypes, similar children;

• A new inference datatype approach to deduce simple datatypes based on four steps: (i) an analysis of predicate information, (ii) an analysis of lexical space values, (iii) a semantic analysis of the predicate, and (iv) a generalization of Numeric and Binary datatypes;

• A method to reduce the complexity of the RDF structure of the data to be published, simplifying the task of analysis, which is performed by the expert user;

• A method to suggest disclosure sources to the expert user, based on a node similarity, reducing the task of data classification; and

• A protection method, based on a generalization operation, to decrease the relations among resources from different datasets, to preserve the main objectives of integration and combination of the Semantic Web.

The different proposals have been tested through experimentation. Experimental results are satisfactory and show an important improvement in the accuracy and high performance for similarity and inference datatype approaches with respect to the existing works. Our protection approach for RDF data overcomes the related work and decreases the expert user interaction.

Resumé

Avec l'avancée du Web Sémantique et des initiatives Open Linked Data, une grande quantité de documents RDF sont disponibles sur Internet. L'objectif est de rendre ces données lisibles pour les humains et les machines, en adoptant des formats spéciaux et en les connectant à l'aide des IRIs (International Resource Identifier), qui sont des abstractions de ressources réelles du monde. L'augmentation du nombre de données publiées et partagées augmente également le nombre d'informations sensibles diffusées. En conséquence, la confidentialité des entités d'intérêts (personnes, entreprises, etc.) est un véritable défi, nécessitant des techniques spéciales pour assurer la confidentialité et la sécurité adéquate des données disponibles dans un environnement où chaque utilisateur a accès à l'information sans aucune restriction (Web).

Ensuite, trois aspects principaux sont considérés pour assurer la protection de l'entité: (i) Préserver la confidentialité, en identifiant les données qui peuvent compromettre la confidentialité des entités (par exemple, les identifiants, les quasi-identifiants); (ii) Identifier l'utilité des données publiques pour diverses applications (par exemple, statistiques, tests, recherche); et (iii) Les connaissances antérieures du modèle qui peuvent être utilisées par les pirates informatiques (par exemple, le nombre de relations, une relation spécifique, l'information d'un nœud). L'anonymisation est une technique de protection de la confidentialité qui a été appliquée avec succès dans les bases de données et les graphes. Cependant, les études sur l'anonymisation dans le contexte des documents RDF sont très limitées. Ces études sont les travaux initiaux de protection des individus sur des documents RDF, puisqu'ils montrent les approches pratiques d'anonymisation pour des scénarios simples comme l'utilisation d'opérations de généralisation et d'opérations de suppression basées sur des hiérarchies. Cependant, pour des scénarios complexes, où une diversité de données est présentée, les approches d'anonymisations existantes n'assurent

pas une confidentialité suffisante. Ainsi, dans ce contexte, nous proposons une approche d'anonymisation, qui analyse les voisins en fonction des connaissances antérieures, centrée sur la confidentialité des entités représentées comme des nœuds dans les documents RDF. Notre approche de l'anonymisation est capable de fournir une meilleure confidentialité, car elle prend en compte la condition de la diversité de l'environnement ainsi que les voisins (nœuds et arêtes) des entités d'intérêts. En outre, un processus d'anonymisation automatique est assuré par l'utilisation d'opérations d'anonymisations associées aux types de données.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The Semantic Web and the Linked Open Data (LOD) initiatives promote the integration and combination of RDF data on the Web [SH01]. RDF describes resources as triples: $\langle$`subject`, `predicate`, `object`$\rangle$, where `subjects`, `predicates`, and `objects` are all resources identified by their IRIs. Objects can also be literals (e.g., a number, a string), which can be annotated with optional type information, called datatype. Since the last decade, RDF is attracting more and more people, and data is gathered and published by different sources (e.g., companies, governments) for many purposes such as statistics, testing, and research proposals. For instance, according to [HDP12], more governments are becoming *e-governments*, since they are part of the LOD initiatives, providing their data to have a more flexible data integration, increasing the data quality, and offering new services. However, as more data is available, sensitive information (e.g., diseases, salaries, or bank accounts) could be sometimes provided or inferred leading to compromise the privacy of related entities (e.g., patients, users, companies).

Data can be analyzed and protected before being published on the Web [RGCGP15, HHD17], or limited in access for queries over controlled scenarios [SLB$^+$17]. In this work, we only focus on the protection of RDF data, expressed as documents, by the analysis of the data before publication. A privacy protection of the RDF data is tricky, since the use of different published heterogeneous datasets could break some protection. For instance, the combination of well-known datasets as DBpedia and Enipedia[1] produces sensitive information of places of interest (e.g., schools, hospitals, production factories), regarding their proximity to nuclear power plants (high contamination resource).

[1] *Enipedia* is a dataset containing data related to the production of energy and its applications. The information available on Enipedia is provided by governments, which support the LOD. `http://enipedia.tudelft.nl`

According to [RGCGP15], anonymization is one common and widely adopted technique for sensitive data protection that has been successfully applied in practice. It consists on protecting the entities of interest by removing or modifying identifiable information to make them anonymous before publication, while keeping the utility of the data. This latter is modified according to certain criteria of the existing values (e.g., taxonomies, ranges) to satisfy some conditions of anonymity (e.g., k-anonymity[1], l-diversity[2]). To apply anonymization, it is necessary to identify and classify the data (see D in Fig. 1.1) into: (i) *main entities*, which are the entities of interest, and (ii) *related data* that is directly or indirectly associated to the main entities and can compromise their privacy. The related data can also be classified as [BWJ06]: (i) *Identifiers*, data that directly identify a main entity (e.g., security social number); (ii) *Quasi-identifiers*, data that can be used to link with other data to identify a main entity (e.g., birthday, postal code, gender); (iii) *Sensitive information*, which is the data that compromise a main entity (e.g., diseases); and (iv) *Unsensitive information* that does not have a particular role or impact.

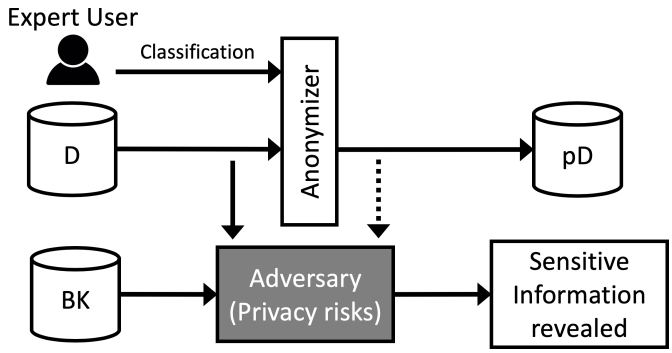


Figure 1.1: Anonymization framework inspired from [MDG14]; D is the data to be published, BK is the Background Knowledge; and pD the anonymous data obtained by the anonymization process, considering the classification made by the Expert User

A classification, which is performed by an expert user (see Expert User in Fig. 1.1) who knows previously the data and is responsible of protecting model, is based on predefined assumptions about how an adversary can take advantage over these data. These assumptions are called *Background Knowledge*. The background knowledge (see BK in Fig. 1.1) is the information related to the published data, which can be used by adversaries to discover sensitive information of the main entities. Due to the huge complexity of the RDF structure, a classification requires a high interaction of the expert user. Moreover, all RDF's elements can be considered as main entities, and they can also be classified

[1]k-anonymity is one of the most used common condition, that consists on making entities undistinguished from at least $k-1$ other entities, because they have similar information [SS98]

[2]l-diversity is an extension of the k-anonymity model that protects the corresponding sensitive values within a homogeneous group.

into identifiers, quasi-identifiers, sensitive information, etc., making the RDF protection complex.

Works on RDF anonymization are limited [RGCGP15, HHD17]. They mainly apply generalization and suppression operations over taxonomies (each RDF's element has a defined taxonomy) to anonymize the RDF document. Defined areas (neighborhood) are also provided [HHD17], where anonymization properties as k-anonymity are satisfied. Various anonymous RDF documents are generated by the combination of all values from the taxonomies and a measure is required to choose the best option. However, the exhaustive method to select the best anonymous RDF document makes these approaches unsuitable for complex cases, since a greater quantity of values to take into account, needs a more elaborate anonymization process (more possible solutions).

Since RDF forms a directed, labeled graph structure with data, where the edges (`predicates`) represent the named link between two resources, represented by the graph nodes (`subjects` and `objects`) [PJH14], databases and graphs anonymization techniques could be applied, but they are limited and inappropriate for privacy protection in the Semantic Web, as we detail in Section 5.2.

Thus, in the context of RDF data, the following limitations are identified:

1. RDF anonymization techniques are limited and designed for a particular and ideal scenario, which is inappropriate when having several linked heterogeneous datasets [Aro13, RGCGP15, HHD17, SLB+17];
2. The non-consideration of IRIs as external and reachable resources makes the current RDF solutions unsuitable for protection on the Web, since other available resources could link or infer sensitive information;
3. The presence and consideration of resources (IRIs and Blank nodes), which are a fundamental part of the RDF data, makes the database-oriented methods [NJ02, MGKV06, LLV07, MJK09, SO14] unsustainable for a large quantity of resources due to the number of JOIN functions needed to satisfy the existing normalized models;
4. Graph anonymization techniques assume simple, undirected and unlabeled graphs [BDK07, HMJ+08, ZP08, CT08, LT08, YW08, LWLZ08, CKR+11, YCYY13]; thus, the reduction of complexity of the RDF structure to a simple graph is necessary for the application of graph solutions, but inappropriate for the Semantic Web, since properties and semantic relations among resources would be ignored;
5. The complexity of the RDF structure requires a high interaction of the expert user

to identify and select the RDF's elements to be protected (main entities), and the ones related to the main entities (identifiers, quasi-identifiers, sensitive information, and unsensitive information); and

6. Approaches based on conceptual RDF representations are needed in order to provide more general solutions that can be serialized later on different formats (e.g., RDF/XML, Turtle, N3, JsonLD).

To overcome these limitations, we propose as the main contribution, a framework called *Ri**A**iR* (Reduction, Intersection, and Anonymization in RDF), which is independent of the serializations formats and providers. Our protection process mainly relies on a four phases approach where the input is converted into a graph representation, used by all modules: (i) *Reducing-Complexity phase* in which the graph is analyzed to reduce its complexity-structure to extract a compressed one; (ii) *Intersection phase*, where similar nodes between the input graph (reduced or not) from the data to be published and the one from the background knowledge are identified as potential keys (identifiers and quasi-identifiers); (iii) *Selecting phase* in which the expert user analyzes and selects the disclosure sources, which contain at least one potential key; and (iv) *protection phase* that executes a protection process over the selected triples. The proposal is designed for RDF documents, considering their elements (IRIs, blank nodes, literals) and the scenario, where a huge quantity of information is available. The complexity of the RDF structure is reduced to make possible the task of classification and to suggest potential disclosure sources to the expert user, decreasing his interaction. Moreover, by a generalization method, we reduce the connections among datasets, preserving the main objectives of the Semantic Web (integration and combination), and protecting the sensitive information at the same time.

As the reduction and intersection phases are based on a similarity function among RDF resources, some limitation related to the comparison among literal nodes were found and studied. For instance, the datatypes, which are associated to the literals, can represent the same information in several formats according to different vocabularies (e.g., a literal value 16.0 can be `float` or `double`). Moreover, a huge quantity of RDF documents is incomplete or inconsistent in terms of datatypes [PHHD10]. Thus, we propose a new hierarchy of datatypes based on the one proposed by the W3C, and a measure to obtain similarity values among different datatypes. Additionally, a inference process is also proposed to provide the datatypes to the literal nodes and perform the similarity.

We continue this chapter by identifying the principal aim and the objectives of the thesis (Section 1.1). Next, in Section 1.2, we explain our research contributions and, in

Section 1.3, we conclude this chapter with the outline of the remainder of this work.

1.1 Research Aims and Objectives

Since the protection of sensitive information on the Web is essential for generating, sharing and publishing data, and there are limitations on existing proposals, a new approach able to ensure the protection of RDF data, is needed.

The ultimate aim of this thesis is to avoid the disclosure of sensitive information by introducing a framework for RDF documents, called *Ri**A**iR*.

Our approach targets RDF protection through the following objectives:

1. Provide an easy classification of the RDF data (keys, sensitive information, etc.);
2. A similarity able to measure the intersection between the data to be published and the background knowledge to suggest disclosure sources; and
3. Select the most appropriate protection taking into account the main objectives of the Semantic Web.

1.2 Research Contributions

Based on the aim and objectives described above, and the related work (developed in Sections 3.2, 4.2 and 5.2), we present the following contributions in this thesis:

1. **Datatype Analysis and Similarity**

   The RDF adopts the XML datatypes defined by the W3C; however, the current hierarchy does not properly capture any semantically meaningful relationship between datatypes. For instance, datatypes `dateTime` and `time` are flattened in the W3C hierarchy. Thus, we analyze datatypes in the context of RDF matching/integration documents, since all information is used to discover similar data. Additionally, similarity measures for datatypes are not adequate for the Semantic Web, since either they are too restrictive (same datatype, then the similarity is 1, otherwise 0), or they are based on specific characteristics from XML and XSD (e.g., constraint facets). In order to perform a study of datatypes for the Semantic Web, we provide:

- An analysis of the current W3C datatype hierarchy, its limitations and adequate applicability for the Semantic Web.
- An extended version of the W3C datatype hierarchy, where a parent-child relationship expresses subsumption (parent subsumes children), which makes it a taxonomy of datatypes.
- A new similarity measure: extending the state-of-the-art works to take into account several aspects related to the new hierarchical relations among compared datatypes such as: distance and depth among datatypes, similar children.

2. **Datatype Inference**

   Datatypes are not always present in the data and according to [ANS09a], the presence of datatype information, constraints, and annotations on an object improves the similarity between two documents up to 14%. Hence, an analysis of the information related to the value, which does not have its respective datatype, is needed. An approach able to infer datatype for the Semantic Web is provided, performing:

   - An analysis of predicate information, such as range property that defines and qualifies the type of the object value.
   - An analysis of lexical space of the object value, by a pattern-matching process.
   - A Semantic analysis of the predicate and its semantic context, which consists in identifying related words or synonyms that can disambiguate two datatypes with similar lexical space.
   - A generalization of Numeric and Binary datatypes, to ensure a possible integration among RDF documents.
   - Besides, an online prototype called **RDF2rRDF** is also provided, in order to test and evaluate the inference process according the accuracy and performance in the context of huge quantity of RDF data.

3. **RDF Protection**

   Existing anonymization solutions in databases and graphs cannot be directly applied to RDF data, and RDF solutions are still in develop process and do not ensure enough privacy; thus, we proposed:

   - A method to reduce the complexity of the RDF structure of the data to be published, simplifying the task of analysis, performed by the expert user;
   - A method to suggest disclosure sources to the expert user, based on node similarity, reducing the task of data classification; and

- A protection operation, based on a generalization method, to decrease the relations among resources from different datasets, to preserve the main objectives of integration and combination of the Semantic Web.

Results have been presented and published in the proceedings of:

- The 28th International Conference on Database and Expert Systems Applications - DEXA 2017 [DAKCC17].
- The 18th International Conference on Web Information Systems Engineering - WISE 2017 [DCAKC17].
- International Journal of Data Science and Engineering - DSE 2018 [DCC18].

1.3 Manuscript Structure

We present an overview of each of the following chapters in this thesis:

Chapter 2 (The Semantic Web: Review) presents the background information regarding the concepts and principles about WWW, Semantic Web, RDF, and its respective definitions to better understanding the anonymization process.

Chapter 3 (The Semantic Web: Datatype Analysis and Similarity) presents the importance of datatypes for the Semantic Web and a motivating scenario to illustrate the limitations of existing approaches on datatype similarity. This chapter also describes our contribution for a better datatype similarity, consisting of a new datatype hierarchy based on the one proposed by the W3C, and a new similarity measure taking into account cross-children similarity. An experimental evaluation to measure the accuracy of our proposal is shown, with respect to existing approaches.

Chapter 4 (The Semantic Web: Datatype Inference) describes our datatype inference proposal. This chapter also includes a motivating scenario to show how inadequate integration among RDF documents can occur if the data types are not present. A formal proposal is described, consisting on four steps: Predicate Information Analysis, Datatype Lexical Space Analysis, Predicate Semantic Analysis, and Generalization of Numeric and Binary Groups. Finally, we detail our prototype, called **RDF2rRDF**, which is used to perform accuracy and performance evaluations, comparing them with existing approaches.

Chapter 5 (The Semantic Web: Privacy Preservation) describes the importance of protection for the Semantic Web in RDF documents. Concepts and definitions related to protection data are presented to formalize the proposal. A motivating scenario, in the context of energy production, is shown to illustrate the generation of sensitive information. A framework called *Ri**A**iR* (Reduction, Intersection, and Anonymization in RDF) based on four phases: (i) *Reducing-Complexity phase*, (ii) *Intersection phase*, (iii) *Selecting phase* and (iv) *protection phase* is also shown. In this chapter, we present our main prototype and the viability and performance evaluations.

Chapter 6 (Conclusions and Future Works) concludes our work, recapitulating our contributions and highlighting future directions.

Chapter 2

The Semantic Web: Review

> "The web as I envisaged it, we have not seen it yet.
> The future is still so much bigger than the past."
> — Tim Berners-Lee

This chapter describes technologies used in this thesis, providing a basic and common background to easy understand the rest of the document. We present the Semantic Web concepts and its associated elements to discern the nature, purpose, and principles of the Web. In section 2.1, we outline a brief overview about the Semantic Web on the Web. In sections 2.1.2 and 2.1.3, we provide a basic definition of the Semantic Web, illustrating the Semantic Web stacks. Section 2.2 contains a description of all the standards linked to the Semantic Web as *Resource Description Framework* (RDF), *eXtensible Markup Language* (XML), *Internationalized Resource Identifier* (IRI), related to the Semantic Web architecture. Thereafter, we present the Semantic Web paradigms related to *Linked Data* (LD) initiatives. The chapter ends with a discussion.

2.1 Semantic Web on the Web

The *World Wide Web* (WWW) marks the end of an era, where the incompatibility and the interaction of computer systems were a real problem. The WWW gives a huge accessibility to the information with many social potential and economic impacts. The idea of people working on a project, in the same space, was a powerful concept of the Web.

The evolution of the Web began as a network of networks-of-documents until become just a network, where documents, information, people, and social data are linked in several

ways over the Web. The Semantic Web appears in 2001 developed by Tim Berners-Lee as a new vision of the Web, where the data is interpreted with a semantic perspective. This perspective provides a "meaning" in which the syntactic and the semantic connection of terms establishes interoperability among systems. The Semantic Web also named as Web of Data, allows a new experience with different interaction among the resources on the Web.

All this evolution was based on some Web technologies to have a remarkable information space, where the resources are linked. These technologies have sufficient efficiency, scalability, and utility to allow this interaction, being also the base of the Semantic Web. We develop some Web technologies in the following section.

2.1.1 Web Technologies

There are several technologies related to the operational infrastructure of the WWW as *Internet*, *Uniform Resource Identifier* (URI) (see more details in Section 2.2.1), *Hypertext Transfer Protocol* (HTTP), *HyperText Markup Language* (HTML), and *Domain Name System* (DNS).

1. *Internet*: Internet is an abstraction from the underlying network technologies and physical address resolution [Sta09].

2. HTTP: Hypertext Transfer Protocol is the Internet protocol for distributed, collaborative, hypermedia information systems [FGM+99].

3. HTML: Hypertext Markup Language is the common language of Internet that allows to publish and retrieve information over the Web [RLHJ+99].

4. DNS: Domain Name System is a distributed database that offers mapping service from domain name into IP address [ML05].

2.1.2 Semantic Web Definitions and Goal

We can find several definitions of the Semantic Web, but the first one was written by his creator Tim Berners-Lee. This definition considers the Semantic Web as an extension of the WWW beyond the Web of Documents (hypertext) to the Web of Data, where documents and data are linked. In [BLHL+01], Berners-Lee say that: *"The Semantic Web*

is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation".

Another definition presented in [DMVH+00] say that: *"The Semantic Web is the next generation of the Web aims at machine-processable information".* Adding more explanation in the definition of [BLHL+01], the authors say: *"The Semantic Web bring structure to the meaningful content of Web pages, creating an environment, where software agents roaming from page to page can readily carry out sophisticated tasks for users."*

The goal of the Semantic Web, described by *World Wide Web Consortium* (W3C)[1], *is to enable computers to do more useful work and to develop systems that can support trusted interactions over the network.*

2.1.3 Semantic Web Architecture

Since the publication of the Semantic Web definition [BLHL+01], Berners-Lee proposed four versions of the Semantic Web stack, which illustrates the architecture of the Semantic Web. These versions were explained in several *presentations* (see Fig. 2.1): Version 1 [BLb] introduced in 2000, Version 2 [BLc, BLd, BLf] presented as part of two presentation in 2003, Version 3 [BLe] presented in WWW2005, and Version 4 [BLa] introduced in his keynote address at AAAI2006. All the versions were never published in the literature or included as part of a W3C. The architectures depict the languages necessary for data interoperability between semantic applications [GVdMB08].

According to all the stacks, we notice that almost the same set of tools and languages compose the different architectures of the Semantic Web architecture. These tools and languages are standards recognized by the W3C. Therefore, we provide detailed information about these standards in the following section.

2.2 Standards approved by the Semantic Web

The standards linked to Semantic Web are related to the different levels of their architecture as we explain in the last section. We only developed the standards definitions related to our research: XML, IRI, RDF, *RDF Schema* (RDF(S)), and *Web Ontology Language* (OWL).

[1] `https://www.w3.org/standards/semanticweb/`

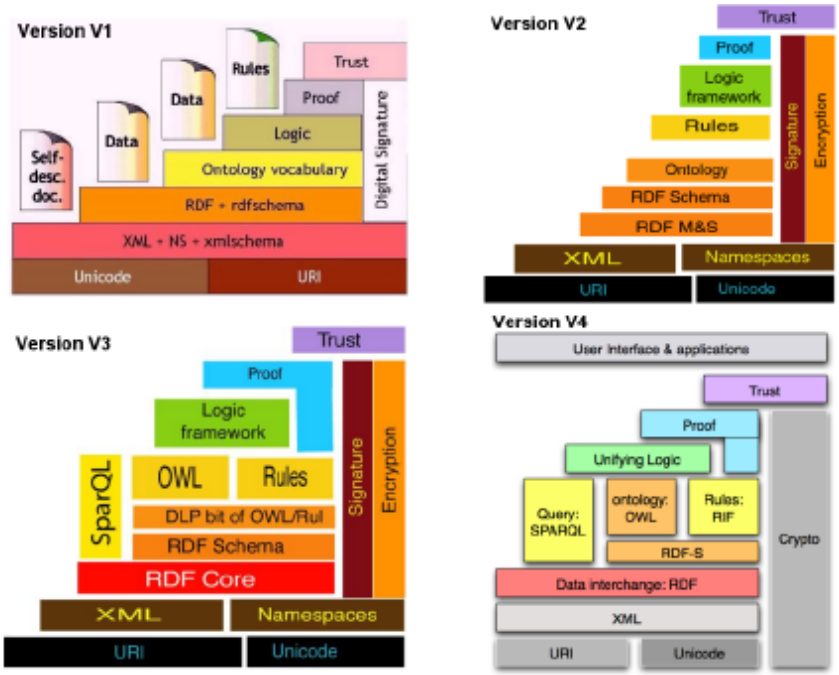


Figure 2.1: Versions of the Semantic Web architecture [GVdMB08]

2.2.1 Internationalized Resource Identifier - IRI

The IRIs constitute the bottom layer that support the Semantic Web architecture (see Fig. 2.1). In this section, we introduce the concepts related to the IRI as *resource* (Def. 1), URI (Def. 2), *Uniform Resource Locator* (URL) (Def. 3), and *Uniform Resource Name* (URN) (Def. 4).

Definition 1. ***Resource (*res*)**: The term **resource** is used in a general sense for whatever might be identified by an IRI [MBLF05, Lew07].* ♦

Following the definition of resource, we can identify anything with an IRI, but it is important to notice that this resource may not be necessarily accessed directly for the users. Indeed, there is a possibility to do not have a human-readable representation associated with the resource identified by an IRI. Due to this possibility, we may have two types of resources [Lew07]:

- Information resources: They are resources that have a human-readable representations that human user can be accessed using HTTP. These resources also are named as information IRI or Web IRI (URL) make up the vast majority of the WWW today.

- Other Web resources: They represent the resources themselves as a non-electronic information like physical entities and abstract concepts. These resources also are named as non-information IRI or Semantic Web IRI.

For instance, a Web page describing the concept Hospital is an information resource, but the Hospital itself (i.e., health care institution) is a non-information resource. Each resource would be identified by: the Web IRI (e.g., `http://live.dbpedia.org/page/Hospital`) and the Semantic Web IRI (e.g., `http://live.dbpedia.org/resource/Hospital`).

Definition 2. ***Uniform Resource Identifier (*URI*)****: A URI is a compact sequence of characters that identifies an abstract or physical resource [MBLF05, Lew07]* ♦

We can find two types of URIs, where the URI can be a locator - URL or a name - URN [MBLF05, Lew07].

Definition 3. ***Uniform Resource Locator (*URL*)****: A URL refers to the subset of URIs that, in addition to identifying a resource, provide a means of locating the resource by describing its primary access mechanism [MBLF05, Lew07].* ♦

Definition 4. ***Uniform Resource Name (*URN*)****: A URN are intended to serve as persistent, location-independent, resource identifiers [Moa97, MBLF05, Lew07].* ♦

Definition 5. ***Internationalized Resource Identifier (*IRI*)****: An IRI is a complement to the Uniform Resource Identifier (URI).An IRI is a sequence of characters from the Universal Character Set (Unicode/ISO 10646) [DS04b].* ♦

IRIs can be used instead of URIs where appropriate to identify resources. There is a diagram to combine all these definitions showing the dependencies between them (see Fig. 2.2).

2.2.2 Extensible Markup Language - XML

XML was created in 1996, under the auspices of the World Wide Web Consortium (W3C) XML is for the Semantic Web what HTML is for the Web. The Semantic Web uses XML as a standard language [DMVH$^+$00] to encode and give a tree structure to the information through some specific tags. All the information in the Semantic Web is encoded in XML.

Ten design goals for XML were proposed by the W3C recommendation as follows [BPSM$^+$97]:

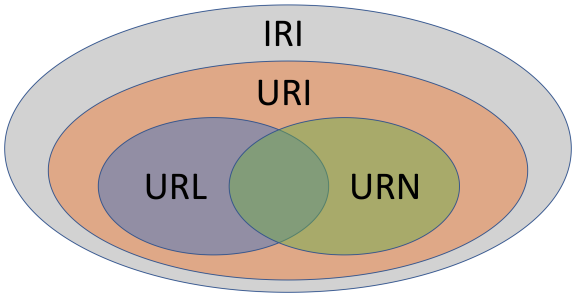


Figure 2.2: Diagram shows that an IRI is a URI, and URI is either a Uniform Resource Locator (URL), a Uniform Resource Name (URN), or both.

1. *XML shall be straightforwardly usable over the Internet.*
2. *XML shall support a wide variety of applications.*
3. *XML shall be compatible with SGML.*
4. *It shall be easy to write programs which process XML documents.*
5. *The number of optional features in XML is to be kept to the absolute minimum, ideally zero.*
6. *XML documents should be human-legible and reasonably clear.*
7. *The XML design should be prepared quickly.*
8. *The design of XML shall be formal and concise.*
9. *XML documents shall be easy to create.*
10. *Terseness in XML markup is of minimal importance*

Due to these goals, there are two main reasons for choosing XML representation for a user [UOVH09]:

- XML encode a wide range of data to be a simple way to send documents across the Web, which is a necessary condition in the Semantic Web, since data can be of any possible type.
- XML is widely used. The different systems have several parsers and writers to make easy the transmission of information between them.

The XML documents can express the same information in different ways and with completely different structures as well. Therefore, it is important to consider some conditions and specifications to have a well-formed document that allows the correct communication between different computer systems. This is where XML Schema comes into play.

XML Schema specifies a general structure and element types of an XML instance by specifying all the nodes required with their data and how they can be nested [FW04]. The data specification is through the definitions and the declaration either built-in data types [BMC+12] (see Fig. 2.3) or user defined data types [BMC+04].

There are several languages developed specifically to express XML schemas as *XML Schema Definition Language* (XSD) [TBM+12], *Document Type Definition* (DTD) [BPSM+08], and Relax NG [VdV03].

2.2.3 Resource Description Framework - RDF

For the Semantic Web, RDF is the *common format* to describe resources, which are abstractions of entities (documents, abstract concepts, persons, companies, etc.) of the real world. It was developed by Ora Lassila and Ralph Swick in 1998 [LSWC98]. RDF uses triples in the form of $\langle$`subject`, `predicate`, `object`$\rangle$ expressions also named statements, to provide relationships among resources. The RDF triples can be composed by the following elements:

– An **IRI**, which is an extension of the Uniform Resource Identifier (URI) scheme to a much wider repertoire of characters from the Universal Character Set (Unicode/ISO 10646), including Chinese, Japanese, and Korean character sets [DS04a] (see Section 2.2.1).
– A **Blank Node**, representing a local identifier used in some concrete RDF syntaxes or RDF store implementations. A blank node can be associated with an identifier (`rdf:nodeID`) to be referenced in the local document, which is generated manually or automatically.
– A **Literal Node**, representing values as strings, numbers, and dates. According to the definition in [CWL14], it consists of two or three parts:

- A **lexical form**, being a Unicode string, which should be in Normal Form C[2] to assure that equivalent strings have a unique binary representation.

[2]It is one of the four normalization forms, which consists on a Canonical Decomposition, followed by a Canonical Composition -http://www.unicode.org/reports/tr15/

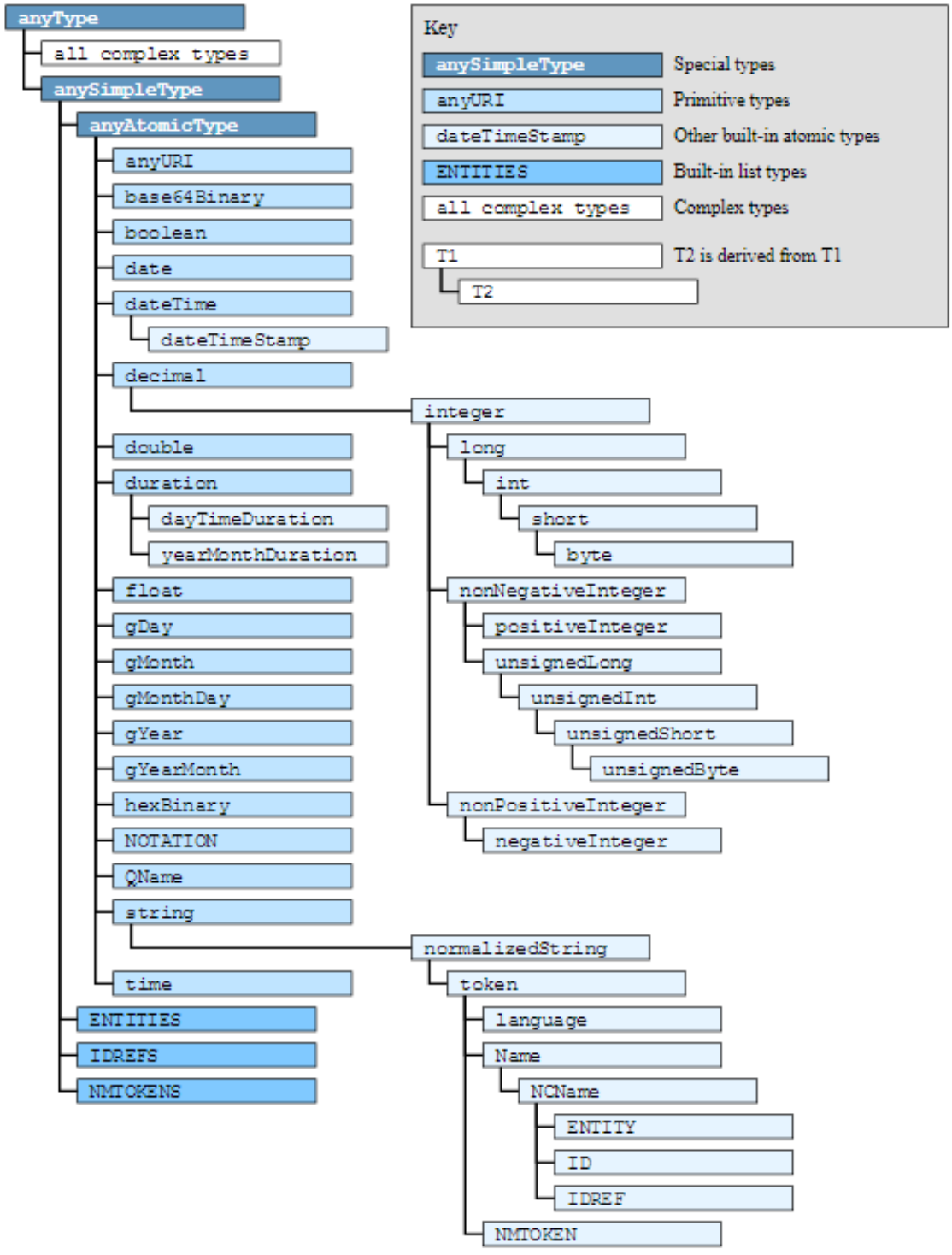


Figure 2.3: Derivation relations in the built-in type hierarchy [BMC+12].

• A datatype **IRI**, being an IRI identifying a **datatype** that determines how the lexical form maps to an object value.
• A **non-empty language tag** as defined by "Tags for Identifying Languages"[AP], if and only if the datatype IRI is http://www.w3.org/1999/02/22-rdf-syntax-ns#langString.

The Semantic Web proposes an implicit representation of the datatype in the literal node as a description of the value (e.g., `"value"^^xml:string`). There are two classes of datatypes: Simple and Complex. Simple datatypes can be primitive (e.g., `boolean`, `float`), derived (e.g., `long`, `int` derived from `decimal`), or user-defined, which are built from primitive and derived datatypes by constraining some of its properties (e.g., range, precision, length, format). Complex datatypes are defined as a set of elements, which can be either simple or complex datatypes.

Def. 6 presents the formal definition of a simple datatype according to W3C [JJC06].

Definition 6. Simple Datatype (*dt*): *In RDF, a simple datatype, denoted as **dt**, is characterized by: (i) a **value space**, denoted as VS(dt), which is a non-empty set of distinct valid values; (ii) a **lexical space**, denoted as LS(dt), which is a non-empty set of Unicode strings; and (iii) a **total mapping** from the lexical space to the value space, denoted as L2V(dt).* ♦

For example, the datatype `boolean` has the following characteristics:
– $VS($`boolean`$) = \{$true,false$\}$; – $LS($`boolean`$) = \{"true","false","1","0"\}$;
– $L2V($`boolean`$) = \{"true" \Rightarrow true, "false" \Rightarrow false, "1" \Rightarrow true, "0" \Rightarrow false\}$.

Following definitions describe the sets of the RDF's elements and datatypes:

Definition 7. Set of IRIs (*I*): *A set of IRIs, denoted as **I**, is a collection of IRIs that can be presented in a given RDF document, defined as: $I = \{i_1, i_2, ..., i_n\} \mid \forall i_i \in I$, i_i is an IRI.* ♦

Definition 8. Set of Literal Nodes (*L*): *A set of literal nodes, denoted as **L**, is a collection of literal nodes that can be presented in a given RDF document, defined as: $L = \{l_1, l_2, ..., l_n\} \mid \forall l_i \in L$, l_i is a literal node.* ♦

Definition 9. Set of Blank Nodes (*BN*): *A set of blank nodes, denoted as **BN**, is a collection of blank nodes that can be presented in a given RDF document, defined as: $BN = \{bn_1, bn_2, ..., bn_n\} \mid \forall bn_i \in BN$, bn_i is a Blank Node.* ♦

Definition 10. Set of Datatypes (*DT*): *A set of Datatypes, denoted as **DT**, is a*

collection of datatypes that can be presented in a given RDF document, defined as: $DT = \{dt_1, dt_2, ..., dt_n\} \mid \forall dt_i \in DT$, dt_i *is a datatype.* ♦

Definition 11. Set of Simple Datatypes (*SDT*)*: A set of Datatypes, denoted as* ***SDT****, is a collection of simple datatypes that can be presented in a given RDF document, defined as:* $DT = \{dt_1, dt_2, ..., dt_n\} \mid \forall dt_i \in SDT$, dt_i *is a simple datatype.* ♦

Table 2.1 summaries the sets of RDF's elements of Section 2.2.3 and 2.2.4, that we use in our formal approach description.

