Knowledge Graph Refinement

Link Detection, Link Invalidation, Key Discovery and Data Enrichment

HABILITATION À DIRIGER DES RECHERCHES

(Spécialité Informatique)

UNIVERSITÉ PARIS SUD

présentée et soutenue publiquement le 20 Juin 2019

By

FATIHA SAÏS

Jury

Rapporteurs:	Oscar Corcho	Professor at Universidad Politécnica de Madrid
	Fabien Gandon	Research Director at INRIA Sophia-Antipolis
	Hala Skaf Molli	Associate Professor HDR at Université de Nantes
Examinateurs:	Sarah Cohen-Boulakia	Professor at Paris Sud University
	Juliette Dibie	Professor at AgroParisTech
	Chantal Reynaud-Delaître	Professor at Paris Sud University (Marraine)

Laboratoire de Recherche en Informatique, UMR 8623 CNRS
Université Paris Sud, Bâtiment Ada Lovelace 650, F-91405, Orsay Cedex, France
Contact: tél.: +33 (0)1 69 15 66 29 – Web: http://www.lri.fr

Abstract

This habilitation thesis outlines some methods and tools resulting from my research activities during the last ten years as well as my scientific projects for a near future. These methods and tools have been developed for knowledge graph refinement in the context of Web of data. We are experiencing an unprecedented production of resources published as Linked Open Data (LOD, for short). This has led to the creation of knowledge graphs (KGs) containing billions of RDF (Resource Description Framework) triples, such as DBpedia, YAGO and Wikidata on the academic side, and the Google Knowledge Graph or eBay Knowledge Graph or Facebook Graph on the commercial side. However, building knowledge graphs while ensuring their completeness and correctness, is a challenging endeavour. For this challenging problem, my research contributions have focused on several issues. First, identity link invalidation problem for which we developed two main approaches relying on either the semantics of ontology axioms to detect inconsistency in the KGs or on the network structure of identity links to assign an error degree for every identity link in the LOD. Second, in the settings of scientific KGs, we defined a generic approach for detecting contextual identity links representing a weak identity relation between entities that is valid in an explicit context expressed as a sub-part of the ontology. This approach is a contribution to the overcoming problem of the strict semantics of *owl:sameAs* predicate, that is not required in all application domains. Third, we proposed a data fusion approach that is able to aggregate data coming from different sources and to compute a unique representation for a set of given linked entities. Furthermore, to deal with missing value prediction, we developed an approach that relies on data linking and case-based reasoning to predict missing values. Finally, to enrich the conceptual level of KGs with new key axioms, that are particularly important for detecting identity links, we defined three efficient methods: KD2R, for discovering exact keys, SAKey for discovering n-almost keys and VICKEY for discovering conditional keys. These three methods are based on computing first the maximal non-keys and then deriving the minimal keys, and apply several strategies to prune the search space.

Overall these approaches have been developed in collaboration with several fellow researchers, in the setting of several PhD theses, post-docs and master theses; some of them in the context of ANR, CNRS and industrial research projects, involving different organisms and companies, such as, INRA, INA, ABES, IGN and Thalès.

Keywords: *Knowledge Graphs, Ontology, Identity Management, Data Linking, Link Invalidation, Key Discovery, Data Fusion and Missing Value Prediction*

Dedication and acknowledgements

Here goes the dedication.

Table of Contents

LIST OF FIGURES

List of Tables

Table **Page**

CHAPTER 1

INTRODUCTION

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

– Leonard da Vinci.

1.1 General Context

The idea of feeding intelligent systems and agents with general formalized knowledge of the world dates back to classic Artificial Intelligence (AI) in 1980s [116]. Indeed, according to the *knowledge principle: "if a program is to perform a complex task well, it must know a great deal about the world in which it operates."* [79], the exploitation of knowledge that describes the environment in which AI applications are operating may lead to breaking the locks and thus allow them to reach their full potential.

Today, we are experiencing an unprecedented production of resources, published as Linked Open Data (LOD, for short). This is leading to the creation of knowledge graphs (KGs) containing billions of RDF (Resource Description Framework) [86] triples, such as DBpedia, YAGO and Wikidata on the academic side, and the Google Knowledge Graph or eBay Knowledge Graph on the commercial side. These KGs contain millions of entities (such as people, proteins, or books), and millions of facts about them. These KGs are either domain-specific, like KnowLife [47] for life sciences, DOREMUS [3] for musical works, or domain-independent, like Yago, DBpedia, Wikidata. They contain knowledge that is typically expressed in RDF , i.e., as statements of the form <*subject*, *predicate*, *object*> such as <*Macron, presidentOf, France*>. By proposing RDF as a standard, researchers from Semantic Web community have promoted graph-based representation of knowledge. In such graphs, the nodes represent entities (e.g. Paris) that may have types (e.g.

Paris is a City), the edges represent relations between entities (e.g., hasMayor). Sometimes, the various types and relations are represented in an OWL2 (Web Ontology Language) [97] ontology, which defines their interrelations and axioms such as, subsumption, disjunction and functionality of properties.

The term *knowledge graph* has been proposed by Google referring to the the use of semantic knowledge in Web search ("Things, not strings"). There is no common formal definition of a knowledge graph [45], but what is commonly established is that [102] a KG: describes entities and relations organized in a graph, defines possible classes of entities and relations in a schema (ontology), enables linking arbitrary entities with each other, may cover different domains and applies reasoners to derive new knowledge [102]. Usually, the number of instance-level (A-Box) statements is by several orders of magnitude larger than the number of schema level (T-Box) statements. For example, ontologies like DOLCE that do not contain instances would not be considered as a knowledge graph. Different approaches have been adopted for constructing knowledge graphs: (i) curated KGs like Cyc, (ii) created by the crowd like freebase and Wikidata or (iii) created using information extraction tools from large-scale semi-structured knowledge bases such as wikipedia, like Yago and DBpedia.

Knowledge graphs are acknowledged and more and more used by different applications and services. Web search through search engines like Google or Bing are enhanced in such a way to be able to provide structured and semantic information about topics in addition of lists of links to other web pages and question answering (e.g. "*Who is the mother of Barack Obama*?") is significantly improved thanks to the use of KGs. Personal assistants like Microsoft Cortana and Google assistant are also empowered by semantic knowledge allowing a natural knowledge access and storage, smart chat bots, personalized and coherent dialogues and a better understanding of the user needs based on the context. All these applications have gained in efficiency and quality of service thanks to the use of KGs. Nevertheless, their efficiency is dependent on the quality of the knowledge graphs they use. Building knowledge graphs while ensuring the good quality of their content is a challenging endeavour. Indeed, whatever approach is used for building knowledge graphs we can not guarantee their completeness, i.e. containing all pieces of information about every entity in the universe, neither their correctness. Existing KGs usually attempt to reach a trade-off between completeness and correction [152].

From a technical point of view KG incompleteness and inconsistency have various causes:

- *heterogeneity of original sources*, due to the use of various dataset schemas and vocabularies, namely different entity types, different sets of properties, and different literal values to describe the same entities in the original sources.

- *uncertainty of the original sources*, due to the inherent imprecision of the information acquisition processes and tools, such as information extraction tools, use of sensors and images processing.

- *evolution of data and knowledge*, due to the permanent changes of the real world entities (e.g. objects movements, object transformations) and consequently to the evolution of the data and knowledge describing them.

- *cost of data and knowledge acquisition*, due to the high cost of some scientific experiments which may involve either costly materials or tremendous human effort, acquiring new data and knowledge requires important fundings. Moreover, curated, valuable and critical data and knowledge are often under private licences.

To address these shortcomings, refinement of Knowledge Graphs has to be dealt with. Conversely to *Knowledge Graph Construction* approaches that apply a set of methods to construct the KG from scratch, *Knowledge Graph Refinement* consists in applying a series of methods to improve the coverage or the correctness of an existing KG [102]. In one hand, these methods deal with the expansion and the enrichment of KGs by focusing on one or several KG component(s), namely, literal values, entity links, entity types, ontology axioms and ontology mappings. In the other hand, these methods address the problem of validating the content of the KGs and propose algorithms and models to deal with errors and ambiguities in KGs.

This HDR thesis outlines my contributions to Knowledge Graph Refinement problem for both KG enrichment and validation. The reported results were obtained from 2008 to early 2018, as an Associate professor at Paris Sud University (LRI, UMR CNRS 8623, LaHDAK team) through collaborations with colleagues and students, in particular, three PhD candidates, three post-doctoral fellows and more than twenty master students.

1.2 Contributions

In direct line with the work carried out during my doctoral thesis [117] on data linking I have focused on issues related on both knowledge graph enrichment and knowledge graph validation. All the approaches that I present here have led to the development of software tools that were used to conduct experiments on big and heterogeneous real datasets and benchmarks.

For knowledge graph enrichment, my contributions are the following:

- Identity link detection methods for instance level enrichment of the KG,

- Data fusion and missing value prediction methods for instance level enrichment of the KG,

- Key discovery methods for the conceptual level (T-Box) enrichment of the KG.

For the knowledge graph validation, I investigated the problem of identity link invalidation, i.e. detection of erroneous identity links, for which I developed different approaches.

Identity link detection and invalidation. In this direction I studied the problem of identity in the Web of data, which is raised because of the erroneousness of some identity links published on the Web [64]. These identity links are mostly represented by `owl:sameAs` predicate proposed in OWL language as a standard for representing identity relation. However, its semantics is as strict as the mathematical identity relation, i.e reflexive, symmetric, transitive and fulfils the property sharing rule ($\forall X,Y,Z$ `owl:sameAs`$(X,Y) \wedge p(X,Z) \Rightarrow p(Y,Z)$). Hence, the use of erroneous `owl:sameAs` links with such a strict semantics may lead to infer and propagate errors in the knowledge graph. That is, to address this problem we explored two directions: (i) define approaches to automatically invalidate links, i.e. detect erroneous `owl:sameAs` links and (ii) define a new data linking approach which, instead of detecting so strict `owl:sameAs` links discovers weaker identity links while providing explicit contexts (a sub-part of the ontology) in which two resources are assumed to be identical and preserving reflexivity, symmetry, transitivity and property sharing, of such links that we called *contextual identity links*. To evaluate and validate the proposed approaches we conducted experiments on real datasets, that are the whole Web of Data content for some approaches of identity link invalidation and complex scientific data for the contextual identity detection, which represent for both approaches some of the worst cases to handle.

Data fusion and missing value prediction. For data fusion problem, I developed a multi-criteria approach [58, 122, 124], which given a set of identity links between resources computes a unique representation of the entity referred by these resources. Thus, it uses data quality criteria to compute a confidence degree and selects the best quality property values . To represent the inherent uncertainty I studied several uncertainty models. Finally, to enrich even more knowledge graphs, I investigated the problem of predicting new property values from the existing data and knowledge. That is, I proposed an approach which applies a method, inspired from data reconciliation, to group similar resources and then uses these groups to predict new values. For the evaluation of these methods, I used real datasets from several domains, e.g. life science, scientific publications and bibliographical data.

Key discovery. In order to enrich the ontology and to enhance the knowledge-based data linking approaches I developed different methods for discovering keys (a set of properties that uniquely identify instances) from RDF datasets. I developed, first KD2R [104] which was the first approach that is able to discover keys that are valid in several RDF datasets. Then, I developed SAKey [131], a scalable approach for the discovery of almost keys (i.e. keys with exceptions). After that, I developed the VICKEY [134] method that is able to discover conditional keys, i.e. keys that are valid on a subset of the dataset that fulfil a condition. To allow scalability two datasets of millions of triples, all these methods apply efficient data filtering and search space prunings. Finally, in [11], I proposed a theoretical and experimental comparison of different semantics of

keys that exist in the literature. For the evaluation of these different approaches, experiments on several datasets have been conducted to show both the scalability of the approaches and the quality of the discovered keys.

All the approaches I chosen to present in this HDR form a coherent workflow that when applied as depicted in Figure 1.1 leads to more complete and more consistent knowledge graphs: (1) starting by the discovery of keys, (2) then, the use of these keys for detecting identity links as I proposed in [117, 118], (3) after that an invalidation approach[1] may be applied to detect erroneous identity links and finally, (4) apply a data fusion approach to merge all the considered and linked knowledge graphs and thus obtaining a refined knowledge graph. I believe that the more knowledge graph refinement is iterated the better the quality of the obtained knowledge graphs will be.

1.3 Manuscript Organisation

I chosen to present some of my research work I did in the past ten years, by focusing only on those that are related to knowledge graph refinement, each put in context, explained and discussed.

Chapter 2 (Background). I present the basics of RDF and OWL ontologies, and give formal definitions for knowledge graphs, RDF graphs and identity links.

Chapter 3 (Identity Link Detection and Invalidation). I first present in section 3.1.1 the state of the art on identity management problem. Then I present in section 3.2 my main contributions to the detection of erroneous identity links. Finally, section 3.3 presents my contributions to the problem of detecting contextual identity links in knowledge graphs.

Chapter 4 (Data Enrichment). I first present the related works in data enrichment in section 4.1. Then, I present two main contributions in this area, namely a data fusion approach in section 4.3 and a data reconciliation-based approach for predicting missing property values in a scientific knowledge base, in section 4.4.

Chapter 5 (Key Discovery). After a survey on key discovery problem in Semantic Web that I present in section 5.1, I present in sections 5.2, 5.3 and 5.4 three methods, KD2R, SAKey and VICKEY respectively, that I developed for key discovery in knowledge graphs.

Chapter 6 (Conclusion and Research Perspectives). Finally, I present a summary and my forthcoming work, which mostly corresponds to challenges for enhancing the state-of-the-art on Knowledge Graph Refinement techniques, for enabling improvements in the quality

[1] An approach of contextual identity link detection may be applied at this stage to re-qualify erroneous identity links or in place of stage classical data linking approaches (2).

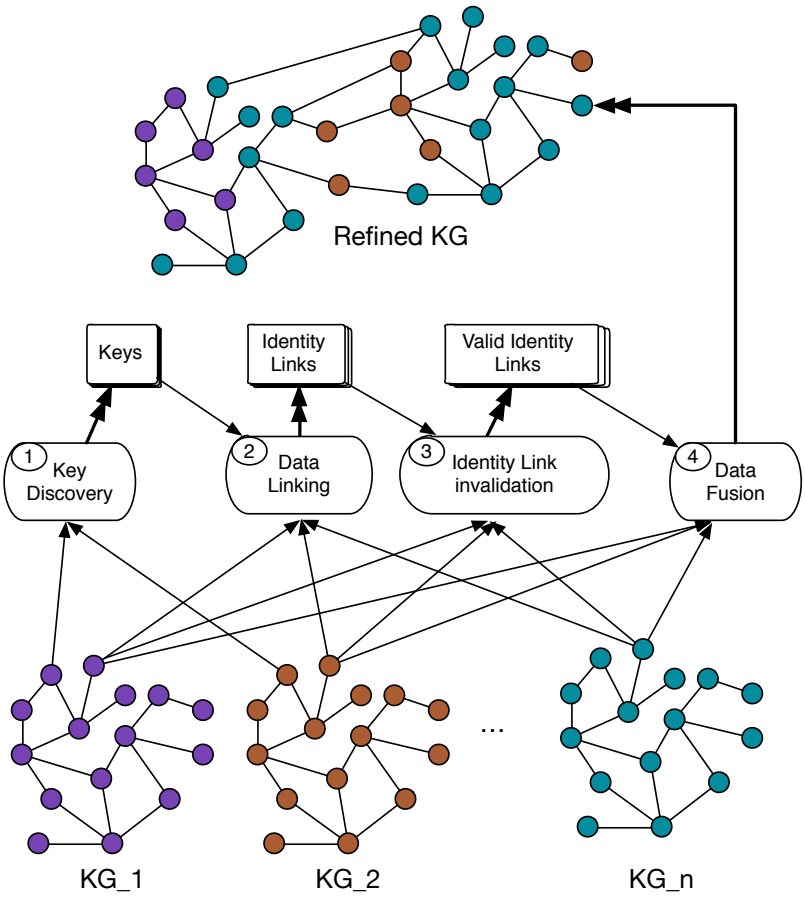


Figure 1.1: General Schema of Knowledge Graph Refinement

of KGs such as their veracity, and allowing new expressive and valuable knowledge to be discovered.

CHAPTER 2

BACKGROUND

This chapter is dedicated to the background notions that are necessary for the description of the bunch of our works that we have selected to be presented in this manuscript. The selected work mainly concerns Knowledge Graphs (KGs) completion. For knowledge representation we rely on the use of semantic Web technologies: (i) **W**eb **O**ntology **L**anguage (OWL) is used to declare and formally represent the knowledge of a given domain and, (ii) **R**esource **D**escription **F**ramework (RDF) is used to uniformly and semantically describe entities (e.g., persons, locations, events). The vocabulary that is used in RDF data may be declared in an OWL ontology. When ontologies are expressive enough, axioms and rules can be declared to represent complex knowledge, such as cardinalities, correlations, negative and positive constraints that are (always) fulfilled in the domain. Thus, thanks to their logical semantics, automated reasoning can be applied to infer new knowledge and/or contradictions in the KGs.

In what follows, we first define RDF graphs. Then we give a definition of ontologies and their axioms (only the ones we exploited) and finally, we give our definition of knowledge graphs.

2.1 Knowledge Graphs

We consider knowledge graph as defined by an ontology $\mathcal{O}$, represented in OWL 2 (Web Ontology Language) [112], its associated dataset $\mathcal{D}$ represented as an RDF (Resource Description Framework) graph composed of a collection of RDF [87] triples, and a set of ontology axioms.

Definition 2.1. (**Knowledge Graph**). A knowledge graph $\mathcal{K}$ is defined by a couple $(\mathcal{O},\mathcal{D})$ where:

- $\mathcal{O} = (\mathcal{C},\mathcal{P},\mathcal{A})$ represents the conceptual knowledge part of the knowledge graph defined by: (i) a set of classes $\mathcal{C}$, (ii) a set of properties $\mathcal{P}$ (*owl*:*DataTypeProperty* and *owl*:*ObjectProperty*) for which the domain and the range are specified, and (iii) a set

of axioms $\mathscr{A}$ declared in the ontology $\mathscr{O}$. These axioms (see Section 2.4) allow representing relations and constraints like subsumption, equivalence and disjunction between classes and properties, (inverse) functionality of properties or key constraints for a given class.

- $\mathscr{D}$ represents the instance level of the knowledge graph. It is expressed through an RDF Graph (see Definition 2.2) which represents a collection of RDF triples $< s,\ p,\ o >$, where the subject s is a URI[1], the property $p \in \mathscr{P}$ is also a URI, and the object o can be either a URI or a Literal (e.g., String, Number, Date). We consider the URIs as belonging to the set of resources R and the set of Literal values as belonging to the set of Literals L. We define a subset I of URIs referring to instances, which corresponds to the set of URIs[2] that appear as an object or a subject in a triple. We note that the triple $< u,\ rdf{:}type,\ c >$ can be declared to state that the URI u, that can appear as a subject or an object, is an instance of the class c. We note that for a given dataset $\mathscr{D}$, we write $p(x,y)$ to mean $< x,\ p,\ y > \in \mathscr{D}$. We also denote $p.V(x)$ to express the value or the set of values of a property p for a resource x.

It is worthwhile to mention that, in this manuscript, we will use the term dataset $\mathscr{D}$ or RDF graph G to refer to the instance level of the knowledge graph.

2.2 RDF Graphs

An RDF *dataset* is a set of triples in the form of <*s, p, o*> that forms the set of *facts* describing certain pieces of knowledge on a given subject. This set of facts can be conceptually represented as an RDF graph that we formally define in Definition 2.2 as follows.

Definition 2.2. (**RDF Graph.**) An RDF graph is a directed and labelled graph $G = (V, E, \Sigma_E, l_E)$. V is the set of nodes. E is the set of node pairs or edges. Σ_E is the set of edge labels. $l_E : E \rightarrow \Sigma_E$ is the mapping from edges to edge labels. $l_E(e)$ denotes the labels of edge e.
We use e_{ij} to denote the edge between nodes v_i and v_j. More precisely, each fact $p(s,o)$ maps to a directed edge e_{ij} where s is represented by the source node v_i, o is represented by the target node v_j, and p is the property name that is represented by the label of e_{ij}.

2.3 OWL Ontologies

In semantic Web the use of ontologies is paramount for knowledge structuring and formalisation. An ontology that is defined as "*An **explicit, formal** specification of a **shared** conceptualization*" by Thomas R. Gruber [62], allows one to represent the vocabulary of a given domain through a

[1]Unique Resource Identifier (URI). Recently, URIs have been generalized to Internationalized Resource Identifier (IRIs) which are Unicode strings that capture more special characters.
[2]In RDF URIs are also used to refer to classes and properties.

set of classes, a set of properties and a set of axioms. The semantics of the knowledge represented in an ontology can be formally expressed by First Order Logic (FOL) formulas. These formulas can be exploited in an automatic logical reasoning to infer new knowledge.

Two languages are recommended by W3C for ontology representation. First, RDF Schema [143], based on RDF, which can be used to represent lightweight ontologies restricted to a hierarchy of classes and a hierarchy of properties. Second, Web Ontology Language (OWL, OWL2 for its second version) [97], that is based on Description Logic and allows one to represent rich and complex knowledge about entities. Thanks to its expressiveness power it can be used to check the consistency of the knowledge.

In most of the work that we present in this manuscript we considered OWL2 ontologies thanks to their ability to represent more expressive knowledge and hence allowing more reasoning capabilities.

An OWL2 ontology is composed of a set of classes representing sets of instances (resources), a set of properties expressing relations between entities (i.e. *owl:ObjectProperty* having a class as domain and as range) or between an entity and a literal value (i.e. *owl:DataTypeProperty* having as domain a class and as range a Literal) and a set of individuals representing the instance-level of the knowledge graph that contains the class instances and property instances declared in the conceptual level. The classes and the properties are organized in a hierarchy using either the relations *owl:subClassOf* or *owl:subPropertyOf*.

In addition to this, a set of axioms can be declared in an OWL2 ontology (see sub-section 2.4).

2.4 Ontology Axioms

OWL2 language allows declaring a set of axioms – statements that express what is true in the considered domain. We consider axioms three kinds of axioms out of eight types provided in OWL2, namely, *class axioms*, *property axioms* and *keys*.

In what follows, we provide the list of ontology axioms that we consider by giving their corresponding logical semantics expressed in First Order Logic (FOL). In what follows we consider a set of variables $x, y, z, ...$ that can take as values class instances (in I) or literals (in L). We use also the predicates $c_1, c_2, ...$ to refer to ontology classes and $op_1, op_2, ..., dp_1, dp_2, ...$ to refer to object properties and data properties, respectively.

Class axioms. We consider three main kinds of axioms that can be set on ontology classes:

- *subsumption* between two classes c_1 and c_2 noted $c_1 \preceq c_2$. It can be expressed using the OWL2 construct *owl:subClassOf*. Its logical semantics is: $\forall x \; c_1(x) \Rightarrow c_2(x)$

- *equivalence* between two classes c_1 and c_2 noted $c_1 \equiv c_2$. It can be expressed using the OWL2 construct *owl:EquivalentClasses*. Its logical semantics is: $\forall x \; c_1(x) \Leftrightarrow c_2(x)$

- *disjunction* between two classes c_1 and c_2 noted $c_1 \not\equiv c_2$. It can be expressed using the OWL2 construct *owl*:*DisjointClasses*. Its logical semantics is: $\forall x c_1(x) \Leftrightarrow \neg c_2(x)$

Data property axioms and Object property axioms. We consider two main kinds of axioms that can be set on ontology properties. We note that, in the formalization, we do not distinguish between Data property axioms and object property axioms:

- *functionality* on a property p denoted by $func(p)$. It can be expressed using the OWL2 construct *owl*:*FunctionalDataProperty* or *owl*:*FunctionalObjectProperty*. Its logical semantics is: $\forall x \forall y \forall z \; p(x,y) \wedge p(x,z) \Rightarrow y = z$

- *inverse functionality* on a property p denoted $inv\text{-}func(p)$. It can be expressed using the OWL2 construct *owl*:*InverseFunctionalDataProperty* or *owl*:*InverseFunctionalObjectProperty*. Its logical semantics is: $\forall x \forall y \forall z \; p(y,x) \wedge p(z,x) \Rightarrow y = z$

- *key* for a given class c and a set $P = \{dp_1, \ldots, dp_n, p_n, op_1, \ldots, op_m\}$ of data properties and/or object properties.