Table 2.1: Description of sets

Set	**Description**
I	A set of IRIs is defined as: I = $\{i_1, i_2, ..., i_n\} \mid \forall i_i \in I$, i_i is an IRI.
L	A set of literal nodes is defined as: L= $\{l_1, l_2, ..., l_n\} \mid \forall l_i \in L$, l_i is a literal node.
BN	A set of blank nodes is defined as: $BN = \{bn_1, bn_2, ..., bn_n\} \mid \forall bn_i \in BN$, bn_i is a Blank Node.
DT	A set of datatypes is defined as: DT= $\{dt_1, dt_2, ..., dt_n\} \mid \forall dt_i \in DT$, dt_i is a datatype.
SDT	The set of simple datatypes proposed by the W3C, is defined as: $SDT = \{$`string`, `boolean`, `decimal`, `datetime`, `base64Binary`, `NOTATION`, *etc.*$\}$

After the definition of sets of RDF'elements, we formally describe a triple in Def 12.

Definition 12. Triple (*t*)*: A Triple, denoted as* ***t****, is defined as an atomic structure consisting of a 3-tuple with a Subject (****s****), a Predicate (****p****), and Object (****o****), denoted as* $t :< s, p, o >$*, where:*

- $s \in I \cup BN$ *represents the subject to be described;*
- $p \in I$ *is a predicate defined as an IRI in the form* `namespace_prefix:predicate_name`*, where* $namespace\_prefix$ *is a local identifier of the* IRI*, in which the predicate (*$predicate\_name$*) is defined. The predicate (p) is also known as the* ***property*** *of the triple;*
- $o \in I \cup BN \cup L$ *describes the object.*

♦

The example presented in Fig. 2.4 underlines five triples with different RDF resources, properties, and literals:

- t_1: `<genid:treatment1,treat:hasPatient,genid:patient1>`
- t_2: `<genid:treatment1,rdf:type,treat:Treatment>`
- t_3: `<genid:patient1,rdf:type,ho:Patient>`

– t_4: <genid:patient1,ho:name,"Bethany Dawson">

– t_5: <genid:patient1,ho:birthday,1985-05-19>

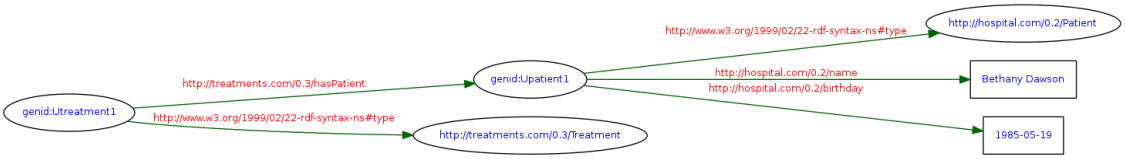


Figure 2.4: Example RDF Document.

A set of triples defines an RDF document, which is formally defined in Def. 13.

Definition 13. ***RDF Document*** **(d)***: An RDF document is defined as an encoding of a set of triples, using a predefined serialization format complying with an RDF W3C standards, such as RDF/XML, Turtle, N3, etc.* ♦

According to the structure of triples, RDF documents are also known as RDF Graphs, since the structure allows a node-edge-node relation. An RDF graph is defined in Def 14.

Definition 14. ***RDF Graph*** **(G)***:An RDF graph of an RDF document is denoted as $G_d(N, E)$, where each triple t_i from d is represented as a node-edge-node link. Therefore, G nodes (N), denoted as n_i, represent subjects and objects, and G edges (E), denoted as e_j, represent corresponding predicates: $n_i \in \bigcup_{t_i.s \cup t_i.o}$ and $e_j \in \bigcup_{t_i.p}$ [THTC+ 15].* ♦

2.2.3.1 Serialization formats

RDF data can be represented in different ways (serializations), i.e., stored in a file system through several formats. The W3C defines four formats: RDF/XML, Turtle, N-Triple, and N3, but there are also other serialization formats as RDFa, microdata, json-ld adopted by the W3C as recommendations.

1. RDF/XML [PJH14]: it is the first serialization format adopted by the W3C. This format serializes the RDF and XML files, where nodes and edges of the RDF document are represented using XML syntax. Their current media type is application/rdf+xml. The RDF document in Fig. 2.4 can be represented in XML as follows:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
```

```
   xmlns:ho="http://hospital.com/0.2/"
xmlns:treat="http://treatments.com/0.3/"
xmlns:vcard="http://www.w3.org/2006/vcard/ns#">

<ho:Patient rdf:nodeID="patient1">
<ho:name> Bethany Dawson </ho:name>
<ho:birthday> 1985-05-19 </ho:birthday>
</ho:Patient>
<treat:Treatment rdf:nodeID="treatment1">
<treat:hasPatient rdf:nodeID="patient1"/>
</treat:Treatment >
</rdf:RDF>
```

2. Turtle (Terse RDF Triple Language) [PJH14]: it is a textual serialization format to encode RDF documents in a compact form and also readable for humans. Their current media type is application/x-turtle. The RDF document in Fig. 2.4 can be represented in turtle format as:

```
@prefix ns0: <http://hospital.com/0.2/> .
@prefix ns1: <http://treatments.com/0.3/> .

_:genid1
  a <http://hospital.com/0.2/Patient> ;
  ns0:name " Bethany Dawson " ;
  ns0:birthday " 1985-05-19 " .
[]
  a <http://treatments.com/0.3/Treatment> ;
  ns1:hasPatient _:genid1 .
```

3. N-Triple (Notation of Triples) [PJH14]: it is simple serialization of RDF but not as compact as Turtle format. Their current media type is text/plain. The RDF document in Fig. 2.4 can be represented in N-triple format as:

```
_:genid1 <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://hospital.com/0.2/Patient> .
_:genid1 <http://hospital.com/0.2/name> " Bethany Dawson " .
_:genid1 <http://hospital.com/0.2/birthday> " 1985-05-19 " .
_:genid2 <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
```

```
<http://treatments.com/0.3/Treatment> .
_:genid2 <http://treatments.com/0.3/hasPatient> _:genid1 .
```

4. N3 (Notation 3) [PJH14]: it is an extension format of turtle language expressing a superset of RDF and has been designed with human readability in mind. Their current media type is text/rdf+n3. The RDF document in Fig. 2.4 can be represented in N3 format as:

```
@prefix ho: <http://hospital.com/0.2/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix treat: <http://treatments.com/0.3/> .
@prefix vcard: <http://www.w3.org/2006/vcard/ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

[] a treat:Treatment ;
    treat:hasPatient [ a ho:Patient ;
            ho:birthday " 1985-05-19 " ;
            ho:name " Bethany Dawson " ] .
```

5. RDFa (Resource Description Framework in Attributes): it is a serialization format that adds structured data to HTML or XHTML documents by extending the attributes of elements. The RDF document in Fig. 2.4 can be represented in RDFa format as:

```
<div xmlns="http://www.w3.org/1999/xhtml"
  prefix="
    rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
    treat: http://treatments.com/0.3/
    ho: http://hospital.com/0.2/
    rdfs: http://www.w3.org/2000/01/rdf-schema#"
  >
  <div typeof="treat:Treatment">
    <div rel="treat:hasPatient">
      <div typeof="ho:Patient">
        <div property="ho:name" content=" Bethany Dawson "></div>
        <div property="ho:birthday" content=" 1985-05-19 "></div>
```

```
      </div>
    </div>
  </div>
</div>
```

6. Microdata: it is a serialization format that describe a simpler way of annotating HTML elements with machine-readable tags. The RDF document in Fig. 2.4 can be represented in Microdata format as:

```
<div>
  <div itemtype="http://treatments.com/0.3/Treatment" itemscope>
    <div itemprop="hasPatient" itemtype="http://hospital.com/0.2/
    Patient" itemscope>
      <meta itemprop="birthday" content=" 1985-05-19 " />
      <meta itemprop="name" content=" Bethany Dawson " />
    </div>
  </div>
</div>
```

7. JSON-LD [MS14]: it is a concrete syntax format that extends the RDF data model to optionally allow JSON-LD to serialize Generalized RDF Datasets. The RDF document in Fig. 2.4 can be represented in JSON-LD format as:

```
{
  "@context": {
    "ho": "http://hospital.com/0.2/",
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "treat": "http://treatments.com/0.3/",
    "vcard": "http://www.w3.org/2006/vcard/ns#",
    "xsd": "http://www.w3.org/2001/XMLSchema#"
  },
  "@graph": [
    {
      "@id": "_:N3daa2c1446df47deac3f9e77aa61c4c2",
      "@type": "ho:Patient",
      "ho:birthday": " 1985-05-19 ",
      "ho:name": " Bethany Dawson "
```