  A generalization of the notion of inverse functionality property is the *key* axiom, which states that for a given class c, a set of properties P uniquely identifies an instance of c. That means, no two distinct (named) instances of c can coincide on the values of P. Unlike, inverse functionality axiom, key axiom needs an additional condition to enforce the considered resources to be named (i.e. the resources must be URIs or literals, but not blank nodes). It can be expressed using the OWL 2 construct *owl*:*hasKey*. If we declare the axiom *owl*:*hasKey* $(c)(dp_1, \ldots, dp_n)(op_1, \ldots, op_m)$ to express that the set of properties $P = \{dp_1, \ldots, dp_n, p_n, op_1, \ldots, op_m\}$ is a key for the class c, its logical semantics is:

$$\left(\forall x, y, \; v_1, \ldots, v_n, u_1, \ldots, u_m \bigwedge_{i=1..n} dp_i(x, v_i) \wedge dp_i(y, v_i)\right) \wedge \left(\bigwedge_{j=1..m} op_j(x, u_j) \wedge op_j(y, u_j)\right) \Rightarrow x = y$$

2.5 Identity Link `owl:sameAs`

Identity links are used to declare that two different descriptions of resources refer to the same real world entity (e.g., same person, same place, same book).

Today, the classical definition of identity has become the canonical one in the Semantic Web (through `owl:sameAs` predicate). Such an `owl:sameAs` statement indicates that two URIs refer to the same $thing$, i.e, the individuals have the same 'identity' [67]. Given an RDF graph G as defined in Definition 2.2, the OWL2 RL rules [67] define the `owl:sameAs` as being reflexive, symmetric, and transitive, and fulfils property sharing axiom as expressed by the rule:

$$\forall XY \; \texttt{owl:sameAs}(X,Y) \wedge p(X,Z) \Rightarrow p(Y,Z)$$

Definition 2.3. SameAs Statement [38]. An owl:sameAs(s,o) statement is an RDF triple $\langle s, \texttt{owl:sameAs}, o \rangle$ in an RDF graph G which connects two RDF resources s and o by means of the *owl:sameAs* predicate.

CHAPTER 3

IDENTITY LINK DETECTION AND INVALIDATION

3.1 State of the Art and Contributions

As the Web of Data continues to grow, more and more large datasets – covering a wide range of topics – are being added to the Linked Open Data (LOD) Cloud. It is inevitable that different datasets, most of which are developed independently of one another, will come to describe (aspects of) the same thing, but will do so by referring to that thing with different names. This situation is not accidental: it is a defining characteristic of the (Semantic) Web that there is no central naming authority that is able to enforce a Unique Name Assumption (UNA). As a consequence, identity link detection, i.e., the ability to determine – with a certain degree of confidence – that two different names in fact denote the same thing, is not a mere luxury but is essential for Linked Data to work. Thanks to identity links, datasets that have been constructed independently of one another are still able to make use of each other's information. The most common predicate that is used for interlinking data on the web is the `owl:sameAs` property (see Definition 2.3). This property denotes a very strict notion of identity that is formalized in model theory. It is defined by Dean et al. [35] as: "an `owl:sameAs` statement indicates that two references actually refer to the same thing". As a result, a statement of the form "x `owl:sameAs` y" indicates that *every* property attributed to x must also be attributed to y, and vice versa.

The problem of detecting identity links has already been studied in several domains, in statistics, relational databases, artificial intelligence and more recently in semantic Web while proposing different approaches probabilistic, logical, similarity-based, and so on (surveys can be found in [52]). In semantic Web several frameworks are now available and are able to handle datasets of a high number of RDF triples with high performance results (see survey [93]) . However, none of them is able to guarantee 100% of precision and 100% of recall for any dataset

of any application domain.

Over time, an increasing number of studies has shown that owl:sameAs is sometimes used incorrectly in practice. For example, Jaffri et al. [75] discussed how erroneous uses of owl:sameAs in the linking of DBpedia and DBLP has resulted in several publications being affiliated to incorrect authors. In addition, Ding et al. [37] discussed a number of issues that arise when linking New York Times data to DBpedia. Specifically, they discuss issues that arise when two things are considered the same in some, but not all contexts. Halpin et al. [64] discussed how the 'sameAs problem', originates from the identity and reference problems in philosophy. In the Semantic Web literature, different families of approaches have been proposed for the limitation of this problem: first, some approaches focus on the (semi-)automatic detection of potentially incorrect owl:sameAs statements [31, 33, 99], other approaches consider the introduction of alternative properties that can replace owl:sameAs [64], finally, a recent approach [19] has been proposed for detecting contextual identity links that captures cases of weak-identity.

In this Chapter, I first present the state of the art on identity problem in section 3.1.1. Then, I will present my contributions for the problems of identity link invalidation in section 3.2, and the problem of contextual identity link detection in section 3.3, in RDF knowledge graphs. In section 3.4, I will give some lessons learned.

3.1.1 State of the Art

3.1.1.1 Identity Link Invalidation

An important aspect of managing identity in the Web of Data is the detection of invalid identity links. It consists in deciding whether a given owl:sameAs link is valid or not, meaning that when the rules expressing its semantics (see Section 2.5) are applied, the knowledge graph remains consistent (i.e., no contradictions are inferred). To detect invalid owl:sameAs links, some recent approaches have been proposed in the literature. These approaches exploit different kinds of information: RDF triples describing the linked resources, domain ontology knowledge automatically acquired or specified by experts, or owl:sameAs network metrics. In what follows, we will give an overview of the related work on detecting invalid identity links by distinguishing three non-exclusive groups of approaches: consistency-based, content-based and network-based.

Consistency-based approaches. Consistency-based approaches are based on the principle of detecting contradictions and/or constraint violations as indications of the erroneousness owl:sameAs links. Some works use ontology axioms to detect contradictions while other exploit constraints on the data and source trustworthiness to measure to what extent an identity link follows the assumptions. idMech [31] is one of the first consistency-based approaches which aims at detecting invalid owl:sameAs links by exploiting two hypotheses: trustworthiness of sources, i.e., assuming that owl:sameAs links published by trusted sources are more likely to be correct and Unique Name Assumption (UNA) which states that every pair of URIs coming from the

same source are necessarily different. The authors developed a probabilistic and decentralized framework for entity disambiguation. The approach detects conflicts between `owl:sameAs` and `owl:differentFrom` assertions by using a graph-based constraint satisfaction solver that exploits the symmetry and the transitivity properties of the `owl:sameAs` relation. The detected conflicts are solved based on the iteratively refined trustworthiness of the sources from which the assertions originate. Other approaches have made use of the hypothesis that individual datasets apply UNA [33, 139], and that violations of the UNA which are caused by cross-dataset linking are indicative of erroneous identity links. De Melo [33] applies a linear programming relaxation algorithm that seeks to delete the minimal number of `owl:sameAs` statements such that the UNA is no longer violated. Valdestilhas et al. [139] efficiently detect the resources that share the same equivalence class and that belong to the same dataset, and rank erroneous candidates based on the number of UNA violations.

Finally, an ontology-based approach, proposed by Hogan et al. [69], exploits the semantics of some ontology axioms to detect inconsistencies in the considered data sources allowing to determine erroneous identity links. It exploits several OWL 2 RL rules in order to express the semantics of axioms such as *owl:differentFrom* and *owl:complementOf* in order to detect inconsistencies. Whenever an inconsistent equality set is detected, the erroneous links are identified by incrementally rebuilding the equality set in a manner that preserves consistency.

Content-based approaches. As it is used for data linking, the content, i.e., the RDF facts describing the resources, may be used to compare the resource descriptions and highlight some inconsistencies or some indications of erroneousness `owl:sameAs` links or of some parts in the data. Paulheim [101] considers that links follow certain patterns, hence, links that violate those patterns are erroneous. The author developed a method which applies a multi-dimensional and scalable outlier detection approach for finding erroneous identity links. It is based on the principle of projecting the links into a vector space in such a way that each link is a point in an n-dimensional vector space. Thus, an identity link is represented as a feature vector in a high dimensional vector space, using direct types and in- and/or outgoing properties. The author has tested different outlier detection methods in order to assign a score to each link, indicating the likeliness of being an outlier. Cuzzola et al. [32] proposed an approach which calculates a similarity score between the names that are involved in a given `owl:sameAs` link, by using the textual descriptions that are associated to these names (e.g., through the `rdfs:comment` property).

Identity Graph Topology-based approaches. Another kind of approaches that have recently been introduced are those exploiting the topological structure of the graph formed by considering only `owl:sameAs` statements, where the nodes represent the linked resources and the edges represent the `owl:sameAs` relation. Gueret et al. [63] hypothesize that the quality of a link can be determined based on how connected a node is within the network in which it

appears. To detect erroneous links, the authors use network metrics, such as, clustering coefficient, centrality and node-degree as well as two additional linked data-specific metrics, namely, owl:sameAs owl:sameAs chains and description richness. The approach relies on the principle of observing changes in the quality of the network with the introduction of new links between datasets as clues of owl:sameAs invalidity. It constructs a local network for a set of selected resources by querying the Web of Data. After measuring the different metrics, each local network is first extended by adding new edges and then analysed again. The results of both analyses are compared to the ideal distribution for the different metrics.

3.1.1.2 Weak and Contextual Identity Links

Instead of seeking for incorrect owl:sameAs links, as discussed in the previous section, some approaches have proposed to represent and/or find weaker identity relations between instances. In this section we present existing alternatives, which either come in the form of simple predicates representing weaker types of identity or similarity, or approaches introducing techniques for representing and detecting contextual identity.

Weak-Identity and Similarity Predicates. One of the main works that proposed an alternative representation of identity relation is Halpin et al. [64]. In this work, the authors presented the Similarity Ontology (SO) in which they hierarchically represent 13 different predicates. This ontology includes five existing linking predicates such as rdfs:seeAlso, SKOS predicates (e.g., skos:exactMatch, skos:closeMatch) and owl:sameAs . The most specific predicate is owl:sameAs and the most general ones are so:claimsRelated and so:claimsSimilar. The predicates prefixed with the word claims express a subjective identity or similarity relation. Their validity depends on the (contextual) interpretation of the user. To define a formal semantics of the SO predicates, the authors have proposed to characterize these predicates by reflexivity, transitivity and symmetry properties. However, this approach does not tackle the problem of how the contexts, in which an identity link is valid, can be explicitly represented.

Some vocabularies acknowledged the abusive use of owl:sameAs and provided alternative identity links. For instance, the UMBEL[1] vocabulary introduced predicates such as the symmetrical property umbel:isLike which is used "to assert an associative link between similar individuals who may or may not be identical, but are believed to be so". Vocab.org[2] introduced the property similarTo to be used when having two things that are not identical (owl:sameAs) but are similar to a certain extent. [33] introduced lvont:nearlySameAs and lvont:somewhatSameAs, two predicates for expressing near-identity in the Lexvo.org[3] vocabulary, with definitions explicitly left vague, “simply because similarity is a very vague notion”. He also introduced lvont:strictlySameAs, a predicate which is declared formally

[1] http://umbel.org
[2] http://vocab.org
[3] http://lexvo.org

equivalent to `owl:sameAs` , but just introduced for the purpose of distinguishing strict identity use from the erroneous use of the latter. Finally, `rdfs:seeAlso`, a predicate which refers to resource that might provide additional information, can still be useful in some cases.

Contextual Identity. The standardized semantics of `owl:sameAs` can be thought of as instigating an implicit context that is characterized by all (possible) properties to have the same values for the linked resources. Weaker kinds of identity can be expressed by considering a subset of properties with respect to which two resources can be considered to be the same. At the moment, the way of encoding contexts on the Web is largely ad hoc, as contexts are often embedded in application programs, or implied by community agreement. The issue of deploying contexts in KR systems has been extensively studied in AI. For the introduction of contexts as formal objects, see [84] for a survey. In the Semantic Web, explicit representation of context has been an important topic of discussion, where the variety and volume of the web poses a new set of challenges than the ones encountered in previous AI systems, see [23], [96] and [59].

With several approaches focusing on representing contexts in the Semantic Web, recent approaches have focused on the specific issue of detecting and/or representing contextual identity. For instance, [19] propose an approach that allows the characterization of the context in which an identity link is valid. A context is represented by a subset of properties for which two individuals must have the same values, with all the possible subsets of properties organized in a lattice using the set inclusion relation. However this approach requires a common vocabulary between the linked resources to represent the contexts.

Finally, [73] presents a system for constructing context-specific identity links between datasets, by allowing annotation of the links. Specifically, this system allows the generation of linksets, and allows one to explicitly represent the provenance information on how each linkset is generated. Then this rich metadata is used to select and combine candidate sets into context-specific *lenticular lenses*, which then serve as a context-specific equality relation for a view over the integrated data. This system uses named graphs and singleton properties [96] to represent the identity contexts. Finally, [73] presents a system for constructing context-specific identity links between datasets.

In order to explicitly represent contexts, it is possible to use a reification[4] mechanism that creates a new resource of type `rdf:Statement`, in which we can associate meta-data (i.e. a context). In addition, it is also possible to use n-ary[5] relations, that allow to represent a property as a class. [96] propose a less costly approach in terms of required number of triples to represent a context. This approach consists in the use of singleton properties, which can represent relations between two entities in a certain context. Finally it is also possible to use named graphs[6] to associate meta-data to a set of triples.

[4]https://www.w3.org/TR/rdf11-mt/
[5]https://www.w3.org/TR/swbp-n-aryRelations
[6]https://www.w3.org/2004/03/trix/

3.1.2 Contributions

As discussed in the related work (section 3.1.1.1), one way of limiting the problem of misuse of identity links consists in detecting the erroneous identity links. In this direction we have developed three different methods. The first one [99] is content and consistency-based which consists in exploiting the semantics of some ontology axioms to generate logical inference rules that when used in a logical reasoning lead to KG inconsistency. The second method is content and similarity-based, and as the logical method, exploits ontology axioms and computes a similarity score based on these axioms and on the resource descriptions, that is low similarity score lead to infer incorrect `owl:sameAs` links. Finally, alike the two first methods that are content and axiom-based, the third method detects erroneous identity links by exploiting only the topology of the identity network formed by the identity links.
In order to capture near/weak identity cases, we developed an approach that detects contextual identity links, that express identity relations between two resources restricted to a context represented as a sub-part of an ontology. This approach is able to consider domain expert constraints to provide the most relevant contexts as possible.
All these approaches have been evaluated on real and complex knowledge graphs, such as the whole linked data and knowledge graphs representing data and knowledge on transformation processes in biology.

3.2 Identity Link Invalidation Approaches

In what follows, weI present three link invalidation methods: the logical method in subsection 3.2.1, the similarity based one in subsection 3.2.2 and finally, the network-based method for erroneous identity link detection, in subsection 3.2.3.

Some notations and definitions used in this section can be found in Chapter 2.

3.2.1 Logical Method for Identity Link Invalidation

In [99] we presented a logical approach for identity link invalidation. The problem we addressed is to check if an `owl:sameAs` statement can be invalidated and eventually explain this deduction. To do so, we formalized the problem in terms of detecting inconsistencies in the KG while applying the `owl:sameAs` semantics, namely, transitivity, symmetry and property sharing rules (see subsection 2.5). To support the inconsistency detection we exploit as much as possible the ontology axioms such as disjunction between classes, functionality, inverse functionality and the local completeness of properties. For the later, there is no OWL 2 predicate that allows declaring it, hence we used SWRL [144] rules to declare that a property is local complete.

The here presented approach performs FOL reasoning based on unit-resolution inference rule applied on knowledge base that is composed of: (i) a set of rules expressing the logical semantics of the considered axioms and (ii) a set of RDF facts (or a data graph) restricted to the facts involving the properties and the classes concerned by the ontology axioms. In particular, we consider only the properties that are declared as (inverse) functional or local complete. This led us to define the notion of contextual graph of depth n (see Definition 3.1 below) representing the aforementioned data graph. A depth n is used to reduce the size of the contextual graph and control the inference propagation phase of the algorithm.

Definition 3.1. *n*-degree Contextual Graph $G_{\{n,s,P\}}$
Given and RDF graph G and a node $s \in G, s \in I$, an integer number n and a set P of properties defined for G, a n-degree Contextual Graph $G_{\{n,s,P\}}$ for s is a sub-graph of G such that every node $v_i \in G_{\{n,s,P\}}$ belongs to a property-based path of length m, with $m \leq n$.

Our approach relies on building two contextual graphs (see Definition 3.1), for x and y respectively and on reasoning on the assertions contained in these two graphs. The building blocks of the problem are the following:

– An RDF graph G
– two resources x and y, such that x, y are resources in G
– the triple $\langle x, owl:sameAs, y\rangle$ (or $sameAs(x,y)$) belonging to G
– a set of properties P in G
– a value n representing the depth of the contextual graphs

– the contextual graphs $G_{\{n,x,P\}}$ and $G'_{\{n,y,P\}}$ of depth n for x and y

More formally, the problem of logical `owl:sameAs` link invalidation can be expressed as follows: $G_{\{n,x,P\}} \wedge G'_{\{n,y,P\}} \wedge sameAs(x,y) \Rightarrow \bot$

The construction of the contextual graphs depends on the predicates (properties) we select and the value n. Indeed in complex RDF graph that can combine data coming from multiple data sources, limiting the depth of a contextual graph could be wise. The main reason is that long property-based paths can lead to not relevant piece of information which can eventually confuse the invalidation process. In Figure 3.1 we show an example of two contextual graphs[7] describing two instances x and y for whom we aim to invalidate the link `owl:sameAs`(x,y). The value $n=2$ has been selected. In this example, the set of properties $P = \{P_0,\ldots,P_4\}$ is considered as involved by the ontology axioms.

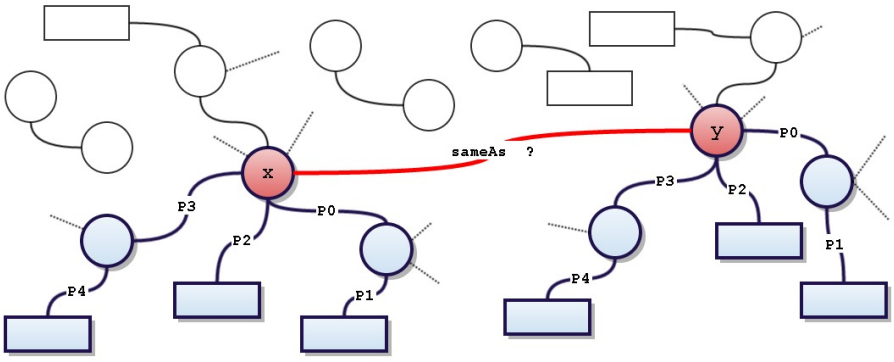


Figure 3.1: The two 2-degree contextual graphs extracted for two instances x and y that are linked by an `owl:sameAs` link.

In the following sub-section we explain how we select the properties.

3.2.1.1 Property Selection and Rule Generation.

To determine the set of properties to consider in the contextual graphs, we choose to select properties that are (inverse) functional and those that are local complete, to allow an efficient and straight forward inconsistency checking. For example, let us consider the property *publication-date* of a book as functional, then for each pair of books having two different values for this property should be considered as different. This kind of reasoning cannot be conducted for the properties that are not functional, except for those an expert can assert their *local completeness*. More precisely, the properties that are instantiated, i.e. for which when the set of values are given for an instance, then we are able to guarantee that this set of values is complete (e.g. the author list of a paper, the directors of a movie, the set of molecules of a drug).

[7] *Circles represent URIs and Rectangles represent Literals.*

In what follows, we define more formally the axioms of (inverse)functionality of properties and local completeness of properties. We also present the inference rules that can be generated from the logical semantics of these considered axioms.

In the sequel we will use the notation ($Z \equiv W$) to indifferently refer to identity relation between Z and W, when Z and W are URIs or to syntactic equality of Z and W, when Z and W are literal values.

Functional and Inverse Functional Properties. Let p be a functional property. It can be expressed logically as follows [67]: $\forall X, Z, W\ p(X,Z) \wedge p(Y,W) \Rightarrow Z \equiv W$

This rule can be written with regards to `owl:sameAs` links invalidation problem defined above. To invalidate $\texttt{owl:sameAs}(x,y)$ while considering that x and y are described by a functional property p with the value w_1 for x and $w2$ for y, and that we can assert that w_1 and w_2 refer to two different entities (or literals), then we can use the following rule which leads to inconsistency:

$$\texttt{owl:sameAs}(x,y) \wedge p(x,w_1) \wedge p(y,w_2) \wedge w_1 \not\equiv w_2 \Rightarrow \bot$$

An analogous reasoning can be done for inverse functional properties. In these situations, if we assume that the assertions already in the RDF graph are true and we have 'doubts' only on the `owl:sameAs` statement, we can conclude that this latter is invalid. In our approach, taking into consideration functional and inverse functional properties, we basically use the following rules for every functional data type property p_i, every functional object property p_j and for every inverse functional object property p_k, in the contextual graphs we are considering.

- $R1_{FDP} : \texttt{owl:sameAs}(x,y) \wedge p_i(x,w_1) \wedge p_i(y,w_2) \rightarrow synVals(w_1,w_2)$
- $R2_{FOP} : \texttt{owl:sameAs}(x,y) \wedge p_j(x,w_1) \wedge p_j(y,w_2) \rightarrow \texttt{owl:sameAs}(w_1,w_2)$
- $R3_{IFP} : \texttt{owl:sameAs}(x,y) \wedge p_k(w_1,x) \wedge p_k(w_2,y) \rightarrow \texttt{owl:sameAs}(w_1,w_2)$

$synVals(w_1,w_2)$ expresses that the two literal values w_1 and w_2 are equal or synonym, and conversely $\neg synVals(w_1,w_2)$ expresses that w_1 and w_2 are different (further description is given in Section 3.2.1.2). Let us notice that, given a property p in the graph G, the fact that p is a functional property can be present among the assertions in G or derived after collecting knowledge from experts or gathering externally (e.g. existing ontologies, additional assertions on the Web.)

Local completeness. The closed-world assumption is in general inappropriate for the Semantic Web due to its size and rate of change [65]. But in some domains and specific contexts, local-completeness for RDF predicates (properties) could be assured. A good example for a multi-valued local complete property could be one representing the authors of a publication. When a

predicate is like that, it should be declared *closed* in the specific knowledge base, making a local completeness assumption. A Local Completeness (LC) rule specifies that the resource is complete for a subset(s) of information (on a particular ontology): the information contained in the resource is all the information for the subset (specified by the rule) of the domain. In an RDF graph G, we declare the following OWL2 RL rule for each property that fulfills LC:

- $R_{4_{LC}} : sameAs(x,y) \wedge p(x,w_1) \rightarrow p(y,w_1)$

where p is a predicate defined in the RDF graph G, x and y are URIs in G ($x,y \in R$) and w_1 is a literal ($w_1 \in L$). This rule will be used to discover inconsistencies since negative facts can be inferred because of the local completeness, as explained in the next Section. Given a property p, the knowledge of 'local completeness' for p can be asserted by an expert or discovered using semi-automatic approaches.

3.2.1.2 The Invalidation Approach

In this Section we present our logical invalidation approach, on the basis of all the Definitions and assumptions made so far. Given G the initial RDF graph with R the set of resources in G. Given `owl:sameAs`(x,y) the input `owl:sameAs` statement to invalidate, where $x,y \in R$. Let F be a set of facts, initialized to an empty set. We consider also L the set of literals in G and a value n indicating the depth of the contextual graphs.

1. Build a set F_1 of $\neg synVals(w_1,w_2)$, for each pair of semantically different w_1 and w_2, with $w_1,w_2 \in L$.

2. Build the contextual graphs for x and y considering (inverse) functional properties and local complete properties

   - For all the (inverse) functional properties $p_{i_{FP}}$ add the relative set of RDF facts to F, considering the rules $R_{1_{FDP}}, R_{2_{FOP}}, R_{3_{IFP}}$ in Section 3.2.1.1.
   - For each $p_{i_{LC}}$ that falls in the contextual graphs and fulfils the local completeness (i.e. $R_{4_{LC}}$ is declared), add to F a set of facts in the form $\neg p_{i_{LC}}(s,w)$ if w is different from all the w' s.t. $p_{i_{LC}}(s,w')$ belongs to F. With $w,w' \in L$ and $s \in R$.

3. Apply iteratively unit resolution until saturation [115] using $F \cup CNF^{8}\{R_{1_{FDP}}, R_{2_{FOP}}, R_{3_{IFP}}, R_{4_{LC}}\}$.

Note also that disjointness of classes can be provided as input and considered in the resolution.