```
    },
    {
      "@id": "_:N175b6312594e45dcbce29f6884b63a81",
      "@type": "treat:Treatment",
      "treat:hasPatient": {
        "@id": "_:N3daa2c1446df47deac3f9e77aa61c4c2"
      }
    }
  ]
}
```

All these formats are interchangeable, since any format can be converted to another one, without losing information. Therefore, a modification and/or protection applied to a specific input (e,g., RDF/XML), can produce a solution in another serialization format for a particular application. The solution can be later converted to another format (keeping the obtained properties) if this is required.

To manage RDF data, there are many tools and frameworks available in the literature, as well as, on the web (online services). One of the most common frameworks is *Apache Jena*, which is a free and open source Java framework for building Semantic Web and Linked Data applications[3]. It allows to create and read RDF graphs, and serialize triples using popular formats such as RDF/XML, Turtle, etc.

Thus, the use of a particular serialization format is independent of the process applied to the RDF data.

2.2.4 RDF Schema

The RDF Schema RDF(S) is a set of classes with certain properties (vocabulary), which are extensions of the basic RDF vocabulary [DB]. RDF(S) defines properties to better describe and determine characteristics of resources. Using RDF(S), we are able to define specific relations between the resources which have a unique meaning [UOVH09] or define the domain and range of their properties. For example, the `rdfs:domain` property designates the type of subject that can be associated to a predicate, while the `rdfs:range` property designates the type of object.

In this way, these RDFS properties allow to extend the description of existing re-

[3]https://jena.apache.org

sources and the meaning of RDF classes and properties. The meaning should be manipulated according to a certain logic to infer/derive new information, but this meaning is not context dependent. For example, if we exchange RDFS statements that are using the property `rdfs:subClassOf`, among different applications, these statements will still keep their meaning because this relation is domain independent [UOVH09]. This last sentence allows us to assume that the result of applying any anonymization process to RDFS properties, can be reused for different applications or as a input of new processes, since they are independent of the domain.

RDF(S) uses the IRI `http://www.w3.org/2000/01/rdf-schema#` with the prefix rdfs. The prefix is concatenated with a suffix (prefix:suffix) for convenience and readability obtaining an abbreviated form. This abbreviated form represents a complete IRI where the suffix should be the property.

2.2.5 Ontology - OWL

The next level after the RDF-Schema in the Semantic Web architecture, is the standard OWL. OWL was created to be used for more complex knowledge about things, groups of things, and relations among things [DSB+04]. This ontology language defines data models in terms of classes, subclasses, and properties to formally express a particular domain.

An ontology allows a better communication between machines, humans, and humans with machines, enabling the reuse of domain knowledge or extending the domain of interests. The main difference between RDF(S) and OWL stands on the higher expressiveness that we can reach with OWL and the complexity to implement it. A definition of ontology is presented in Def 15.

Definition 15. ***Ontology (*onto*)****: An ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members). The definitions of the representational primitives include information about their meaning and constraints on their logically consistent application. [LÖ09]* ♦

In order to introduce some definitions and concepts related to our approach, we defined a Class of resource in an ontology (Def. 16).

Definition 16. ***Class (*Class*)****: Given an entity e and an onto o, Class is a function that returns the class of e defined in o.*

$$Class(e,o) = class\ of\ e \in o. \qquad \blacklozenge$$

The following section 2.3 describes the Semantic Web paradigms.

2.3 Semantic Web paradigm

Linked Data is one of the big paradigms and pillars of the Web of Data (Semantic Web). The Web of Data works with links between datasets understandable not only to humans but also to machines. Linked Data also provides the best practices for making those links.

We have to precise that the Open Data is not equal to Linked Data. Open Data can be made available the data to everyone without links, the data also can be freely use and distributed. So, *Linking Open Data* (LOD) project merge the Linked Data with the Open Data based on metadata collected and curated by contributors to the Data Hub. The authors in [ALNZ13] give a schema to represent the differences of the representation and the degree of openness between Linked Data, Open Data and Linked Open Data. We present this comparison in Table 2.2.

Representation \ degree of openness	**Possibly closed**	**Open (cf. opendefinition.org)**
Structured data model (i.e. XML, CSV, SQL etc.)	Data	Open Data
RDF data model (published as Linked Data)	Linked Data (LD)	Linked Open Data (LOD)

Table 2.2: Comparison of the concepts Linked Data, Linked Open Data and Open Data [ALNZ13]

2.3.1 Benefits

There are significant benefits using the Linked Data, as the authors in [ALNZ13] shown:

- *Uniformity: all the information in the Linked Data is represented as triples using RDF statement data model. Almost all the elements represented by this structure are unique IRI/URI.*
- *De-referencability: IRIs are used for two purposes, for identifying entities and for*

locating and retrieving resources. When the IRI is used to identify an entity, there is another IRI to describe and represent the entity on the Web.

- *Coherence: all the IRIS use in an RDF triple as a subject or object position are linked through the predicate. This triple has a coherence in the RDF context developed*
- *Integrability: RDF data model represents all the information in the Linked Data sources facilitating the syntactic and semantic integration through the schema and instance matching techniques.*
- *Timeliness: using linked data sources facilitate a timely availability, due to publishing and updating of data in simple way.*

2.3.2 Best Practices for Publishing and Linking Structured Data

Linked Data is related to a set of best practices for publishing structured data on the Web. These practices are based on the principles established by Tim Berners-Lee [4] in [BLa]. These principles help the data became one big data space with linked information. The principles are: *(i) use URIs as names for things, (ii) use HTTP URIs, so that people can look up those names, (iii) when someone looks up a URI, provide useful information, and (iv) include links to other URIs, so they can discover more things* [5].

The practices are recommendations to make data interconnected, giving the possibility to re-use the information, which is the added value by the Web. The interpretation of these practices becomes rules. The first rule is related *to identify things with URIs*, the second rule is related *to use HTTP URIs* for following the standard, the third rule is related *to give information on the Web against a URI*, and the fourth is related *to make links elsewhere* for connecting the data [BLa].

2.3.3 Community projects

LOD community was founded in 2007 [BHIBL08], which the goal is to convert the datasets to RDF according to the principles and publishing them on the Web [BHBL09]. Inside of this community, we found several projects and open datasets as:

- BBC Music: it is a dataset about Artists, Releases and Reviews. Largely based upon

[4] The inventor of the WWW, *Semantic Web* (SW), and the Linked Data
[5] `https://www.w3.org/wiki/LinkedData`

MusicBrainz and the Music Ontology [6].

- DBpedia: it is a community to transform Wikipedia in a Linked Data version [7].
- Enipedia: it is an active exploration into the applications of wikis and the semantic web for energy and industry issues. [8].
- Freebase: it is an open-license database for all things in the world, has released a Linked Data interface [9].
- Geonames: it is a community based on add geospatial semantic information to the Word Wide Web [10].
- FOAF: it is a vocabulary for describing people and their social network on the Word Wide Web [11].
- DBLP Bibliography Server Berlin: it is a dataset with 800.000 articles and 400.000 authors, approx. 15 million triples about scientific papers [12].

The growing of LOD is exponential as we can compare LOD of 2007 in Fig. 2.5 and the one of 2017 in 2.6. LOD started with 12 smaller datasets until having more than 1163 datasets in these days. This project has a huge importance and impact for the Web community and their applications.

2.4 Summary

In this chapter, we have introduced all the background necessary for understanding the definitions, concepts, and Web technologies linked to the Semantic Web and RDF.

We began this chapter with a brief introduction about Semantic Web on the Web (Section 2.1) describing web technologies in Section 2.1.1, some definitions of Semantic Web in Section 2.1.2 and the Semantic Web architecture in Section 2.2. We then described the standards linked to Semantic Web: IRIs (Section 2.2.1), XML (Section 2.2.2), RDF (Section 2.2.3), RDF-schema (Section 2.2.4), and OWL (Section 2.2.5). These standards

[6] https://www.bbc.co.uk/music
[7] http://dbpedia.org/about
[8] http://enipedia.tudelft.nl/wiki/Main_Page
[9] https://developers.google.com/freebase/
[10] http://www.geonames.org/ontology/documentation.html
[11] http://www.foaf-project.org/
[12] http://wifo5-03.informatik.uni-mannheim.de/dblp/

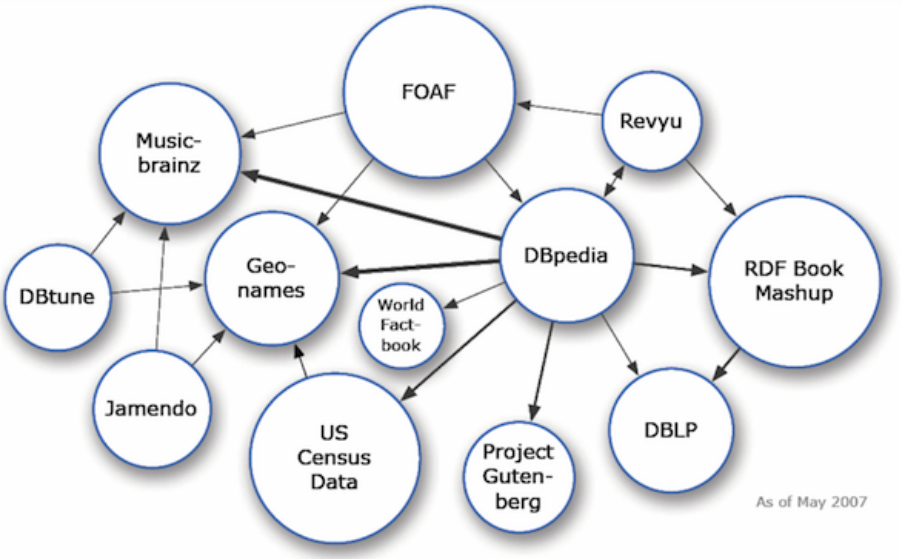


Figure 2.5: Linked Datasets as 2007 (Source: http://lod-cloud.net/,2007).

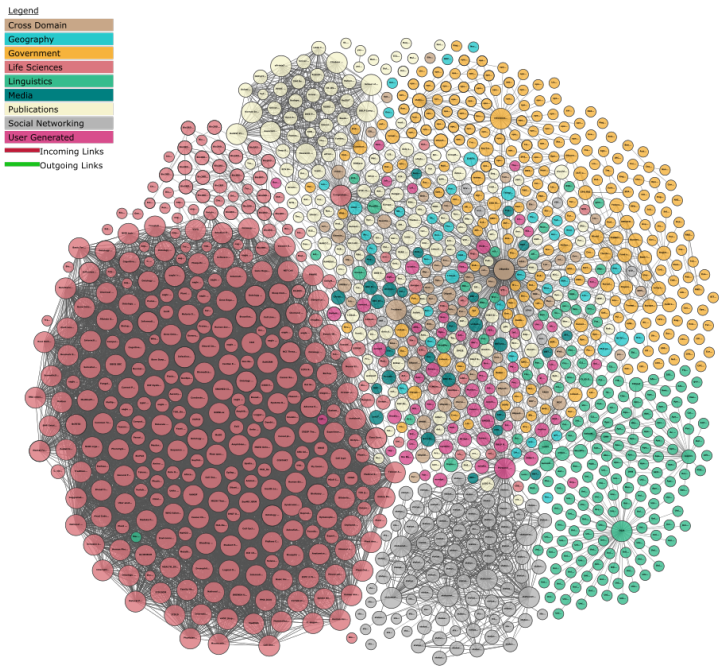


Figure 2.6: Linked Datasets as 2017 (Source: http://lod-cloud.net/,2017).

allow a representation of real resources on the Web, which can be linked among themselves through the use of IRIs.

Finally, in Section 2.3, we discussed the Semantic Web paradigm related to Linked data movement as consequence for the increase of RDF triples and their use on the Web. Thus, we concluded that currently a huge quantity of RDF data is available, because of the need to link different resources and the role of international communities as part of the Linked Open Data initiatives (e.g., W3C, e-governments, companies). Moreover, the combination of several heterogeneous datasets can disclose sensitive information, since resources are linked and thus, it is more sensitive to find disclosure sources in other datasets that breaches the protection.

Against this background, in the next chapter, we introduce the Datatype analysis and similarity for the Semantic Web in order to discover better similar resources.

Chapter 3

The Semantic Web: Datatype Analysis and Similarity

> "The advance of technology is based on making it fit in so that you don't really even notice it, so it's part of everyday life."
>
> — Bill Gates

As we mentioned in Chapter 2, one of the benefits offered by the Semantic Web initiative is the increased support for data sharing and the description of real resources on the web, by defining standard data representation models such as RDF, the Resource Description Framework. Particularly, heterogeneous RDF documents can express similar concepts using different vocabularies. Hence, many efforts focus on describing the similarity between concepts, properties, and relations to support RDF document matching/integration [MAL+15, ANS09b, Aea08].

Indeed, RDF describes resources as triples: ⟨`subject`, `predicate`, `object`⟩, where subjects, predicates, and objects are all resources identified by their IRIs. Objects can also be literals (e.g., a number, a string), which can be annotated with optional type information, called a **datatype**; RDF adopts the datatypes from XML Schema. A **datatype** is a classification of data, which defines types for RDF, adopted from XML Schema [PVB04]. There are two classes of datatypes: Simple and Complex. Simple datatypes can be primitive (e.g., `boolean`, `float`), derived (e.g., `long`, `int` derived from `decimal`), or user-defined, which are built from primitive and derived datatypes by constraining some of its properties (e.g., range, precision, length, format). Complex datatypes contain elements defined as either simple or complex datatypes. Simple datatypes are formally defined in

Def. 6.

The W3C Recommendation (proposed in [PJH14]) points out the importance of the existence of datatype annotations to detect entailments between objects that have the same datatype but a different value representation. For example, if we consider two distinct triples containing the objects `"20.000"` and `"20.0"`, then these objects are considered as different, because of the missing datatype. However, if they were annotated as follows: `"20.000"^^xml:decimal` and `"20.0"^^xml:decimal`, then we can conclude that both objects are identical. Moreover, works on XML Schema matching proved that the presence of datatype information, constraints, and annotations on an object improves the similarity between two documents (up to 14%) [ANS09a].

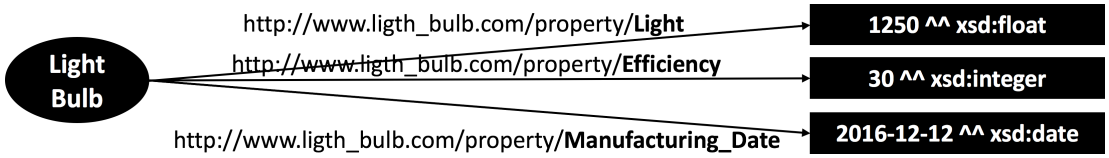
Another W3C Recommendation [JJC06] proposes a simple method to determine the similarity of two distinct datatypes: the similarity between two primitive datatypes is `0` (disjoint), while the similarity between two datatypes derived from the same primitive datatype is `1` (compatible). Obviously, this method is straightforward and does not capture the degree of similarity of datatypes; for instance, `float` is more similar to `int` than to `date`. This observation lead to the development of *compatibility tables*, that encodes the similarity ($\in [0, 1]$) of two datatypes. They were used in several studies [BMR01, NT07] for XML Schema matching. These compatibility tables were either populated manually by a designated person, as in [BMR01, NT07] or generated automatically using a similarity measure that relies on a hierarchical classification of datatypes, as in [HMS07, TLL13].

Hence, in the context of RDF document matching/integration, these works present the following limitations:

1. The Disjoint/Compatible similarity method as proposed by the W3C is too restrictive, especially when similar objects can have different, yet related, datatypes (e.g., `float` and `int` *vs* `float` and `double`).

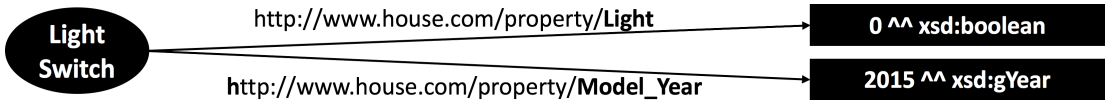
2. The use of a true similarity measure, expressed in a *compatibility table*, is very reasonable; however, we cannot rely on an arbitrary judgment of similarity as done in [BMR01, NT07]; moreover, for 44 datatypes (primitive and derived ones, according to W3C hierarchy), there are 946 similarity values ($n \times (n-1)/2$, n=44), which makes the *compatibility table* incomplete as in [BMR01]; a similarity measure that relies on a hierarchical relation of datatypes is needed.

3. The W3C datatype hierarchy, used in other works, does not properly capture any semantically meaningful relationship between datatypes (see, for instance, how datatypes related to `dateTime` and `time` are flattened in Fig. 3.2).



(a) Light Bulb concept and its properties

(b) Lamp concept and its properties



(c) Light Switch concept and its properties

Figure 3.1: Three concepts from three different RDF documents

From these limitations, there is a need to provide a better solution for any RDF document matching approach, where simple datatype similarity is considered. To achieve this, we propose:

1. An extended version of the W3C datatype hierarchy, where a parent-child relationship expresses subsumption (parent subsumes child), which makes it a taxonomy of datatypes.

2. A new similarity measure: extending the one presented in [HMS07], to take into account several aspects related to the new hierarchical relations between compared datatypes (e.g., children, depth of datatypes).

This chapter is organized as follows. In Section 3.1, we present a motivating scenario to illustrate the limitations of the state-of-the-art. In Section 3.2, we survey the literature on datatype similarity and compare them using our motivating scenario. In Section 3.4, we describe the new datatype hierarchy and the new similarity measure. In Section 3.5, we present the experiments we performed to evaluate the accuracy of our approach. And finally, we finish the chapter with Section 3.6, in which some reflections and discussions are presented.

3.1 Motivating Scenario

In order to illustrate the limitations of existing approaches for datatype similarity, we consider a scenario in which we need to integrate three RDF documents with similar concepts (resources) but based on different vocabularies. Fig. 3.1 shows three concepts from three different RDF documents to be integrated. Fig. 3.1a describes the concept of a `Light Bulb` with properties (predicates) `Light`, `Efficiency`, and `Manufacturing_Date`, Fig. 3.1b describes the concept of `Lamp` with properties `Light` and `MFGDT` (manufacturing date), and Fig. 3.1c shows the concept of `Light Switch` with properties `Light` and `Model_Year`.

To integrate these RDF documents, it is necessary to determine the similarity of the concepts expressed in them. For this, we use the similarity of their properties. More precisely, we can determine the similarity of two properties by inspecting the datatypes of their *ranges*[4] (i.e., of their objects).

Intuitively, considering the datatype information of the properties, we can say that:

1. `Light Bulb` and `Lamp` are similar, since their properties are similar: the `Light` property, representing the intensity of light, has the datatype `float` for `Light Bulb` and `double` for `Lamp`. We know that both `float` and `double` express floating points, and they differ only by their precisions; a similar analysis can be done for the properties `Manufacturing_Date` and `MFGDT`, both represent the manufacturing date, the datatypes of both properties are related to dates.

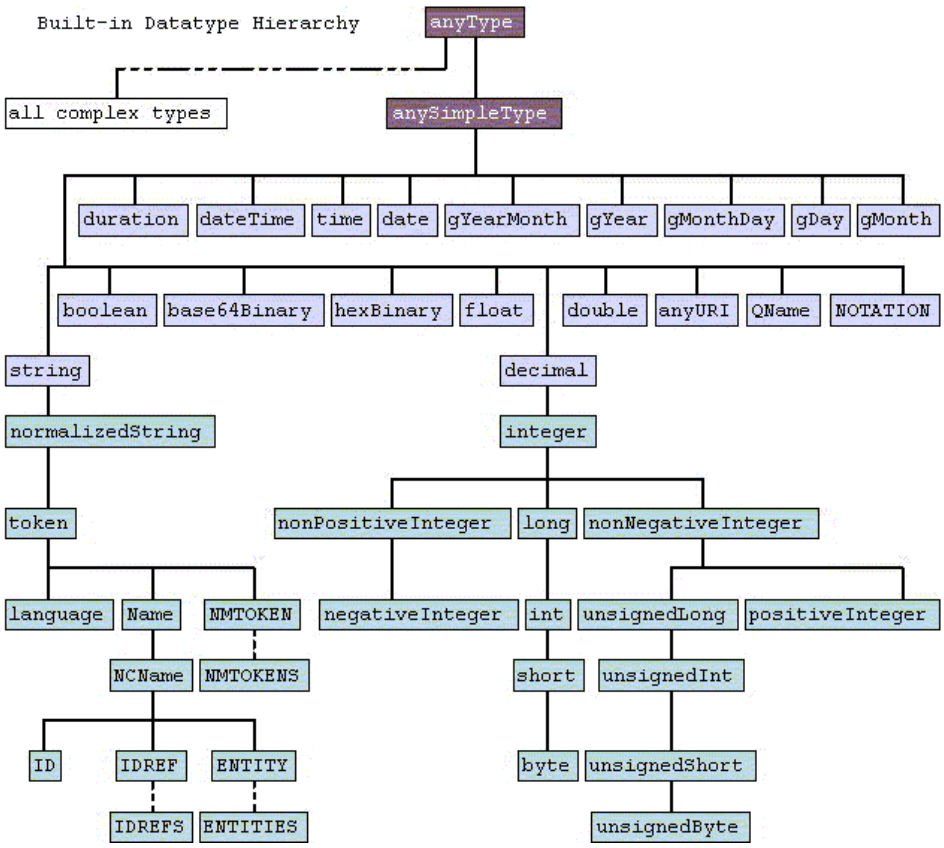
2. `Light Switch` is different from the other concepts, since it is about a switch and not a Bulb as the other concepts; indeed, the `Light` property is expressed in `binary`, and can hold one of two values, namely 0 and 1, expressing the state of the light switch (i.e., on and off, respectively).

Hence, to support automatic matching of RDF documents based on their concepts similarity, it is necessary to have a datatype hierarchy establishing semantically meaningful relationship among datatypes and a measure able to extract these relations from the hierarchy. In the following section, we survey the literature on datatype similarity and compare them using this motivating scenario.

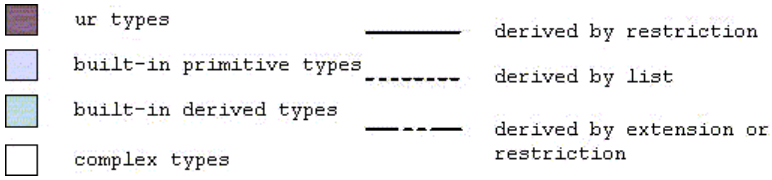
[4] A range (*rdfs:range*) defines the object type that is associated to a property.

3.2 Related Work

To the best of our knowledge, there is no existing work tackling datatype similarity specifically targeting RDF documents. Hence, we review works on datatype similarity described for XML and XSD, since RDF uses the same XML datatypes proposed by the W3C (the datatype hierarchy is shown in Fig. 3.2), and we also consider works in the context of ontology matching. We evaluate these works in an RDF document matching/integration scenario in the discussion.



(a) Datatype Hierarchy



(b) Type of Datatype (c) Type of Derivative

Figure 3.2: W3C Datatype Hierarchy

Most of the existing works in the XML and XSD area are focused on schema matching in contexts of, for example, XML message mapping, web data sources integration, and data warehouse loading. The main approaches taken to establish the datatype similarity

are mainly based on:

1. User-defined compatibility tables [ANS10, Aea08, ASS09, BMR01, DR02, NT07, NX04, TN10];
2. Constraining facets[5] [TLL13];
3. Extended W3C hierarchy and measures [ABF12, ARTW09, HMS07].

User-defined compatibility tables, as the one presented in Table 3.1 (taken from [BMR01]), express the judgment and perception of users regarding the similarity between each pair of datatypes. Hence, these tables present similarity values that are not objective, complete, or reliable.

When constraining facets are considered as in [TLL13], the similarity value between two different datatypes is calculated by the number of common facets divided by the union of them. For example, datatypes `date` and `gYearMonth` have the same facets (i.e., pattern, enumeration, whiteSpace, maxInclusive, maxExclusive, minExclusive, and minInclusive), thus, their similarity is equal to 1. This method allows to create an objective, complete, and reliable *compatibility table*; however, suitability is still missing: besides facets, which are only syntactic restrictions, other information should be considered for the Semantic Web (e.g., common datatypes attributes[6] – datatype subsumption).

Table 3.1: Datatype compatibility table of work [BMR01]

Type (s)	Type (t)	Compatibility coefficient (s, t)
string	string	1.0
string	date	0.2
decimal	float	0.8
float	float	1.0
float	integer	0.9
integer	short	0.8

Other works have proposed a new datatype hierarchy by extending the one proposed by the W3C. This hierarchy describes two classes of datatypes: Simple and Complex. Simple datatypes can be primitive (e.g., `duration`, `dateTime`), derived (e.g., `integer`, `long` derived from `decimal`), or user-defined, which are built from primitive and derived datatypes by constraining some of its properties (e.g., range, precision, length, format). Com-

[5] Constraining facets are sets of aspects that can be used to constrain the values of simple types (e.g., length, pattern, fractionDigits) (https://www.w3.org/TR/2001/REC-xmlschema-2-20010502/#rf-facets).

[6] An attribute is the minimum classification of data, which does not subsume another one. For example, datatype `date` has the attributes year, month, and day.

plex datatypes contain elements defined as either simple or complex datatypes (see Section 2.2.3). In [HMS07], the author proposes five new datatype groups: `Text, Calendar, Logic, Numeric`, and `Other` (see Figure 3.3). They also propose a new datatype similarity function that relies on that hierarchy and takes into account the proximity of nodes to the root and the level of the Least Common Subsumer[7] (LCS) of the two compared datatypes. The works presented in [ABF12, ARTW09], combine semantic similarity, structural similarity, and datatype compatibility of XML schemas in a function, by using the hierarchy and similarity function proposed by [HMS07]. Even though these works improve the similarity values, we will see their limitations in the context of our motivational scenario, concerning to misdefined datatype relations in the datatype hierarchy.

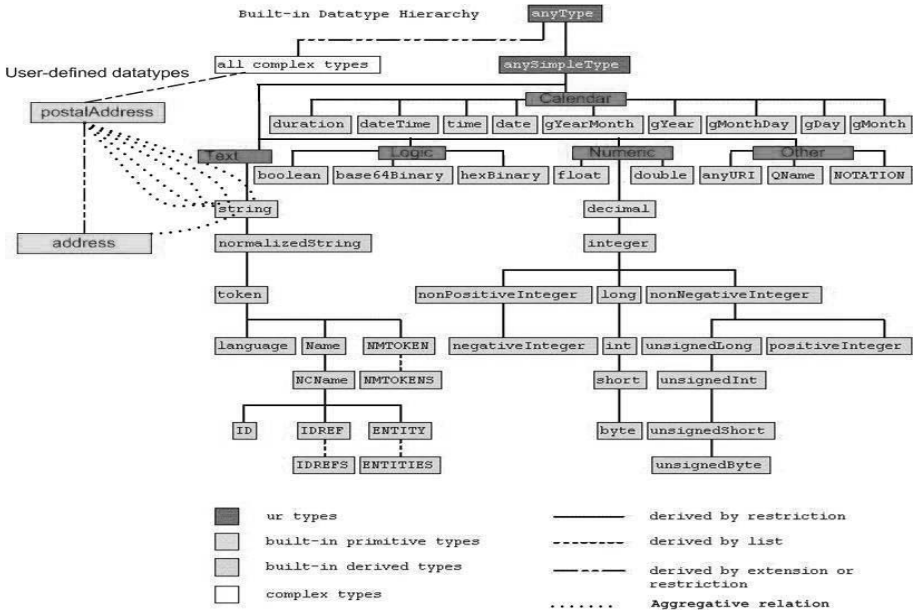


Figure 3.3: Extended Hierarchy from the work [HMS07]

In the context of ontology matching, most of the works classify datatypes as either Disjoint or Compatible (similarity $\in \{0, 1\}$). Some of them are based on the W3C hierarchy, such as [ES07, JMSK09], while others take into account properties of the datatypes (domain, range, etc.) [CAS09, HA09, HQC08, JLD15, LT06, LTLL09, MAL+15, NB16, SSK05]. When domain and range properties are considered, if two datatypes have the same properties, the similarity value is 1, otherwise it is 0. In the context of RDF matching, in which similar objects can have different but related datatypes, this binary similarity is too restrictive. The authors in [EYO08] generate a vector space for each ontology by extracting all distinct concepts, properties, and the ranges of datatype properties. To calculate the similarity between the two vectors, they use the cosine similarity measure. However, as the measure proposed in [HMS07], the problem remains in the datatype hierarchy that

[7]It is the most specific common ancestor of two concepts/nodes, found in a given taxonomy/hierarchy.

does not represent more semantically meaningful relationships between datatypes.

Table 3.2: Related Work Classification

Group	Work	Datatype Similarity	Datatype Similarity Approach Requirements			
			Simple datatype	Common Attributes	SW Context	
					XML XSD	RDF OWL
1	W3C [JJC06] [CAS09, EYO08, ES07] [HA09, HQC08, JMSK09] [JLD15, LT06, LTLL09] [MAL+15, NB16, SSK05]	Disjoint/Compatible (binary values)	✓	X	✓	✓
2	[ANS09a, ANS10, Aea08] [ASS09, BMR01] [DR02, NT07, NX04, TN10]	User-defined Compatibility Table	✓	X	✓	X
3	[TLL13]	Constraining Facets	✓	X	✓	X
4	[ABF12, ARTW09, HMS07]	Formula on extended W3C Hierarchy	✓	X	✓	X

According to this review of existing works, we classify them into four groups: Group 1, those works are based on Disjoint/Compatible similarity; Group 2, where works apply user-defined compatible tables; Group 3, datatype similarity values are obtained by the used of constraining facets; and Group 4, where the works use a formula applied to an extension of the W3C hierarchy (see Table 3.2). We evaluate them in our motivating scenario in the upcoming section.

3.3 Resolving Motivating Scenario and Discussion

Now, we evaluate our scenario using the defined groups in Table 3.2. In our motivating scenario, we have the datatypes `float` and `date` from the concept `Light Bulb` (Fig. 3.1a), datatypes `double` and `gYearMonth` from the concept `Lamp` (Fig. 3.1b), and `boolean` and `gYear` from concept `Light Switch` (Fig. 3.1c).

According to the Disjoint/Compatible similarity, either defined by the W3C or not (Group 1 in Table 3.2), the similarity between the three pairs of datatypes related to `Light` property (`float`–`double`, `float`–`boolean`, and `double`–`boolean`) is 0, because the three datatypes are primitives. We have the same similarity result regarding `Manufacturing_Date`, `MFGDT`, `Model_Year` properties, since their datatypes are also primitives. It means that there is no possible integration for these concepts using this Disjoint/Compatible similarity method. However, the concepts `Light Bulb` and `Lamp` are strongly related according to our scenario.

Based on the user-defined compatibility table shown in Table 3.1 (as works in Group 2 do), the similarity between `float`–`double` is a given constant > 0 (as `decimal`–`float` has in the compatibility table), however the similarity values of `double`–`boolean`, `date`–`gYearMonth`, `date`–`gYear`, and `gYearMonth`–`gYear` are not present in the compatibility table, therefore leading to a similarity value of 0 as in [HMS07] do. In this case, concepts `Light Bulb` and `Lamp` have their respective properties `Light` considered similar, while `Manufacturing_Date` and `MFGDT` are considered disjoint, even though they are clearly related.

According to the methods of Group 3 (based on constraining facets), similarity values for `float`–`double`, `date`–`gYearMonth`, `date`–`gYear`, and `gYearMont`–`gYear` are all equal to 1 (because they have the same facets), and for `float`–`boolean` and `double`–`boolean`, the similarities are equal to 0.29 (2 common facets divided by the union of them, which is 7). Thus, the three concepts can be integrated as similar, which is incorrect. Additionally, datatypes `date`, `gYearMonth`, and `gYear` are related but not equal: besides their facets, other information (such as datatype attributes - year, month, day) should count to decide about their similarities.

Finally, according to the works in Group 4, which are based on similarity measures applied on a datatype hierarchy extended from the W3C hierarchy [HMS07], similarity between `float`–`double` is 0.30, similarity between `float`–`boolean` and `double -boolean` is 0.09, for `date`–`gYearMonth`, `date`–`gYear`, and `gYearMonth`–`gYear` the similarity value is 0.296[8]. Even though these works manage in a better way the datatype similarity than all other Groups, there is still the issue of considering common datatypes attributes (as for work in Group 3). We can note that `date`–`gYearMonth` share year and month as common attributes, while `date`–`gYear` only have year as common attribute; thus, similarity between `date`–`gYearMonth` should be bigger than the other.

Table 3.3 summarizes the integration results of the motivating scenario. Column *Appropriate* shows the correct integration according to our intuition. One can note that existing works cannot properly determine a correct integration. With this analysis, we can observe the importance of datatypes for data matching/integration and the limitations of the existing works, from which, the following requirements for a more appropriated datatype similarity approach, were identified:

1. The similarity measure should consider at least all simple datatypes (primitive and

[8]We show the results according the measure proposed on [HMS07], all other works in Group 4 propose similar measures.

Table 3.3: Integration Results for our Motivating Scenario

Concept Integration	**G. 1 (Sim)**	**G. 2 (Sim)**	**G. 3 (Sim)**	**G. 4 (Sim)**	**Appropriate Integration**
`Light Bulb` and `Lamp`	NI (0.00)	NI (0.40)	I (1.00)	NI (0.30)	**I**
`Lamp` and `Light Switch`	NI (0.00)	NI (0.00)	I (0.65)	NI (0.19)	**NI**
`Light Bulb` and `Light Switch`	NI (0.00)	NI (0.00)	I (0.65)	NI (0.19)	**NI**
Sim is the average between the similarity of properties within concepts; Results were obtained by applying a threshold 0.50 for average of properties; NI = Not Integrable, I = Integrable.					

derived datatypes); complex datatypes are out of the scope in this work.

2. The datatype hierarchy and similarity measure should consider common datatype attributes (subsumption relation) in order to establish a more appropriate similarity.

3. The whole approach should be objective, complete, reliable, and suitable for the Semantic Web.

Table 3.2 compares the existing works based on these requirements. We can note that all works consider primitive and derived datatypes and are suitable in XML and XSD contexts. Only the works in the context of ontology matching (Group 1) consider RDF data. None of these works consider common datatype attributes. Hence, it is clear that a new datatype similarity approach is need for the Semantic Web in order to satisfy the defined requirements. The following section describes our approach, based on a new hierarchy and a new similarity measure, that overcomes the limitations of existing works and addresses these requirements.

3.4 Our Proposal

In this section, we describe our datatype similarity approach that mainly relies on an extended W3C datatype hierarchy and a new similarity measure.

3.4.1 New Datatype Hierarchy

As we mentioned before, the W3C datatype hierarchy does not properly capture any semantically meaningful relationship between datatypes and their common attributes. This

issue is clearly identified in all datatypes related to date and time (e.g., `dateTime`, `date`, `time`, `gYearMonth`), which are treated as isolated datatypes in the hierarchy (see Fig. 3.2).

Our proposed datatype hierarchy extends the W3C hierarchy as it is shown in Fig. 3.4. White squares represent our new datatypes, black squares represent original W3C datatypes, and gray squares represent W3C datatypes that have changed their location in the hierarchy. We propose four new primitive datatypes: `period`, `numeric`, `logic`, and `binary`. Thus, we organize datatypes into eight more coherent groups of primitive datatypes (`string`, `period`, `numeric`, `logic`, `binary`, `anyURI`, `QName`, and `NOTATION`). All other datatypes are considered as derived datatypes (e.g., `duration`, `dateTime`, `time`) because their attributes are part of one particular primitive datatype defined into the eight groups.

We also add two new derived datatypes (`yearMonthDuration` and `dayTimeDuration`), which are recommended by W3C to increase the precision of `duration`, useful for `XPath` and `XQuery`. We classify each derived datatype under one of the eight groups (e.g., `Period` subsumes `duration`, `numeric` subsumes `decimal`) and, in each group, we specify the proximity of datatypes by a sub-hierarchy (e.g., `date` is closer to `gYearMonth` than to `gYear`).

The distribution of the hierarchy for derived datatypes is established based on the subsumption relation and stated in the following assumption:

Assumption 1. *If a datatype d_1 contains at least all the attributes of a datatype d_2 and more, d_1 is more general than d_2 (d_1 subsumes d_2).*

As a consequence of Assumption 1, the hierarchy designates datatypes more general to more specific, from the root to the bottom, which in turn defines datatypes more related than others according to their depths in the hierarchy. With regards to this scenario, we have the following assumption:

Assumption 2. *Datatypes in the top of the hierarchy are less related than datatypes in the bottom, because datatypes in the top are more general than the ones in the bottom.*

Thus, according to Assumption 2, the datatype similarity value will depend on their position (depth) in the hierarchy (e.g., `gYearMonth`–`gYear` are more similar than `period`–`dateTime`), as we show in the next section.