The set of $\neg synVals(w_1,w_2)$ with $w_1,w_2 \in L$ can be obtained using different strategies. It is possible, for example, to perform a pre-processing step in which we build a clustering of the values according to specific criteria. To clarify, consider a simple example of names

[8] CNF: Conjunctive Normal Form

of cities in a specific domain: it is possible to pre-process all the possible values and assert that $synVals('Paris','ParisCity')$ and that $\neg synVals('Paris','Milan')$ and so on. Thus, the evaluation is based on determining if two values w_1, w_2 belong to the same cluster. Another situation arises when the values are 'well defined' as in the case of enumeration, dates, years, geographical data or some types of measures. In these cases, the evaluation is again a simple syntactic comparison of the values. If they are the same, they are equivalent, otherwise they are not equivalent.

3.2.1.3 Experiments and Results

To evaluate the efficiency of our approach we considered three sets of initial `owl:sameAs` links that are computed by three different data linking tools: [118], [131] and [150]. In [118] the `owl:sameAs` statements are computed according to similarity measures over specific property descriptions, as in [150] where similarity between entities is iteratively calculated by analysing specific features. In [131], instead `owl:sameAs` statements are computed on the basis of a novel algorithm used to evaluate the quality of keys that are discovered by SAKey.

All the above methods have produced results on the Person-Restaurants (PR) dataset available for the instance matching contest OAEI 2010 (IM@OAEI2010) [26]. For this experiment, we considered as functional properties *phone_number* and *has_address* that describe a restaurant and *city* that describes an address[9]. Thus, given a `owl:sameAs` statement in the form $sameAs(x,y)$ we computed the contextual graph of degree 2 considering the three functional properties listed before.

Linking Method (LM)	LM Precision	IM Recall	IM Precision	IM Accuracy	LM+IM precision
[131]	95.55%	75%	37%	93.34%	98.85%
[118]	69.71%	88.4%	88.4%	92.9%	95.19%
[150]	90.17%	100%	42.30%	86.60%	100%

Table 3.1: The results of the logical invalidation approach on a set of `owl:sameAs` links provided by the three linking methods.

To assess the quality of the `owl:sameAs` statements computed by different linking methods, we computed the recall and the precision of the invalidation approach. We considered as correct links those that are in the link set provided by [118], [131] or [150], and do not belong to the invalidation method output. In table 3.1, we report the recall and the precision for the invalidation approach (IM) and the overall precision (LM+IM) when the data linking approach is followed by the logical invalidation approach. We can observe that the precision of our method (IM) is lower than the linking methods precision respectively presented in [118], [131] and [150]. This is due

[9]Note that both the previous methods aligned the two initial datasets in order to compute the `owl:sameAs` statements. We considered the same alignment in the explanation of the results.

to the fact, the logical reasoning does not capture the possible syntactic variations (e.g. the phone number values `+142353658` different from `0142353658`) and in the OAEI dataset contains some cases of a restaurant having two phone-numbers which do not fit with the functionality of the property phone-number. In addition to this, there can be errors in the data (for example `'los angeles'` and `'los feliz'`) and the computation of the $\neg synVals$ has been imprecise. That is, the invalidation approach can be used to highlight the inconsistency to the user (expert) and ask for confirmation or correction.

In summary, the results presented in Table 3.1, showed that, the accuracy of the approach is rather good for all datasets 86.6% for [150] links, 92.9% for [131] links and 93.34% for [118] links. We also may observe that when the logical invalidation method is applied after one of the linking tool, the precision may increase significantly (up to 25% of increase). Indeed, for [131] we pass from a precision of 95.55% to 98.85%, for [118] from a precision of 69.71% to 95.19% and finally for [150] from a precision of 90.17% to 100%.

3.2.2 Numerical Method for Identity Link Invalidation

In order to improve even more the recall of the invalidation process, we have proposed in [98] an extension of the logical approach [99] by using similarity measures to compare literal values instead of equality of values. Thus, the heterogeneity of the datasets can be captured and taken into account.

Approach. This numerical method is similarity based and relies on the same principle of exploiting ontology axioms when building the contextual graphs.

Definition 3.2. (**Contextual similarity between two resources**). Let $G_{\{n,x,P\}}$ and $G'_{\{n,y,P\}}$ two contextual graphs (see Definition 3.1) of depth n for x and y, with $P = DP \cup OP$ the subset of properties of type *owl*:*DatatypeProperty* (DP) or *owl*:*ObjectProperty* (OP) delimiting the context in the two graphs G and G'. The contextual similarity $CSim_{\{P,m\}}(x,y)$ for two resources x and y is defined as follows:

$$CSim_{\{P,n\}}(x,y) =$$

$$F\Big(\bigcup_{\forall p_i \in DP} Sim(p_i.V(x), p_i.V(y)) \cup \bigcup_{\forall p_j \in OP} CSim_{\{P,n\}}(p_j.V(x), p_j.V(y))\Big)$$

where :

– $p_i.V(x)$ represents the property values of a property p_i for a resource x in $G_{\{m,x,P\}}$,

– $Sim(v_x, v_y)$ is a function that computes a similarity score in [0..1] for the literal values v_x and v_y. It consists either on an elementary similarity measure (e.g. Jacard, Jaro, Levenstein), or a measure which computes a similarity score between two sets of values for non-functional properties,

– F is an aggregation function (e.g. average or minimum) that is applied on the set of elementary similarity scores obtained for the set of considered properties.

The problem of numerical invalidation of `owl:sameAs` links can be expressed as the ability to determine whether for a pair of resources x and y, we have $CSim_{\{P,m\}}(x,y) \leq T$. (with T a similarity threshold $\in$ [0..1]).

Comparative experiments of the logical and the numerical invalidation approaches. We conducted an experiment on the PR track of OAEI2010 and we compared the results obtained by the logical approach [99] and the numerical [98] approach for link invalidation. The Table 3.2 presents the results obtained by the two approaches on the three datasets *Person1*, *Person2* and *Restaurant* of PR@OAEI2010. The results of the numerical approach are those obtained at the best similarity threshold and using the average as aggregation function. For the three datasets the results of the numerical approach in terms of F-measure and precision are better than the results of the logical approach. Indeed, we get an average gain of 23% of F-measure using the numerical approach, thanks to a very significant increase in precision while having a comparable result in terms of recall. It suffices to have a single property with different values for the logical approach to invalidate the identity link.

	Logical approach [99]			Numerical approch [98]			
Datasets	Precision	Recall	F-measure	Precision	Recall	F-measure	Threshold
Person1	0.69	0.98	**0.81**	1.0	0.98	**0.99**	0.3
Person2	0.5	**1.0**	**0.67**	0.994	**0.989**	**0.99**	0.2
Restaurant	0.63	0.97	**0.77**	0.97	1.0	**0.98**	0.4

Table 3.2: Comparison between the logical approach[99] and the numerical approach[98]

3.2.3 Community Detection approach for Identity Link Invalidation

In the linked open data (LOD) where millions of `owl:sameAs` links are declared between resources coming from thousands of sources, it is not realistic to use content or axiom based approaches, to invalidate identity links. Indeed, content based approaches assume having access to the RDF description of the linked resources. This can not be guaranteed for every resource in the LOD (e.g. freebase is not any more accessible). Axiom based approaches assume beforehand declared axioms. This assumption is not scalable to the LOD context because there is no expert who can assert such axioms and discovering automatically axioms that are valid in hundreds or thousands of sources infeasible. All these reasons have led us to believe that there is a need for methods for identity link invalidation that are source content and ontology axiom agnostic.

In this direction we explored the possibility of using only the topology of the data graph restricted to `owl:sameAs` links. We studied how the communities that can be formed in the graph of `owl:sameAs` links can help to detect erroneous links. That is, in [107] we proposed an approach for erroneous identity link detection by relying on the community structure of the identity network itself. It uses the Louvain [22] community detection algorithm to compute the set of non-overlapping communities of the identity network. Based on that communities, an error degree is computed for each intra-community and inter-community identity link.

The method we proposed consists in two main steps: firstly, the extraction and compaction of the identity network, and secondly, the ranking of each identity link based on the community structure.

3.2.3.1 Identity Network Construction

The first step of our approach consists in extracting the identity network from a given data graph (see Definition 2.2). From a given data graph G, we extract the explicit identity network N_{ex}, which is a directed labelled graph that only includes the edges whose labels are `owl:sameAs`.

Definition 3.3. (**Explicit Identity Network**). Given a graph $G = (V, E, \Sigma_E, l_E)$, the related explicit identity network is the edge-induced subgraph $G[\{e \in E \mid \{\texttt{owl:sameAs}\} \subseteq l_E(e)\}]$.

For scalability reasons, we reduce the size of the explicit identity network N_{ex} into a more concise undirected and weighted identity network I (defined in Definition 3.4), without losing any significant information. Since reflexive `owl:sameAs` statements are implied by the semantics of identity, there is no need to represent them explicitly. In addition, since the symmetric statements e_{ij} and e_{ji} express the same assertion: that v_i and v_j refer to the same thing, we can represent them more efficiently, by including only one undirected edge with a weight of 2 (edges for which either e_{ij} or e_{ji}, but not both, is present in N_{ex} we assign weight 1.)

Definition 3.4. (**Identity Network**). The identity network is an undirected labeled graph $I = (V_I, E_I, \{1,2\}, w)$, where V_I is the set of nodes, E_I is the set of edges, $\{1,2\}$ are the edges labels, and $w : E_I \rightarrow \{1,2\}$ is the labeling function that assigns a weight w_{ij} to each edge e_{ij}.

For each explicit identity network $N_{ex} = (V_{ex}, E_{ex})$, the corresponding identity network I is derived as follows:

– $E_I := \{e_{ij} \in E_{ex} \mid i \neq j\}$

– $V_I := V_{ex}[E_I]$, i.e., the vertex-induced subgraph.

– $w(e_{ij}) := \begin{cases} 1, & \text{if } e_{ij} \in E_{ex} \text{ and } e_{ji} \notin E_{ex} \\ 2, & \text{if } e_{ji} \in E_{ex} \text{ and } e_{ji} \in E_{ex} \end{cases}$

3.2.3.2 Links Ranking

Given $I = (V_I, E_I, \Sigma_{E_I}, w)$, a partitioning of V_I is a collection of non-empty and mutually disjoint subsets $Q_k \subseteq V_I$ that together cover V_I. Since the closure of E_I forms an equivalence set (the semantics of the `owl:sameAs` property states that it is reflexive, symmetric, and transitive), it also induces a unique partitioning. Adopting terminology from [60], we call members of this partition *equality sets*. These partition members correspond to the connected components of I (Definition 3.5).

Definition 3.5. (**Equality Set**). Given an identity network $I = (V_I, E_I, \{1,2\}, w)$, an equality set Q_k is a connected component of I.

In this work our aim consists in detecting erroneous identity links based on the community structure of each connected component of the identity network. While the number of potential identity links is quadratic in the size of the domain, the representation of equality sets is only linear in terms of the size of the domain. With equality sets, we implemented our algorithm with the following requirements:

- The computation of erroneous identity links must not have a large memory footprint, since it must be able to scale to very large identity networks, and preferably to all identity statements that appear in the LOD Cloud.
- It must be possible to perform computation in parallel, to allow errors to be detected relatively quickly, preferably directly after the publication of the potential error into the LOD Cloud.
- Calculation must be resilient against incremental updates. Since triples are added to and removed from the LOD Cloud constantly, adding or removing a `owl:sameAs` link must only require a re-ranking of the links within the equality sets that are directly involved in this link.

In order to compute a ranking for the `owl:sameAs` links, our method first partitions the identity network into different equality sets (several graph partitioning techniques could be applied such as [17]). Then it detects a set of non overlapping communities by applying the *Louvain* algorithm [22] for each equality set.

Given an equality set Q_k, the *Louvain* algorithm returns a set of non overlapping communities $C(Q_k) = \{C_1, C_2, \ldots, C_n\}$ where:

- a community C of size $|C|$ (i.e. the number of nodes) is a subgraph of Q_k such that the nodes of C are densely connected (i.e. the modularity of the Q_k is maximized).
- $\bigcup_{1 \leq i \leq n} C_i = Q_k$ and $\forall C_i, C_j \in C(Q_k) s.t. i \neq j, C_i \cap C_j = \emptyset$.

We then evaluate the quality of each identity link by relying on its weight and the structure of the communities it occurs in. More precisely, to compute the error degree, we distinguish between two types of edges: the *intra-community links* and *inter-community links*.