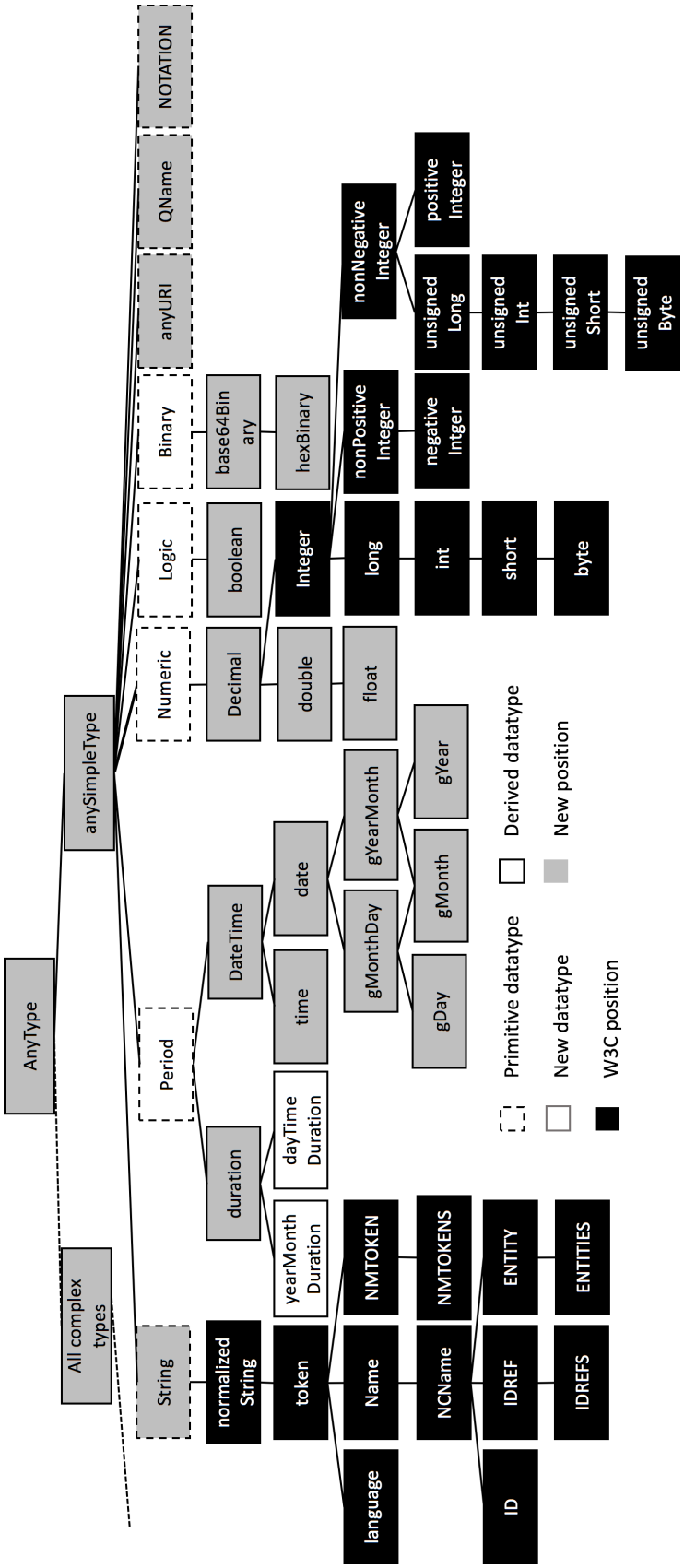


Figure 3.4: New Datatype Hierarchy

3.4.2 Similarity measure

Our proposed similarity measure is inspired by the one presented in [HMS07]. The authors establish the similarity function based on the following intuition:

"The similarity between two datatype d_1 and d_2 is related to the distance separating them and their depths in the datatype hierarchy. The bigger the distance separating them, the less similar they are. The deeper they are the more similar they are, since at deeper levels, the difference between nodes is less significant [HMS07]."

The authors state the similarity between two datatypes d_1 and d_2 as:

$$c(d_1, d_2) = \begin{cases} f(l) \times g(h) & \text{if } d_1 \neq d_2 \\ 1 & \text{otherwise} \end{cases} \tag{3.1}$$

where:

- l is the shortest path length between d_1 and d_2;
- h is the depth of the Least Common Subsumer (LCS) datatype which subsumes datatype d_1 and d_2.
- $f(l)$ and $g(h)$ are defined based on Shepard's universal law of generalization [JC97] in Eq. 3.2 and Eq. 3.3, respectively.

$$f(l) = e^{-\beta l} \tag{3.2}$$

$$g(h) = \frac{e^{\alpha h} - e^{-\alpha h}}{e^{\alpha h} + e^{-\alpha h}} \tag{3.3}$$

  where α and β are user-defined parameters.

The work in [HMS07] does not analyze the common attributes (children) of compared datatypes. For example, the datatype pair `date`–`gYearMonth` (with 2 attributes, namely `year` and `month`, in common) involves more attributes than `date`–`gYear` (with only 1 attribute, namely `year`, in common). The authors of [HMS07] consider that the similarity values of both cases are exactly the same.

In order to consider this analysis, we assume that:

Assumption 3. *Two datatypes d_1 and d_2 are more similar if their children in the datatype hierarchy are more similar.*

Furthermore, the depth of the LCS is not enough to calculate the similarity according to Assumption 2. Notice that the difference in levels in the hierarchy is also related to

similarity. For example, according to [HMS07], we have c(`time`, `gYearMonth`) = c(`dateTime`, `gYear`), because in both cases the distance between the datatypes is $l = 3$, and the LCS is `dateTime`, whose $h = 3$ (see Fig. 3.4). However, the difference between levels of `time` and `gYearMonth` is smaller than the one of `dateTime` and `gYear`, thus the similarity of `time-gYearMonth` should be bigger than the second pair (i.e., c(`time`, `gYearMonth`) > c(`dateTime`, `gYear`)). Hence, we assume:

Assumption 4. *The similarity of two datatypes d_1 and d_2 is inversely proportional to the difference between their levels.*

Based on Assumption 3 and Assumption 4, we defined the cross-children similarity measure in the following.

To consider the cross-children similarity, we first calculate the children similarity vector V_{d_1p,d_2q} of a datatype d_1, with respect to datatype d_2 in levels p and q, respectively. In d_1 sub-hierarchy, d_1 has i children in level p and in d_2 sub-hierarchy, d_2 has j children in level q. Thus, V_{d_1p,d_2q} is calculated as in Eq. 3.4.

$$V_{d_1p,d_2q} = [c(d_1, d_{1p}^1), \ldots, c(d_1, d_{1p}^i), c(d_1, d_{2q}^1), \ldots, c(d_1, d_{2q}^j)] \tag{3.4}$$

where d_{1p}^x represents the child x of d_1 (with x from 1 to i) in level p and d_{2q}^y represents the child y (with y from 1 to j) of d_2 in level q.

Similarly, V_{d_2q,d_1p} is the children similarity vector of a datatype d_2, with respect to datatype d_1 in the levels q and p respectively, defined as in Eq. 3.5.

$$V_{d_2q,d_1p} = [c(d_2, d_{1p}^1), \ldots, c(d_2, d_{1p}^i), c(d_2, d_{2q}^1), \ldots, c(d_2, d_{2q}^j)] \tag{3.5}$$

For each pair of vectors V_{d_1p,d_2q} and V_{d_2q,d_1p}, we formally define the cross-children similarity for level p and q, in Def. 17.

Definition 17. *The cross-children similarity of two datatypes d_1 and d_2 for levels p and q, respectively, is the cosine similarity of their children similarity vectors V_{d_1p,d_2q} and V_{d_2q,d_1p}, calculated as:*

$$CCS_{d1p,d2q} = \frac{V_{d_1p,d_2q} \cdot V_{d_2q,d_1p}}{\|V_{d_1p,d_2q}\| \|V_{d_2q,d_1p}\|}$$ ♦

Now, considering all pairs of V (i.e., all levels of both sub-hierarchies), we define the total cross-children similarity between d_1 and d_2 in Def. 18.

Definition 18. *The total cross-children similarity of two datatypes d_1 and d_2 is calculated as:*

$$S(d_1, d_2) = \frac{1}{L_1} \times \sum_{p=1}^{L_1} \sum_{q=1}^{L_2} m(d1p, d2q) \times CCS_{d1p,d2q}$$

where $m(d1p, d2q)$ is a Gaussian function based on Assumption 4: L_1 and L_2 are the number of levels of sub-hierarchies of d_1 and d_2, respectively. ♦

The Gaussian function is defined as follows:

$$m(d1p, d2q) = e^{-\pi \times (\frac{(depth(d1p) - depth(d2q))}{H-1})^2}$$

where $depth(d_{1p})$ and $depth(d_{2q})$ are the depths of the levels p and q respectively. H is the maximum depth of the hierarchy. Note that the depth of the hierarchy starts from 0. We denote the cross-children similarity, named $S'(d_1, d_2)$, as the average between $S(d_1, d_2)$ and $S(d_2, d_1)$ to obtained a symmetric equation.

$$S'(d_1, d_2) = 0.5 \times S(d_1, d_2) + 0.5 \times S(d_2, d_1) \tag{3.6}$$

Finally, we define similarity between datatypes d_1 and d_2 in Def. 19 as an extension of Eq. 3.1.

Definition 19. *Similarity between two datatypes d_1 and d_2, denoted as $sim(d_1, d_2)$, is determined as:*

$$sim(d_1, d_2) = \begin{cases} (1 - \omega) \times f(l) \times g(h) + \omega \times S'(d_1, d_2) & \text{if } d_1 \neq d_2 \\ 1 & \text{otherwise} \end{cases}$$

where $\omega \in [0, 1]$ is a user-defined parameter that indicates the weight to be assigned to the cross-children similarity. ♦

If the user-defined parameter ω is zero (ω=0), we have the original measure of the authors in [HMS07]. With our RDF similarity approach, we satisfy all identified requirements. This measure generates similarity values based on a hierarchy (objective, complete, and reliable) for simple datatypes. The whole approach is more suitable for the Semantic Web, because common attributes among datatypes are taking into account both in the hierarchy by Assumption 1 and in the similarity measure by Def. 17.

The following section illustrates how our approach is applied to calculate similarity between the properties of the concepts `Light Bulb` and `Lamp` from our motivating scenario and, it is compared with the work in [HMS07].

3.4.3 Illustrative Example

To better understand our similarity approach, we illustrate step by step the process to obtain the similarity between datatypes `date` from `Light Bulb` and `gYearMonth` from `Lamp`. We compare it with the one obtained by [HMS07]. To do so, we fix the parameters with the following values: $\alpha = \beta = 0.3057$ (taken from [HMS07]), and $\omega = 0.20$, which means a weigh of 20% for cross-children similarity and 80% for the distance between datatypes and their depths (i.e., $f(l)$ and $g(h)$).

According to our new datatype hierarchy, we have $l = 1$, as the distance between `date-gYearMonth`, and $h = 4$ the depth of `date`, which is the LCS. Fig. 3.5(a) shows these values and the sub-hierarchy from the LCS, according to our new hierarchy. For [HMS07], the distance between `date-gYearMonth` is $l = 2$ and $h = 2$ is the depth of the LCS, which is `Calendar`. Fig. 3.5(b) shows these values and the sub-hierarchy, according to the hierarchy in [HMS07].

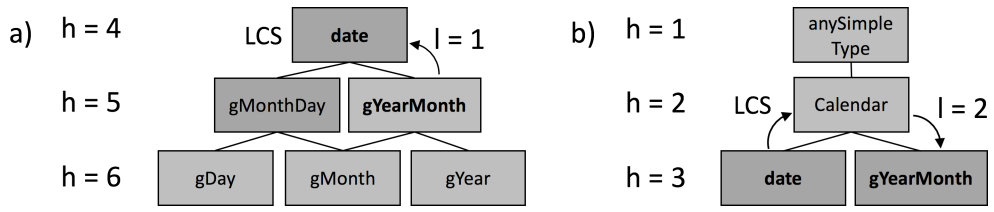


Figure 3.5: a) sub-hierarchy from our new hierarchy; b) sub-hierarchy from [HMS07]

Then, the similarity value is calculated as:

- For our similarity approach is (see Def.19):
$$sim(\texttt{date}, \texttt{gYearMonth}) = 0.80 \times f(1) \times g(4) + 0.20 \times S'(\texttt{date}, \texttt{gYearMonth}).$$

- For [HMS07] is (see Eq. 3.1):
$$c(\texttt{date}, \texttt{gYearMonth}) = f(2) \times g(2);$$

According to Eq. 3.2 and Eq. 3.3, $f(1) = 0.74$, $g(4) = 0.84$ (for our similarity approach) and $f(2) = 0.54$, $g(2) = 0.55$ (for [HMS07]). Hence, for [HMS07] the similarity value between `date-gYearMonth` is: $c(\texttt{date}, \texttt{gYearMonth}) = 0.297$.

For our similarity approach, the cross-children similarity is taken into account to finally calculate the similarity between `date-gYearMonth` (see Eq.3.6):

$$S'(\texttt{date}, \texttt{gYearMonth}) = 0.5 \times S(\texttt{date}, \texttt{gYearMonth}) + 0.5 \times S(\texttt{gYearMonth}, \texttt{date})$$

To calculate $S'(\texttt{date}, \texttt{gYearMonth})$, we have to calculate before the total cross-children similarities, S(date,gYearMonth) and S(gYearMonth,date). From Def. 18, we obtain:

$$S(\texttt{date}, \texttt{gYearMonth}) = \frac{1}{2} \times \sum_{p=1}^{2} \sum_{q=1}^{1} e^{-\pi \times (\frac{(depth(d1p) - depth(d2q))}{9-1})^2} \times CCS_{\texttt{date}p,\texttt{gYearMonth}q}$$

The two needed vectors for $CCS_{\texttt{date1,gMonthDay1}}$ are presented as follows:

$$V_{\texttt{date1,gMonthDay1}} = \langle c(\texttt{date}, \texttt{gMonthDay}), c(\texttt{date}, \texttt{gYearMonth}), c(\texttt{date}, \texttt{gDay}), c(\texttt{date}, \texttt{gMonth}) \rangle$$

$$V_{\texttt{gMonthDay1,date1}} = \langle c(\texttt{gMonthDay}, \texttt{gMonthDay}), c(\texttt{gMonthDay}, \texttt{gYearMonth}), c(\texttt{gMonthDay}, \texttt{gDay}), c(\texttt{gMonthDay}, \texttt{gMonth}) \rangle$$

Note that date has two levels of children (thus, $p = 1$ to 2 in the sum), while gYearMonth has one level of children (thus, $q = 1$ to 1 in its sum). According to Eq. 3.1, we calculate the $c(d_1, d_2)$ for each element of the vectors and we obtain the cross-children similarity of $CCS_{\texttt{date1,gMonthDay1}}$.

$$CCS_{\texttt{date1,gMonthDay1}} = \frac{\langle 0.62, 0.62, 0.46, 0.46 \rangle \cdot \langle 1.00, 0.46, 0.67, 0.67 \rangle}{\langle 0.62, 0.62, 0.46, 0.46 \rangle \cdot \langle 1.00, 0.46, 0.67, 0.67 \rangle}$$

$$CCS_{\texttt{date1,gMonthDay1}} = 0.960$$

Similarly, we calculate $CCS_{\texttt{date2,gMonthDay1}} = 0.977$. Replacing values, we have $S(\texttt{date}, \texttt{gYearMonth}) = 0.945$. An equivalent process is done to calculate $S(\texttt{gYearMonth},\texttt{date}) = 0.978$. Now, we replace the obtained values in the equation:

$$S'(\texttt{date}, \texttt{gYearMonth}) = 0.5 \times 0.945 + 0.5 \times 0.978 = 0.961.$$

The $S'(\texttt{date}, \texttt{gYearMonth})$ is replaced by the respective value in the similarity equation to finally have: $sim(\texttt{date}, \texttt{gYearMonth}) = 0.497 + 0.20 \times 0.961 = 0.688$.

Using our approach, the similarity value between date-gYearMonth has increased from 0.30 (according to [HMS07]) to 0.69. Table 3.4 compares our approach and [HMS07], with other pairs of datatypes and their respective similarity values. Note that datatypes with attributes in common (e.g., dateTime and time have in common *time*) have greater similarity value than the ones obtained by [HMS07]. Furthermore, Table 3.5 compares the similarity of some datatypes that are part of String, Period and Numeric groups. We use the new proposed hierarchy for both similarity measures ([HMS07] and our similarity). Note that the level of datatype gMonthDay (h=5) is different from the one of datatype

Table 3.4: Datatypes similarity using the proposal of [HMS07] and our approach

$Datatype_1$	$Datatype_2$	**Similarity Value [HMS07]**	**Our Similarity Value**
date	gYearMonth	0.30	0.69
date	gYear	0.30	0.46
dateTime	duration	0.30	0.37
dateTime	time	0.30	0.53
dateTime	gDay	0.30	0.29
decimal	float	0.30	0.39
double	float	0.30	0.62

`date` (h=4), and the level of datatype `gYearMonth` (h=5) is also different from the one of datatype `gYear` (h=6). However, both similarity values are the same (0.46) according to the similarity measure from [HMS07], but for our similarity measure, the similarity values are different (0.53 and 0.46, respectively). The same situation is noted in some datatypes from Numeric and String groups, observing a more adequate similarity value obtained by our similarity measure.

Table 3.5: Datatype similarity using the measure of [HMS07] applied to our new hierarchy, and our whole new approach

Datatype1	*Datatype2*	**Similarity Value New Hierarchy + Measure of [HMS07]**	**Our Similarity Value**
gMonthDay	gYearMonth	0.46	0.53
date	gDay	0.46	0.46
short	integer	0.34	0.43
int	nonPositiveInteger	0.34	0.38
long	negativeInteger	0.34	0.34
NCName	token	0.46	0.55
Name	NMTOKEN	0.46	0.51
token	NMTOKENS	0.46	0.46

Following section evaluates the accuracy of our approach.

3.5 Experimental Evaluation

In order to evaluate our approach, we adopted the experimental set of datatypes proposed in [HMS07], since there is not a benchmark available in the literature for datatype similarity. This set has 20 pairs of datatypes taken from the W3C hierarchy. These pairs were chosen according to three criteria: (i) same branch but at different depth levels (e.g.,

`int-long`); (ii) different branches with different depth levels (e.g., `string-int`); and (iii) identical pairs (e.g., `int-int`).

In [HMS07], the authors used the human perception as reference values for the 20 pairs. The closer their similarity measure is to the human perception, the better the measure performs. We used the *Human Average* similarity values presented by [HMS07] to benchmark our approach and a new *Human Average-2* dataset that we obtained by surveying 80 persons that have under- and pots-graduate degrees in computer science[9]. We also compared our work with the similarity values obtained from the compatibility table found in [BMR01] and [TLL13], and with the disjoint/compatible similarity from W3C.

To compare how close are the similarity values to the human perception, we calculate the correlation coefficient (CC) of every work (i.e., [BMR01, HMS07, TLL13], and our approach) with respect to *Human Average* and *Human Average-2*. A higher CC shows that the approach is closer to the human perception (*Human Average* and *Human Average-2*), and viceversa. The CC is calculated as follows:

$$CC = \frac{1}{n-1} \sum_{i=1}^{n} \frac{(x_i - \bar{x})}{\sigma_x} \frac{(y_i - \bar{y})}{\sigma_y}$$

where n is the number of datatype pairs to compare ($n = 20$ in this case), x_i is the similarity value between datatype pair i, and y_i is its respective human average value, $\bar{x}$ and $\bar{y}$ are averages, and σ_x and σ_y are standard deviations with respect to all similarity values x and all human average values y. Results are shown in Table 3.6.

Since the similarity measures for work [HMS07] and our work depend on the values of α and β, we evaluate the results under different assignments of α and β. To that end, we devised four experiments:

1. In the first experiment, we fix $\alpha = \beta = 0.3057$ as chosen by [HMS07], which they report to be the optimal value obtained by experimentation. We calculated the similarity values as in Eq. 3.1 to: (i) the W3C extended hierarchy [HMS07] (column 6 in Table 3.6); and (ii) our proposed datatype hierarchy (column 7 in Table 3.6). We calculated the CC for both scenarios with respect to *Human Average* and *Human Average-2*. With this experiment, we evaluated the quality of our proposed datatype hierarchy.

2. In the second experiment, we fix $\alpha = \beta = 0.3057$ as chosen by [HMS07], but instead

[9]Results are available: http://cloud.sigappfr.org/index.php/s/yRRbUQUeHs0NJnW

Table 3.6: Experimental Results: for the first and second experiments

Datatype 1	Datatype 2	Work [BMR01] (Cupic)	Work [TLL13]	W3C	Work [HMS07]	**Measure [HMS07] + our Hierarchy**	**Our Mea. + our Hierarchy**	**H. Avg. from** [HMS07]	**Our H. Avg-2**
string	normalizedString	0.00	1.00	1.00	0.53	**0.40**	**0.47**	0.27	0.77
string	NCName	0.00	1.00	1.00	0.21	**0.16**	**0.29**	0.11	0.55
string	hexBinary	0.50	1.00	0.00	0.09	**0.09**	**0.09**	0.36	0.23
string	int	0.40	0.25	0.00	0.03	**0.05**	**0.08**	0.28	0.13
token	boolean	0.00	0.17	0.00	0.05	**0.05**	**0.05**	0.37	0.15
dateTime	time	0.90	1.00	0.00	0.30	**0.53**	**0.53**	0.70	0.71
boolean	time	0.00	0.58	0.00	0.09	**0.06**	**0.06**	0.04	0.13
int	byte	0.00	1.00	1.00	0.52	**0.52**	**0.52**	0.71	0.58
int	long	0.00	1.00	1.00	0.67	**0.67**	**0.73**	0.79	0.72
int	decimal	0.00	1.00	0.00	0.29	**0.29**	**0.38**	0.59	0.55
int	double	0.00	0.83	0.00	0.12	**0.21**	**0.23**	0.51	0.50
decimal	double	0.00	0.83	0.00	0.30	**0.53**	**0.55**	0.60	0.72
byte	positiveInteger	0.00	1.00	1.00	0.13	**0.13**	**0.13**	0.57	0.49
gYear	gYearMonth	0.00	1.00	0.00	0.30	**0.67**	**0.67**	0.65	0.65
int	int	1.00	1.00	1.00	1.00	**1.00**	**1.00**	1.00	1.00
string	byte	0.00	0.25	0.00	0.02	**0.03**	**0.03**	0.34	0.21
token	byte	0.00	0.25	0.00	0.01	**0.01**	**0.01**	0.46	0.24
float	double	0.00	1.00	0.00	0.30	**0.62**	**0.62**	0.60	0.75
float	int	0.00	0.83	0.00	0.12	**0.16**	**0.16**	0.46	0.47
gYear	negativeInteger	0.00	0.83	0.00	0.03	**0.01**	**0.01**	0.02	0.10
CC. wrt. H. Avg [HMS07]		40.33%	38.32%	27.45%	69.45%	**80.21%**	**77.15%**	100.0%	-
CC. wrt. our H. Avg-2		29.48%	70.69%	51.09%	83.93%	**90.23%**	**92.39%**	-	100.0%

of using their measure (Eq. 3.1), we used our cross-children similarity measure (see Def. 19) with our proposed datatype hierarchy (column 8 in Table 3.6). We fixed the $\omega = 0.20$[10]. With this experiment, we compared the quality of our approach against all other works.

3. In the third experiment, we chose values for α and β from the range $(0, 1]$, with a 0.02 step. In this case, 2010 possibilities were taken into account.

4. The fourth experiment is similar to the third one, except that a smaller step of 0.001 is considered. Therefore, there were 999181 possibilities.

As shown in Table 3.6, for experiments 1 and 2, we obtained a CC of 80,21% and 77.15% respectively, with respect to the *Human Average*. With respect to our *Human Average-2*, we obtained even better CC (90,23% and 92.39%).

In the third experiment, we obtained our best results for $\alpha = 0.20$ and $\beta = 0.02$, $CC = 82.60\%$ with respect to the *Human Average* (see Table 3.7, row 1). For $\alpha = 0.50$ and $\beta = 0.18$, $CC = 95.13\%$ with respect to our *Human Average-2* (see Table 3.7, row 2). In general, the similarity values generated by our work were closer to both human perception values than the other works (99.90% of the 2010 possible cases).

Table 3.7: Third experiment with step = 0.001

	α	β	CC.
Human Average [HMS07]	0.20	0.02	82,60%
Human Average-2	0.50	0.18	95.13%

Similarly, for the fourth experiment, we obtained our best results for $\alpha = 0.208$ and $\beta = 0.034$ with a CC=82.76% with respect to the *Human Average* of the work [HMS07] (see Table 3.8, row 1). With respect to our *Human Average-2*, we obtained the best results for $\alpha = 0.476$ and $\beta = 0.165$, with a CC=95.26% (see Table 3.8, row 2). In general, the similarity values generated by our work were closer to both human perceptions (99.97% of the 999181 possible cases).

Table 3.8: Forth experiment with step = 0.01

	α	β	CC.
Human Average [HMS07]	0.208	0.034	82.76%
Human Average-2	0.476	0.165	95.26%

In conclusion, our approach outperforms all other works that we surveyed by considering a new hierarchy that captures a semantically more meaningful relation among

[10]By experimentation, we determined this value as the optimal one.

datatypes, in addition to a measure based on cross-children similarity. Note that our work is not exclusive to RDF data; it can be also applied to XML data similarity and XSD/ontology matching.

3.6 Summary

In this chapter, we analyzed the datatypes, the current datatype hierarchy proposed by the W3C, and its limitations for the Semantic Web. We also investigated the issue of datatype similarity for the application of RDF matching/integration. In this context, we proposed a new simple datatype hierarchy aligned with the W3C hierarchy, containing additional types to cope with `XPath` and `XQuery` requirements in order to ensure an easy adoption by the community. Also, a new datatype similarity measure inspired by the work in [HMS07], is proposed to take into account the cross-children similarity.

This similarity measure is independent of the values within the nodes, therefore, it can be applied to any hierarchy/taxonomy. For instance, we apply this contribution in our protection approach (see Chapter 5) where a new predicate is returned based on a hierarchy provided by the expert user.

We experimentally compare the effectiveness of our proposal (datatype hierarchy and similarity measure) against existing related works. Our approach produces better results (closer to what a human expert would think about the similarity of compared datatypes) than the ones described in the literature.

Include complex datatypes in this contribution is the next challenge. Also, the analysis of semantic types, which are more specific for the Semantic Web, can complement the scape of this work.

Chapter 4

The Semantic Web: Datatype Inference

> "For a lot of companies, it's useful for them to feel like they have an obvious competitor and to rally around that. I personally believe it's better to shoot higher. You don't want to be looking at your competitors. You want to be looking at what's possible and how to make the world better."
>
> — Larry Page

As we described in Chapter 3, datatypes play an important role on RDF matching/integration. However, a huge quantity of RDF documents is incomplete or inconsistent in terms of datatypes [PHHD10]. Hence, when this information is missing, datatype inference emerges as a new challenge in order to obtain more accurate RDF document matching results. We recall that datatypes can be classify as simple and complex, and the W3C proposes a hierarchy.

In the context of XSD, works such as [Chi02, HNW06] infer simple datatypes by a pattern-matching process on the format of the values; i.e., the characters that make unique a datatype, which is called *lexical space* according to the W3C Recommendation [PVB04]. These works consider a limited number of simple datatypes (`date`, `decimal`, `integer`, `boolean`, and `string`), thus for other datatypes, as `year` (e.g., `1999`), this method cannot determine its correct datatype, since it is identified as an `integer`.

Others works in the context of programming languages and OWL are focused on inferring complex datatype through axioms, assigned operations, and inference rules [FP06,

Hol13, PB13], which are elements not present in an RDF document for simple datatypes. Thus, in the context of RDF document matching/integration, these works are not suitable mainly for two reasons:

1. Lexical space based methods cannot infer all simple datatypes, since there are intersections between datatype lexical spaces (e.g., 1999 can be an `integer` or a `gYear` according to the lexical space of both W3C datatypes); and
2. Complex datatype inference methods cannot be applied to simple datatypes, since in RDF, a simple datatype is an atomic value associated to a predicate.

To overcome these limitations, we propose a new approach that considers, in addition to the lexical space analysis, the analysis of the predicate information related to the object. It consists of four steps:

1. Analysis of predicate information, such as *range property* that defines and qualifies the type of the object value;
2. Analysis of lexical space of the object value, by a pattern-matching process;
3. Semantic analysis of the predicate and its semantic context, which consists in identifying related words or synonyms that can disambiguate two datatypes with similar lexical space; and
4. Generalization of Numeric and Binary datatypes, to ensure a possible integration among RDF documents.

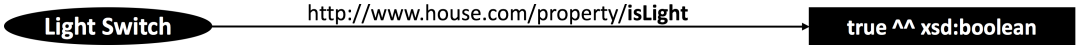
The rest of this chapter is organized as follows: Section 4.1 presents a motivating scenario to illustrate the importance of datatypes. Section 4.2 surveys the related literature. Section 4.3 describes our inference approach. Section 4.5 shows the experiments to evaluate the accuracy and performance of our approach. Finally, we present some discussion and reflections in Section 4.6.

4.1 Motivating Scenario

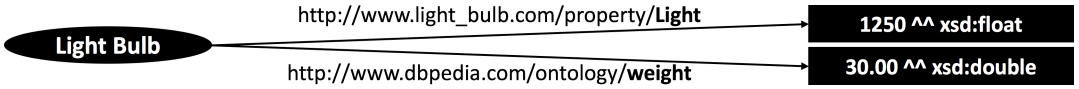
In the motivating scenario presented in Chapter 3, we show the importance of datatypes for the similarity between concepts, properties, and relations in the context of RDF document matching/integration. In this chapter, we complement this aspect showing another

scenario with more ambiguous information, when datatype information is not presented. Then, we need to integrate three RDF documents with similar concepts (resources) but based on different vocabularies. Fig. 4.1 shows three concepts from three different RDF documents that we want to integrate. Figs. 4.1a and 4.1b describe the concept `Light Switch`, with property (predicate) `isLight`, whose datatype is `boolean`. However, they are represented with different lexical spaces: binary lexical space with value `1` in Fig. 4.1a and string lexical space with value `true` in Fig. 4.1b. In both cases, `isLight` property expresses the state of the light switch (i.e., turned on or turned off). Fig. 4.1c shows the concept `Light Bulb`, with property `Light`, whose datatype is `float`, and property `weight` with datatype `double`.

(a) Light Switch. Datatype boolean: Binary Lexical Space $(0, 1)$



(b) Light Switch. Datatype boolean: String Lexical Space $(false, true)$



(c) Light Bulb

Figure 4.1: Three concepts from three different RDF documents

For the integration, it is necessary to analyze the information of their concept properties. Intuitively, considering the datatype information, we can say that:

1. Both `Light Switch` concepts from Figs. 4.1a and 4.1b are similar, since their properties are similar: the `isLight` property is `boolean` in both cases, and boolean literals can be expressed either as binary values (0 or 1) or string values (true or false) according to the W3C [PVB04].

2. `Light Bulb` concept is different from the other ones. Indeed, the `Light` property is expressed with `float` values, expressing the light intensity, that has nothing to do with light switch state (i.e., turned on or turned off).

If the datatype information is missing and the integration is made only based on literals, we have problems related to the *ambiguity* of properties. Contrary to our intuition,

concepts in Figs. 4.1a and 4.1b are incompatible because of the use of different lexical spaces (i.e., value `1` is not compatible with value `true`, which can be considered as a `string` datatype instead of `boolean`). The integration of concept `Light Switch` from Fig. 4.1a with concept `Light Bulb` from Fig. 4.1c will be possible, even though it is incorrect. The `Light` properties of both respective documents are compatible because the lexical spaces of their values are the same (1 and 1275 respectively, can be `integer`). With the presence of datatype information, we can avoid this ambiguity even if the lexical spaces of the values are compatible.

In this scenario, we can realize the role of datatype inference, when this information is missing, for matching/integration of RDF documents. Thus, an approach capable of inferring the datatype from the existing information is needed.

In the following section, we survey existing works on datatype inference. We highlight their limitations and discuss their possible applications on RDF document matching/integration.

4.2 Related Work

To the best of our knowledge, no prior work manages simple datatype inference for RDF documents. However, datatype inference has been addressed in the context of XSD, programming languages, and OWL (theoretical approaches) and there are tools for XSD available on the Web. To evaluate the existing works, we have identified the following criteria of comparison:

1. Consideration of *simple* datatypes, since this is the scope of the work;

2. Analysis of *local information*, such the object value, and *external information*, since the Semantic Web allows the integration of resources available on the Web; and

3. *Suitability for the Semantic Web*, the whole method should be objective, complete and applicable for any domain.

Following sections describes the theoretical and tools approaches

4.2.1 Theoretical Approaches

For **theoretical approaches**, we classify the existing works, according to similar solutions, into four groups:

Lexical space based approaches

In the inference of XSD from XML documents, datatypes are reduced to a small set of values (`date`, `decimal`, `integer`, `boolean`, and `string`) or to only `string` datatypes [Chi02, HNW06]. The authors in [HNW06] propose a hierarchy between the reduced datatypes according to the *lexical spaces* of the W3C Recommendation (see Fig. 4.2). The *lexical space* of a datatype describes the representation format and restricts the use of characters for the object values. The proposal returns the most specific datatype that subsumes the candidate datatypes obtained from the patter-matching of the values. However, a `gYear` value is reduced to `integer`, which is incorrect. Table 4.1 shows the *lexical spaces* of simple datatypes according to the W3C.

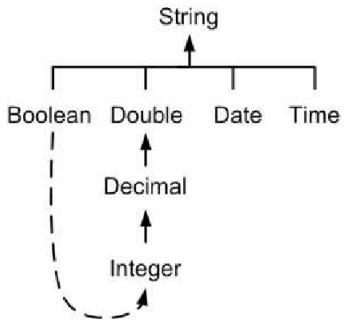


Figure 4.2: Hierarchical structure to recognize datatypes. Solid lines describe strictly hierarchical relations, the dotted line a loose relation [HNW06]

Axioms, constructors, and operations based approaches

In the context of programming languages, the authors in [FP06] focus on inferring complex datatypes, modelling them as a collection of constructor, destructor, and coercion functions. Other works [Hol13, WGMH13], also use axioms and pattern matching over the constructors of the datatype during the inference process. In [ACH08, Bou15], operations and a syntax associated to datatypes are analyzed to infer complex datatypes. Simple datatypes such as `date` and `integer` are mainly inferred by a pattern-matching process of the value format using the *lexical spaces*. However, some simple datatypes

Table 4.1: Lexical Space for Simple Datatypes (W3C Recommendation [PVB04])

Datatype	**Lexical Space**	**Examples**
string	Any character	”Example 123”
duration	PnYnMnDTnHnMNS	P1Y2M3DT10H30M
dateTime	CCYY-MM-DDThh:mm:ss-UTC	1999-05-31T13:20:00-05:00
time	hh:mm:ss	13:20:00-05:00
date	CCYY-MM-DD	1999-05-31
gYearMonth	CCYY-MM	1999-05
gYear	CCYY	1999
gMonthDay	–MM-DD	–05-31
gDay	–DD	–31
gMonth	–MM–	–05–
boolean	true, false, 1, 0	false
base64Binary	Base64-encoded	0YZZ
hexBinary	Hex-encoded	0FB7
float	32-bit floating point type	12.78e-2, 1999
decimal	Arbitrary precision	12.78e-2, 1999
double	64-bit floating point type	12.78e-2, 1999
integer	[0-9]	1999

have intersection among their *lexical spaces* as `gYear` and `integer`, therefore, and this pattern-matching process is not able to infer a correct datatype.

Inference rules based approaches

In the context of OWL, the authors in [PB13] propose a method to heuristically generate type information by exploiting axioms in a knowledge base. They assign type probabilities to the assertions. In the domain of health-care, [SFJ15] proposes a type recognition approach (inference type) by associating a weight to each predicate, using support vector machines to model types and by building a dictionary to map instances. For [LHLZ15], the Semantic Web needs an incremental and distributed inference method because of the long ontology size. The authors use a parallel and distributed process (MapReduce) to “reduce” the “map” of new inference rules. The authors in [KMK15] state that DBpedia only provide 63.7% of type information. Hence, they propose an approach to discover complex datatypes in RDF datasets by grouping entities according to the similarity between incoming and outgoing properties. They also use a hierarchical clustering and the confidence of types for an entity. The use of inference rules helps to infer datatypes where a specific information is known (e.g., type of properties, knowledge database). However, RDF data is not always available with its respective ontology, which makes impossible the task of

formulating inference rules.

Semantic analysis based approaches

In [GTSC16], the authors analyze two types of predicates: object property (semantic type, e.g., `dbr:Barack Obama`) and datatype property (syntactic type, e.g., `xsd:string`). They propose an approach to infer the semantic type of string literals using the word detection technique called Stanford CoreNLP [11] to identify the principal term and the UMBC[12] semantic similarity service to discover the semantic class. However, a semantic type is not always related to the same datatype, since it depends on the datatype defined in the structure. A value can be expressed as a `string` or `integer` according to two different ontologies.

4.2.2 Tools

On the other hand, there are **tools** that generate XSD from XML documents, inferring the type of data from existing values (*lexical spaces*), such as XMLgrid [XML10], FreeFormatted [fre11], and XmlSchemaInference by Microsoft [Mic]. However, they do not share a standard process to infer datatypes. For example, the attribute `weight` and `isLight` from the following XML document extracted from Fig. 4.1, have different inferred datatypes according to these three tools.