Definition 3.6. (**Intra-Community Link**). Given a community C, an intra-community link in C noted by e_C is a weighted edge e_{ij} where v_i and $v_j \in C$. We denote by E_C the set of intra-community links in C.

Definition 3.7. (**Inter-Community Link**). Given two non overlapping communities C_i and C_j, an inter-community link between C_i and C_j noted by $e_{C_{ij}}$ is an edge e_{ij} where $v_i \in C_i$ and $v_j \in C_j$. We denote by $E_{C_{ij}}$ the set of inter-community links between C_i and C_j.

In what follows, we consider the non-overlapping community structure as an approximation of *equivalence class structure* that can be built from the set of considered `owl:sameAs` links. Indeed, an equivalence class structure represents a partition of the resources involved in the `owl:sameAs` links in such a way that each equivalence class contains the set of resources that are pairwise identical. Each equivalence class fulfils transitivity, reflexivity and symmetry properties of `owl:sameAs` relation. Moreover, two different equivalence classes are considered to be disjoint, i.e. their intersection is empty. For our link invalidation aim, we compute an error degree for each `owl:sameAs` link by considering the density of the community(ies) to which the link belongs, as well as symmetry degree of the link (1 if not symmetrical and 2 if symmetrical). More precisely, we considered the density of the community as expressing how the transitivity property is fulfilled in the community structure. Moreover, inter-community links express how the community structure do not fulfil the disjunction between communities. By construction, these links will be evaluated as erroneous.

To compute the *intra-community links*, our method relies on the density of the community containing the edge as well as the weight of the considered edge. The lower the density of this community and the weight of an edge are, the higher the $error - degree$ will be.

Definition 3.8. (**Intra-Community Link Error-degree**.) Let e_C be an intra-community link of the community C, the intra-community error-degree of e_c denoted by $err(e_C)$ is defined as follows:

$$\text{a) } err(e_C) = \frac{1}{w(e_C)} \times \left(1 - \frac{W_C}{|C| \times (|C| - 1)}\right)$$

where $W_C = \sum_{e_C \in E_C} w(e)$

To compute the *inter-community links*, we rely both on the density of the inter-community connections and the weight of this edge. The less the two communities are connected to each other and the lower the weight of an edge is, the higher the $error-degree$ degree of inter-edge will be.

Definition 3.9. (**Inter-Community Link Error-degree**.) Let $e_{C_{ij}}$ be an inter-community link of the communities C_i and C_j, the inter-community error-degree of $e_{C_{ij}}$ denoted by $err(e_{C_{ij}})$ is defined as follows:

$$\text{b) } err(e_{C_{ij}}) = \frac{1}{w(e_{C_{ij}})} \times \left(1 - \frac{W_{C_{ij}}}{2 \times |C_i| \times |C_j|}\right)$$

where $W_{C_{ij}} = \sum_{e_{C_{ij}} \in E_{C_{ij}}} w(e)$

3.2.3.3 Experiments

In order to test and evaluate our approach we have conducted an empirical evaluation on the whole set `owl:sameAs` links in the LOD in 2015. Bellow, we describe the dataset that is used; and the quantitative and qualitative results of our community structure-based invalidation approach.

Dataset. We have tested our approach on the LOD-a-lot dataset [51][10], a compressed data file that contains the 28B distinct triples from the 2015 LOD Laundromat Linked Data crawl [18]. This large subset of the LOD Cloud represents our data graph (Definition 2.2).

Quantitative Results. We have extracted the explicit identity network (Definition 3.3) from the data graph described above, by performing a Triple Pattern query of the form ⟨?, `owl:sameAs`,?⟩ with the HDT C++ library[11]. This returns a stream of distinct identity pairs, as described in [17]. Extracting the explicit identity network from the RDF graph described above takes around four hours. It results on an explicit identity network of 558.9M edges and 179.73M nodes. The explicit identity network is publicly available at `https://sameas.cc/triple`.

Identity Network Construction. From the explicit identify network described above, we built an identity network (Definition 3.4) containing ~331M weighted edges and 179.67M terms. We leaved out ~2.8M reflexive edges, ~225M *duplicate* symmetric edges and their corresponding nodes (67,261 nodes). For the symmetric edges 68% we assigned the value 2 in the identity network.

The next step consists in partitioning the identity network into several equality sets (Definition 3.5). We have deployed an efficient algorithm described in [17] that partitions the identity network into ~49M equality sets, in just under 5 hours. The set of equality terms of each resulted

[10] `http://lod-a-lot.lod.labs.vu.nl`
[11] `https://github.com/rdfhdt/hdt-cpp`

equality set are publicly available at `http://sameas.cc/id`.

Links Ranking. Once the identity network has been partitioned, we apply the *Louvain* algorithm to detect communities in each equality set. We then assign an error degree to all edges of each community. This process takes around 2 hours[12], resulting an error score to each `owl:sameAs` statement (~556M statements) in the explicit identity network. The 179.67M terms of the identity network were assigned into a total of 24.35M communities, with the communities size varying between 2 and 4,934 terms (averaging ~7 terms per community).

Community Structure Analysis. In this following we provide a first analysis of the community structures obtained from two equality sets (the largest one and the one about Barack Obama) based on the IRIs contained in the communities.

Community Structure in the Largest Equality Set. The largest equality set Q_{max} contains 177,794 terms and 2,849,650 undirected and weighted edges. This equality set is the result of the compaction of 5,547,463 distinct `owl:sameAs` statements (~ 1% of the total number of `owl:sameAs` in the LOD) and is available at `https://sameas.cc/term?id=4073`. As shown by de Rooij et al. [34], the social meaning encoded in IRI names significantly coincides with the formal meaning of IRI-denoted resources. Hence, by looking at the IRIs of this equality set, we observed that it contains a large number of terms denoting different countries, cities, things and persons (e.g. Bolivia, Dublin, Coca-Cola, Albert Einstein, *Literals*, and so on). So, it is clear that this equality set contains many erroneous `owl:sameAs` statements. Applying the *Louvain* algorithm on Q_{max} resulted in 924 non-overlapping communities, with a size varying from 29 to 2,267 terms per community. As a first interpretation on the community results, we have solely looked at the IRIs form in order to evaluate at the coarse-grained level how the communities contain erroneous `owl:sameAs` links.

Community Structure in the 'Barack Obama' Equality Set. The equality set Q_{obama} containing the term `http://dbpedia.org/resource/Barack_Obama` is composed of 440 terms and 7,615 undirected and weighted edges. It is built from an explicit identity network of 14,917 `owl:sameAs` statements. Applying the *Louvain* algorithm on Q_{obama} resulted in 4 non-overlapping communities, with a size varying from 34 to 166 terms per community. The resulting community structure of Q_{obama} is presented in Figure 3.2. The four detected communities are distinguished by their nodes' color [13].

– C_0 **(purple)** includes 166 terms, with 98% of the links of this community representing cross-language symmetrical links between DBpedia IRIs (e.g. `http://fr.dbpedia.org/resource/Barack_Obama`) referring to the person Barack Obama.

[12] on an 8GB RAM Windows 10 machine, with an Intel Core 4 × 2.6 GHz process

[13] The full figure is available at `https://github.com/raadjoe/LOD-Community-Detection/blob/master/Communities-Graph-Obama.svg`

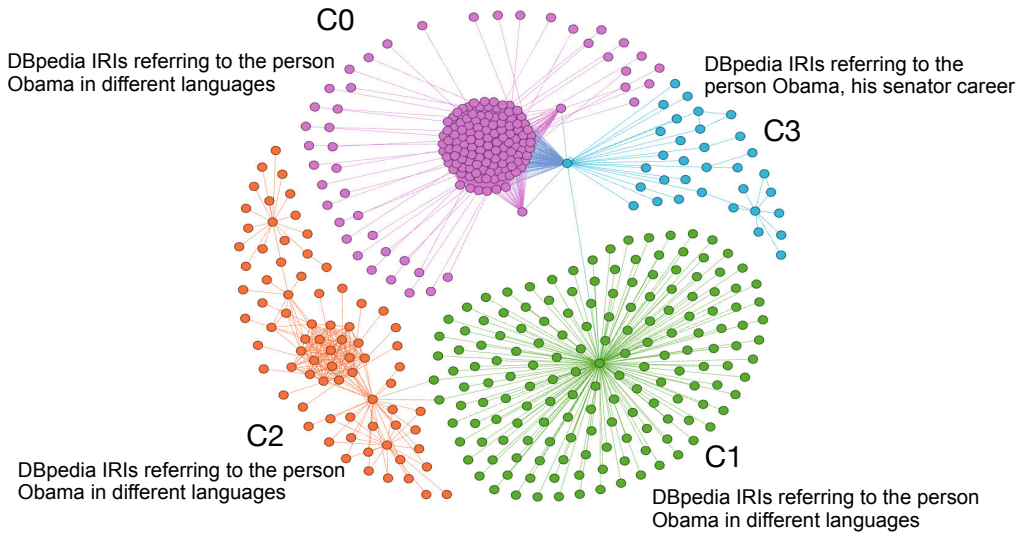


Figure 3.2: The communities detected from the equality set containing the term referring to Barak Oabama using the *Louvain* algorithm

– C_1 **(green)** includes 162 terms, mostly DBpedia IRIs of the person Obama in his different roles and political functions (e.g. `http://dbpedia.org/resource/President_barack_obama`, `http://dbpedia.org/resource/senator_obama`).

– C_2 **(orange)** includes 78 terms, mostly referring to the presidency and administration of Barack Obama (e.g. `http://dbpedia.org/resource/Obama_cabinet`, `http://dbpedia.org/resource/Barack_Hussein_Obama_administration`)

– C_3 **(blue)** includes 34 terms from different datasets denoting various entities such as: Barack Obama the person, his senate career, and a misused literal (`"http://dbpedia.org/resource/United_States_Senate_career_of_Barack_Obama`, `"http://dbpedia.org/resource/Barack_Obama"^^xsd:string`).

3.2.3.4 Links Ranking Evaluation

In order to evaluate the accuracy of our ranking approach, five computer scientists judges have conducted several manual evaluations without any information on the error-degree of the links. The judges relied on the descriptions associated to the terms in the *LOD-a-lot* dataset [51], and did not have any prior knowledge about each link's error degree (i.e. whether they are evaluating a well-ranked link or not). In order to avoid any incoherence between the evaluations, the judges were asked to justify all their evaluations and were given the following instructions: **(a) the same:** if two terms denote the same entity (e.g. Obama and the First Black US President), **(b)**

related: not intended to refer to the same entity but closely related (e.g. Obama and the Obama Administration), **(c) unrelated:** not the same nor closely related (e.g. Obama and the Indian Ocean), **(d) can't tell:** in case there are no sufficient descriptions available for the terms.

Accuracy Evaluation in the 'Barack Obama' Equality Set. In our first evaluation, we have relied on the previous observations, made on the community structure presented in Figure 3.2, to interpret and evaluate the accuracy of our approach:

1. an `owl:sameAs` statement in C_0 has an average error rate of 0.24. The manual evaluation of 30 random `owl:sameAs` statements shows that they are all true identity links.

2. the low density of C_1 has led to several correct `owl:sameAs` statements to have a high error degree (0.9). This is due to the fact that there is only one term linking to all the 161 other terms in this community, with most of these edges being non-symmetrical links.

3. the only two `owl:sameAs` statements in this equality set with an error value $\simeq 1$ are the edges in the graph connecting the IRI v_1: `http://rdf.freebase.com/ns/m.05b6w1g` from C_2 to `http://dbpedia.org/resource/President_Barack_Obama` and `http://dbpedia.org/resource/President_Obama` from C_1. Relying on their descriptions in the *LOD-a-lot* dataset, we can see that v_1 refers to the presidency of Obama, while the two other IRIs refer to the person Obama, indicating that indeed both statements are incorrect. This has led to the false equivalences between the 78 terms of C_2 and the rest of the network's terms.

Accuracy Evaluation on a Subset of the Identity Network. In order to evaluate the accuracy over the whole identity network, four semantic Web researchers were asked to evaluate a subset of the identity network. The judges were asked to evaluate 200 `owl:sameAs` links (50 links each), representing in an equal manner, each bin of the error degree distribution.

Table 3.3: Evaluation of 200 `owl:sameAs` links, with each 40 links randomly chosen from a certain range of error degree

	0-0.2	0.2-0.4	0.4-0.6	0.6-0.8	0.8-1	total
same	35(100%)	22(100%)	18(85.7%)	7(77.7%)	15(68.1%)	**97**(88.9%)
related	0	0	2	2	2	**6**
unrelated	0	0	1	0	5	**6**
related + unrelated	0(0%)	0(0%)	3(14.2%)	2(22.2%)	7(31.8%)	**12**(11%)
can't tell	5	18	19	31	18	**91**
Total	**40**	**40**	**40**	**40**	**40**	**200**

The obtained results presented in Table 3.3, showed the more the error degree is high the more the probability of the links to be erroneous is high. More precisely, we can observe that:

- our error degree is able to identify true `owl:sameAs` links with a high accuracy, since 100% of the evaluated links with an error degree ≤ 0.4. are correct (without considering the "*can't tell*" cases).
- when the error degree is between 0.4 and 0.8, 83% `owl:sameAs` links are correct. However, in 17% of the cases, such links might have been used to refer to two different, but related terms.
- an `owl:sameAs` link with an error degree > 0.8 is an unreliable identity statement, referring in ~31% of the cases to two different, and mostly unrelated terms.

We have further investigated the 13 evaluated identity links with an error degree over 0.9. Two features were observed from the incorrect identity statements: (i) their error degree is higher than the false positives one, and (ii) they all belong to equality sets with a higher number of terms than the false positives. This evaluation showed that: (1) considering solely equality sets with a high number of terms improves the accuracy of our ranking (50% of the links are incorrect), (2) considering solely the top ranked links slightly improves the accuracy of our approach (40% are incorrect) and (3) considering the top ranked links in the large equality sets significantly improves the accuracy of our approach in detecting erroneous identity links (88.2% are incorrect among those having an error degree ~ 1 S3).

Recall Evaluation. In order to compute the recall of our approach, we have verified how our approach can rank new erroneous `owl:sameAs` statements. Firstly, we have chosen 15 random terms in the explicit identity network, while making sure that these terms are all different, by looking at their descriptions, and that they are not explicitly `owl:sameAs` . From these 15 terms, we have generated all the possible 105 undirected edges between them. Then we added separatly, each edge e_{ij} to the identity network with $w(e_{ij})$=1, calculated its error degree, and removed it from the identity network. On average, the introduced identity link has an error degree of 0.84, ranging from 0.2 to 0.99. When the threshold is fixed to 0.8 (resp. 0.9) the recall is 93% (resp. 89%).

The whole evaluation suggests that the error degree is more accurate when the equality sets are large and the error degree is high. In addition, even when a high threshold is chosen, the recall of detecting incorrect identity statements remains high. Since, many `owl:sameAs` statements (~1.26M) have an error degree between 0.99 and 1 and many `owl:sameAs` statements belong to large equality sets (e.g., ~5.5M belong to the largest equality set) the proposed approach can obtain effective results for detecting erroneous `owl:sameAs` links on the LOD.

3.3 Contextual Identity Links Detection

Context-dependency of identity relation is one of the characteristics that cannot be captured by `owl:sameAs` predicate. Indeed, while comparing two medicines described by their name, their chemical molecule and their price, one may reach different decision whether if he/she is interested in the chemical or the economic information of the two medicines. To deal with the problem of discovering contextual identity links, as proposed by Beek et al. [19], we presented in [110] a new ontology-based approach for detecting contextual identity links. Our approach aims at detecting identity links that are valid in contexts that can be defined as sub-parts of the domain ontology.

3.3.1 Problem statement

The problem of detecting contextual identity links can be defined as follows: given a knowledge graph $\mathscr{K} = (\mathscr{O}, \mathscr{D})$ and a set I^{tc} of instances of a target class tc of the ontology $\mathscr{O}$, find for the set of all instance pairs $(i_1, i_2) \in (I^{tc} \times I^{tc})$ the most specific contexts in which (i_1, i_2) are identical.

3.3.1.1 Illustrative example

Let us consider the example depicted in 3.3 which shows a domain ontology for drug description and four different instances of the target class drug.

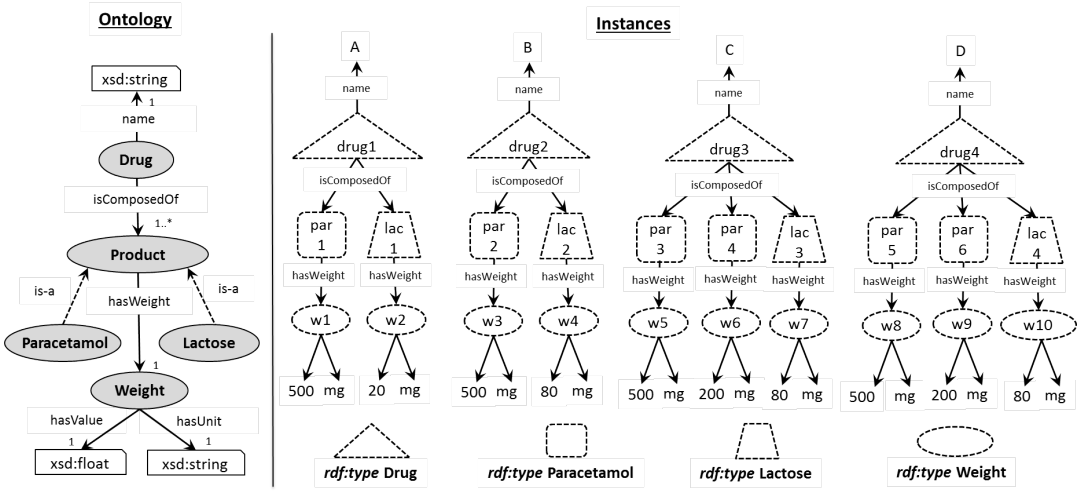


Figure 3.3: An extract of ontology $\mathscr{O}$, four instances $drug1$, $drug2$, $drug3$ and $drug4$ of the target class $Drug$.

In this example the two instances $drug3$ and $drug4$ of the target class $Drug$ can be considered as identical in the context where we consider all the ontology's properties except of the property name for the drugs. On other hand, the two instances $drug1$ and $drug2$ can be considered as identical in two most specific contexts. The first context is the one in which we

consider all the ingredients composing the drugs and for every ingredient we consider its weight. However, in this first context, the description of a weight is reduced to the measure unit without considering the quantity (property $hasValue$). A second context in which these instances are identical is the context where we consider the weight of *Paracetamol* described by its value and its measure unit, and only the presence of $Lactose$ in the drugs without considering its weight.

3.3.1.2 Expert constraints

We are interested in the search of the most specific contexts involving a subset of classes, and for each class a subset of properties. Some contexts can be obviously more relevant than others (e.g. a value of the weight without its measure unit is not meaningful). Hence, we also aim to take into account some expert knowledge that can be represented as a set of constraints on the classes and/or properties that should or should not be involved in the considered contexts. In our approach we aim to consider three kinds of expert constraints to filter out the irrelevant contexts to consider:

– $Unwanted\ properties$ (UP): this refers to properties that an expert wants to discard in the detection of contextual identity links.

– $Necessary\ properties$ (NP): a necessary property p is a constraint that allows one to consider only contexts that contain the property $p \in OP$ or $p \in DP$. – $Co\text{-}occurring\ properties$ (CP): a co-occurrence constraint can be declared to guarantee that a certain class c_i will be either declared as the domain (or range) of all the properties indicated in the constraint, or $none$ of them.

3.3.2 Contexts for Identity Relation

The contextual identity link detection is based on identifying the most specific contexts where two instances are identical. We consider global context as a connected sub-part of the ontology $\mathcal{O}$ which represents the set of classes and properties in which two instances are considered as identical. We consider the notion of global context as opposed to the notion of local context, which defines locally [14] the sub-part of the ontology $\mathcal{O}$ in which the two instances are identical.

A global context is represented as a subset of classes and properties of the ontology, and a set of axioms which is limited to property domains and ranges typing constraints. As in RDF datasets we may have instances that are typed by one or several classes of different levels of abstraction, we defined the set $DepC$ to represent the abstraction level of the classes selected to be allowed to appear in the contexts. This level is automatically determined depending on the instances direct types; thus we keep the most general one. We automatically choose the abstraction level of the

[14] If we transpose these two notions in an ordinary graph the global context will correspond to sub-graphs of depth n that can be ≥ 1 and local context correspond to sub-graphs of depth ≤ 1.

classes involved in a global context by selecting, from the instantiated classes (direct types), the most general ones.

Definition 3.10. Selected classes. The set of selected classes $DepC$ that can be involved in the contextual identity links is the subset of instantiated classes c_i of $\mathscr{K}$ such that:

$$DepC = \{c_i \in \mathscr{C} \mid \nexists c_j \in \mathscr{C}\ s.t.\ \exists x, directType(x, c_j)\ and\ c_i \sqsubset c_j\}$$

Example. In Figure 3.3, $DepC$ will contain all the classes of the graph except $Product$ which is not instantiated. Therefore, $par1$ and $lac1$ will be considered as of type $Paracetamol$ and $Lactose$ respectively.

Definition 3.11. Global Context. A global context is a sub-ontology GC_u=(C_u,DP_u,OP_u,A_u) of an ontology $\mathscr{O}$ such that $C_u \subseteq DepC$, $DP_u \subseteq DP$, $OP_u \subseteq OP$ and A_u is a set of domain and range constraints that are more specific than those described in A: $\forall op \in OP_u$, $domain_u(op) \sqsubseteq domain_{\mathscr{O}}(op)$ and $range_u(op) \sqsubseteq range_{\mathscr{O}}(op)$, and $\forall dp \in DP_u$, $domain_u(dp) \sqsubseteq domain_{\mathscr{O}}(dp)$.

Example. In Figure 3.3, there exist many possible global contexts. We present one:
GC_1=($C = \{Drug, Paracetamol, Lactose, Weight\}$,
$OP = \{isComposedOf, hasWeight\}$, $DP = \{hasValue\}$,
$A = \{domain(isComposedOf) = Drug$,
$range(isComposedOf) = Lactose \sqcup Paracetamol$,
$domain(hasWeight) = Lactose \sqcup Paracetamol$,
$range(hasWeight) = Weight\}$)

Definition 3.12. Order relation between global contexts. Let $GC_u = (C_u, OP_u, DP_u, A_u)$ and $GC_v = (C_v, OP_v, DP_v, A_v)$ be two global contexts. The context GC_u is more specific than GC_v, noted $GC_u \leq GC_v$, if $C_v \subseteq C_u$, $OP_v \subseteq OP_u$, $DP_v \subseteq DP_u$ and
$\forall op \in OP_v$, $domain_u(op) \sqsubseteq domain_v(op)$ and $range_u(op) \sqsubseteq range_v(op)$, and $\forall dp \in DP_v$, $domain_u(dp) \sqsubseteq domain_v(dp)$.

3.3.2.1 Contextual identity links.

In our approach, two instances are considered as identical in a given context, when all the properties involved in the global context are instantiated in the considered instances' descriptions, and when their respective values are equal. Therefore, we firstly define the contextual description that is considered for one instance in one context. Then we will define the conditions that must hold to consider that two RDF descriptions refer to identical instances in a given context.

Definition 3.13. Contextual instance description according to a global context. Given a RDF dataset $\mathscr{D}$, a global context $GC_u = (C_u, OP_u, DP_u, A_u)$ and an instance i, a contextual description G_i of i in GC_u is the maximal set of triples that describe i in $\mathscr{D}$ such that:

– G_i forms a connected graph that contains at least one triple where i is a subject or an object
– $\forall\ t = <s, p, o> \in G_i$ then $p \in OP_u \cup DP_u$ and $type(s) \sqsubseteq domain_u(p)$ and $type(o) \sqsubseteq range_u(p)$
– $\forall\ j$ a class instance of G_i, and $\forall dp \in DP_u$ such as $type(j) \sqsubseteq domain_u(dp)$, then $\exists\ t_a = <j, p, v> \in G_i$, with v of type literal
– $\forall\ j$ a class instance of G_i, and $\forall op \in OP_u$ such as $type(j) \sqsubseteq domain_u(op)$, and $c_1 \cup c_2 \sqsubseteq range_u(op)$ then $\exists\ t_a = <j, op, k>$ and $t_b = <j, op, l> \in G_i$ with $type(k) = c_1$ and $type(l) = c_2$

From two contextual descriptions of two class instances defined in a given context, we can define if they can be considered as identical. In this work we will consider that properties are local complete: if a property p is instantiated for a given class instance i, we consider that all the property instances are known for i. Since a local completeness is assumed, two instances can be considered as identical when the contextual graphs, formed by the contextual descriptions, are isomorphic up to a renaming of the instance URI. Note that since some classes can be removed from the global context, this constraint can in fact be considered class by class.