```
<Light_Bulb>
  <Light>1250</Light>
  <weight>30.00</weight> </Light_Bulb>
<Light_Switch>
  <isLight>1</isLight>
</Light_Switch>
```

- XMLgrid infers `weight` as `double` and `isLight` as `int`;
- Using FreeFormatted, the datatype for `weight` is `float` and for `isLight` is `byte`;
- While according to XmlSchemaInference `weight` is `decimal` and `isLight` is `unsignedByte`.

[11] CoreNLP is a natural language analysis tool for text that extract particular relations, datatypes, etc. - http://stanfordnlp.github.io/CoreNLP/

[12] Semantic similarity service that analyses semantic relations between words/phrases extracted from Wordnet - http://swoogle.umbc.edu/

The criteria used to infer the datatype are unknown since these tools do not describe their algorithms. Thus, the direct application of existing approaches presents limitations in the context of RDF document integration/matching.

Table 4.2: Related Work Classification

Work	**Inference Method**	**Requirements**				
			Information		**SW**	
		Simple Datatypes	**Local**	**External**	**XML XSD**	**RDF OWL**
[Chi02, HNW06]	Lexical Space	Reduced Set	✓	X	✓	X
[FP06, Hol13, WGMH13] [ACH08, Bou15]	Axioms, operations, constructors	Only Complex	✓	X	✓	X
[PB13, SFJ15, KMK15]	Inference rules	Only Complex	✓	X	X	✓
[GTSC16]	Semantic Analysis	Only `string`	✓	✓	X	✓
Tools:[XML10, fre11, Mic]	Not provided	Not provided	✓	X	✓	X

Table 4.2 shows our related work classification. Note that none of the works satisfies all the defined requirements. Following section describes our datatype inference process.

4.3 Inference Process: Our Proposal

Our datatype inference approach mainly relies on a four step process that considers the annotations on the predicate, the specific format of literal object values, the semantic context of the predicate, and the generalization of datatype for Numeric and Binary groups. Fig. 4.3 shows the framework of our inference process composed by the four steps. Each step can be applied independently and in different orders according to user parameters.

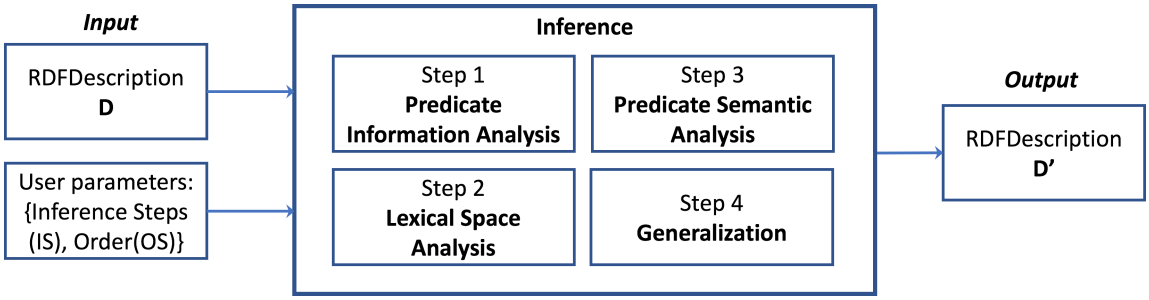


Figure 4.3: Framework of our RDF Inference process

The input of our framework is an RDF Description which can be represented in different serializations formats (such as RDF/XML, Turtle, N3) and the user parameters (inference steps and their order). The output is an RDF Description with its respective inferred datatypes. A description of each step is presented as follows:

4.3.1 Predicate Information Analysis (Step 1)

In a triple `t: <s,p,o>`, the predicate `p` establishes the relationship between the subject `s` and the object `o`, making the object value `o` a characteristic of `s`. Information (properties) such as `rdfs:domain` and `rdfs:range` can be associated to each predicate to determine the type of subject and object. As one of the steps to deduce the simple datatype of a particular literal object, we propose to inspect the property `rdfs:range`, if this information exists. We formally describe this Step 1 with the following definitions and rule.

Definition 20. ***Predicate Information (*PI*)****: Given a triple* $t :< s, p, o >$*, Predicate Information is a function, denoted as* ***PI(t)****, that returns a set of triples defined as:* $PI(t) = \{t_i \mid t_i =< s_i, p_i, o_i >\}$*, where:*

- s_i *is the predicate of t (t.p), acting as a subject on each* t_i *triple;*
- p_i *is an RDF defined property* $\in \{rdfs{:}type, rdfs{:}label, rdfs{:}range, ...\}$*;*
- o_i *is the value of* p_i*.* ♦

Table 4.3 shows the set of triples (PI), returned by the function Predicate Information, for property `dbp:weight`, which is presented in Fig. 4.1c.

Table 4.3: Example of the set of triples of Predicate information (PI) for `dbp:weight`

Subject	**Predicate (Property)**	**Object (Value)**
dbp:weight	rdf:type	owl:DatatypeProperty
dbp:weight	rdfs:label	Gewicht (*g*) (de)
dbp:weight	rdfs:label	gewicht (*g*) (nl)
dbp:weight	rdfs:label	peso (*g*) (pt)
dbp:weight	rdfs:label	poids (*g*) (fr)
dbp:weight	rdfs:label	weight (*g*) (en)
dbp:weight	rdfs:label	weight (*g*) (en)
dbp:weight	**rdfs:range**	**xsd:double**
dbp:weight	prov:wasDerivedFrom	http://mappings.dbpedia.org/ OntologyProperty:weight

Definition 21. Predicate Range Information (*PRI*)**:** *Given a triple* $t :< s, p, o >$*, Predicate Range Information is a function, denoted as* ***PRI(t)****, that returns the value associated to the* `rdfs:range` *property, defined as:*

$$PRI(t) = \begin{cases} t_i.o & if\ \exists t_i \in PI(t) \mid t_i.p = \texttt{rdfs:range}, \\ null & \text{otherwise.} \end{cases}$$ ♦

Applying Def. 21 to the set of predicate information (PI) of property `dbp:weight` (see Table 4.3), the Predicate Range Information function returns the value `xsd:double`.

Definition 22. Is Available (*IA*): *Given a predicate* ***p****, Is Available is a boolean function, denoted as* ***IA(p)****, that verifies if* ***p*** *is an IRI available on the web:*

$$IA(p) = \begin{cases} True & if\ p\ returns\ code\ 200; \\ False & \text{otherwise.} \end{cases}$$ ♦

Using the three previous definitions, we formalize our first inference rule.

Rule 1. Datatype Inference by Predicate Information Analysis:
Given a triple $t :< s, p, o >$*, in which* $o \in L$ *(Def. 8), the datatype of o is determined as follows:* ***R1:*** *if* $IA(p) \implies$ *datatype* $= PRI(t)$.

Rule 1 verifies if the predicate of the triple is an IRI available (Def. 22) and Def. 21 determines if the `rdfs:range` property exists from the set of triples extracted by Def. 20.

Alg. 1 is a pseudo-code of how this rule can be implemented in high level programming language.

Algorithm 1: Predicate Information Analysis

```
   Input: Triple t=< s,p,o >
   Output: Datatype dt
1  if IA(t.p) then
2      Graph triples = PI(t);          //Set of triples with information from
         predicate.
3      foreach triple in triples do
4          if triple.p == rdfs:range then
5              dt = t.o;               //If range information exists, the datatype is
                 returned.
6              return dt;
7      return dt=null;                 //Range information does not exist.
8  return dt=null;                     //There is not external information available.
```

As the input, the algorithm receives the triple t :`<s,p,o>`, from which we want to determine the datatype of its object value. If the IRI representing the predicate exists (line 1 – Def. 22), we access the link to extract all available information as triples (line 2 – Def. 20). For example, if we have `dbp:weight` (more specifically `http://www.dbpedia.org/-ontology/weight`) as the predicate, we can get the list of triples shown in Table 4.3 (each row in Table 4.3 represents a triple). If among these triples there is the property `rdfs:range`, then its associated object value, which is the datatype, is returned (lines 3 to 5), otherwise null is returned (lines 7 – (Def. 21). According to Fig. 4.1c, the output of the algorithm will be the datatype `xsd:double`.

This algorithm examines external information and it is independent of the query language. Rule 1 can also be implemented as a simple SPARQL query:

```
select distinct ?datatype where
  {?subject ?predicate ?literal.
   ?predicate rdfs:range ?datatype }
```

Where `?subject ?predicate ?literal` is the triple to be analyzed; `?predicate` is the analyzed predicate (`dbp:weight` in example of Table 4.3); and `?datatype` is the returned result. Following step analyzed the lexical space of a literal object in order to infer its datatype.

4.3.2 Datatype Lexical Space Analysis (Step 2)

According to Def. 6, a datatype is a 3-tuple consisting of: (i) a set of distinct valid values, called *value space*; (ii) a set of lexical representations, called *lexical space*; and (iii) a total mapping from the lexical space to the value space. In some cases, the datatype can be inferred from its *lexical space*, when it is uniquely formatted (e.g., value `1999-05-31` matches with the format `CCYY-MM-DD`, which is the *lexical space* of datatype `date`). However, in several cases (such as `boolean`, `gYear`, `decimal`, `double`, `float`, `integer`, `base64Binary`, and `hexBinary`), the *lexical spaces* of datatypes have common characteristics, leading to ambiguity (e.g., value `1999` matches with *lexical spaces* of `gYear` and `float` – see Table 4.1). Figure 4.4 illustrates graphically the *lexical space* intersections of W3C simple datatypes.

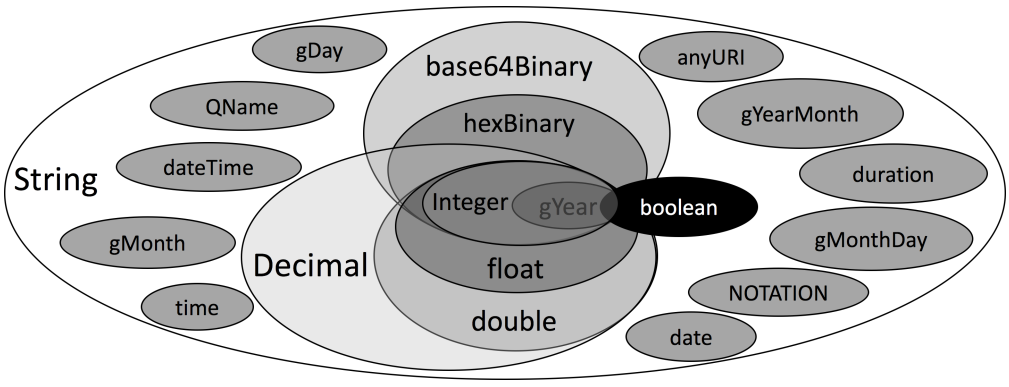


Figure 4.4: Datatype Lexical Space Intersection

To compare the datatype lexical spaces with the literal values, we classify the datatypes in a hierarchy (see Fig. 4.5) based on the lexical spaces intersections (from a general

lexical space to a specific one). Since all literal values can be strings, the `string` datatype is the root of the hierarchy.

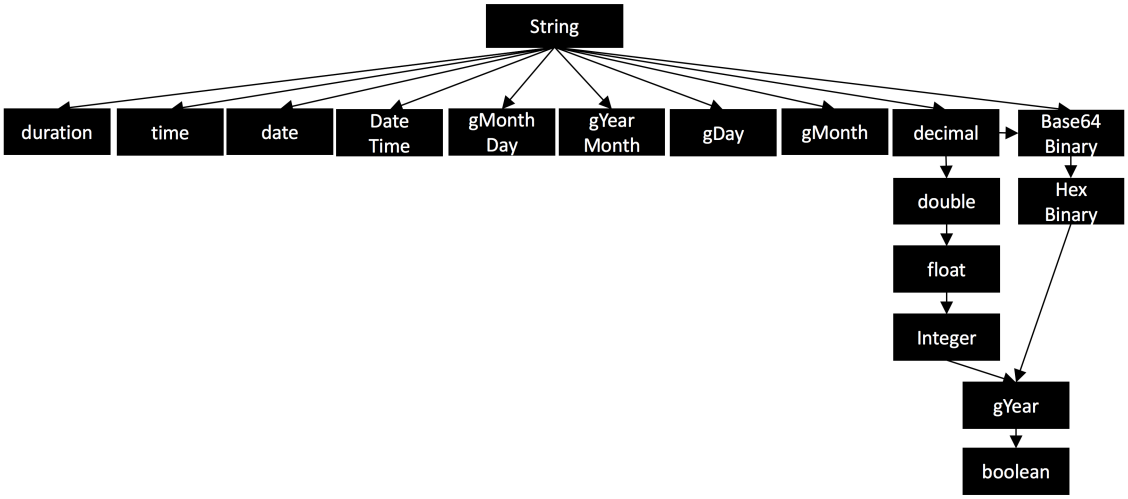


Figure 4.5: Lexical Space Hierarchy

To analyze the *lexical spaces*, we propose the following definition.

Definition 23. Candidate Datatypes (*CDT*): *Given a literal object o, the set of its candidate datatypes is determined by the function Candidate Datatypes, defined as:*

$$CDT(o) = \{dt \mid dt \in SDT\ (Def.\ 11) \wedge o \in LS(dt)\ (Def.\ 6)\} \qquad \blacklozenge$$

By Def. 23, the set of candidate datatypes of the object literal value `1` presented in Fig. 4.1a is: `CDT(1)={float, decimal, double, hexBinary, base64Binary, integer, boolean, string}`. Based on this definition, we formally define our second inference rule.

Rule 2. Datatype Inference by Lexical Space:
Given a triple $t :< s, p, o >$, in which $o \in L$, the datatype of o is determined as follows:

$$\textbf{\textit{R2:}} \qquad datatype = \begin{cases} \texttt{string} & \text{if } |CDT(o)| = 1, \\ dt \mid dt \in CDT(o) \wedge dt \neq \texttt{string} & \text{if } |CDT(o)| = 2, \\ null & \text{otherwise.} \end{cases}$$

Rule 2 analyzes the number of possible datatypes of a literal object value. The order to analyze the lexical space of each datatypes is established by a lexical-space hierarchy (see Fig. 4.5). In all cases, the datatype `string` is a candidate datatype, since it has the most general *lexical space* (see Fig. 4.4 or Fig. 4.5); if the number of candidate datatypes is one, then the only datatype, which is `string`, is returned. If the number of candidate datatypes is two, then the other datatype is returned. Otherwise, we have an **ambiguous case**

and any decision cannot be provided. Hence, the inference process remains incomplete due to the ambiguous cases and further analysis is needed.

A pseudo-code following the definition of Rule 2 is proposed in Alg. 2.

Algorithm 2: Lexical Space Analysis

```
   Input: Triple t=< s, p, o >
   Output: Datatype dt
1  DT dt_list = new [Datatype(String)];   // Every value is a string (see Fig. 2).
2  dt_list = CDT(t.o); // Pattern-matching verification with all simple datatype
     lexical spaces.
3  if size(dt_list) > 2 then
4  |  return dt=null;                    // Ambiguous case.
5  else
6  |  if size(dt_list) == 1 then
7  |  |  dt = first value of dt_list;     // Non ambiguous case, it is string.
8  |  else
9  |  |  dt = second value of dt_list;   // Non ambiguous case, string and
   |  |    other datatype.
10 |  return dt;
```

The Algorithm receives a triple t :<s,p,o> and returns a list of candidate datatypes that can be associated to the object. This list is initialized with the datatype `string`, because any object value is a `string` (line 1). According to the *lexical spaces* defined by the W3C (see Table 4.1), the list of candidate datatypes is generated by a pattern matching process (line 2 in Alg. 2 – Def. 23) following the order obtained from the hierarchy shown in Figure 4.5.

If the number of candidate datatypes is more than 2, we are under an ambiguous case, since the lexical space of the literal value matches with several lexical spaces of the datatypes (lines 3 and 4 of Alg. 2). If we have only `string` as a candidate datatype, then this is the returned information (line 7 of Alg. 2). If we get two candidate datatypes, they are `string` and another one; thus, the datatype of the object value is the second one (line 9 of Alg. 2).

The following step analyses semantically the predicate of the literal object through the definition of context rules.

4.3.3 Predicate Semantic Analysis (Step 3)

In presence of ambiguous cases, a semantic analysis of the predicate can be done to resolve ambiguity. The predicate name can define the context of the information in a scenario where the data is consistent. Regarding the W3C datatype *lexical spaces*, the datatypes `boolean`, `gYear`, `decimal`, `double`, `float`, `integer`, `base64Binary`, and `hexBinary` datatypes are ambiguous. However, the ambiguity of `boolean`, `gYear`, and `integer`, in some specific scenarios, can be resolved by examining the context of its predicate according to a knowledge base. For example, the predicate `dbp:dateOfBirth` has the context *date*, then it is possible to assume `gYear` as the datatype; the predicate `dbp:era` has the context *period* and the datatype assigned can be `integer`; however, for predicate `dbp:salary`, it is possible to assign datatypes `decimal`, `double`, or `float`; the ambiguous case persists. In order to describe our inference process in this step, we formalize a knowledge base as follows:

Definition 24. Knowledge Base (*KB*): *Knowledge bases (thesaurus, taxonomies, and ontologies) provide a framework to organize entities (words/expressions, generic concepts, etc.) into a semantic space. A knowledge base has the following defined functions:*

– **Similarity (*sim*)**: *Given two word values* $\boldsymbol{n}$ *and* $\boldsymbol{m}$*, Similarity is a function, denoted as* ***sim(n,m)****, that returns the similarity value among the words:*
$sim(n, m) = A\ similarity\ value \in [0, 1]\ between\ \boldsymbol{n}\ and\ \boldsymbol{m}\ according\ to\ KB.$

– **IsPlural (*IP*)**: *Given a string value* $\boldsymbol{n}$*, IsPlural is a function, denoted as* ***IP(n)****, that returns True if the word* $\boldsymbol{n}$ *is plural:*

$$IP(n) = \begin{cases} True & if\ n\ is\ plural\ according\ to\ KB; \\ False & \text{otherwise.} \end{cases}$$

– **IsCondition (*IC*)**: *Given a string value* $\boldsymbol{n}$*, IsCondition is a function, denoted as* ***IC(n)****, that returns True if the word* $\boldsymbol{n}$ *is a condition:*

$$IC(n) = \begin{cases} True & if\ n\ is\ a\ condition\ according\ to\ KB; \\ False & \text{otherwise.} \end{cases}$$ ♦

The semantic context is formalized, based on the knowledge base, as follows:

Definition 25. Context (*ct*): *A context is a related word or synonym, which clarifies or generalizes the domain of a word. It is associated to a similarity value according to a*

knowledge base. A context is denoted as a 3-tuple $ct :< w, y, v >$*, where* w *is a word;* y *is a related word of* w*; and* v *is sim(w,y)* $\in [0, 1]$. ♦

Definition 26. Set of Contexts (*CT*): *Given a word* w*, its set of contexts is defined as* $CT = \{ct_i \mid ct_i :< w, y_i, v_i > \text{ is a context of } w\}$. ♦

For example, from Fig. 4.1c, the set of contexts of predicate `weight` is: `CT = {<weight,load,0.8>, <weight,heaviness,0.5>, <weight,obesity,0.4>, <weight,-size,0.3>}`

Definition 27. Predicate Context (*PC*): *Given a triple* $t :< s, p, o >$ *and a threshold* $\boldsymbol{h}$*, Predicate Context is a function, denoted as* ***PC(t,h)****, that returns a set of contexts defined as:*

$$PC(t, h) = \{ct_i \mid ct_i :< p.property\_name_i, y_i, v_i >, v_i \geq h\}.$$ ♦

The context can determine the datatype for some literal objects through a semantic analysis, then we assume two scenarios for an ambiguous case:

– If date is in the context ($< word, \textbf{date}, 0.5 >$) and the literal value is a number (e.g., 1999), then the datatype is `gYear` because `gYear` (1999) is a part of datatype `date` (**1999**-05-31);
– If period is in the context ($< word, \textbf{period}, 0.5 >$) and the literal value is a number (e.g., 3 months), then the datatype is `integer` because it is about quantity.

However, if the context is date, the word from which we obtain the context, cannot be plural, since plural words express quantities. Thus, in this case the word is related to the datatype `integer` according to our scenarios. Def. 28 generalizes our scenarios to assign a datatype to a literal object, according to the context of its corresponding predicate name.

Definition 28. Predicate Name Context (*PNC*): *Given a triple* $t :< s, p, o >$*, in which* $o \in L$ $(Def.\ 8)$*, and a threshold* $\boldsymbol{h}$*, Predicate Name Context is a function, denoted as* ***PNC(t,h)****, that returns a datatype defined as:*

$$PNC(t, h) = \begin{cases} \texttt{gYear} & if \exists ct_i \in PC(t, h) \mid ct_i.y_i = date \wedge \texttt{gYear} \in CDT(o); \\ \texttt{integer} & if \exists ct_i \in PC(t, h) \mid ct_i.y_i = date \wedge \texttt{integer} \in CDT(o) \\ & \wedge IP(p.property\_name); \\ \texttt{integer} & if \exists ct_i \in PC(t, h) \mid ct_i.y_i = period \wedge \texttt{integer} \in CDT(o); \\ null & \text{otherwise.} \end{cases}$$ ♦

In addition, to determine a datatype as `boolean`, we assume that a word is defined as condition in a knowledge base (e.g., Wordnet).

Using the previous definitions, we formally define our third inference rule.

Rule 3. Datatype Inference by Semantic Analysis*:*
Given a triple $t :< s, p, o >$*, in which* $o \in L$*, and a threshold* $\boldsymbol{h}$ *the datatype of* o *is determined as follows:*

$$\textbf{\textit{R3:}} \qquad datatype = \begin{cases} \texttt{boolean} & if\, \texttt{boolean} \in CDT(o) \wedge IC(p.name\_property); \\ PNC(t, h) & \text{otherwise.} \end{cases}$$

Rule 3 returns the datatype of the object value when a defined context associated to the predicate exists. If that is not the case, we are still under an ambiguous case. Note that Rule 3 is proposed for a scenario where the data is consistent with the W3C recommendations (e.g., self-descriptive names).

Alg. 3 is a pseudo-code of our semantic analysis step. The algorithm receives the triple t :<s,p,o> to be analyzed. For the analysis of the predicate name, an external service is required in order obtained the synonyms of the predicate name, called contexts (line 2 in Alg. 3). If more than one defined context is available in the set of contexts (Def. 27), the algorithm returns the one which has more similarity value (lines 16 – 17 in Alg. 3). A null datatype is returned if none defined context is present.

Algorithm 3: Predicate Semantic Analysis

Input: Triple $t =< suject, predicate, object >$, float h
Output: Datatype *dt*

```
1  DT dt_list = CD(t)
2  PC contexts = get_context_SERVICE(t.predicate.name, h);
3  DT candidates = {}
4  if contexts contains date then
5  |  if t.predicate.name is plural and dt_list contains integer then
6  |  |  dt = new Datatype(integer);
7  |  |  candidates.add(dt);
8  |  else if dt_list contains gYear then
9  |  |  dt = new Datatype(gYear);
10 |  |  candidates.add(dt);
11 if context is period and dt_list contains integer then
12 |  dt = new RDFDatatype(integer);
13 |  candidates.add(dt);
14 if context is condition and dt_list contains boolean then
15 |  dt = new Datatype(boolean);
16 |  candidates.add(dt);
17 candidates.ORDER_BY_DESC()
18 return candidates.first or dt = null;
```

Following step describes the generalization method for literal values that are part of

numeric and binary groups.

4.3.4 Generalization of Numeric and Binary Groups (Step 4)

If we still have ambiguity, as an alternative to disambiguate the datatypes `decimal`, `double`, `float`, `integer`, `base64Binary`, and `hexBinary`, we propose two groups of datatypes: **Numeric** and **Binary**. In each group, we define an order among the datatypes by considering *lexical space* intersection (see Fig. 4.4). Hence, for the Numeric Group, we have `decimal` > `double` > `float` > `integer` and in the Binary Group, `base64Binary` > `hexBinary`. According to these groups, we return the most general datatype, if all candidate datatypes belong only to one of these two groups.

Definition 29. Generalization (G): *Given a literal object o, the set of its candidate datatypes is reduced by the function Generalization, defined as:* $G(o) = \{dt \mid dt \in CDT(o) \wedge (dt$ *is the most general datatype according to* ***Numeric*** *and* ***Binary*** *groups)*$\}$

♦

Note that datatype `string` is always part of candidate datatypes. We formally define our fourth inference rule as follows:

Rule 4. Datatype Generalization:
Given a triple $t :< s, p, o >$, in which $o \in L$ ($Def.$ 8), the datatype of o is determined as follows:

$$\textbf{\textit{R4:}} \qquad datatype = \begin{cases} dt \mid dt \in G(o) \wedge dt \neq \texttt{string} & \text{if } |G(o)| = 2, \\ null & \text{otherwise.} \end{cases}$$

However, we can have a case where an object value has `decimal` and `base64Binary` as candidate datatypes and our inference approach cannot determinate the most appropriate datatype.

Alg. 4 is a pseudo-code of a possible implementation of Rule 4. The algorithm receives the triple t :`<s,p,o>` to be analyzed. The list of candidate datatypes is reduced removing specific datatypes and keeping the most general ones (`decimal` and `base64Binary`) (line 2 in Alg. 4). If the list of candidate datatypes has only a value, the datatype is `string` (line 2 in Alg. 4), but if there are two, the datatype is the second one (line 2 in Alg. 4), since the first one is `string`. If there are more than two datatypes, the ambiguity persists and this step is not able to produce a result.

Algorithm 4: Generalization of Numeric and Binary Groups

Input: Triple $t =< s, p, o >$
Output: Datatype *dt*

```
1 DT dt_list = CDT(t);                                  //Candidate Datatypes
2 DT generalDT = get_general_datatypes(dt_list); //General datatypes from
    Numeric and Binary groups.
3 if size(generalDT) == 1 then
4     return dt = new Datatype(string);
5 if size(generalDT) == 2 then
6     return dt = second value of generalDT;
7 return dt=null;                                       // Ambiguous case.
```

Our inference approach allows to improve the datatype analysis for RDF matching/integration by complying with the identified requirements (see Section 4.2): (i) the use of local available information, as the predicate value in *Step 1* and *Step 3* and the datatype lexical space in *Step 2*, as well as external available information, such the predicate information in *Step 1* and the predicate context in *Step 3*); and (ii) this method is objective and complete for the Semantic Web, since all simple datatypes are considered, which are available in the most common Semantic Web databases as DBpedia.

Alg. 5 shows a global vision of our inference process, composed by the four steps. Each step can be applied independently according to user-preferences as we mentioned it before. However, we suggest an order starting from a general solution (*Step 1*), that can be applied to all datatypes, until a specific one for particular cases (*Step 4*). This order obtained the best results during experimentation. If the predicate information analysis (lines 1 to 3 in Alg. 5) cannot infer the datatype, then datatype lexical space inference is used (lines 4 to 7 in Alg. 5). The semantic analysis is processed if we obtain a datatype *null* from previous inference (lines 8 to 11 in Alg. 5). The last step is applied if once again we obtain a datatype *null* from previous inference (lines 12 to 13 in Alg. 5).

In the following section, we analyze the complexity of our datatype inference process in order to measure the applicability for real cases.

4.4 Complexity Analysis

A complexity analysis of our inference approach indicates a linear order performance in terms of the number of triples ($O(n)$).

Algorithm 5: Datatype Inference Process

```
Input: Triple t =< s, p, o >, float h
Output: Datatype dt
1  dt = predicate_information_analysis(t);                    //Step 1
2  if dt is not null then
3  |  return dt;
4  else
5  |  dt = datatype_lexical_space_analysis(t);                //Step 2
6  |  if dt is not null then
7  |  |  return dt;
8  |  else
9  |  |  dt = predicate_semantic_analysis(t,h);              //Step 3
10 |  |  if dt is not null then
11 |  |  |  return dt;
12 dt = generalization(t);                                    //Step 4
13 return dt;
```

- For Step 1, the predicate information of each triple is extracted to search the `rdfs:range` property, since the number of properties associated to the predicate of each triple (Def. 20) is constant, then its execution order is $O(n)$.
- In the case of Step 2, for each triple a pattern-matching is executed for all simple datatypes (finite number of execution) thus, it is of linear order ($O(n)$).
- In Step 3, for each triple, its set of contexts is extracted to determine the best related work (in a constant time), thus its time complexity is also $O(n)$.
- Finally, Step 4 reduces the finite set of candidate datatypes (generalization) in a linear order ($O(n)$).

As the four steps are executed sequentially, the whole inference datatype process exhibits a linear order complexity, $O(n)$. The following section evaluates the accuracy and demonstrate the linear order performance of our proposal.

4.5 Experimental Evaluation

To evaluate and validate our inference approach, an online prototype system, called ***RDF2rRDF***[13], was developed using PHP and Java. Fig. 4.6 shows the graphic user

[13] http://rdf2rrdf.sigappfr.org/

interface of the prototype, where the inference steps can be selected according to user preferences.

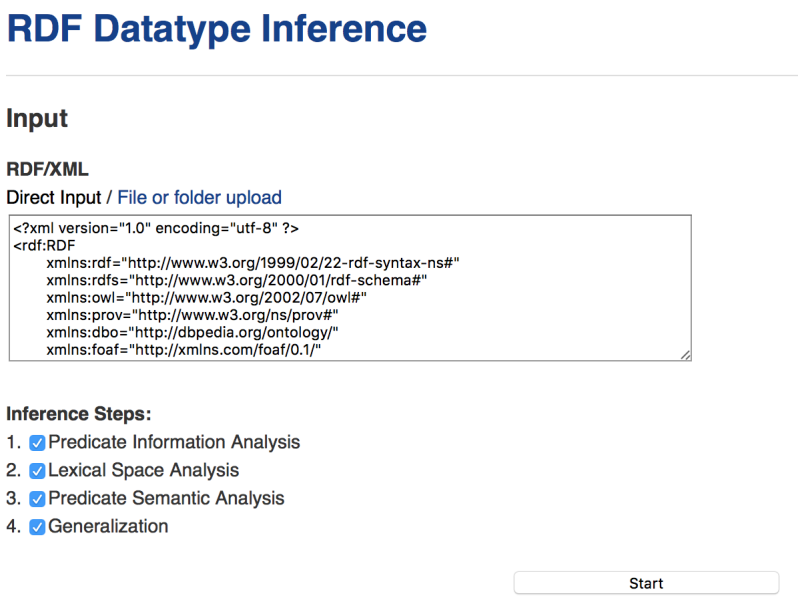


Figure 4.6: Graphic User Interface of our Inference Approach

For *Step 3*, we implemented our assumptions of contexts using the semantic similarity service *UMBC: Semantic Similarity Service Computing*, which is based on distributional similarity and Latent Semantic Analysis (Def. 27). UMBC service is available online and an API is provided[14]. Also, we used Wordnet[15] to recognize if a word is *plural* assuming that every word has a root lemma where the default plurality is singular. Additionally, we assume that a word is a *condition* if it has the prefix *"is"* or *"has"*. All these assumptions compose our knowledge base.

Table 4.4 shows the different datatypes available in several semantic web databases. Note that DBpedia has more variety of datatypes compared with the others, thus our experiments were made with DBpedia database.

Table 4.4: Semantic Web databases

DataBase	**Datatypes**
DBpedia	`integer`, `gYear`, `date`, `gMonthDay`, `float`, `nonNegative`, `double`, `Integer` and `decimal`
Wordnet	`string`
GeoLinked data	point (complex datatype)

Experiments were undertaken on a MacBook Pro, 2.2 GHz Intel Core(TM) i7 with

[14]http://swoogle.umbc.edu/SimService/api.html
[15]WordNet is a large lexical database of English (nouns, verbs, adjectives, etc.)

16.00GB, running a MacOS Sierra and using a Sun JDK 1.7 programming environment.

Our prototype was used to perform a large battery of experiments to evaluate the accuracy and the performance (execution time) of our approach in comparison with the related work. To do so, we considered two datasets:

- **Case 1**: 5603 RDF documents gathered from *DBpedia person data*[16], in which 1059822 triples, 38292 literal objects, and 8 different datatypes are available.
- **Case 2**: the whole *DBpedia person data* as a unique RDF document with 16842176 triples, in which only datatypes `date`, `gMonthDay`, and `gYear` are presented.

For **Case 1**, we evaluated the accuracy and performance of each step of our datatype inference approach, *Step 1 + Step 2*, *Step 1 + Step 3*, *Step 2 + Step 3*, and the whole inference process. The order of the whole inference process was established starting from a general solution (*Step 1*), that can be applied to all simple datatypes, until a specific solution for particular cases (*Step 3* and *Step 4*). In **Case 2,** we only evaluated the whole inference process, since it is mainly used for performance because the high number of triples.

4.5.1 Accuracy evaluation

To evaluate the accuracy of our approach, we calculated the F-score, based on the Recall (R) and Precision (PR). These criteria are commonly adopted in information retrieval and are calculated as follows:

$$\textbf{PR} = \frac{A}{A+B} \in [0,1] \qquad \textbf{R} = \frac{A}{A+C} \in [0,1] \qquad \textbf{F-score} = \frac{2 \times PR \times R}{PR+R} \in [0,1]$$

where A is the number of correctly inferred datatypes; B is the number of wrongly inferred datatypes; C is the number of correct datatypes not inferred by our inference approach (datatypes that should be inferred but were not because of ambiguity).

Test 1: In Table 5.12, for *Step 1*, 24059 datatypes were inferred (45.35% of the total, 38292) with a Precision, Recall, and F-score of 99.89%, 62.81%, and 77.12% respectively. This process inferred 26 invalid simple datatypes because inconsistencies on the data. In *Step 2*, 17435 datatypes were inferred (45.35% of the total) with a Precision, Recall, and

[16]Information about persons extracted from the English and Germany Wikipedia, represented by the FOAF vocabulary - http://wiki.dbpedia.org/Downloads2015-10

F-score of 96.91%, 44.76%, and 61.24% respectively. This process inferred 537 invalid datatypes (14 simple and 523 complex datatypes) and it could not determine the datatype for 20857 literal objects. Combining *Step 1* and *Step 2*, the Precision, Recall and F-score values increased considerably (99.17%, 88.85%, and 93.73% respectively). In *Step 3*, only 2480 datatypes were inferred (Recall 6.85%), since it is proposed for particular cases (context rules). Precision in *Step 4* is less than all other Steps; however, the Recall is greater than *Step 2* and it makes a F-score similar to *Step 2*. Other combinations as *Step 1* and *Step 3* and *Step 2* and *Step 3* have high Precision but low Recall, because the Recall of *Step 3* (specific cases). We can noted that the combination of *Step 2* and *Step 4* has the same Precision and Recall as the ones of *Step 4*. According to the definition of *Step 4*, it uses the datatype candidates in order to keep the most general datatypes. The candidates are obtained by a lexical-space matching process, which is the *Step 2*. The same situation is noted between the results of *Step 1*, *Step 4*, and *Step 1*, *Step 2*, *Step 4*.

Table 4.5: Accuracy Evaluation

Inference Process	**Accuracy Evaluation**					
	Valid	**Invalid**	**Ambiguity**	**Precision**	**Recall**	**F-score**
Case 1: Step 1	24033	26	14233	99.89%	62.81%	77.12%
Case 1: Step 2	16898	537	20857	96.92%	44.76%	61.24%
Case 1: Step 3	2480	119	35812	95.20%	6.85%	11.62%
Case 1: Step 4	16899	1962	19431	89.60%	46.52%	61.24%
Case 1: Step 1 + Step 2	33771	281	4240	99.17%	88.85%	93.73%
Case 1: Step 1 + Step 3	26394	145	11753	99.45%	69.19%	81.61%
Case 1: Step 2 + Step 3	19259	656	18377	96.71%	51.17%	66.93%
Case 1: Step 1 + Step 4	33772	999	3521	97.13%	90.56%	93.73%
Case 1: Step 2 + Step 4	16899	1962	19431	89.60%	46.52%	61.24%
Case 1: Step 1 + Step 2 + Step 4	33772	999	3521	97.13%	90.56%	93.73%
Case 1: Whole Approach	36132	551	1609	97.71%	96.50%	97.10%
Case 2: Whole Approach	2250402	710234	0	76.01%	100.00%	86.37%

Executing the whole approach, 37066 datatypes were inferred (96.80%). The Precision, Recall and F-score are 97.71%, 96.50%, and 97.10% respectively.

The best F-score was obtained with the whole inference process; however, the Precision decreased from 99.89% (*Step 1*) to 97.71% because of *Step 3* and *Step 4* (Precision 95.20% and 89.60% respectively). Table 4.6 shows the Precision, Recall, and F-score for each datatype available in the Case 1. In this table, the datatype `date` was not correct inferred 7 times; however, its *lexical space* is unique according to the W3C recommendation; regarding the data, these 7 cases have the format YY-MM-DD instead of CCYY-MM-DD

(inconsistencies of the data).

Table 4.6: A detailed Inference per Datatype (Case 1) - Whole Approach

Datatype	Valid	Not - Valid	Ambiguity	Precision	Recall	Case 1: F-score
integer	13567	424	1311	96.37%	91.72%	93.99%
gYear	5067	1	0	99.98%	100%	99.99%
date	16446	7	0	99.91%	100%	99.98%
gMonthDay	459	0	0	100%	100%	100%
float	0	142	0	0%	NaN	NaN
double	266	1	0	100%	99.63%	99.81%
nonNegativeInteger	77	0	0	100%	100%	100%
decimal	0	0	1	NaN	0%	NaN
Complex	250	273	0	47.80%	100%	64.68%
Total	**36132**	**934**	**1226**	**97.71%**	**96.50%**	**97.10%**

For Case 2, the Precision decreased to 76.01%. It is caused by the noise and inconsistencies of the DBpedia datasets [PHHD10] (e.g., `dbo:deathDate` should have the datatype property `date`, but in the queried datasets, it was set as `gYear`).

Test 2: We also evaluated the accuracy of our approach in comparison with alternative methods and tools, namely Xstruct, XMLgrid, FreeFormated, and XMLMicrosoft [XML10, fre11, HNW06, Mic]. Since these works infer datatypes in XML documents, we transformed all literal nodes to XML format by using the value and its relation. Table 4.7 shows the accuracy results obtained for Case 1. Note that our approach has the best Precision and F-score. Our Recall is less than the other ones because we consider a bigger number of datatypes and thus, there are more ambiguous cases (*lexical space* intersections).

Table 4.7: Accuracy Comparison with the Related Work for Case 1

Work	Precision	Recall	F-score
Xstruct	83.28%	100%	90.88%
XMLgrid	83.61%	100%	91.07%
FreeFormated	43.32%	100%	60.45%
XMLMicrosft	43.23%	100%	60.36%
RDF2rRDF	**97.71%**	**96.50%**	**97.10%**

Test 3: For Case 1, we performed an extra experiment to measure the behavior of our inference approach when a partial number of datatypes is missed (25%, 50% and 75%). Table 4.8 shows the results obtained for this experiment. Precision, Recall and F-score were measured with respect to the number of missed datatypes. Since each document has at most two same predicates, the results have not increased significantly. However, in a scenario where a huge number of same predicates are presented, the known datatype of a literal node is added to all the literal nodes associated to its predicate, then a better and

easy inference is performed.

Table 4.8: Availability of datatypes for Case 1

Availability of Datatypes	Precision	Recall	F-score
0%	97.71%	96.50%	97.10%
25%	97.78%	96.47%	97.12%
50%	97.66%	96.66%	97.16%
75%	97.64%	96.91%	97.27%

4.5.2 Performance evaluation

To evaluate the performance, we measured the average time of 10 executions for each test. Table 4.9 shows the results obtained in our performance evaluation.

Test 4: In Case 1, the execution time of *Step 1* was greater than *Step 2*, because the use of external calls increased the execution time. However, the execution time of *Step 1 + Step 2* was similar to *Step 1*, since *Step 1* works as a filter of triples and leaves less analysis for *Step 2*. *Step 3* has the greatest execution time, since it depends of an external service. *Step 4* depends of the list of candidate datatypes; thus, its execution time should be greater than *Step 2* because the use of extra operations to reduce the set of datatypes (generalization).

Table 4.9: Performance Evaluation

Inference Process	Performance Evaluation	
	Execution Time	Cache Building Time
Case 1: Step 1	31.336s	11.582s
Case 1: Step 2	15.939s	15.939s
Case 1: Step 3	243.826s	40.764s
Case 1: Step 4	17.879s	17.879s
Case 1: Step 1 + Step 2	33.216s	13.966s
Case 1: Whole Approach	53.247s	14.236s
Case 2: Whole Approach	-	59.282s

Test 5: Additionally, we implemented in *Step 1* and *Step 3* the use of cache to store predicate information and predicate contexts, respectively (see Table 4.9 - column 3). This cache is reused for consequential analysis of triples, since same predicates are available in different triples. For Case 1, the use of cache in Step 1 reduced the execution time in more than 65% and made the execution time of *Step 1 + Step 2* less than *Step 1* and *Step 2*, separately. The cache in the whole inference approach represented more than 70% of improvement in the performance and an average of 157×10^{-7}sec. per triple. Moreover,

for more than 16 million of triples (Case 2), the execution time remained in the order of seconds (59.28s) and the average execution time per tripe was reduced to 35×10^{-7}sec. We presume in Case 2 that the majority of triples were inferred in *Step 1*, which uses cache.

Fig. 4.7 shows the execution time with respect to the number of triples. The performance obtained confirms the linearity of our inference approach. Note that the use of cache makes the function stable for high number of triples because of the finite number of predicates available in the DBpedia database.

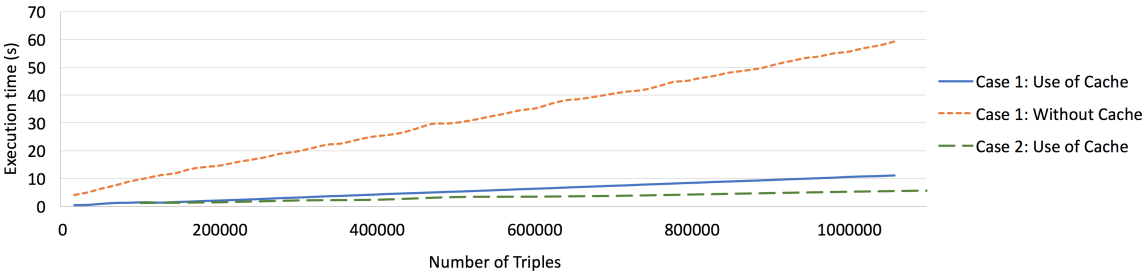


Figure 4.7: Execution Time of our Inference Approach

4.6 Summary

In this chapter, we investigated the issue of datatype inference for RDF documents matching/integration. We proposed an approach, consisting of four steps: the analysis of the predicate information associated to the object value, analysis of the lexical space of the value itself, semantic analysis of the predicate name, and generalization of datatypes.

Each step in the inference process can infer different datatypes, for example, a number (e.g., 12) is consider as datatype `Decimal` for our Step 4, while for Step 1 is `int`. We recommend the use of Step 1, as well as, Step 2, since both inference steps have a high Precision.

We evaluated the accuracy and performance of our inference process with DBpedia datasets (*DBpedia person data*). Results show that the inference approach increases the F-score up to 97.10% (accuracy) and it does not incur in high execution time (performance).

Extending this work to include other datatypes as complex ones and propose more context rules to resolve extra ambiguity, are some of our next steps.

Chapter 5

The Semantic Web: Sensitive Data Preservation

> "Obviously, everyone wants to be successful, but I want to be looked back on as very innovative, very trusted and ethical and ultimately making a big difference in the world."
>
> — Sergey Brin

As we mentioned in Chapter 1, the Semantic Web and the Linked Open Data (LOD) initiatives promote the integration and combination of RDF data on the Web [SH01]. RDF data is gathered and published by different sources (e.g., companies, governments) for many purposes such as statistics, testing, and research proposals. However, as more data is available, sensitive information (e.g., diseases) could be sometimes provided or inferred compromising the privacy of related entities (e.g., patients).

For the authors in [RGCGP15], anonymization is one common and widely adopted technique for sensitive data protection that has been successfully applied in practice. It consists on protecting the entities of interest by removing or modifying identifiable information to make them anonymous before publication, while keeping the utility of the data. To apply anonymization, it is necessary to classify the data into: (i) *main entities*, which are the entities of interest, and (ii) *related data* that compromise the main entities. The related data is also classified as: (i) *Identifiers*, data that identify a main entity (e.g., security social number); (ii) *Quasi-identifiers*, data that can be used to link with other data to identify a main entity (e.g., birthday, postal code); (iii) *Sensitive information*, data which compromises a main entity (e.g., diseases); and (iv) *Unsensitive information*

that does not have a particular role. A classification, which is performed by an expert user who knows previously the data and is responsible of protecting the model, is based on predefined assumptions about how an adversary can take advantage over this data. These assumptions are called *Background Knowledge*. Due to the huge complexity of the RDF structure, a classification requires a high interaction of the expert user. Moreover, all RDF's elements can be considered as main entities, and they can also be classified into identifiers, quasi-identifiers, sensitive information, etc., making the RDF protection complex.

Thus, in the context of RDF data, several limitations are identified: (i) RDF anonymization techniques are limited and designed for a particular and ideal scenario, which is inappropriate when having several linked heterogeneous datasets [Aro13, RGCGP15, HHD17, SLB+17]; (ii) the non-consideration of IRIs as external and reachable resources makes the current RDF solutions unsuitable for protection on the Web, since other available resources could link or infer sensitive information; (iii) the presence and consideration of resources (IRIs and Blank nodes), which are a fundamental part of the RDF data, makes the database oriented methods [NJ02, MGKV06, LLV07, MJK09, SO14] unsustainable for a large quantity of resources due to the number of JOIN functions needed to satisfy the existing normalized models; (iv) graph anonymization techniques assume simple, undirected and unlabeled graphs [BDK07, HMJ+08, ZP08, LT08, YW08, LWLZ08, CKR+11, YCYY13], which are inappropriate for the Semantic Web, since properties and semantic relations among resources would be ignored; and (v) the complexity of the RDF structure requires a high interaction of the expert user to classify the RDF's elements to be protected, and their related data.

To overcome these limitations, we propose a framework, called *Ri**A**iR* (Reduction, Intersection, and Anonymization in RDF), which is independent of serialization formats and providers. The proposal is designed for RDF documents, considering all their elements and a scenario of a huge quantity of information. The complexity of the RDF structure is reduced to make possible the task of classification and to suggest potential disclosure sources to the expert user, decreasing his interaction. By a generalization method, we reduce the connections among datasets to protect the data and to preserve the objectives of the Semantic Web (integration and combination).

The chapter is organized as follows. Section 5.1 presents a motivating scenario to illustrate the disclosure of sensitive information on the Web. Section 5.2 surveys the related literature. The main problematic of this study is formalized in Section 5.3. Section 5.4 describes our approach. Section 5.5 shows the experiments to evaluate the viability and

performance of our approach. Finally, we present our conclusions in Section 5.6.

5.1 Motivating Scenario

The goal of the Semantic Web is to publish datasets, mainly as RDF, describing and combining resources on the Web for an open access. The datasets are usually treated and protected before being published; however, sensitive information could be deduced using related information available from other datasets. To illustrate this, let's consider a scenario in which a data manager X works for a government to publish a *dataset A*, related to energy production and its applications, on the Web[1].

An extract of the *dataset A* to be published is shown in Table 5.1.

Table 5.1: An example of the data extracted from *Enipedia* dataset

N°	**cat:Fuel_-type**	**cat:radio-active**	**rdfs:label**	**prop:City (rdfs:label)**	**prop:Country (rdfs:label)**	**prop:Lat.**	**prop:Long.**
1	art:Nuclear	true	Hartlepool	Hartlepool Cleveland	United Kingdom	54.6824	-1.2166
2	art:Nuclear	true	Limerick	Pottstown	United States	40.2257	-75.5866
3	art:Nuclear	true	Neckar	Neckarwestheim	Germany	49.0411	9.1780
4	art:Nuclear	true	Beaver Valley	Shippingport	United States	40.6219	-80.4336

Figure 5.1 shows the schema of the *dataset A* to be published. Note that the properties `prop:Latitude`, `prop:Longitude`, `rdfs:label`, and `cat:radioactive` define values, while the properties `prop:City`, `prop:Country`, and `cat:Fuel_type` define resources.

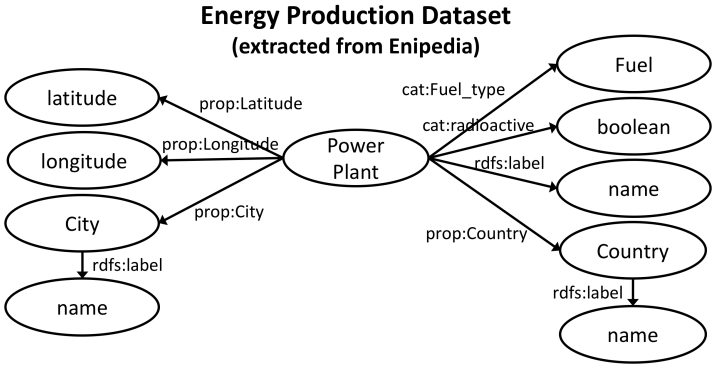


Figure 5.1: Structure of the data extracted of the Enipedia dataset

As a data manager, X should pay attention about the side effect of publishing the *dataset A* on the Web, since it can produce sensitive information for entities already

[1]The example provided uses an extract from *Enipedia* dataset.

published. For instance, *DBpedia*[2], which is a linked open dataset extracted from Wikipedia, can be used as background knowledge in order to discover sensitive information related to places of interest. This dataset can be easily connected by the use of properties, such as `prop:Latitude` and `prop:Longitude` present in the *dataset A* as well. Table 5.2 shows some places of interest available in the *DBpedia* dataset.

Table 5.2: Some places of interest available in the *DBpedia* dataset

Nº	**rdf:type**	**rdfs:label**	**prop:Lat.**	**prop:Long.**
1	dbo:School	Hartlepool College of Further Education	54.6839	-1.2109
2	dbo:School	English Martyrs School and Sixth Form College	54.6754	-1.2362
3	dbo:School	Coventry Christian Schools	40.2505	-75.5930
4	dbo:School	Hölderlin-Gymnasium Lauffen am Neckar	49.0704	9.1394
5	dbo:School	Pennsylvania Cyber Charter School	40.6385	-80.4549

By the intersection among coordinates (`prop:Latitude` and `prop:Longitude`) of nuclear power plants (*dataset A*) and the ones of places of interest (*dataset DBpedia*), one can easily identify their proximity in a defined `Region`. A `Region` is an area obtained by the maximum distance between a nuclear power plant and a place of interest. The following SPARQL query produces the intersection between the *dataset A* to be published and the *dataset DBpedia*. Note that a `Region` of 100km was used to obtain the results.

```
SELECT DISTINCT
?Place ?g bif:st_distance(?g,bif:st_point(".$long.",".$lat."))
AS ?distance
FROM
<http://dbpedia.org> WHERE {?p rdfs:label ?Place ;
geo:geometry ?g ; rdf:type dbo:School .
FILTER
(bif:st_intersects (?g, bif:st_point (".$long.", ".$lat."), 100)
&& (lang(?Place) = \"en\"))}
ORDER BY ASC(?distance)
```

Table 5.3 is the result of the intersection between the *dataset A* and *dataset DBpedia*. It shows in row 1 that a *school* is less than 500 meters distance from a nuclear power plant in United Kingdom. It also shows which hospitals, universities, and any other crowded places are close to power nuclear plants in a defined area. One can even identify which are the dirtiest power nuclear plants (`prop:Carbonemissions`) and the places next to them. If this combined information is available on the Web, it can be misused against the nuclear power plants to stop their production and management, and even against the places of interest near to them.

[2]DBpedia does not contain sensitive information, since all data correspond mainly to well-known entities (e.g., places, governments, actors, singers).

Table 5.3: Some places of interest next to Nuclear Power Plants

Nuclear PowerPlant	City	Country	School	Distance (Km)
Hartlepool	Hartlepool Cleveland	United Kingdom	Hartlepool College of Further Education	0.40244
Hartlepool	Hartlepool Cleveland	United Kingdom	English Martyrs School and Sixth Form College	1.48812
Beaver Valley	Shippingport	United States	Pennsylvania Cyber Charter School	2.5761
Limerick	Pottstown	United States	Coventry Christian Schools	2.81988
Neckar	Neckarwestheim	Germany	Hölderlin-Gymnasium Lauffen am Neckar	4.2998

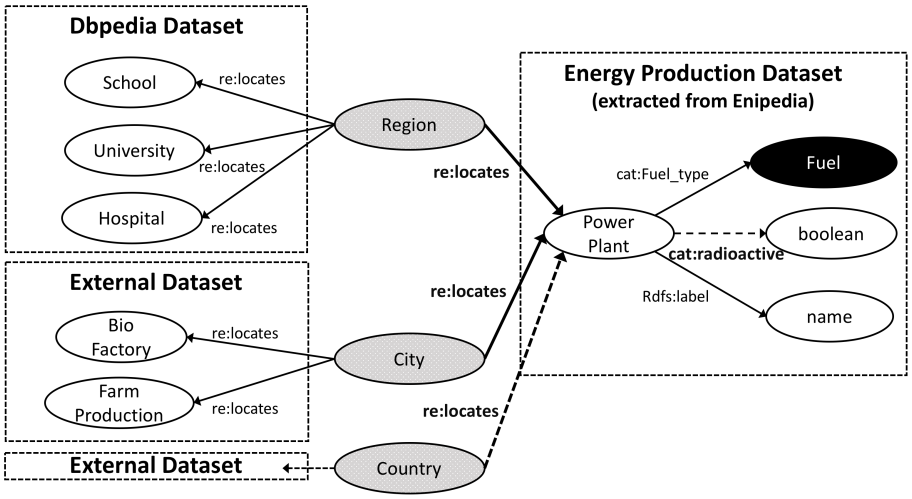


Figure 5.2: Intersection between *Energy Production* dataset and other datasets

Figure 5.2 illustrates graphically the intersection between *dataset DBpedia* and the *dataset A*. The resource `Region` links `School`, `University`, `Hospital` and `Power Plant` resources.

To protect the *dataset A* to be published, *X* needs to identify and classify the data, according to the assumptions of how an adversary can obtain or produce sensitive information, using the background knowledge, as follows. *The information of a* `Power Plant` *resource of type nuclear (*`art:Nuclear`*) is sensitive, if there is at least a place of interest (e.g.,* `School`*) in a defined* `Region`[3].

- *Keys: (Identifiers/Quasi-Identifiers)*: Properties `prop:Longitude` and `prop:Latitude` are keys since both values indicate the position of a `Power Plant`, which belongs to a defined `Region`.

- *Sensitive Information*: An resource (`dbo:School`) and its properties are sensitive

[3]Considering only *DBpedia* dataset as external related information (background knowledge).

information, since it defines the place of interest.

- *Unsensitive Information*: Other values and properties, which are not considered in the previous types, are unsensitive information.

Once X has established the classification, a protection technique based on this classification, should be used to protect the disclosure of sensitive information. Thus, the following challenges are defined in this study. Thus, the following challenges are defined in this study:

- Provide an easy classification of the RDF data (keys, sensitive information and unsensitive information);
- A similarity measure able to evaluate the intersection between the data to be published and the background knowledge, to suggest disclosure sources; and
- Select the most appropriate protection taking into account the complexity of the RDF data and the objectives of the Semantic Web.

Our contribution in this study is as follows:

- A general framework designed for RDF documents, independent of the serialization formats, in a scenario where linked and heterogeneous resources are presented; i.e., the Web;
    1. A method to reduce the complexity of the RDF structure of the data to be published, simplifying the task of analysis, performed by the expert user;
    2. A method to suggest disclosure sources to the expert user, based on node similarity, reducing the task of data classification; and
    3. An anonymization operation, based on a generalization method, to decrease the relations among resources from different datasets, to preserve the main objectives of integration and combination of the Semantic Web.

The following section presents the related work of RDF anonymization.

5.2 Related Work

In this work, we focus on anonymization thecniques as a solution to protect the sensitive information since it has been widely adopted for sensitive data protection.

To the best of our knowledge, works on RDF document anonymization are limited [RKKT13, RKKT14a, RKKT14b, RKKT14c, RGCGP15, SLB+17, HHD17]; however, due to the particularity of the RDF data, other domains where anonymization has been extensively studied could be applied, such as: databases [MGKV06, LLV07, MJK09, SO14, YLZY13, GLLW17] and graphs [BDK07, CKR+11, HMJ+08, ZP08, LT08, YW08, CT08]. To evaluate and classify the existing works, we identified the following criteria of comparison according to the challenges and objectives of this work:

1. *The complexity of the data*, which should be aligned with the one of RDF structure, considering heterogeneous nodes and relations, increasing the expressibility and difficulty of the representation;

2. *The type of classification method* for identifiers, quasi-identifiers, sensitive and unsensitive information due to the high quantity of entities, properties and values available on the Web, making difficult the task of the expert user; and

3. *The conditions of anonymity* that are proposed in the current proposals to identify the most appropriate ones for the Semantic Web.

Following sections describe the RDF, databases, and graph approaches in the context of anonymization.

5.2.1 RDF Document Anonymization

For RDF documents, the authors in [RGCGP15] provide an overview of RDF's elements over the role in anonymization (e.g., explicit identifiers, quasi-identifiers, sensitive data). They propose a framework to anonymize RDF documents, which satisfies the k-anonymity condition. They consider the use of taxonomies for values and relations (each type of value and relation has its own taxonomy). Generalization and suppression operations are applied over these taxonomies to anonymize the RDF document. Once the operations are applied, several anonymous RDF documents are produced by the use of all value combinations from the taxonomies. A measure for anonymous solutions that satisfied the k-anonymity condition, is proposed to select the best option. In [HHD17], the authors extend the previous work defining an area (neighborhood), where the k-anonymity condition is satisfied. The exhaustive method to select the best option makes these approaches unsuitable for complex cases, since a greater quantity of values to take into account, needs a more elaborate anonymization process (more possible solutions). Moreover, the authors

assume a classification of the data provided by the model and they do not specify how this classification was performed.

Additionally, there are some works on the context of statistical queries [Aro13, SLB+17] based on grouping operators (e.g.,SUM, AVG,MAX) and others based on expert-defined sanitization queries [RKKT13, RKKT14c, RKKT14b, RKKT14a] to remove identifiers, but we only focus on the protection of RDF documents.

5.2.2 Database Anonymization

In some cases when one has small RDF data, a common practice can be to convert the RDF to a structured dataset as tables to reuse existing techniques. Anonymization in databases has been extensively studied and many works are available in the literature. One of the most used work is proposed in [Sam01], the authors define a condition, called *k-anonymity*, where an entity cannot be identified, since there are at least $k-1$ other similar entities. However, the problem of satisfying the k-anonymity condition is NP-hard, producing different studies where the complexity and an efficient solution are addressed. For instance, to anonymize the data, the authors in [NJ02] apply techniques based on neural networks, the authors in [AS16] apply genetic algorithms, while in [NJ02] the authors use matching learning. Non-perturbative operations, such as generalization and suppression methods, where data is modified according to certain criteria of the existing values (e.g., taxonomies, ranges), are mainly used to satisfy the k-anonymity condition [AW16]. Other studies use *perturbative* operations, such as Micro-aggregation/clustering methods, where the entity values are replaced or modified by the centroid of the clusters, adding in some cases new entities to satisfy conditions of anonymization in each cluster [SYLL12, ZZYY14].

According to [MGKV06], k-anonymity condition does not protect the sensitive values, since k similar entities can have the same sensitive information, which is the one required by the adversary. For that, the authors in [MGKV06] extend the k-anonymity condition considering a diversity (l) of sensitive values for each set of similar entities (l-diversity). However, the disclosure is still possible due to the attribute distribution of the dataset. The authors in [LLV07] propose a condition where the distribution of each sensitive attribute should be close/similar to the whole attribute distribution in the dataset (t-closeness). Other studies extend the previous mentioned conditions to address particular assumptions of the background knowledge. The authors in [MJK09] propose a (k,T)-anonymization model over spatial and temporal dimensions. Other works apply the conditions of anonymity to different values as the authors [SO14] do, where l-diversity con-

dition is satisfied by the sensitive information as well. The l-diversity condition is extended in the clustering proposal work [YLZY13], defining a (k, l, θ)-diversity model, which takes into account the cluster size, the distinct sensitive attribute values, and the privacy preserving degree of the model. An improvement of certain conditions is made for special scenarios; for instance, the authors in [HYY08] divide numerical sensitive values into several levels, getting a better protection for numerical values. Also, properties of the data such as utility, value distribution, etc., are considered to propose anonymization models. The work in [GLLW17] takes into account the association between quasi-identifiers and the sensitive information as a criterion to control the use of generalization hierarchy. Some semantic features are added in recent works. The authors in [OTSO17] provide a (l, d)-semantic diversity model based on a clustering method. They analyze the distance among sensitive values (d) to consider more actual diversity. According to [SOTO17], a value can be quasi-identifier and sensitive information at the same time, proposing a method that can treat "sensitive quasi-identifier" and satisfying the conditions of l-diversity and t-closeness.

Differential Privacy as k-anonymity is another well-used technique to provide privacy. The authors in [DMNS06] propose a perturbation method for true answer of a database query by the addition of a small amount of distributed random noise. This method is extended by other authors as in [HRMS10], where they improve the accuracy of a general class of histogram queries while satisfying differential privacy. The work in [MCFY11] is a non-interactive setting model, generalizing probabilistically the raw data and adding noise to guarantee differential privacy. Other studies are focused on the privacy of anonymized datasets, since a dataset, in the context of databases, can be affected by updating and removing operations, which can expose the sensitive information. The authors in [SI12] propose an architecture which protects the main entities for databases that require removing operations frequently. They apply generalization operations based on hierarchies (non-perturbative method). The model satisfies k-anonymity condition; however, the architecture needs to verify each new deleting request the anonymized data in order to protect the privacy of the original datasets. A centralized scenario is required to apply this proposal.

Work on database anonymization approaches that satisfy k-anonymity and its variations, assume that the classification of data into identifiers, quasi-identifiers, sensitive and un-sensitive information is provided by an expert user, who knows the data, focusing mainly on the method to satisfy the conditions of anonymization. In the Semantic Web, it is unable to understand the detailed characteristics of external datasets and assume all the background knowledge possessed by adversaries. Moreover, as more information is

involve, more complex is the task of converting the RDF data to a structured normalized model, since a high granularity (many tables) is produced due to the use of IRIs, acting as foreign-keys.

Following section describes the works related to graph anonymization.

5.2.3 Graph Anonymization

RDF data can be represented as a graph structure, having labeled-nodes, and directed and labeled-edges. In the literature, there are several works in the context of social media, where the authors assume a simple network as undirected, node-unlabeled and edge-unlabeled structure [LT08, YW08, CT08] (see Group 9 in Table 5.4). These works focus on the privacy through the number of edges among nodes, since an adversary can have the information about the relations, which can be the only one with a particular number (k-degree condition). The work in [CT08] proposes a greedy algorithm to satisfy the k-degree by partitioning all nodes to n clusters. Each cluster becomes uniform with respect to the quasi-identifier attributes and the quasi-identifier relationship (generalization). To choose the best n values, two criteria are taken into account: (i) each cluster has to contain at least k nodes and (ii) minimize the information loss of the data. The authors in [LT08] propose an algorithm to satisfy the k-anonymity condition over the number of edges of each node. They also rename the k-anonymity as k-degree condition. The proposal consists in two steps: (i) Degree Anonymization, where a degree sequence of the graph (descending order) is generated to group similar nodes with the same degree and (ii) Graph construction, where an algorithm decides among which nodes a new edge is added according to satisfy the k-degree condition. In [YW08], the authors anonymize a graph by adding random edges. They provide an analysis on the spectrum of the graph to measure the impact of the anonymization solution. The spectrum is directly related to the topological properties such as diameter, presence of cohesive clusters, long paths and bottlenecks, and randomness of the graph. Works in this group only take into account the number of relations as a condition of anonymity (k-degree), but in a scenario where a diversity of nodes is present, the number of operations to satisfy the k-degree condition increases exponentially. Moreover, diversity of edges values is not analyzed and the authors assume that the classification of the data is provided by the expert user.

Other works manage more complex graphs by assuming labeled-node structure as in [BDK07, CKR$^+$11, HMJ$^+$08, ZP08] (see Group 10 in Table 5.4). The authors in [BDK07] demonstrate assuming several attacks that removing identifiers and renaming

the nodes in an arbitrary manner, from a social graph, is an ineffective anonymization mechanism. Walk-based attacks are able to compromise the privacy for modest numbers of node (around 90%); thus, it has been proven for the authors that removing identifiers of the data is not a well protection. The authors in [CKR+11] assume that the adversary knows only degree-based information, which is the number of relations (edges) that has each node. To anonymize the graphs, they add new nodes instead of edges, since they affirm that *"introducing new nodes does not necessarily have an adverse effect. To the contrary, adding new nodes with similar properties could better preserve aggregate measures than will distorting the existing nodes"*. To satisfy the k-anonymity condition, an algorithm following four steps is provided: (i) Optimally partition degree sequence (descending order), (ii) Augment graph with new dummy nodes, (iii) Connect original graph nodes to new dummy nodes, and (iv) Insert inter-dummy-node edges to anonymize dummies. In [HMJ+08], the authors propose an anonymization technique that protects against re-identification by generalizing the input graph. They generalize the graph by grouping nodes into partitions, and then publishing the number of nodes in each partition, along with the density of edges that exist within an across partitions. To preserve the privacy of individuals, which are represented as nodes in a social network, the authors in [ZP08] assume that an adversary may have the background knowledge about the neighborhood of some target individuals. Two properties are taking into account: (i) node degree in power law distribution [FFF99] and (ii) small-world phenomenon [WF94] to ensure a low loss of data. They greedily organize nodes into groups and anonymize the neighborhoods of nodes in the same group to satisfy the k-anonymity condition.

Works in this group have the same drawbacks as the previous one, which are related to the modeling of social graphs as a simple structure (even if the graph is node-labeled), and the assumption of the classification, which is provided by the expert user.

The authors in [LWLZ08] work also on the context of social networks by ensuring the privacy of main entities, which are the nodes in the graph (see Group 11 in Table 5.4). They consider a weight over edges, since it can represent affinity among two nodes, frequency among two persons, or similarity between two organizations. They propose a Gaussian Randomization Multiplication strategy due to its simple implementation in practice and responds to the dynamic-evolution nature of social networks, since it is very hard and costly to collect the information in advance in a huge and dynamic scenario. This work represents in a better way the scenario present in the Semantic Web. However, edges-labeled are reduced to values and they are not considered as reachable resources which can be used to disclosure the sensitive information. Also, this work assumes that the classification of the data is provided by an expert user.

Another interesting work is presented in [YCYY13] (see Group 12 in Table 5.4), the authors assume a more complex graph than the previous described groups. In fact, in addition of the node degree, they also assume the values of the nodes as sensitive data. They propose a framework, which satisfies k-anonymity and l-diversity conditions. They generate a sequence of 3-tuples (id, node-degree, and its respective sensitive value). A grouping algorithm is applied over the list to group similar triples, following certain criteria to satisfy the conditions of anonymization (k-anonymity and l-diversity). The sequence is called *KDLD sequence*, when all the defined conditions are satisfied. From the KDLD sequence, the graph is rebuilt. Then, they propose a graph construction technique adding nodes to preserve utilities of the original graph. Two key properties are considered: (i) Add as few noise edges as possible; (ii) Change the distance between nodes as less as possible.

In general, graph anonymization approaches assume a simple structure of the data as an undirected and unlabeled-edge social media graph. Also, k-degree is a one of the common conditions of anonymity used for the authors; however, considering a diversity of nodes as in RDF and using the existing solutions to satisfy the k-degree condition, the complexity increases considerably.

The following section summarizes and discusses the works related to anonymization.

5.2.4 Summary and Discussion

Existing techniques in the context of RDF document anonymization are really limited. In [RGCGP15, HHD17], the authors reduce the complexity of RDF structure to micro-data, where a huge quantity of information such as heterogeneous nodes and relations is simplified and anonymized. However, in a scenario where thousands of heterogeneous resources are present, the current solutions are not appropriate due to the greedy algorithm to generate all possible solutions (anonymous RDF) and then, their measure to evaluate and select the most adequate one.

Since RDF data can be converted, in some cases, to a structured data as databases, database anonymization techniques could be also applied. Small RDF data can be managed by these solutions; however, reducing the complexity of big RDF data into structured models can produce a high semantic information loss (properties), and a huge granularity of the structured normalized-model. Moreover, solutions are proposed for simple cases where data satisfy conditions of anonymity, but when a diversity of values is present, the complexity of the solutions increases exponentially.

Table 5.4: Related Work Classification

G	Work	Requirements		
		Conditions of Anonymity	**Complexity of data**	**Classification Method**
1	[RGCGP15]	k-anonymity	RDF	Manual (I, QI, SI, USI)
2	[HHD17]	k-anonymity neighborhood	RDF	Manual (I, QI, SI, USI)
3	[SLB+17]	Differential privacy	RDF	Manual (SI)
4	[Aro13]	Differential privacy	RDF	Manual (SI)
5	[NJ02]	k-anonymity	Structured data	Manual (I, QI, SI, USI)
6	[MGKV06, LLV07, MJK09, SO14]	k-anonymity and variations	Structured data	Manual (I, QI, SI, USI)
7	[DMNS06]	Differential privacy	Structured data	Manual SI
8	[HRMS10, MCFY11, SI12]	Differential privacy and variations	Structured data	Manual (SI)
9	[CT08, LT08, YW08]	k-degree	Undirected, node-unlabeled, edge-unlabeled	Manual (I, QI, SI, USI)
10	[BDK07, CKR+11, HMJ+08, ZP08]	k-degree	Undirected, node-labeled, edge-unlabeled	Manual (I, QI, SI, USI)
11	[LWLZ08]	k-degree	Undirected, node-labeled, edge-labeled (weight)	Manual (I, QI, SI, USI)
12	[YCYY13]	k-degree l-diversity	Undirected, node-labeled, edge-unlabeled	Manual (I, QI, SI, USI)
13	[RKKT13, RKKT14a] [RKKT14b, RKKT14c]	Sanitization	RDF	Manual (I, QI, Si, USI)

As RDF data can be also represented as a graph, anonymization graph approaches have been explored in this work. The simplicity of the graph structure assumption makes the current approaches not adequate for the Semantic Web, where heterogeneous nodes and relations are present. Some criteria of anonymization, such as k-degree, can be adopted to the Semantic Web, but the solutions to satisfy these criteria have to be modified according to the complexity of the RDF structure.

Most of the works in RDF documents, databases and graphs anonymization assume that the classification of the data required to satisfy the conditions of anonymity, is provided by expert user. However, the scenario of the Semantic Web complicates the task of classification, since it is difficult to understand the detailed characteristics of external datasets and assume all the background knowledge possessed by adversaries.

Table 5.4 shows our analysis in this regard. Note that none of the works on database and graph anonymization satisfies the criteria of complexity of data (heterogeneous nodes and relations). Moreover, the classification on the data is mainly provided by the proposals and there is no information about how it was performed. We assume that the process to classify the data has been manual. Thus, a new anonymization approach able to cope all requirements is needed to provide an appropriate protection of sensitive information for the Semantic Web.

Before describing how our approach addresses these requirements, the following section introduces some common terminologies and definitions of anonymization in the context of RDF.

5.3 Problem Definition

As we show in the motivating scenario, there are cases in which sensitive information can be disclosed through the data published from different sources on the Web (due to data intersection). Thus, the data to be published, denoted as D, should be protected before, in order to avoid compromising the disclosure or production of sensitive information.

The available information on the Web is called background knowledge. It can be provided automatically or semi-automatically by the expert user and can contain simple or complex resources (e.g., one RDF resource, RDF graph, text files). The background knowledge is formally defined in Def. 30.

Definition 30. Background Knowledge (*BK*): *It is a set of IRIs, considered as nodes and denoted as BK:* $\{n_1, n_2, ..., n_i \mid \forall n_i, n_i$ *is a IRI*$\}$. ♦

In this work, we assume that the intersection between D and BK can disclose or produce sensitive information, hence identifiers and quasi-identifiers appear in D due to the connection among its subjects and objects. We rename both concepts to keys, defined in Def. 31, since they allow the disclosure of sensitive information.

Definition 31. Keys (*K*): *Keys are identifiers and quasi-identifiers, denoted as* $K : \{k_i \mid \forall k_i \in I \cup BN \cup L, k_i$ *produces sensitive information*$\}$. ♦

We formally define our assumption concerning the intersection between D and BK datasets in Ass.5.

Assumption 5. Key Detection (Intersection) (*IN*): *The intersection between a set of triples T and a set of IRIs I is defined as a set of nodes (subjects and objects of triples) that belong to the RDF graph of T (G_T), denoted as IN, where each node of IN has another similar one in I. The similarity among the two nodes is measured by a similarity function (simFunc), whose value is equal or greater than an established threshold.*

$$IN : T \sqcap I = \bigcup_{\{n_i \in G_T | sim(n_i \in T, n_j \in I, \alpha, \beta, \gamma) \geq threshold\}}$$

Where:

– $\sqcap$ *is an operator that defines the intersection between triples and IRIs;*

– n_i *is a subjects or object that belong to T;*

– n_j *is a IRI that belong to I;*

– *sim is the similarity function defined in Def. 32.*

The similarity function between two nodes is defined in Def. 32.

Definition 32. Similarity function (*simFunc*): *The similarity between two nodes is defined as a float value, denoted as simFunc that takes into account tree different aspects of the nodes: (i) syntactic; (ii) semantic; and (iii) context analysis, such that:*

$$\begin{aligned} simFunc(n_i, n_j, \alpha, \beta, \gamma) = \alpha \times syntactic\_similarity(n_i, n_j) + \\ \beta \times semantic\_similarity(n_i, n_j) + \\ \gamma \times context\_similarity(n_i, n_j)) \end{aligned}$$

Where:

– $n_i \in I \cup BN \cup L$ *and* $n_j \in I$;

– *Syntactic_similarity is a function which considers the syntactic aspect of the node, whose values are in [0, 1];*

– *Semantic_similarity is a function which considers the semantic aspect of the nodes, whose values are in [0, 1];*

– *Context_similarity is a function which considers the incoming and outgoing relations of the nodes, whose values are in [0, 1];*

– $\alpha + \beta + \gamma = 1$. ♦

According to the type of nodes of BK (IRIs), different similarity functions should be provided to discover similar nodes. The nodes belonging to the intersection between D and BK (IN), are potential keys according to our assumption, then $K = IN$. The triples from D that contain at least a key are considered as disclosure sources, defined in 33, since the triples are connected to other resources.

Definition 33. Disclosure Sources (*DS*): *It is a set of triples, which contains at least a key from K, denoted as $DS : \{ds_i \mid \forall ds_i \in D \wedge (ds_i.s \in K \vee ds_i.o \in K)$, ds_i is a disclosure source that disclose or produce sensitive information$\}$, .* ♦

However, all triples in D that contain at least a key, cannot be considered as *disclosure sources*, since it depends of the scenario; thus, the interaction of the expert user is needed to identify only the ones that compromise the data to be published. Def. 34 formally explains the result of the expert interaction.

Definition 34. Disclosure-Source Query (*EU*): *It is a selection/projection query applied over DS ($\prod_{DS}$), that returns triples considered as disclosure sources by the expert user according to the scenario. This set of triples is denoted as* $EU : \{eu_i \mid \forall eu_i \in DS$, eu_i *is considered as a disclosure source by the expert user*$\}$. ♦

Using the classification of the expert user, anonymization methods can be applied on the selected triples in order to prevent the disclosure of sensitive information. Note that even the original set of triples (D) could be protected, it should be re-protected considering the already published data (BK) and their intersections with the original one. An anonymization operation is formalized in Def. 35.

Definition 35. Protection Function (*ProtFunc*): *It is a function applied on a triple that returns another similar one, by modifying either the subject, the predicate, the object, or all the three RDF elements, to avoid the disclosure of sensitive information. It is denoted as* $ProtFunc(t \in D, op, par)$, *where op is a protection operation (e.g., generalization, suppression) and pr are the parameters of configuration (e.g., level of generalization).* ♦

By the result of applying the protection process on the set of triples selected by the expert user, the protected data is obtained. This latter is formalized in Def. 36 and it does not allow the disclosure of sensitive information.

Definition 36. Protected data to be published (*pD*): *It is a set of triples denoted as pD, which is the result of applying any protection technique on the set of triples selected by the expert user (EU) of D; i.e., the data to be published are protected if their intersection with the BK does not produce the triples selected by the expert user, using the same threshold established during the intersection:*

$$pD = D \sqcap \{ProtFunc(eu_i) \mid eu_i \in EU\}$$

Where:

– *D is the data to be published;*

– $\sqcap$ *defines the replacement of the set* $EU \subset D$ *with the one obtained by applying a operation function over its elements;*

– *EU is the set of triples considered as disclosure sources by the expert (see Def. 34);*

– *ProtFunc is a function that applies a protection operation (e.g., generalization, suppression) on either the subject, predicate, object, or all three values.* ♦

The following section describes our protection process.

5.4 Protection Process: Our Proposal

Our protection process mainly relies on a four phases approach (see Fig. 5.3), called *Ri**A**iR*, where the input, a set of RDF documents in any serialization format (D), is converted into a graph representation, used by all modules: (i) *Reducing-Complexity phase* in which the graph is analyzed to reduce its complexity-structure to extract a compressed one; (ii) *Intersection phase*, where similar nodes between the input graph (reduced or not) from D and the one from the BK are identified as potential keys (IN); (iii) *Selecting phase* in which the expert user analyzes and selects the disclosure sources (EU), which contains at least one potential key; and (iv) *protection phase* that executes a protection process over the selected triples (EU).

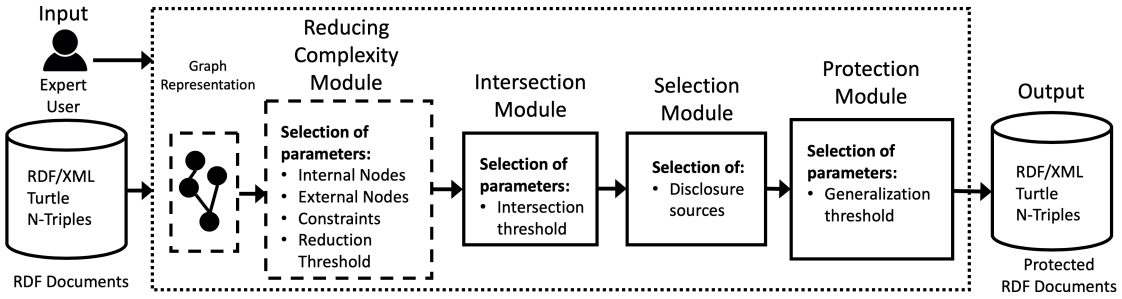


Figure 5.3: Framework of our RDF protection process

A description of each phase is presented in the following sections.

5.4.1 Reducing-Complexity Phase

Since the expert user needs to classify thousands of triples available in D, a reduction step is needed in order to simplify the interaction and make easy the task of classification. Some triples are essential to describe concepts; therefore, they cannot be removed from the data and are considered as constraints. These latter are a set of triples, defined by the expert user, that have an important role over the data. The set of constraints is defined in Def. 37.

Definition 37. Constraints (*C*): *It is a selection/projection query applied over D ($\prod_D$) the indicate the triples to be preserved. It is denoted as C:*$\{c_i{:}< s_i, p_i, o_i >| \forall c_i \in D, c_i$ *is a triple to be preserved*$\}$. ♦

For example, we define as a constraint the triples whose predicates are equal to the value *http://www.w3.org/1999/02/22-rdf-syntax-ns#type*, since it describes the concept of a resource.

The set of triples T=$\{\mathtt{t_i} : \langle \mathtt{s_i}, \mathtt{p_i}, \mathtt{o_i} \rangle\}$ of D is analyzed by the similarity function *simFunc* defined in Def. 32, considering the set of IRIs as a simple node (e.g., a resource). This similarity should take into account the context of the value (e.g., a similarity function based on the incoming and outgoing relations) instead of the analysis of the value itself in order to identify a more general resource. From two similar nodes, the one that subsumes the other is kept. A sorting step to organize the triples in a defined order is needed to return a unique output (e.g., Depth-Subject-Predicate-Object order). As sensitive information can be present in resources and literal values as well, we classify the nodes into two categories: *internal nodes*, which are the ones that are subjects and objects at the same time, and *external nodes* that are only objects in the set of triples (T).

We propose Algo. 6 and Algo.7 to reduce the complexity of each category of nodes. The reducing-complexity algorithm applied on internal nodes, receives a set of triples T=$\{\mathtt{t_i} : \langle \mathtt{s_i}, \mathtt{p_i}, \mathtt{o_i} \rangle\}$, a threshold th_1, and returns another set of triples T'=$\{\mathtt{t'_i} : \langle \mathtt{s'_i}, \mathtt{p'_i}, \mathtt{o'_i} \rangle\}$. In Algo. 6, each triple (t_i) in T is analyzed by the *simFunc* applied to its subject (node) with other subjects from T (lines 4-5 of Algo. 6). If the *simFunc* is equal or greater than the defined threshold (th_1), the triple (t_i) is added to the list *processedListTriples* and the subject of t_i will be replaced by the one from t_j in all triples from T (lines 8, 9 of Algo. 6). The replacing function is performed in line 11 of Algo. 6 and the modified set of triples is returned in line 13.

The algorithm for external nodes receives a set of triples T=$\{\mathtt{t_i} : \langle \mathtt{s_i}, \mathtt{p_i}, \mathtt{o_i} \rangle\}$, a threshold th_1, and returns another set of triples T'=$\{\mathtt{t'_i} : \langle \mathtt{s'_i}, \mathtt{p'_i}, \mathtt{o'_i} \rangle\}$, according to the threshold (th_1) provided by the expert user. A list, called *removeListTriples*, is used to store temporally the triples to be removed in the last step of the algorithm (line 1 in Algo. 7). As the previous algorithm, a sorting step is needed to return an unique output. Each subject (node) from triple t_i in T is compared with other subjects that belong to the triples in T, using the similarity function *simFunc* defined in Def. 32 for simple nodes. To verify if the triple has an external node, its depth[4] is calculated. If the depth of t_i is different than 0, then the object node is not external, and we move forward to the next triple in T (lines 4-5 of Algo. 7). If the *simFunc* between t_i and t_j is equal or greater than the defined threshold and t_i does not belong to the set of constraints (C in Algo. 7) defined by the expert user (see Def. 37), the triple t_i is added to the *removeListTriples* list

[4]The depth of a triple is considered as the biggest path of its object to a terminal node.

Algorithm 6: Reducing Complexity - Internal Nodes

Input: Set of triples T={t:⟨s,p,o⟩}, threshold th_1

Output: Set of triples T'

```
1  processedListTriples = {};       //List of processed triples.
2  replaceListNodes = {};       //List to replace nodes in the set of triples.
3  T = T.sort(HSPO);        //Sorting by depth-subject-predicate-object order.
4  foreach t_i in T do
5      foreach t_j in T-{t_i} do
6          if t_j ∉ processedListTriples then
7              if simFunc(t_i.s, t_j.s)≥ th_1 then
8                  processedListTriples.add(t_i);
9                  replaceListNodes.add(Pair(t_i.s,t_j.s));
10                 break; //Since a similar node was found, the next t_i is analyzed.
11 T' = T.replaceNodes(replaceListNodes);  // Nodes are replaced.
12 T' = T'.removeDuplicateTriples();       //Duplicate triples are removed.
13 return T';
```

(lines 8-10 in Algo. 7). Finally, the triples are removed in line 11 in Algo. 7).

Note that Algo. 6 and Algo. 7 are independent and they can be used in any order.

The reducing-complexity algorithms are applied to the data to be published (D). Once the reductions are obtained, the intersection among this set and the BK can be performed. Following phase describes the intersection phase.

5.4.2 Intersection Phase

The previous phase reduces the complexity-structure of D; the number of triples of D to decrease the interaction of the expert user over the data. However, identifying the triples that are disclosure sources in the reduced set of D, is still a difficult task for the expert user. To identify the nodes of the reduced set D that belong to the intersection with the background knowledge (BK), we propose Algo. 8, based on the *intersection among two datasets* assumption (see Ass. 5) and using the similarity function defined in Def.32.

Algo. 8 receives a set of triples T={$\mathtt{t_i} : \langle \mathtt{s_i}, \mathtt{p_i}, \mathtt{o_i} \rangle$}, a set of IRIs I, a threshold th_2, and returns a set of nodes IN, according to the threshold defined by the expert user.

Algorithm 7: Reducing Complexity - External Resource

Input: Set of triples T=$\{t{:}\langle s,p,o\rangle\}$, threshold th_1

Output: Set of triples T'

```
1  removeListTriples = {};    //List to remove triples.
2  T = T.sort(HSPO);          //Sorting by depth-subject-predicate-object order.
3  foreach t_i in T do
4      if t_i ∈ removeListTriples ∨ depth of t_i ≠ 0 then
5          continue;          //Next triples is analyzed.
6      foreach t_j in T - {t_i} do
7          if t_j ∉ removeListTriples then
8              if simFunc(t_i.s, t_j.s)≥ th_1 and t_i ∉ C then
9                  removeListTriples.add(t_i);    //Adding triples to be removed.
10                 break; //Since a similar node was found, the next t_i is analyzed.
11 T' = T.removeTriples(removeListTriples);   //Triples of removeListTriples list
       are removed.
12 return T';
```

Each subject and object from triple t_i in T are analyzed by using the similarity function (*simFunc*) with the IRI i_j in I. If *simFunc* is equal or greater than the defined threshold (*th*), the subject or object from triple t_i in T are added to the list IN (lines 4-9 in Algo. 8). TThe set IN is returned in line 10.

The nodes of IN are considered as potential keys (see Def.31), since they allow the connection of the data to be published with other datasets. Following section presents the selecting phase which is executed by the expert user.

5.4.3 Selecting Phase

According to Def. 33, triples that contain at least a key are disclosure sources and can disclose or produce sensitive information; however, not all triples that belong to this definition can reveal sensitive information; therefore, the interaction of the expert user is needed to select only the triples that compromise the data. The selection can be performed by a query or any other method.

To further simplify the expert user interaction, we propose the use of a Graphic User Interface (GUI) based on the set of potential disclosure sources (DS). By a visual interface,

Algorithm 8: Intersection among two datasets

Input: Set of triples T=$\{t_i$:$\langle s,p,o\rangle\}$, I, threshold th_2

Output: Set of nodes IN

```
1  IN = {};      //Set of nodes.
2  foreach t_i in T do
3      foreach i_j in I do
4          if simFunc(t_i.s, i_j)≥ th_2 then
5              if t_i.s ∉ IN then
6                  IN.add(t_i.s);      //The subject of T is added.
7          if simFunc(t_i.o, i_j)≥ th_2 then
8              if t_i.o ∉ IN then
9                  IN.add(t_i.o);      //The object of T is added.
10 return IN;
```

the expert user can analyze and select only the triples which are disclosure sources for the scenario. The set of triples obtained by the selection of the expert user, is the set EU (see Def. 34).

Following section describes the protecting phase applied over the set of triples EU.

5.4.4 Protection Phase

Once the disclosure sources are selected by the expert user, a protection process on these triples can be performed. We propose the use of generalization operations on the predicate of each triple, to only reduce the connections among datasets (D and BK), preserving the objectives of integration and combination of the Semantic Web. A taxonomy for each type of relation from the set of triples EU (see Def. 34), has to be provided by the expert user. Moreover, a measure to calculate the level of generalization, applied to the taxonomies (to choose a predicate form a set of values), is needed (e.g., hierarchical and taxonomy measures) in order to provide an appropriate, customized and measured protection according to different scenario. Algo. 9 describes the protection process by applying a generalization operation on each selected triple of EU (see Def.35).

Algo. 9 receives a set of triples T=$\{\mathtt{t_i} : \langle \mathtt{s_i}, \mathtt{p_i}, \mathtt{o_i}\rangle\}$, a set of taxonomies TA, a level of generalization g, which is a value among [0, 1], and returns a set of modified triples T'=$\{\mathtt{t'_i} : \langle \mathtt{s'_i}, \mathtt{p'_i}, \mathtt{o'_i}\rangle\}$, according to the taxonomies and the level of generalization provided

Algorithm 9: Protection Process

Input: Set of triples T=$\{t_i$:$\langle s,p,o\rangle\}$, Set of taxonomies TA, Level of generalization g

Output: Set of triples T'

```
1 T' = {};                                  // Set of triples.
2 foreach t_i in T do
3     Taxonomy ta = TA.getTaxonomy(t_i.p);  // Taxonomy of predicate t_i.p.
4     t_i.p = ta.getPredicate(g);           // Predicate from taxonomy ta.
5     T'.add(t);                            // The modified triple is added to T'.
6 return T';
```

by the expert user. From the set of taxonomies provided by the expert user (TA), the taxonomy which corresponds to the predicate of t_i ($t_i.p$) is used to obtain another predicate that satisfy the level of generalization (g) (lines 3 and 4 in Algo. 9). The modified triple is added to the list T' (line 5 in Algo. 9) and the whole list is returned in line 6.

Note that to obtain the protected RDF data, the compressed triples selected by the expert user, have to be released to apply the protection process over their triples.

Our whole proposal overcomes the limitations identified in the context of RDF protection. The proposal is designed for RDF data, considering their elements (IRIs, blank nodes and literals) and the scenario, where linked and heterogeneous resources are available. The complexity of the RDF structure is reduced in order to decrease the interaction of the expert user and to make easy the task of classification. Potential keys are identified and disclosure sources are provided to the expert user. Moreover, by a generalization method, we reduce the connections among datasets, preserving the main objectives of the Semantic Web (integration and combination), and protecting the sensitive information at the same time.

The following section evaluates the complexity of our proposal.

5.4.5 Complexity Analysis of the whole Anonymization Process

A complexity analysis of our anonymization approach indicates a quadratic order performance in terms of number of triples of the data to be published (n) and the ones from the background knowledge (m), i.e., $O(n^2 + m^2)$. A detailed complexity analysis was done on each phase of the process to get the complexity of the whole process:

- For the Reducing-complexity phase, each triple (n) is analyzed by searching another similar one in the set of triples, then their execution order is $O(n^2)$.
- The Intersection phase based on the reduced set of triples from D, has an execution order $O(n \times m)$, D and BK respectively, for the worst case where no triple was removed by reducing-complexity phase.
- The Configuration phase, which is made by the expert user, depends of the number of triples from D that contain potential keys, which are obtained by the intersection between D and BK. Thus, this phase has an execution order $O(n)$ where all triples are considered as disclosure sources.
- The anonymization phase, applied over the triples selected by the expert user, has an execution order of $O(n)$, if all triples from D are considered as disclosure sources.

As the four phases are executed sequentially, the whole anonymization approach exhibits a quadratic order complexity, i.e., $(O(n^2 + m^2 + n \times m + 2 \times n))$.

The following section evaluates the viability and demonstrate the quadratic order performance of our proposal.

5.5 Experimental Evaluation

5.5.1 Prototype and Implementation

To evaluate and validate our anonymization approach, a desktop prototype system, called *Ri**A**iR*, was developed using Java. Fig. 5.4 shows a visual interface of our prototype, which has several customizable options according to user-preferences. For example, the expert user can apply the reducing-complexity process to either internal, external nodes, or only one of them. The thresholds for the reduction, intersection, and anonymization processes can be also customized by the expert user, selecting a value among [0,1] in the left area of the visual interface.

For the reducing-complexity and intersection phases, we implemented the similarity function, called *simFunc* (Def. 32), considering only the context similarity by using the incoming and outgoing properties (relations) from the nodes, since the behavior of a node can be defined through its relations (context). We present the similarity function as follows.

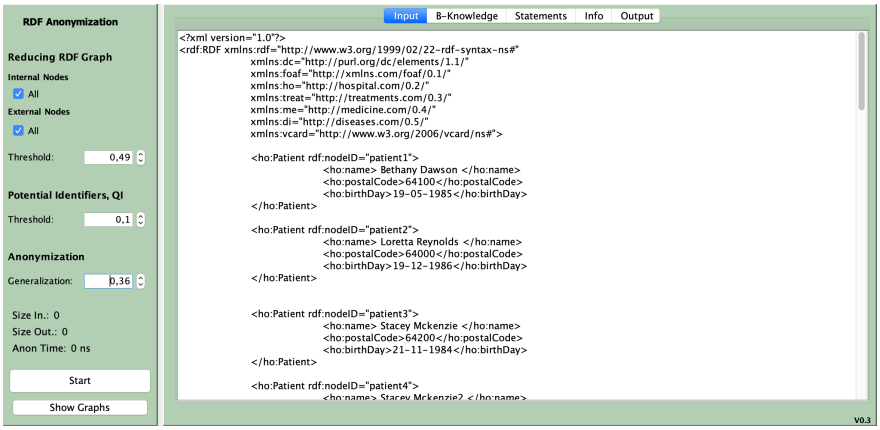


Figure 5.4: Visual interface of our Anonymization Approach

$$simFunc(n_i, n_j, \alpha{=}0, \beta{=}0, \gamma{=}1) = \alpha \times syntactic\_similarity + \beta \times semantic\_similarity + \gamma \times (0.5 \times \frac{|incomingProperties(n_i) \cap incomingProperties(n_j)|}{|incomingProperties(n_i) \cup incomingProperties(n_j)|} + 0.5 \times \frac{|outgoingProperties(n_i) \cap outgoingProperties(n_j)|}{|outgoingProperties(n_i) \cup outgoingProperties(n_j)|})$$

Where:

– $incomingProperties$ is a function that returns the incoming relations of a node;

– $outgoingProperties$ is a function that returns the outgoing relations of a node;

Note that for the reducing-complexity phase, the intersection and union among properties is made by a syntactic string comparison; while for the intersection phase (see Def. 31), since the datasets are provided from different sources, the syntactic comparison is performed to only the *property name* of the incoming and outgoing properties (e.g., http://www.domain1.com/**nameProp** is equal to http://www.domain2.com/**nameProp**, since both property names are equals - nameProp).

For the anonymization phase, we implemented a generalization operation based on taxonomies provided by the expert user. The taxonomies are processed by the approach through the use of a simple document in XML format, presented as follows.

```
<taxonomies>
    <taxonomy_1>
        <taxonomy_1a>
        </taxonomy_1a>
        <taxonomy_1b>
        </taxonomy_1b>
```

```
    </taxonomy_1>
    <taxonomy_2>
        <taxonomy_2a>
        </taxonomy_2a>
    </taxonomy_2>
    ...
</taxonomies>
```

A taxonomy for each triple of the set EU (see Def.34) is analyzed by applying a similarity measure that returns another similar relation (predicate) according to a defined threshold. We use the similarity measure of Chapter 4, since it takes into account the deepness, the distance, and the children in common of the taxonomies.

5.5.2 Datasets and Environment

Our prototype was used to perform several experiments to evaluate the viability and the performance (execution time) of our approach in comparison with the related work. To do so, we considered three datasets:

- **Data 1**: the *DBpedia person data*[5] with 16'842,176 triples (used to evaluate the reducing-complexity phase due to the huge number of triples);
- **Data 2 (BK)**: the *DBpedia geo coordinates* [6] with 151,205 triples; and
- **Data 3 (D)**: an extraction of Enipedia dataset (power plants), considering properties `cat:Fuel_type`, `rdfs:label`, `cat:radioactive`, `prop:City`, `prop:Country`, `prop:lat`, `prop:long`, and `prop:year`, with 568 triples.

Using **Data 1**, **Data 2**, and **Data 3**, we evaluated the viability and performance of the reducing-complexity process, while for the intersection phase, we used **Data 2** and **Data 3**. The protection phase is applied over the reduced set of triples obtained by the reducing-complexity phase and the set of nodes of the intersection phase between **Data 3** and **Data 2**. Since in this particular case the BK is also a set of triples (a complex node), we applied the reducing-complexity process over the dataset as well. Experiments were undertaken on a MacBook Pro, 2.2 GHz Intel Core(TM) i7 with 16.00GB, running a MacOS High Sierra and using a Sun JDK 1.7 programming environment.

[5]Information about persons extracted from the English and Germany Wikipedia, represented by the FOAF vocabulary - `http://wiki.dbpedia.org/Downloads2015-10`.

[6]Geographic coordinates extracted from Wikipedia - `https://wiki.dbpedia.org/downloads-2016-10`.

5.5.3 Evaluation metrics

5.5.3.1 Accuracy in disclosure sources

In order to evaluate the accuracy of our approach when a set of triples are suggested as disclosure sources to the expert user, we calculated the F-score, based on the Recall (R) and Precision (PR). These criteria are commonly adopted in information retrieval and are calculated as follows:

$$\textbf{PR} = \frac{A}{A+B} \in [0,1] \qquad \textbf{R} = \frac{A}{A+C} \in [0,1] \qquad \textbf{F-score} = \frac{2 \times PR \times R}{PR+R} \in [0,1]$$

where A is the number of correctly suggested triples; B is the number of wrongly suggested triples; and C is the number of triples not suggested by our approach but considered as disclosure sources.

According to our scenario, **Data 3** contains eight properties, from which only two properties (prop:lat and prop:long) are considered as disclosure sources. Thus, 142 triples need to be selected by the expert user, since 71 power plants are present (a total of 568 triples). We describe the accuracy evaluation in subsection Configuration Phase.

5.5.3.2 Protection Data Verification

To consider the data as a protected one, it should not contain disclosure sources which compromise the data; thus, to verify the data, we propose a measure based on the sensitive triples returned by applying a query over the datasets. The verification is performed as the relation between the sensitive information produced by the original data with respect to the one produced by the protected data; i.e.,

$$\textbf{AnonV(D,pD)} = \frac{N.\ of\ sensitive\ triples\ from\ D\ -\ N.\ of\ sensitive\ triples\ from\ pD}{N.\ of\ sensitive\ triples\ from\ D} \in [0,1].$$

where D is the data to be published and pD the protected one (see Def. 36).

For our evaluation, we use the query presented in our motivating scenario, considering any type of resources (e.g., `dbo:School`, `dbo:Hospital`) as places of interest. A total of 364 entities, represented by 1456 triples, are sensitive information.

```
SELECT DISTINCT
```

```
?Place ?g bif:st_distance(?g,bif:st_point(".$long.",".$lat.")) AS ?distance
FROM <http://dbpedia.org>
WHERE {?p rdfs:label ?Place ; geo:geometry ?g.
FILTER (bif:st_intersects (?g, bif:st_point (".$long.", ".$lat."), 100)
&& (lang(?Place) = \"en\"))}
ORDER BY ASC(?distance)
```

This metric evaluates the protected RDF data in the subsection Protection Phase. We describe and evaluate as follows each process to obtain a protected RDF data.

5.5.4 Reducing-Complexity Phase

We performed the reducing-complexity process over three real datasets available on the Web (**Data 1**, **Data 2**, and **Data 3**). We evaluated the Jena parsing-time (ms) and the size (bytes) of the input and output to compare the improvement of working over the output in terms of viability and performance.

5.5.4.1 Viability Evaluation

Test 1: We chose randomly the value *0.44* as the threshold for the reducing-complexity process. We extracted 1,000 triples from each dataset and increased the number of triples by a step of 1,000 for the next iterations. Table 5.5 shows the results obtained for **Data 1**. This process reduced the complexity of more than 16 million of triples to only 132 triples, since the values were extracted from *Wikipedia* following a schema with a finite number of properties. The Jena parsing-time of the input is reduced to 1.01 ms (132 triples) and its size to 9333 bytes. Note that applying the same threshold for different sets of triples extracted from **Data 1**, we obtain the same output for all the cases, showing that the general schema of the resources (finite number of properties) is returned by this process.

For **Data 2**, Table 5.6 shows the results of applying the reducing-complexity process. The dataset of 151,205 triples is reduced to only 4 triples, i.e., the 151,205 triples follow the schema represented by the 4 returned triples. The Jena parsing-time and the size of the input were reduced to 0.40 ms and 455 bytes, respectively. In **Data 3**, the output contains only 8 triples from 568 triples as we can observe in Table 5.7. The Jena parsing-time and the size of the dataset was reduced to 0.68 ms and 769 bytes, respectively. Similarly to the two previous data sets, the 8 returned triples represents the scheme of all triples in the set.

Table 5.5: **Test 1**: Reducing-Complexity process for **Data 1**, using a threshold 0.44

Data 1	**Input**			**Output**		
Threshold	**Triples**	**Jena Time (ms)**	**Size (bytes)**	**Triples**	**Jena Time (ms)**	**Size (bytes)**
0.44	1,000	7.99	68958	132	1.10	9333
0.44	2,000	16.89	138108	132	1.08	9333
0.44	3,000	23.95	207036	132	1.12	9333
0.44	4,000	30.41	276070	132	1.05	9333
0.44	5,000	36.50	345687	132	1.07	9333
0.44	6,000	42.75	414809	132	1.15	9333
0.44	7,000	48.23	484719	132	1.06	9333
0.44	8,000	53.11	553507	132	1.10	9333
0.44	9,000	56.93	622646	132	1.01	9333
0.44	10,000	61.12	666224	132	1.09	9333
0.44	**16'842,176**	–	–	132	1.03	9333

Table 5.6: **Test 1**: Reducing-Complexity process for **Data 2**, using a threshold 0.44

Data 2	**Input**			**Output**		
Threshold	**Triples**	**Jena Time (ms)**	**Size (bytes)**	**Triples**	**Jena Time (ms)**	**Size (bytes)**
0.44	1,000	9.45	77144	4	0.40	455
0.44	2,000	17.94	154729	4	0.35	455
0.44	3,000	25.37	232222	4	0.39	455
0.44	4,000	31.49	309952	4	0.44	455
0.44	5,000	38.63	387289	4	0.36	455
0.44	6,000	44.98	464888	4	0.41	455
0.44	7,000	51.81	543737	4	0.37	455
0.44	8,000	57.41	622768	4	0.36	455
0.44	9,000	62.74	700421	4	0.39	455
0.44	10,000	69.89	778651	4	0.42	455
0.44	**151,205**	–	–	4	0.40	455

Test 2: In order to select the best threshold for the reducing-complexity process of each dataset, we evaluated the number of triples, Jena parsing-time, and the size of the output by using a threshold value between [0.01 - 1.00] with a step of 0.01. Table 5.8 shows the results obtained for **Data 1**. As we can observe, we obtained the best result for the thresholds from 0.01 to 0.29, where only nine properties are used in the whole database. The Jena parsing-time of the output was reduced to 0.49 ms, while the size was reduced to 834 bytes.

For **Data 2** and **Data 3** (see Tables 5.9 and 5.10), the best results were obtained for a wide range of thresholds [0.01 - 0.49]. By regarding the datasets, in **Data 2** and **Data 3**, all resources were described by the same properties (four and eight properties, respectively), while in **Data 1**, there are some resources described by only three or four properties from a total of nine, therefore in **Data 1**, the optimal threshold was obtained in a smaller range [0.01 - 0.29], since for the range [0.30 - 0.49], some resources were not

Table 5.7: **Test 1**: Reducing-Complexity process for **Data 3**, using a threshold 0.44

Data 3	**Input**			**Output**		
Threshold	**Triples**	**Jena Time (ms)**	**Size (bytes)**	**Triples**	**Jena Time (ms)**	**Size (bytes)**
0.44	568	4.99	37645	8	0.68	769

considered as similar to the general schema due to their smaller number of properties.

Table 5.8: **Test 2**: Reducing-Complexity process for **Data 1** with a step 0.01

Data 1	**Input**			**Output**		
Threshold	**Triples**	**Jena Time (ms)**	**Size (bytes)**	**Triples**	**Jena Time (ms)**	**Size (bytes)**
[**1.00 , 0.50**]	10,000	63.62	666224	10,000	62.56	666224
[**0.49 , 0.45**]	10,000	61.54	666224	148	1.17	10420
0.44	10,000	62.21	666224	132	1.12	9333
0.43	10,000	65.32	666224	111	0.96	7934
0.43	10,000	62.59	666224	75	0.86	5423
[**0.41 , 0.40**]	10,000	61.98	666224	55	0.80	4040
0.39	10,000	60.81	666224	39	0.72	3069
0.38	10,000	62.44	666224	26	0.63	2174
[**0.37 , 0.36**]	10,000	62.86	666224	33	0.65	2617
[**0.35 , 0.34**]	10,000	61.12	666224	18	0.56	1523
[**0.33 , 0.30**]	10,000	63.29	666224	12	0.51	1047
[**0.29 , 0.01**]	10,000	63.58	666224	9	0.49	834
0.29	**16'842,176**	–	–	9	0.49	834

Table 5.9: **Test 2**: Reducing-Complexity process for **Data 2** with a step 0.01

Data 2	**Input**			**Output**		
Threshold	**Triples**	**Jena Time (ms)**	**Size (bytes)**	**Triples**	**Jena Time (ms)**	**Size (bytes)**
[**1.00 , 0.50**]	10,000	69.25	778651	10,000	69.42	778651
[**0.49 , 0.01**]	10,000	70.91	778651	4	0.39	455
0.49	**151,205**	–	–	4	0.39	455

5.5.4.2 Performance Evaluation

To evaluate the performance of of the reducing-complexity phase, we measured the average time of 10 executions for each test.

Test 3: We evaluated the time of the reducing-complexity process of 10,000 triples from **Data 1** by using several thresholds between [0.01 - 1.00] in order to observe the influence of the threshold over the reduction time. Figure 5.5 shows that from a threshold 0.49, where the number of triples is reduced to only 148, the reduction time decreases to 4,977.91 ms until 3,668.54 ms for a threshold value of 0.01. As more triples are reduced during the reducing-complexity process, less comparisons are performed, since for each iteration less

Table 5.10: **Test 2**: Reducing-Complexity process for **Data 3** with a step 0.01

Data 3	**Input**			**Output**		
Threshold	**Triples**	**Jena Time (ms)**	**Size (bytes)**	**Triples**	**Jena Time (ms)**	**Size (bytes)**
[1.00 - 0.50]	568	4.92	37645	568	4.89	37645
[0.49 -0.01]	568	4.71	37645	8	0.39	769
0.49	**568**	–	–	8	0.39	769

operations of similarity are needed to discover another similar node.

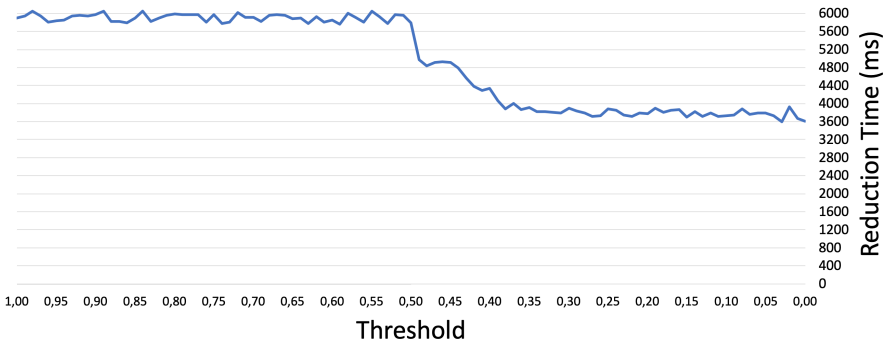


Figure 5.5: **Test 3:** Execution time of the Reducing-complexity process using a threshold between 0.01 and 1.00

Test 4: In this test, we evaluated the impact of the number of triples, from **Data 1**, on the execution time of the reducing-complexity phase. We used a threshold value of 0.29, which was one of the thresholds that reduced more triples, and a step of 10,000 triples for the iterations. Figure 5.6 shows the execution time with respect to the number of triples. For 60,000 triples, the execution time is 302.65s. The result obtained confirms the quadratic performance of this process. The following section evaluates the intersection phase.

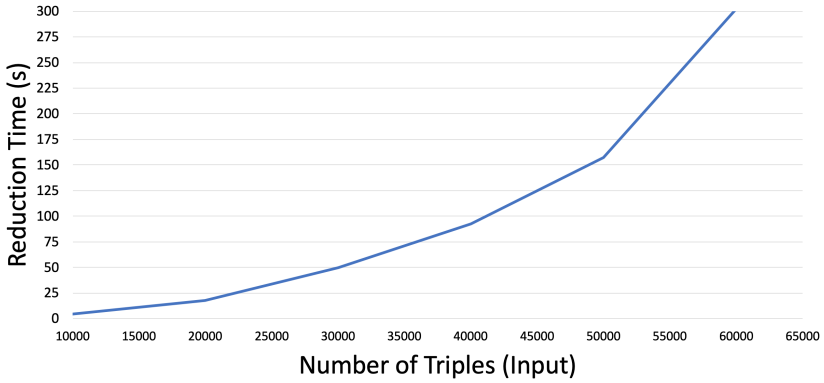


Figure 5.6: **Test 4:** Execution time of the Reducing-complexity process using a threshold value of 0.29

5.5.5 Intersection Phase

Using the reduced datasets of **Data 2** and **Data 3**, obtained by the reducing-complexity process (4 and 8 triples, respectively), we perform the intersection process considering **Data 3** as the data to be published (D), while **Data 2** as the background knowledge (BK).

5.5.5.1 Viability Evaluation

To evaluate the viability of applying this process over real scenarios, we chose randomly a threshold value (0.65) and later we analyzed the behavior of this process with respect to several threshold values.

Test 5: By using a threshold value of 0.65, the intersection process did not return any intersection node. Regarding the reduced datasets, the nodes that represent the latitude and longitude properties are terminal nodes, thus they do not have outgoing properties and its similarity is 0.50. Additionally, The similarity between the node which represents a power plant (**Data 3**) and the one which represents a place of interest (**Data 2**) is calculated based on the properties in common (intersection – only latitude and longitude) from a total of ten properties (union – eight properties in D and four properties in BK), thus their similarity is 0.20.

Test 6: We evaluated the viability of this process using several thresholds from 0.01 to 1.00 with a step of 0.01. From a threshold value between 1.00 and 0.50, no node was returned. For [0.49, 0.21], two nodes which represent the coordinates of the power plant resource in D, are returned as potential keys, which is what we expect. For [0.20, 0.01], three nodes are returned (coordinates and the node which represents the power plant).

Table 5.11: **Test 6:** Intersection process between **Data 2** and **Data 3** with a step 0.01

Threshold	Number of Nodes
[1.00 - 0.50]	0
[0.49 - 0.21]	2
[0.20 - 0.01]	3

5.5.5.2 Performance Evaluation

Test 7: The time required to discover the nodes that can be potential keys, was measured. An average of 10 execution indicates a time of 0.24 ms for this process.

5.5.6 Selecting Phase

A GUI based on triples was built to reduce the effort of the expert user. The interface selects automatically the triples which contain at least one potential key (IN) which are returned by the intersection process.

Test 8: We measured the average of verifying the selected triples, which contain the nodes detected during the intersection process, of 10 people that have under- and pots-graduate degrees in computer science. Since only eight triples are available in the reduced dataset of **Data 3**, the verifying average time was 8.23 s.

Test 9: We evaluated the accuracy of the set of triples suggested as disclosure sources by our approach, using the F-score measure. Table 5.12 shows that for a threshold between [0.49 , 0.21] all triples which compromise the data to be published are suggested (Data 3), obtaining a F-score 100%. For a threshold between [0.20 , 0.01] also the triples which compromise the data are suggested, but other triples were suggested as well. These thresholds have a F-score of 40%.

Table 5.12: **Test 9:** Accuracy evaluation for the set of triples suggested as disclosure sources to the Expert User

Intersec. Thres-hold	**N. of potential keys**	**Triples suggested as disclosure sources (Expert User Interface)**	**Triples suggested as disclosure sources (Internal Mapping)**	**Valid**	**Not Valid**	**Not sugges-ted**	**Prec. (%)**	**Rec. (%)**	**F-score (%)**
[1.00 , 0.50]	0	**0**	0	0	0	142	0	0	**0**
[0.49 , 0.21]	2	**2**	142	142	0	0	100	100	**100**
[0.20 , 0.01]	3	**8**	568	142	426	0	25	100	**40**

5.5.7 Protection Phase

The relations (properties) that belong to the triples considered as disclosure source by the expert user, have to be protected in order to reduce the risk of disclosure of sensitive information. According to the configuration process, the eight triples from the reduced set of **Data 3** were pre-selected in the selecting interface, showing that they can be used to disclosure sensitive information. By the verification of the expert user, the anonymization process is performed. Since there are eight triples with different properties (predicates), eight taxonomies need to be provided by the expert user.

Test 10: We measured the average time of 10 executions, by using a random threshold

of generalization (0.36). A time of 1.12 ms was required to perform this process.

Test 11: Additionally, we evaluated the protected data by using the $AnonV$ function defined in the subsection evaluation metrics. Table 5.13 shows that for a threshold less than 0.50 in the intersection phase, the protected data (pD) does not produce sensitive information, obtaining the maximum evaluation value (100%).

Table 5.13: **Test 11:** Protection Data Evaluation according to the number of sensitive triples produced by the D and pD

Intersec. Threshold	**Sensitive Triples in D**	**Sensitive Triples in pD**	**Protected Data Verification (%)**
[1.00 , 0.50]	1456	1456	0
[0.49 , 0.21]	1456	0	100
[0.20 , 0.01]	1456	0	100

In these subsections, we evaluated the viability and performance of our approach by using datasets available on the Web. We demonstrated a huge reduction of the expert-user interaction suggesting disclosure sources. Also, a high performance was obtained for all the phases. Following subsection evaluates our approach with respect to related work.

5.5.8 Related Work Comparison

In order to measure the viability and the performance of our approach with respect to the state of the art, we selected a work for each identified group of the related work section. For RDF data, we selected the work in [RGCGP15], for structured data (database) the work in [SO14], while for graph data the work in [YCYY13]. Thresholds of 0.49, 0.10, and 0.36 were used for the reducing-complexity (D and BK), intersection, and generalization processes, respectively in our approach.

Test 12: We evaluated the average time of 10 executions of the anonymization processes. From **Data 2**, 10,000 triples are considered as the background knowledge (BK) and the whole **Data 3** as the data to be published (D). Table 5.14 shows the results obtained for this comparison. The non-viability of the works in [RGCGP15, SO14, YCYY13] for real scenarios, was clearly demonstrated in this evaluation, since the interaction of the expert user to classify the data, required a high effort (more than three hours), making this task almost impossible. Moreover, the execution time of the protection processes, without considering the classification, was greater than one hour for [RGCGP15, SO14, YCYY13] (the executions were stopped after one hour of processing), while for our solution was only

5.28 s. Note that we considered the time of classification similar to the time of verification which was obtained in our configuration-phase evaluation (∼1 second for triple).

Following section presents our conclusions of this chapter.

5.6 Summary

In this chapter, we investigated the protection of sensitive information for RDF documents before publication on the Web. We proposed an protection approach, consisting on four phases: (i) *Reducing-Complexity phase*, where the input, a set of RDF documents (D) in any serialization format, is analyzed to reduce its graph complexity; (ii) *Intersection phase*, where similar nodes (IN) between the reduced graph from the data to the published (D) and the one from the background knowledge (BK) are identified as potential keys; (iii) *Configuration phase* in which the expert user analyzes and selects the triples that contain at least one potential key, considered as disclosure sources (EU); and (iv) *protection phase* that executes an generalization operation over the selected triple.

We evaluated the viability and performance of our protection approach with several datasets available on the Web. Results show that our approach decreases the interaction of the expert user by reducing the complexity of the graph structure (reducing-complexity phase), identifying potential keys (intersection phase), and suggesting potential disclosure sources through a graphic user interface to the expert user. Moreover, we evaluated our approach with respect to the state of the art, demonstrating that our proposal overcome existing solutions and these later are not able to manage linked and heterogeneous resources.

Following chapter describes the summary of all chapters, conclusions and future directions of this work.

Table 5.14: **Test 12:** Related Work Comparison

Work	**Complexity of data**	**Triples**	**Classification**		**Anonymization Time (s)**			**Total Time (s)**
			Type	**Time (s)**				
[RGCGP15]	RDF	D: 568 BK: 10,000	Manual (I, QI, SI, USI)	∼10,568 (*)	>3,789.24 (+)			>14,357.24 (>3.99 h)
[SO14]	Structured data	D: 528 BK: 10,000	Manual (I, QI, SI, USI)	∼10,568 (*)	>3,632.67 (+)			>14,200.67 (>3.94 h)
[YCYY13]	Graph	D:568 BK:10,000	Manual (I, QI, SI, USI)	∼10,568 (*)	>3,721.34 (+)			>14,289.34 (>3.97 h)
Our Approach	RDF	D: 568 BK: 10,000	Automatic (I, QI)	8.23 (Verification)	**Reducing Complexity**	**Intersection**	**Anonymization**	13.51 (0.00375 h)
					D: 0.82 BK: 4.46	0.00024	0.00112	

(*) An estimation of 1 second for each triple.
(+) The approach was stopped after an hour of execution.

Chapter 6

Conclusions and Future Works

> "If you want to live your life in a creative way, as an artist, you have to not look back too much. You have to be willing to take whatever you've done and whoever you were and throw them away."
>
> — Steve Jobs

In this thesis, we proposed and evaluated an RDF protection framework, called *Ri**A**iR*. The proposal is designed for RDF documents, considering all their elements and a scenario of a huge quantity of information. The complexity of the RDF structure is reduced to make possible the task of classification and to suggest potential disclosure sources to the expert user, decreasing his interaction. By a generalization method, we reduce the connections among datasets to protect the data and to preserve the objectives of the Semantic Web (integration and combination).

We investigated several similarity functions in order to provide the most adequate one. A similarity function based on the context of the resources (incoming and outgoing properties) was used to perform the experiments. However, in syntactic similarities, some limitations corcening the datatypes were found and studied in this thesis. A similarity and an inference approach among datatypes were proposed.

The experimental evaluation of accuracy, viability, and performance through several databases available on the Web, reflects the effectiveness of your approach in comparison with existing works.

In this chapter, we present the conclusions of our work and a discussion regarding the limitations around the challenges. We conclude with the future works that can extend

the scope of the approach to ensure a better protection for the Semantic Web.

6.1 Synopsis

Chapter 2 described the background information regarding the WWW, Web technologies, and its principles. A Semantic Web definition, its architecture and the standard frameworks as the RDF to describe real resources on the Web, were reported. We discussed the relation between the principles of the Web and how RDF fulfill them. We analyzed the Web of Data, Linked Open Data and the initiatives to convert the datasets to RDF data according to the reported principles.

Chapter 3 analyzed the datatypes proposed by the W3C in the context of RDF matching/integration. We also discussed about the current datatype hierarchy, which does not properly capture any semantically meaningful relationship between datatypes. In additional, we noticed that existing similarity measures among datatypes are not suitable for the Semantic Web, since either they are too restrictive, based on arbitrary judgment, or formulas applied to the W3C hierarchy. In this context, we proposed:

- An analysis of the current W3C datatype hierarchy, its limitations and adequate applicability for the Semantic Web.
- A new datatype hierarchy, extending the one proposed by the W3C.
- A new similarity measure, extending the existing works to take into account the children of each datatype (cross-children similarity).

Experiments showed an important improvement in the accuracy of the approach with respect to the existing works.

Chapter 4 highlights the importance of the datatypes for RDF matching/integration. However, datatypes are not available in some RDF documents; thus, we analyzed the state-of-the-art about datatype inference in order to deduce the datatype from existing information. For the Semantic Web, we provided an inference based on the following steps:

- An analysis of predicate information, such as range property that defines and qualifies the type of the object value.
- An analysis of lexical space of the object value, by a pattern-matching process.
- A Semantic analysis of the predicate and its semantic context.

- A generalization of Numeric and Binary data- types, to ensure a possible integration among RDF documents.

In addition, an online prototype called **RDF2rRDF** was developed, in order to test and evaluate the inference process, according the accuracy and performance, in the context of huge quantity of RDF data. Results showed better accuracy (up to 97.10%) than existing works and a lineal order performance.

Chapter 5 described our main contribution of this study. As more RDF data is published and shared on the Web, sensitive information such as diseases, salaries, or bank accounts, can be also provided. Thus, we proposed a new approach able to avoid the disclosure of sensitive information.

The protection approach was based on four phases: (i) Reducing-Complexity phase in which the graph is analyzed to reduce its complexity-structure to extract a compressed one; (ii) Intersection phase, where similar nodes between the input graph (reduced or not) from D and the one from the BK are identified as potential keys; (iii) Selecting phase in which the expert user analyzes and selects the disclosure sources, which contains at least one potential key; and (iv) protection phase that executes a protection process over the selected triples. Mainly, we provided:

- A general framework designed for RDF documents, independent of the serialization formats, in a scenario where linked and heterogeneous resources are presented; i.e., the Web;
    1. A method to reduce the complexity of the RDF structure of the data to be published, simplifying the task of analysis, performed by the expert user;
    2. A method to suggest disclosure sources to the expert user, based on node similarity, reducing the task of data classification; and
    3. A protection operation, based on a generalization method, to decrease the relations among resources from different datasets, to preserve the main objectives of integration and combination of the Semantic Web.

A desktop prototype was developed in order to test and evaluate the protection process using our motivating scenario. Results showed a viable approach with high performance.

6.2 Future Works

In this section, we discuss possible directions for future works that would advance our research and provide a better study. Future works directions include improvements into similarity and inference of datatypes, similarity and inference in matching tools, and similarity measure among resources. These directions are described as follows:

6.2.1 Complex Datatypes

In chapters 3 and 4, we restricted the scope of our study to simple datatypes. However, the analysis of datatypes can be extended to complex datatypes, since as we mentioned in Chapter 2, complex datatypes contain elements defined as either simple or complex datatypes. Thus, a complex datatype can be treated as a set of simple datatypes in some cases. An average of their elements can be used to measure the similarity among complex datatypes. In the case of inference, to deduce complex datatypes, extra context rules can be proposed according to the type of database to resolve the ambiguity.

6.2.2 Matching Tools

Chapters 3 and 4 propose two different approaches in the context of RDF document matching/integration. Then, these approaches should be evaluated in real matching tools in order to measure the improvement with respect to current methodologies. For example, according to the related work classification from Chapter 3, works from Group 1 ([CAS09, EYO08, ES07, HA09, HQC08, JMSK09, JLD15, LT06, LTLL09, MAL$^+$15, NB16, SSK05]) use binary similarity (similarity among equal datatypes is one, otherwise the similarity is zero), if we replace the existing similarity with the one proposed in this thesis, we can observe the importance of an adequate datatype similarity. For our inference approach, an evaluation between a database without datatypes, a database with their respective datatypes, and another one with the datatypes inferred by our approach, will motivate the need of datatypes for the Semantic Web, as well as, the contribution of our proposal.

6.2.3 Inferring Semantic Datatypes

Our inference approach presented in Chapter 4, is focuses on simple datatypes for literal nodes. There are two types of properties (predicates), object property and datatype

property, then our proposal can be considered as an inference approach for datatype properties, since a datatype property is always related to a literal node. This property is known in the literature as Syntactic datatype, because it describes the format of the value it-self that is related to it. However, semantic datatypes (nodes related to object property) are also present in RDF documents; thus, a new approach able to infer them is needed. Semantic datatype is a complex challenge, since a node (IRI and blank node) can have different semantic datatypes according to the context of the data.

6.2.4 Similarity measure among Resources

The accuracy of our protection approach depends on the similarity measure applied in the reduction and intersection phases. Since heterogeneous and linked RDF datasets are provided using several vocabularies, semantic similarities are needed in order to compare similar resources with different properties.

Bibliography

[ABF12] Maythm Al-Bakri and David Fairbairn. Assessing similarity matching for possible integration of feature classifications of geospatial data from official and informal sources. *International Journal of Geographical Information Science*, 26(8):1437–1456, 2012.

[ACH08] Thomas Arts, Laura M. Castro, and John Hughes. Testing erlang data types with quviq quickcheck. In *Proc. of the 7th ACM SIGPLAN Workshop on ERLANG*, pages 1–8, NY, USA, 2008. ACM.

[Aea08] Alsayed Algergawy and et al. A sequence-based ontology matching approach. In *Proc. of European Conference on Artificial Intelligence Workshops*, pages 26–30, 2008.

[ALNZ13] Sören Auer, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, and Amrapali Zaveri. Introduction to linked data and its lifecycle on the web. In *Reasoning Web. Semantic technologies for intelligent data access*, pages 1–90. Springer, 2013.

[ANS09a] Alsayed Algergawy, Richi Nayak, and Gunter Saake. *On the Move to Meaningful Internet Systems: OTM 2009: Confederated International Conferences, CoopIS, DOA, IS, and ODBASE 2009, Vilamoura, Portugal, November 1-6, 2009, Proceedings, Part II*, chapter XML Schema Element Similarity Measures: A Schema Matching Context, pages 1246–1253. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[ANS09b] Alsayed Algergawy, Richi Nayak, and Gunter Saake. *XML Schema Element Similarity Measures: A Schema Matching Context*, pages 1246–1253. Berlin, Heidelberg, 2009.

[ANS10] Alsayed Algergawy, Richi Nayak, and Gunter Saake. Element similarity measures in xml schema matching. *Inf. Sci.*, 180(24):4975–4998, December 2010.

[AP] M. Davis A. Phillips. Tags for Identifying Languages. `https://tools.ietf.org/html/bcp47`. Online; accessed 2017-09-11.

[Aro13] Yotam Aron. *Information privacy for linked data*. PhD thesis, Massachusetts Institute of Technology, 2013.

[ARTW09] Nuttakan Amarintrarak, SK Runapongsa, Sissades Tongsima, and Nuwee Wiwatwattana. Saxm :semi-automatic xml schema mapping. *The 24th International Technical Conference on Circuits/Systems, Computers and Communications, ITC-CSCC*, 2009.

[AS16] Ainur Abdrashitov and Anton Spivak. Sensor data anonymization based on genetic algorithm clustering with l-diversity. *2016 18th Conference of Open Innovations Association and Seminar on Information Security and Protection of Information Technology (FRUCT-ISPIT)*, pages 3–8, 2016.

[ASS09] Alsayed Algergawy, Eike Schallehn, and Gunter Saake. Improving xml schema matching performance using prüfer sequences. *Data Knowl. Eng.*, August 2009.

[AW16] Olivia Angiuli and Jim Waldo. Statistical tradeoffs between generalization and suppression in the de-identification of large-scale data sets. In *Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual*, volume 2, pages 589–593. IEEE, 2016.

[BDK07] Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. Wherefore art thou r3579x?: Anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 181–190, New York, NY, USA, 2007. ACM.

[BHBL09] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data-the story so far. *Semantic services, interoperability and web applications: emerging concepts*, pages 205–227, 2009.

[BHIBL08] Christian Bizer, Tom Heath, Kingsley Idehen, and Tim Berners-Lee. Linked data on the web (ldow2008). In *Proceedings of the 17th international conference on World Wide Web*, pages 1265–1266. ACM, 2008.

[BLa] Tim Berners-Lee. Artificial Intelligence and the Semantic Web: AAAI2006 Keynote. W3C Website. https://www.w3.org/2006/Talks/0718-aaai-tbl/Overview.html. Online; accessed 2017-09-28.

[BLb] Tim Berners-Lee. Semantic Web - XML 2000. W3C Website. https://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html. Online; accessed 2017-09-28.

[BLc] Tim Berners-Lee. Semantic Web and Challenges 2003. W3C Website. https://www.w3.org/2003/Talks/01-sweb-tbl/slide15-0.html. Online; accessed 2017-09-28.

[BLd] Tim Berners-Lee. Standards, Semantics and Survival 2003. W3C Website. https://www.w3.org/2003/Talks/01-siia-tbl/slide18-0.html. Online; accessed 2017-09-28.

[BLe] Tim Berners-Lee. WWW 2005 Keynote. W3C Website. https://www.w3.org/2005/Talks/0511-keynote-tbl/. Online; accessed 2017-09-28.

[BLf] Tim Berners-Lee. WWW Past and Future 2003. W3C Website. https://www.w3.org/2003/Talks/0922-rsoc-tbl/slide30-0.html. Online; accessed 2017-09-28.

[BLHL+01] Tim Berners-Lee, James Hendler, Ora Lassila, et al. The semantic web. *Scientific american*, 284(5):28–37, 2001.

[BMC+04] Paul V Biron, Ashok Malhotra, World Wide Web Consortium, et al. Xml schema part 2: Datatypes, 2004.

[BMC+12] Paul V Biron, Ashok Malhotra, World Wide Web Consortium, et al. W3c xml schema definition language (xsd) 1.1 part 2: Datatypes, 2012.

[BMR01] Philip A. Bernstein, Jayant Madhavan, and Erhard Rahm. Generic schema matching with cupid. Technical Report MSR-TR-2001-58, Microsoft Research, August 2001.

[Bou15] Dmitry Boulytchev. Combinators and type-driven transformers in objective caml. *Science of Computer Programming*, 114:57 – 73, 2015.

[BPSM+97] Tim Bray, Jean Paoli, C Michael Sperberg-McQueen, Eve Maler, and François Yergeau. Extensible markup language (xml). *World Wide Web Journal*, 2(4):27–66, 1997.

[BPSM+08] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, and François Yergeau. Extensible Markup Language (XML) 1.0. `https://www.w3.org/TR/REC-xml/#dt-doctype`, 2008. Online; accessed 2017-09-28.

[BWJ06] Claudio Bettini, Xiaoyang Sean Wang, and Sushil Jajodia. The role of quasi-identifiers in k-anonymity revisited. *CoRR*, abs/cs/0611035, 2006.

[CAS09] Isabel F. Cruz, Flavio Palandri Antonelli, and Cosmin Stroe. Agreementmaker: Efficient matching for large real-world schemas and ontologies. *Proc. VLDB Endow.*, 2(2):1586–1589, August 2009.

[Chi02] Boris Chidlovskii. Schema extraction from xml collections. In *Proceedings of the 2Nd ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '02, pages 291–292, New York, NY, USA, 2002. ACM.

[CKR+11] Sean Chester, Bruce Kapron, Ganesh Ramesh, Gautam Srivastava, Alex Thomo, and S. Venkatesh. k-anonymization of social networks by vertex addition. In *IN PROC. 15TH ADBIS (2), VOLUME 789 OF CEUR WORKSHOP PROCEEDINGS*, pages 107–116, 2011.

[CT08] Alina Campan and Traian Marius Truta. A clustering approach for data and structural anonymity in social networks, 2008.

[CWL14] Richard Cyganiak, David Wood, and Markus Lanthaler. RDF 1.1 Concepts and Abstract Syntax. Technical report, 2014. Online; accessed 2016-12-06.

[DAKCC17] Irvin Dongo, Firas Al Khalil, Richard Chbeir, and Yudith Cardinale. *Semantic Web Datatype Similarity: Towards Better RDF Document Matching*, pages 189–205. Springer International Publishing, Cham, 2017.

[DB] R.V. Guha Dan Brickley. RDF Schema 1.1. `https://www.w3.org/TR/rdf-schema/`. Online; accessed 2016-12-06.

[DCAKC17] Irvin Dongo, Yudith Cardinale, Firas Al-Khalil, and Richard Chbeir. *Semantic Web Datatype Inference: Towards Better RDF Matching*, pages 57–74. Springer International Publishing, Cham, 2017.

[DCC18] Irvin Dongo, Yudith Cardinale, and Richard Chbeir. Rdf-f: Rdf datatype inferring framework. *Data Science and Engineering*, 3(2):115–135, Jun 2018.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. *Calibrating Noise to Sensitivity in Private Data Analysis*, pages 265–284. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[DMVH+00] Stefan Decker, Sergey Melnik, Frank Van Harmelen, Dieter Fensel, Michel Klein, Jeen Broekstra, Michael Erdmann, and Ian Horrocks. The semantic web: The roles of xml and rdf. *IEEE Internet computing*, 4(5):63–73, 2000.

[DR02] Hong-Hai Do and Erhard Rahm. Coma: A system for flexible combination of schema matching approaches. In *Proc. of the 28th International Conference on Very Large Data Bases*, pages 610–621, 2002.

[DS04a] Martin Duerst and Michael Suignard. Internationalized Resource Identifiers (IRIs). Technical report, Microsoft Corporation, 2004.

[DS04b] Martin Dürst and Michel Suignard. Internationalized resource identifiers (iris). Technical report, 2004.

[DSB+04] Mike Dean, Guus Schreiber, Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L McGuinness, Peter F Patel-Schneider, and Lynn Andrea Stein. Owl web ontology language reference. *W3C Recommendation February*, 10, 2004.

[ES07] Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching*, volume 18. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[EYO08] Zahra Eidoon, Nasser Yazdani, and Farhad Oroumchian. Ontology matching using vector space. In *European Conference on Information Retrieval*, pages 472–481. Springer, 2008.

[FFF99] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. *SIGCOMM Comput. Commun. Rev.*, 29(4):251–262, August 1999.

[FGM+99] Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. Hypertext transfer protocol–http/1.1. `http://www.hjp.at/doc/rfc/rfc2616.html`, 1999. [Accessed 10-09-2017].

[FP06] Matthew Fluet and Riccardo Pucella. Practical datatype specializations with phantom types and recursion schemes. *Electronic Notes in Theoretical Computer Science*, 148(2):211 – 237, 2006.

[fre11] Free Formatter - Free Online Tools For Developers. `https://www.freeformatter.com/xsd-genearator.html`, 2011. Online; accessed 2017-05-03.

[FW04] David C Fallside and Priscilla Walmsley. Xml schema part 0: primer second edition. *W3C recommendation*, 16, 2004.

[GLLW17] Y. Gao, T. Luo, J. Li, and C. Wang. Research on k anonymity algorithm based on association analysis of data utility. In *2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pages 426–432, 2017.

[GTSC16] Kalpa Gunaratna, Krishnaprasad Thirunarayan, Amit Sheth, and Gong Cheng. Gleaning types for literals in rdf triples with application to entity summarization. In *Proc. of the 13th International Conference on The SW.*, pages 85–100, NY, USA, 2016.

[GVdMB08] Aurona Gerber, Alta Van der Merwe, and Andries Barnard. A functional semantic web architecture. *The Semantic Web: Research and Applications*, pages 273–287, 2008.

[HA09] Md. Seddiqui Hanif and Masaki Aono. An efficient and scalable algorithm for segmented alignment of ontologies of arbitrary size. *J. Web Sem.*, 7(4):344–356, 2009.

[HDP12] Michael Hausenblas, Li Ding, and Vassilios Peristeras. Linked open government data. *IEEE Intelligent Systems*, 27:11–15, 2012.

[HHD17] B Heitmann, Felix Hermsen, and S Decker. k- rdf-neighbourhood anonymity: Combining structural and attribute-based anonymisation for linked data. In *5th Workshop on Society, Privacy and the Semantic Web–Policy and Technology (PrivOn2017)(PrivOn), C. Brewster, M. Cheatham, M. d'Aquin, S. Decker and S. Kirrane, eds, CEUR Workshop Proceedings, Aachen*, 2017.

[HMJ+08] Michael Hay, Gerome Miklau, David Jensen, Don Towsley, and Philipp Weis. Resisting structural re-identification in anonymized social networks. *Proc. VLDB Endow.*, 1(1):102–114, August 2008.

[HMS07] T. Hong-Minh and D. Smith. Hierarchical approach for datatype matching in xml schemas. In *24th British National Conference on*, pages 120–129, 2007.

[HNW06] Jan Hegewald, Felix Naumann, and Melanie Weis. Xstruct: Efficient schema extraction from multiple and large xml documents. In *Proc. of the 22Nd International Conference on Data Engineering Workshops*, pages 81–, Washington, DC, USA, 2006.

[Hol13] Stefan Holdermans. Random testing of purely functional abstract datatypes: Guidelines for dealing with operation invariance. In *Proc. of the 15th Sympo-*

sium on Principles and Practice of Declarative Programming, pages 275–284. ACM, 2013.

[HQC08] Wei Hu, Yuzhong Qu, and Gong Cheng. Matching large ontologies: A divide-and-conquer approach. *Data Knowl. Eng.*, 67(1):140–160, October 2008.

[HRMS10] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. Boosting the accuracy of differentially private histograms through consistency. *Proc. VLDB Endow.*, 3(1-2):1021–1032, September 2010.

[HYY08] Jianmin Han, Huiqun Yu, and Juan Yu. An improved l-diversity model for numerical sensitive attributes. In *Communications and Networking in China, 2008. ChinaCom 2008. Third International Conference on*, pages 938–943. IEEE, 2008.

[JC97] Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc. of 10th International Conference on Research in Computational Linguistics*, 1997.

[JJC06] Jeff Z. Pan Jeremy J. Carroll. XML Schema Datatypes in RDF and OWL, W3C Working Group Note 14 March 2006. `https://www.w3.org/TR/swbp-xsch-datatypes/#sec-values`, 2006. Online; accessed 2016-12-06.

[JLD15] Shangpu Jiang, Daniel Lowd, and Dejing Dou. Ontology matching with knowledge rules. *CoRR*, abs/1507.03097, 2015.

[JMSK09] Yves R. Jean-Mary, E. Patrick Shironoshita, and Mansur R. Kabuka. Ontology matching with semantic verification. *Web Semant.*, 7(3):235–251, September 2009.

[KMK15] Kenza Kellou-Menouer and Zoubida Kedad. Discovering types in rdf datasets. In *European Semantic Web Conference*, pages 77–81. Springer, 2015.

[Lew07] Rhys Lewis. Dereferencing http uris. *Draft Tag Finding (May 31, 2007 (retrieved July 25, 2007), http://www. w3. org/2001/tag/doc/httpRange-14/2007-05-31/HttpRange-14. html*, 2007.

[LHLZ15] B. Liu, K. Huang, J. Li, and M. Zhou. An incremental and distributed inference method for large-scale ontologies based on mapreduce paradigm. *Transac. on Cybernetics*, 45(1):53–64, 2015.

[LLV07] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 106–115. IEEE, 2007.

[LÖ09] Ling Liu and M Tamer Özsu. *Encyclopedia of database systems*, volume 6. Springer Berlin, Heidelberg, Germany, 2009.

[LSWC98] Ora Lassila, Ralph R. Swick, World Wide, and Web Consortium. Resource description framework (rdf) model and syntax specification, 1998.

[LT06] Patrick Lambrix and He Tan. Sambo-a system for aligning and merging biomedical ontologies. *Web Semant.*, 4(3):196–206, September 2006.

[LT08] Kun Liu and Evimaria Terzi. Towards identity anonymization on graphs. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 93–106, New York, NY, USA, 2008. ACM.

[LTLL09] Juanzi Li, Jie Tang, Yi Li, and Qiong Luo. Rimom: A dynamic multistrategy ontology alignment framework. *IEEE Transactions on Knowledge and Data Engineering*, 21(8):1218–1232, August 2009.

[LWLZ08] Lian Liu, Jie Wang, Jinze Liu, and Jun Zhang. Privacy preserving in social networks against sensitive edge disclosure. Technical report, Technical Report Technical Report CMIDA-HiPSCCS 006-08, Department of Computer Science, University of Kentucky, KY, 2008.

[MAL+15] Lauri Mukkala, Jukka Arvo, Teijo Lehtonen, Timo Knuutila, et al. Current state of ontology matching. a survey of ontology and schema matching. 2015.

[MBLF05] Larry Masinter, Tim Berners-Lee, and Roy T Fielding. Uniform resource identifier (uri): Generic syntax. 2005.

[MCFY11] Noman Mohammed, Rui Chen, Benjamin C.M. Fung, and Philip S. Yu. Differentially private data release for data mining. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 493–501, New York, NY, USA, 2011. ACM.

[MDG14] Maria Laura Maag, Ludovic Denoyer, and Patrick Gallinari. Graph anonymization using machine learning. In *Advanced Information Networking and Applications (AINA), 2014 IEEE 28th International Conference on*, pages 1111–1118. IEEE, 2014.

[MGKV06] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*, pages 24–24. IEEE, 2006.

[Mic] Microsoft. Xml Schema Inference - Developer Network. `https://msdn.microsoft.com/en-us/library/system.xml.schema.xmlschemainference.aspx`. Online; accessed 2017-05-03.

[MJK09] Amirreza Masoumzadeh, James Joshi, and Hassan A. Karimi. Lbs (k, t)-anonymity: A spatio-temporal approach to anonymity for location-based service users. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '09, pages 464–467, New York, NY, USA, 2009. ACM.

[ML05] Youngsong Mun and Hyewon K Lee. Domain name system (dns). *Understanding IPv6*, pages 151–172, 2005.

[Moa97] Ryan Moats. Urn syntax. Technical report, 1997.

[MS14] Gregg Kellogg-Markus Lanthaler Niklas Lindström Manu Sporny, Dave Longley. JSON-LD 1.0, A JSON-based Serialization for Linked Data, W3C Recommendation 16 January 2014. `https://www.w3.org/TR/json-ld/`, 2014. Online; accessed 2017-10-27.

[NB16] DuyHoa Ngo and Zohra Bellahsene. Overview of yam++(not) yet another matcher for ontology alignment task. *Web Semantics: Science, Services and Agents on the WWW*, 41:30 – 49, 2016.

[NJ02] Andrew Nierman and H. V. Jagadish. Evaluating structural similarity in XML documents. In Mary F. Fernandez and Yannis Papakonstantinou, editors, *Proceedings of the Fifth International Workshop on the Web and Databases, WebDB 2002*, pages 61–66. University of California, 2002.

[NT07] Richi Nayak and Tien Tran. A progressive clustering algorithm to group the xml data by structural and semantic similarity. *International Journal of Pattern Recognition and Artificial Intelligence*, 21(04):723–743, 2007.

[NX04] Richi Nayak and Fu Bo Xia. Automatic integration of heterogenous xml-schemas. In *The Sixth International Conference on Information Integration and Web Based Applications & Services*, Jakarta, Indonesia, 2004.

[OTSO17] Keiichiro Oishi, Yasuyuki Tahara, Yuichi Sei, and Akihiko Ohsuga. Proposal of l-diversity algorithm considering distance between sensitive attribute values. *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8, 2017.

[PB13] Heiko Paulheim and Christian Bizer. Type inference on noisy rdf data. In *International Semantic Web Conference*, pages 510–525. Springer, 2013.

[PHHD10] Axel Polleres, Aidan Hogan, Andreas Harth, and Stefan Decker. Can we ever catch up with the web? *Semantic Web*, 1(1, 2):45–52, 2010.

[PJH14] Peter F. Patel-Schneider Patrick J. Hayes. RDF 1.1 Semantics, W3C Recommendation 25 February 2014. `https://www.w3.org/TR/rdf11-mt/#literals-and-datatypes`, 2014. Online; accessed 2016-12-06.

[PVB04] Ashok Malhotra Paul V. Biron. XML Schema Part 2: Datatypes Second Edition, W3C Recommendation 28 October 2004. `https://www.w3.org/TR/xmlschema-2/#built-in-datatypes`, 2004. Online; accessed 2016-12-06.

[RGCGP15] Filip Radulovic, Raúl García-Castro, and Asunción Gómez-Pérez. Towards the anonymisation of rdf data. In *SEKE*, 2015.

[RKKT13] Jyothsna Rachapalli, Vaibhav Khadilkar, Murat Kantarcioglu, and Bhavani Thuraisingham. Redact: A framework for sanitizing rdf data. In *Proceedings of the 22Nd International Conference on World Wide Web*, WWW '13 Companion, pages 157–158, New York, NY, USA, 2013. ACM.

[RKKT14a] Jyothsna Rachapalli, Vaibhav Khadilkar, Murat Kantarcioglu, and Bhavani Thuraisingham. Rdf-x: A language for sanitizing rdf graphs. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, pages 363–364, New York, NY, USA, 2014. ACM.

[RKKT14b] Jyothsna Rachapalli, Vaibhav Khadilkar, Murat Kantarcioglu, and Bhavani Thuraisingham. Redaction based rdf access control language. In *Proceedings of the 19th ACM Symposium on Access Control Models and Technologies*, SACMAT '14, pages 177–180, New York, NY, USA, 2014. ACM.

[RKKT14c] Jyothsna Rachapalli, Vaibhav Khadilkar, Murat Kantarcioglu, and Bhavani Thuraisingham. Towards fine grained rdf access control. In *Proceedings of the 19th ACM Symposium on Access Control Models and Technologies*, SACMAT '14, pages 165–176, New York, NY, USA, 2014. ACM.

[RLHJ+99] Dave Raggett, Arnaud Le Hors, Ian Jacobs, et al. Html 4.01 specification. *W3C recommendation*, 24, 1999.

[Sam01] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Trans. on Knowl. and Data Eng.*, 13(6):1010–1027, November 2001.

[SFJ15] Jennifer Sleeman, Tim Finin, and Anupam Joshi. Entity type recognition for heterogeneous semantic graphs. *AI Magazine*, 36(1):75–86, 2015.

[SH01] Phil Archer Sandro Hawke, Ivan Herman. W3C Semantic Web Activity. `https://www.w3c.org/2001/sw/`, 2001. Online; accessed 2016-12-06.

[SI12] Dipalee Shah and Rajesh Ingle. Privacy-preserving deletion to generalization-based anonymous database. In *Proceedings of the CUBE International Information Technology Conference*, CUBE '12, pages 459–463, New York, NY, USA, 2012. ACM.

[SLB+17] Rôney Reis C. Silva, Bruno C. Leal, Felipe T. Brito, Vânia M. P. Vidal, and Javam C. Machado. A differentially private approach for querying rdf data of social networks. In *Proceedings of the 21st International Database Engineering & Applications Symposium*, IDEAS 2017, pages 74–81, New York, NY, USA, 2017. ACM.

[SO14] Yuichi Sei and Akihiko Ohsuga. Randomized addition of sensitive attributes for l-diversity. *2014 11th International Conference on Security and Cryptography (SECRYPT)*, pages 1–11, 2014.

[SOTO17] Yuichi Sei, Hiroshi Okumura, Takao Takenouchi, and Akihiko Ohsuga. Anonymization of sensitive quasi-identifiers for l-diversity and t-closeness. *IEEE Transactions on Dependable and Secure Computing*, 2017.

[SS98] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, 1998.

[SSK05] Giorgos Stoilos, Giorgos Stamou, and Stefanos Kollias. A string metric for ontology alignment. In *Proc. of the 4th International Conference on The SW*, pages 624–637, 2005.

[Sta09] W3C Standards. Help and faq. `https://www.w3.org/Help/#webinternet`, 2009. [Accessed 21-09-2017].

[SYLL12] Moonshik Shin, Sunyong Yoo, Kwang H Lee, and Doheon Lee. Electronic medical records privacy preservation through k-anonymity clustering method. In *Soft Computing and Intelligent Systems (SCIS) and 13th International Symposium on Advanced Intelligent Systems (ISIS), 2012 Joint 6th International Conference on*, pages 1119–1124. IEEE, 2012.

[TBM+12] Henry S Thompson, David Beech, Murray Maloney, Noah Mendelsohn, et al. W3c xml schema definition language (xsd) 1.1 part 1: Structures. `https://www.w3.org/TR/xmlschema11-1/`, 2012. Online; accessed 2017-09-28.

[THTC+15] Regina Ticona-Herrera, Joe Tekli, Richard Chbeir, Sébastien Laborie, Irvin Dongo, and Renato Guzman. *Toward RDF Normalization*, pages 261–275. Springer International Publishing, Cham, 2015.

[TLL13] Pham Thu Thuy, Young-Koo Lee, and Sungyoung Lee. Semantic and structural similarities between xml schemas for integration of ubiquitous healthcare data. *Personal Ubiquitous Comput.*, 17(7):1331–1339, October 2013.

[TN10] Huynh Quyet Thang and Vo Sy Nam. Xml schema automatic matching solution. *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 4(3):456 – 462, 2010.

[UOVH09] Jacopo Urbani, Eyal Oren, and Frank Van Harmelen. Rdfs/owl reasoning using the mapreduce framework. *Science*, pages 1–87, 2009.

[VdV03] Eric Van der Vlist. *Relax NG: A Simpler Schema Language for XML*. " O'Reilly Media, Inc.", 2003.

[WF94] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[WGMH13] Meng Wang, Jeremy Gibbons, Kazutaka Matsuda, and Zhenjiang Hu. Refactoring pattern matching. *Science of Computer Programming*, 78(11):2216 – 2242, 2013.

[XML10] XML Grid - Online XML Editor. `http://xmlgrid.net/xml2xsd.html`, 2010. Online; accessed 2017-05-03.

[YCYY13] Mingxuan Yuan, Lei Chen, Philip S. Yu, and Ting Yu. Protecting sensitive labels in social network data anonymization. *IEEE Trans. on Knowl. and Data Eng.*, 25(3):633–647, March 2013.

[YLZY13] Gaoming Yang, Jingzhao Li, Shunxiang Zhang, and Li Yu. An enhanced l-diversity privacy preservation. In *Fuzzy Systems and Knowledge Discovery (FSKD), 2013 10th International Conference on*, pages 1115–1120. IEEE, 2013.

[YW08] Xiaowei Ying and Xintao Wu. Randomizing social networks: a spectrum preserving approach. In *SDM*, pages 739–750. SIAM, 2008.

[ZP08] Bin Zhou and Jian Pei. Preserving privacy in social networks against neighborhood attacks. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, ICDE '08, pages 506–515, Washington, DC, USA, 2008. IEEE Computer Society.

[ZZYY14] Jianpei Zhang, Ying Zhao, Yue Yang, and Jing Yang. A k-anonymity clustering algorithm based on the information entropy. In *Computer Supported Cooperative Work in Design (CSCWD), Proceedings of the 2014 IEEE 18th International Conference on*, pages 319–324. IEEE, 2014.

Appendix A

Appendix

A.1 Introduction

Avec les progrès des techniques du Web sémantique, actuellement une énorme quantité de données est disponible sur Internet. Ces données sont collectées et publiées par différentes sources (par exemple, les entreprises, les gouvernements) à de nombreuses fins, telles que des services, des statistiques, des tests, de la recherche, etc. Le Web sémantique permet l'intégration et la combinaison de ces données en fournissant des modèles standards tel que RDF et OWL [SH01].

Selon [HDP12], plus en plus de gouvernements deviennent des gouvernements électroniques, puisqu'ils font partie des initiatives Linked Open Data, fournissant leurs données pour une intégration des données plus flexible, augmentant la qualité des données et fournissant de nouveaux services et une réduction des coûts. Linked Data est un ensemble de bonnes pratiques pour la publication et la connexion de données structurées sur le Web [BHB09]. L'idée est de rendre les données lisibles pour les humains et les machines, en adoptant des formats spéciaux et en les connectant en utilisant des identifiants de ressources internationales (IRI), qui sont des abstractions de ressources réelles du monde.

Cependant, vu que davantage de données sont publiées et partagées, des informations sensibles telles que des maladies, des salaires ou des comptes bancaires sont également fournies et, par conséquent, compromettent la vie privée des entités (patients, utilisateurs, entreprises). Ainsi, pour protéger la vie privée des entités principales, il est nécessaire d'identifier, dans les données publiées, les informations permettant de découvrir directement ou indirectement la relation entre les entités principales et les informations sensibles.

Dans un environnement interne (par exemple, une entreprise particulière), l'accès aux données est limité aux utilisateurs disposant de droits appropriés, grâce à des techniques de contrôle d'accès [BSB + 05]. En revanche, le Web est une plate-forme ouverte, où tout utilisateur a accès sans aucun contrôle ou privilège. Ainsi, dans le contexte des données disponibles sur le Web, d'autres techniques sont nécessaires pour gérer les problèmes de confidentialité. De plus, il est nécessaire de garder la disponibilité et l'utilité de l'information, tout en préservant la vie privée. Ensuite, trois aspects principaux sont considérés pour assurer la protection de l'entité :

1. Préserver la confidentialité, en identifiant et en traitant les données qui peuvent compromettre la confidentialité des entités (par exemple, les identifiants);

2. Identifier l'utilité des données publiques pour diverses applications (par exemple, statistiques, tests, recherches);

3. Connaître les connaissances de base qui peuvent être utilisées par les adversaires (par exemple, le nombre de relations, une relation spécifique, l'information d'un nœud). Les connaissances antérieures peuvent être facilement obtenues, en raison de l'accès gratuit à l'information sur le Web et l'utilisation des IRI, ce qui permet des connexions entre différentes données qui peuvent compromettre la confidentialité des entités.

Selon [RGCGP15], l'anonymisation est une technique de protection de la vie privée qui a été appliquée avec succès en condition réelle. Il consiste à protéger les entités principales en supprimant ou en modifiant les informations identifiables pour les rendre anonymes, tout en conservant l'utilité des données. Ces données sont modifiées en fonction de certains critères des valeurs existantes (par exemple, taxonomies) pour satisfaire aux conditions d'anonymat.

L'anonymisation dans les bases de données a été bien étudiée et plusieurs propositions sont disponibles dans la littérature [AFK + 06, XWP + 06, LDR06, KG06, BS08, CJT13, MKM13]; Cependant, ces techniques ne sont pas directement applicables à RDF, car il s'agit d'un modèle plus complexe, dont les données sont représentées sous forme d'un graphe orienté, avec des informations sur les nœuds et les arêtes qui ont des propriétés et des restrictions. Cela signifie que les relations entre les ressources ont des significations et peuvent être utilisées pour déduire des données supplémentaires. Par exemple, une propriété `dbo:works` qui lie un créateur à une peinture, permet de déduire que le créateur est un `dbo:painter`, alors que la même propriété liant un créateur à un livre, fait

inférer que le créateur est un `dbo:writer`. De plus, les IRI sont des ressources internationales et uniques, donc ils gardent leurs propriétés dans toutes les données, même si elles proviennent de différentes sources.

Actuellement, dans de nombreux domaines (réseaux sociaux, données de communication, traces de mobilité et graphiques Web), les informations sont générées, partagées et fournies au public, à la communauté de recherche et aux partenaires commerciaux sous forme de données graphiques [JMB17]. Par conséquent, les techniques d'anonymisation appliquées aux structures de graphes deviennent populaires de nos jours. Le problème avec ces approches de graphes d'anonymisations est que des solutions sont proposées pour modéliser des hypothèses particulières de la structure du graphe et des connaissances antérieures, qui sont vraiment restrictives pour les applications RDF. Dans les graphes classiques, il existe un seul type de nœuds, tandis que dans RDF, il existe différents types de nœuds (par exemple, IRI, nœuds vides et nœuds littéraux). Dans les graphes RDF, il existe des données dans les nœuds ainsi que dans les arêtes, et toutes sortes de données peuvent être présentées (c'est-à-dire, des identifiants, des quasi-identifiants et des données sensibles).

Par conséquent, plusieurs techniques d'anonymisation des graphes doivent être combinées (par exemple, graphes non dirigés [LT08a, YW08a, CT08, BDK07, ZP08a, HMJ + 08a, ZG08, ZP11, CSYZ08], graphes de vertex étiquetés [BDK07, ZP08a, HMJ + 08a, ZG08 , ZP11, CSYZ08]) afin de couvrir la structure et les besoins RDF. De plus, RDF est plus sensible à la perte d'informations, puisque le masquage des nœuds, des arêtes et la modification de la structure du graphe compromettent considérablement la certitude de l'inférence.

L'anonymisation dans les données RDF n'a pas été bien étudiée jusqu'à présent. L'étude récente [RGCGP15], est un travail initial de protection des individus sur les données RDF, car elle montre une approche pratique d'anonymisation pour des scénarios simples comme l'utilisation d'opérations de généralisation et de suppression basées sur des hiérarchies. Mais, elle est encore inadéquate pour les scénarios complexes, où une énorme quantité de données et de connaissances antérieures sont utilisées, et pour la protection des informations sensibles.

Pour résoudre ces limitations, nous proposons comme contribution principale un cadre appelé RiAiR (Réduction, intersection et anonymisation dans RDF), indépendant des formats et des fournisseurs de sérialisation. Notre processus de protection repose principalement sur une approche en quatre phases dans laquelle l'entrée est convertie en une représentation graphique utilisée par tous les modules: (i) Phase de réduction de la

complexité dans laquelle le graphique est analysé afin de réduire sa structure de complexité pour en extraire un comprimé; (ii) phase d'intersection, où des nœuds similaires entre le graphe d'entrée (réduit ou non) issu des données à publier et celui issu des connaissances de base sont identifiés en tant que clés potentielles (identifiants et quasi-identifiants); (iii) Sélection de la phase dans laquelle l'utilisateur expert analyse et sélectionne les sources d'informations qui contiennent au moins une clé potentielle. et (iv) un phaset de protection qui exécute un processus de protection sur les triples sélectionnés. La proposition est conçue pour les documents RDF, en tenant compte de leurs éléments (IRI, nœuds vides, littéraux) et du scénario dans lequel une quantité énorme d'informations est disponible. La complexité de la structure RDF est réduite afin de rendre possible la tâche de classification et de suggérer des sources de divulgation potentielles à l'utilisateur expert, diminuant ainsi son interaction. De plus, par une méthode de généralisation, les connexions entre les jeux de données sont réduites, en préservant les objectifs principaux de SemanticWeb (intégration et combinaison) et en protégeant les informations sensibles au même moment.

Comme les phases de réduction et d'intersection sont basées sur une fonction de similarité entre les ressources RDF, certaines limitations liées à la comparaison entre les nœuds littéraux ont été trouvées et étudiées. Par exemple, les types de données, qui sont associés aux littéraux, peuvent représenter les mêmes informations dans plusieurs formats en fonction de vocabulaires différents (par exemple, une valeur littérale 16.0 peut être flottante ou double). De plus, une quantité énorme de documents RDF est incomplète ou incohérente en termes de types de données [PHHD10]. Ainsi, nous proposons une nouvelle hiérarchie de types de données basée sur celle proposée par le W3C, ainsi qu'une mesure permettant d'obtenir des valeurs de similarité entre différents types de données. De plus, un processus d'inférence est également proposé pour fournir les types de données aux nœuds littéraux et effectuer la similarité.

A.2 Contributions à la recherche

Nous présentons les contributions suivantes dans cette thèse :

A.2.1 Analyse et similitude de type de données

Le RDF adopte les types de données du XML définis par le W3C; cependant, la hiérarchie actuelle ne capture pas correctement toute relation sémantiquement significative entre les types de données. Par exemple, les types de données dateTime et time sont mises

au même niveau dans la hiérarchie W3C. Ainsi, nous analysons les types de données dans le contexte des documents de correspondance/intégration RDF, puisque toutes les informations sont utilisées pour découvrir des données similaires. De plus, les mesures de similarité pour les types de données ne sont pas adéquates pour le Web sémantique, car elles sont trop restrictives (même type de données, la similarité est 1, sinon 0) ou basées sur des caractéristiques spécifiques de XML et XSD (par exemple, constraint facets). Afin d'effectuer une étude des types de données pour le Web sémantique, nous fournissons :

- Une analyse de la hiérarchie actuelle des types de données du W3C, ses limites et son applicabilité adéquate pour le Web sémantique.
- Une version étendue de la hiérarchie des types de données W3C, où une relation parent-enfant exprime la subsomption (parent subsume les enfants), ce qui en fait une taxonomie des types de données.
- Une nouvelle mesure de similarité : étendre les travaux de pointe pour prendre en compte plusieurs aspects liés aux nouvelles relations hiérarchiques entre les types de données comparés, tels que: la distance et la profondeur entre les types de données, les enfants similaires, etc.

A.2.2 Inférence du type de données

Les types de données ne sont pas toujours présents dans les données et selon [ANS09a], la présence d'informations de type de données, de contraintes et d'annotations sur un objet améliore la similarité entre deux documents jusqu'à 14%. Par conséquent, une analyse de l'information liée à la valeur, qui n'a pas son type de données respectif, est nécessaire. Une approche capable d'inférer le type de données pour le Web sémantique est fournie, effectuant:

- Une analyse des informations de prédicat, telles que la propriété de plage qui définit et qualifie le type de la valeur de l'objet.
- Une analyse de l'espace lexical de la valeur de l'objet, par un processus d'appariement de formes.
- Une analyse sémantique du prédicat et de son contexte sémantique, qui consiste à identifier des mots apparentés ou des synonymes pouvant désambiguïser deux types de données ayant un espace lexical similaire.

- Une généralisation des types de données numériques et binaires, pour assurer une intégration possible entre les documents RDF.

- En outre, un prototype en ligne appelé RDF2rRDF est également fourni, afin de tester et d'évaluer le processus d'inférence en fonction de la précision et des performances dans le contexte d'une énorme quantité de données RDF.

A.2.3 Anonymisation de documents RDF

Les solutions d'anonymisation existantes dans les bases de données et les graphes ne peuvent pas être directement appliquées aux données RDF, et les solutions RDF sont encore en cours de développement et n'assurent pas une confidentialité suffisante; Nous avons donc proposé:

- Une méthode pour réduire la complexité de la structure RDF des données à publier, simplifiant la tâche d'analyse, effectuée par l'utilisateur expert.

- Une méthode pour suggérer des sources de divulgation à l'utilisateur expert, basée sur la similarité de nœud, réduisant la tâche de classification des données.

- Une opération de protection, basée sur une méthode de généralisation, visant à réduire les relations entre les ressources de différents jeux de données, afin de préserver les objectifs principaux d'intégration et de combinaison du Web sémantique.

A.3 Structure du Manuscrit

Nous présentons un aperçu de chacun des chapitres suivants dans cette thèse:

A.3.1 Le Chapitre 2 - Le Web Sémantique: Révision

Présente les informations générales sur les concepts et les principes de WWW, Web Sémantique, RDF et ses définitions respectives pour mieux comprendre le processus d'anonymisation.

A.3.2 Le Chapitre 3 - Le Web Sémantique: Analyse et Similarité de Types de Données

Présente l'importance des types de données pour le Web sémantique et un scénario motivant pour illustrer les limites des approches existantes sur la similarité des types de données. Ce chapitre décrit également notre contribution pour une meilleure similarité de type de données, consistant en une nouvelle hiérarchie de types de données basée sur celle proposée par le W3C, et une nouvelle mesure de similarité prenant en compte la similarité entre enfants. Une évaluation expérimentale pour mesurer l'exactitude de notre proposition est montrée, en ce qui concerne les approches existantes.

A.3.3 Le Chapitre 4 - Le Web Sémantique: Inférence de Type de Données

Décrit notre proposition d'inférence de type de données. Ce chapitre comprend également un scénario de motivation pour montrer comment une intégration inadéquate entre les documents RDF peut se produire si les types de données ne sont pas présents. Une proposition formelle est décrite, consistant en quatre étapes: l'analyse des informations de prédicat, l'analyse de l'espace lexical de type de données, l'analyse sémantique des prédicats et la généralisation des groupes numériques et binaires. Enfin, nous détaillons notre prototype, appelé RDF2rRDF, qui est utilisé pour effectuer des évaluations de précision et de performance, en les comparant aux approches existantes.

A.3.4 Le Chapitre 5 - Le Web Sémantique: Préservation de la Confidentialité

Décrit l'importance de la confidentialité pour le Web sémantique dans les documents RDF. Des concepts et des définitions sur les données d'anonymisation sont présentés pour formaliser la proposition. Un scénario motivant, dans le contexte du domaine des centrale nucléaire, s'avère analyser l'applicabilité des approches existantes et leurs limites. L'approche de protection reposait sur quatre phases: (i) Phase de réduction de complexité dans laquelle le graphique est analysé afin de réduire sa structure de complexité afin d'en extraire une structure comprimée; (ii) Phase d'intersection, où des nœuds similaires entre le graphe d'entrée (réduit ou non) de D et celui de BK sont identifiés comme clés potentielles; (iii) Phase de sélection dans laquelle l'utilisateur expert analyse et sélectionne les sources d'informations, qui contient au moins une clé potentielle; et (iv) la phase de

protection qui exécute un processus de protection sur les triples sélectionnés.

A.3.5 Le Chapitre 6 - Conclusions et travaux futurs

Dans cette thèse, nous avons proposé et évalué un cadre de protection RDF, appelé RiAiR. La proposition est conçue pour les documents RDF, en tenant compte de tous leurs éléments et d'un scénario contenant une quantité énorme d'informations. La complexité de la structure RDF est réduite afin de rendre possible la tâche de classification et de suggérer des sources de divulgation potentielles à l'utilisateur expert, diminuant ainsi son interaction. Par une méthode de généralisation, réduisez les connexions entre les jeux de données pour protéger les données et préserver les objectifs du Web sémantique (intégration et combinaison).

Nous avons étudié plusieurs fonctions de similarité afin d'en fournir la plus adéquate. Une fonction de similarité basée sur le contexte des ressources (propriétés entrantes et sortantes) a été utilisée pour effectuer les expériences. Cependant, dans les similitudes syntaxiques, quelques délimitations ont été trouvées et analysées dans cette thèse.

L'évaluation expérimentale de la précision, de la viabilité et des performances de plusieurs bases de données disponibles sur le Web reflète l'efficacité de notre approche par rapport aux travaux existants. Dans ce chapitre, nous présentons les conclusions de nos travaux et une discussion sur les limites des défis. Nous concluons avec les travaux futurs susceptibles d'étendre la portée de l'approche afin d'assurer une meilleure confidentialité pour le Web sémantique.

Les directions futures des travaux comprennent des améliorations dans la similarité et l'inférence des types de données, la similarité et l'inférence dans les outils de correspondance et mesure de similarité entre ressources. Ces directions sont décrites comme suit:

Types de Données Complexes

Dans les chapitres 3 et 4, nous avons limité la portée de notre étude à des types de données simples. Cependant, l'analyse des types de données peut être étendue à des types de données complexes, car comme nous l'avons mentionné au chapitre 2, les types de données complexes contiennent des éléments définis comme des types de données simples ou complexes. Ainsi, un type de données complexe peut être traité comme un ensemble

de types de données simples dans certains cas. Une moyenne de leurs éléments peut être utilisée pour mesurer la similarité entre les types de données complexes. Dans le cas de l'inférence, pour déduire des types de données complexes, des règles de contexte supplémentaires peuvent être proposées en fonction du type de base de données pour résoudre l'ambiguïté.

Outils de correspondance

Les chapitres 3 et 4 proposent deux approches différentes dans le contexte de correspondance/intégration de documents RDF. Ensuite, ces approches devraient être évaluées dans de vrais outils d'appariement afin de mesurer comment est l'amélioration par rapport aux méthodologies actuelles. Par exemple, selon la classification de travail connexe du chapitre 3, les travaux du groupe 1 ([CAS09, EYO08, ES07, HA09, HQC08, JMSK09, JLD15, LT06, LTLL09, MAL + 15, NB16, SSK05]) utilisent une similarité binaire (la similarité entre les types de données égaux est d'un, sinon la similarité est nulle), si nous remplaçons la similarité existante par celle proposée dans cette thèse, nous pouvons observer l'importance d'une similarité de type de données adéquate. Pour notre approche d'inférence, une évaluation entre une base de données sans types de données, une base de données avec leurs types de données respectifs, et une autre avec les types de données déduits par notre approche, motivera le besoin de types de données pour le Web sémantique ainsi que la contribution de notre proposition.

Inférer des Types de Données sémantiques

Notre approche d'inférence présentée au chapitre 4, se concentre sur des types de données simples pour les nœuds littéraux. Comme nous l'avons décrit dans le chapitre 2, il existe deux types de propriétés (prédicats), propriété d'objet et propriété de type de données (propriété d'entité renommée et propriété de valeur, respectivement), notre proposition peut être considérée comme une approche d'inférence pour le type de données propriétés, car une propriété de type de données est toujours liée à un nœud littéral. Cette propriété est connue dans la littérature sous le nom de type de données Syntactic, car elle décrit le format de la valeur elle-même qui lui est associée. Cependant, les types de données sémantiques (nœuds liés à la propriété de l'objet) sont également présents dans les documents RDF; ainsi, une nouvelle approche capable de les inférer est requise. Le type de données sémantique est un défi complexe, car un nœud (IRI et nœud vide) peut avoir différents types de données sémantiques selon le contexte des données.

Mesure de similarité entre les ressources

La précision de notre approche de protection dépend de la mesure de similarité appliquée dans les phases de réduction et d'intersection. Etant donné que les ensembles de données RDF hétérogènes et liés utilisent plusieurs vocabulaires, des similitudes sémantiques sont nécessaires pour comparer des ressources similaires avec des représentations différentes.