Definition 3.14. Identity in a global context. Given a global context GC_u, a pair of instances (i_1, i_2) are identical in GC_u, noted $identiConTo_{<GC_u>}(i_1, i_2)$, only if the two labelled graphs G_{i_1} and G_{i_2} that represent the contextual descriptions of i_1 and i_2 are isomorphic up to a rewriting of the URI of the class instances (literals must be equal).

Example. $drug1$ and $drug2$ are considered as identical according to the global context GC_1 defined in Example 2.
(i.e. $identiConTo_{<GC_1>}(drug1, drug2)$).

The contextual identity relations will only be specified for the most specific global context(s), but can be inferred for the more general ones using the order relation between global contexts: given GC_u and GC_v two global contexts, with $GC_u \leqslant GC_v$, then $identiConTo_{<GC_u>}(i_1, i_2) \Rightarrow identiConTo_{<GC_v>}(i_1, i_2)$.

3.3.3 Contextual Identity Detection Method

The goal of our method named $DECIDE$ (DEtection of Contextual IDEntity) is to determine for each pair of individuals $(i_1, i_2) \in I^{tc} \times I^{tc}$ of a target class tc given by the user, the set of the most specific global contexts in which the identity relation $identiConTo$ is true. $DECIDE$ requires to have the set of facts $\mathscr{F}$ of the considered knowledge base, the target class tc as inputs, and may consider different constraint lists UP, NP, CP given by an expert. Here, we restrict the description of this algorithm to its three main functions, nonetheless a more detailed description with different use-cases is available in [109, 110]:
– collects the selected classes, in order to indicate the level of abstraction to be considered in building the identity graphs and generating the most specific global contexts.
Then for each pair of individuals of the target class tc:

– constructs the identity graph(s), using a depth-first search algorithm. When different mappings between instances of the same class can be considered, a new identity graph is constructed.

– generates the most specific global context(s) by relying on the constructed identity graphs. A global context GC is constructed using the set of local contexts by insuring the presence of a single[15] one local context per class in each global context. The most specific global contexts are generated using the function $generateGC$, which traverses the identity graph IG using also a depth-first search algorithm.

3.3.4 Empirical evaluation

We have conducted an experimental evaluation in real datasets in life science domain. We first evaluated quantitatively our contextual identity approach. The we showed how the discovered contexts may be used to generate prediction rules for missing observations.

3.3.4.1 Datasets description.

Our approach has been evaluated on two scientific datasets exploited using the 1.4 version[16] of the ontology PO^2 [72], which aims at modelling transformation processes. Each process can be conducted over several itineraries, with each itinerary representing a sequence of transformation steps (e.g. drying, heating). The ontology PO^2 also represents the set of observations conducted during each step. These observations contain a large number of missing information.

–The first dataset describes the process of micro-organisms' stabilization, conducted in 20 different itineraries in the context of the INRA[17] CellExtraDry project. This dataset contains 1 721 979 statements, 208 instantiated selected classes, 415 136 instances and 159 properties (83 object properties).

– The second dataset describes the process of the dairy gels' transformation, conducted in 12 itineraries in the context of the INRA Carredas project. This dataset contains 237 838 statements, 555 instantiated selected classes, 42 269 instances, and 159 properties (83 object properties).

We have tested our algorithm $DECIDE$ separately on each of these datasets, in order to detect the most specific global contexts in which the instances of the target class $Mixture$ are identical. A mixture is composed of a set of products and is transformed during the different steps of the process.

[15]This constraint guarantees semantic coherence of identity detection procedure, e.g. to compare two instances of the class Researcher, the same local context will be used.

[16]The core ontology of PO^2 is available at: `http://agroportal.lirmm.fr/ontologies/PO2`

[17]The French National Institute for Agriculture

Table 3.4: Results of $DECIDE$ on the CellExtraDry and Carredas datasets with the target class $Mixture$

	CellExtraDry	***Carredas***
# Individuals of target class	210	619
# Possible Pairs	21 945	191 271
# Graph Nodes per pair	11	7
# Different Global Contexts	28	231
# Identity Links	31 092	239 410
# Identity Links per pair	1.41	1.25
Execution Time (approx. minutes)	2	26

3.3.4.2 Discovered contextual identity links.

Table 3.4 presents the results of $DECIDE$ applied on these two scientific datasets, without considering their observations (i.e. the properties related to the observations have been declared as unwanted properties). In the $CellExtraDry$ dataset, the 210 instances of the target class $Mixture$ which can form 21945 pairs, have resulted in 31092 contextual identity links valid in 28 global contexts in total, while the 191 271 pairs of mixtures in the $Carredas$ dataset have resulted in 239 410 identity links valid in 231 different global contexts in total. Some of the detected contexts contain up to 20 classes and 35 properties, while less specific ones contain only one class and one property.

We repeated the experiments on each dataset, while taking into account a constraint cp that expresses that a weight value cannot be considered without its unit of measure and vice versa. While the number of distinct most specific global contexts remained unchanged in both datasets, we noticed a change in around 40 % of the generated most specific global contexts.

3.3.4.3 Use of contextual identity links for prediction.

The goal of this experimentation is to show if contextual identity links can be exploited for prediction tasks. More precisely, we want to find out the probability of two experiments, identical in a certain context, to have similar observations. Therefore, we will be able to predict to a certain degree of certainty, some experiments' unobserved measures. Table 3.4 indicates that the instances of the target class $Mixture$ are connected to most of the datasets' instantiated classes, 191 out of 208 in $CellExtraDry$ and 488 out of 555 classes in $Carredas$, thus showing that an identity between two mixtures can also indicate an identity between the experiments' steps in which these two mixtures exist.

By exploiting the property sharing of identity relation (see Definition 2.5 in Chapter 2) we attempted to detect for each context GC_i, the set Ψ of properties $\{p_1,...,p_n\}$, where $identiConTo_{<GC_i>}(x,y) \cap p(x,z_1) \rightarrow p(y,z_2)$ with $z_1 \simeq z_2$ and $\Psi \cap (OP^{GC_i} \cup DP^{GC_i}) = \emptyset$. Such rules can be expressed as r: $identiConTo_{<GC_i>}(x,y) \rightarrow$ same(m), with m representing a certain measure (e.g. pH measure). Since the detected contextual identity links are only stated for the most specific contexts of each pair, we have exploited the global contexts' order relation (see

Table 3.5: Examples of Detected Rules in the Carredas dataset

Rule	***Error Rate***	***Support***
$identiConTo_{<GC_{102}>}(x,y)$ → same(Adhesiveness)	2.2 %	23 %
$identiConTo_{<GC_{74}>}(x,y)$ → same(Sweetness)	4.5 %	13 %
$identiConTo_{<GC_{202}>}(x,y)$ → same(Bitterness)	7.1 %	29 %
$identiConTo_{<GC_{124}>}(x,y)$ → same(Acidity)	8.2 %	21 %

Definition 3.12) to obtain the complete set of contextual identity links for each global context.

In order to evaluate the quality of a rule r we used two measures:

– **the rule's average error rate:** for each instance pair (x, y) identical in GC_i, we calculate the error rate for their m measure values (i.e., the value of an observation property like temperature) based on the maximum and the minimum value of the observation property in the dataset.

– **the rule's support:** represents the number of instance pairs identical in GC_i that have the same value for the measure m, divided by the total number of pairs in GC_i.

We have generated 112 rules in the $CellExtraDry$ dataset (averaging 4 rules per context), and 3677 rules in the $Carredas$ dataset (averaging 15 rules per context). The error rate on average is around 7.3% in CellExtraDry and 20% for Carredas. On average, in both datasets the rule's support is very low around 0.4% and 1%. This low support in both datasets shows the large number of observational measures that are missing in each experiment. As a result we have also observed that the error rate of a rule decreases by 22% when a global context is replaced by a more specific global context in the $CellExtraDry$ dataset, and decreases by 31.5% in the $Carredas$ dataset. This decrease shows that the rules discovered in more specific global contexts are more precise than the rules discovered in more general ones, and that the contextual identity links can for example be exploited to predict missing properties values with different confidence level. We asked the domain experts to evaluate the plausibility of the 20 best detected rules (in terms of error rate and support combined) on a scale of "Plausible", "Probably Plausible", "Not Sure", "Probably Not Plausible", and "Not Plausible". 9 rules were evaluated as plausible, 4 as probably plausible and 1 rule as not plausible. The experts were not sure of the plausibility of the 6 remaining rules. Table 3.5 presents some of the rules evaluated as plausible, in the $Carredas$ dataset. For instance, the first rule indicates that there is a high probability that mixtures with the same weight of Rennet, Sardine, and Sodium Chloride, and mixtures containing Lipids, Water, and Proteins (not necessarily the same weight), to have similar adhesiveness.

Our collaboration with the domain experts, and the experiments' results conducted on these scientific datasets have shown us that:

- the use of genuine identity links such as the *owl:sameAs* link is rarely required in scientific datasets, since the experiments' environment tend to change, even slightly from one experiment to another, which could result in a propagation of incorrect observational measures;
- asking domain experts to specify the contexts in which two objects are considered identical is not an intuitive task, as the identity contexts can differ from one expert and task to another. Instead, specifying some constraints on these contexts is a more effective way to benefit from the experts' knowledge;
- thousands of explicit contextual identity links can be detected in a reasonable time, despite the high connectivity between all these graph's instances;
- the contextual identity links can for instance be used to generate rules that can help predict some of the missing observational measures;
- the relevance of a certain context can vary depending on the conducted observations. For instance, the identity of the mixtures' composition is required in tasks that study the acidity, while the identity of the mixtures' steps is required in tasks studying the experiments' environmental impact;
- rules that are detected in more specific contexts have a better error rate than the ones detected in less specific contexts.

3.4 Lessons Learned

In this chapter I drew up an overview of existing approaches for identity link invalidation and near/weak identity while describing some of our contributions in this field of research. Hence, I proposed different solutions to address some facets of identity problem, namely, erroneousness of available `owl:sameAs` links and the subjectivity and context-dependency of identity relation.

As regards to link invalidation problem, to achieve higher precision and scalability results, there is a real need for more hybrid link invalidation methods that are able to combine different kinds of information about the resources themselves and the original data sources. Even though, the rather efficiency and scalability of the proposed approaches, there are still improvements needed for achieving higher accuracy results. Evaluation protocols are still considering manual evaluation on samples, hence, benchmarks are needed to be built and proposed to global evaluation contests as OAEI. As firstly proposed in [4], crowdsourcing evaluation processes should be considered in such evaluation protocols. Moreover, as it has been highlighted by some of works [139] in the state of the art, links that are produced by automatic data linking tools fulfil better some consistency constraints (e.g. UNA) than ones published on repositories like `sameas.org`. Actually, the rules of link publication on the LOD should be strengthen and restricted to links satisfying some quality criteria. Moreover, link quality assessment is not matter of one unique dimension. Link quality consists in not only its validity but also in link added-value and meta-data, like information gain [125], reachability, availability [94] as well as link evolution throughout the time. Up to our knowledge, none of the existing link invalidation methods propose a full repairing approaches. Although the invalidation reasons of an identity link can be manifold (e.g. errors in the values, data freshness, ontology axioms uncertainty), existing approaches need to be extended by a repairing phase where several possible repairs are proposed.

Main Collaborations and projects of the Chapter

The research work I presented in this chapter has been achieved thanks to the collaboration with several colleagues. For the identity link detection and invalidation I co-supervised a PhD student and a post-doc candidate:

Laura Papaleo's post-doc project (2014–2016) co-supervised with **Nathalie Pernelle** (Paris Sud University) and funded by the Qualinca ANR project (2012-2016).

Joe Raad's PhD (2015–2018) co-supervised with **Nathalie Pernelle** (Paris Sud University), **Juliette Dibie** (AgroParisTech) and **Liliana Ibanescu** (AgroParisTech) and funded by the LIONES project of CDS-Paris Saclay. In the setting of Joe Raad PhD project, I also had the opportunity to work with **Frank van Harmelen** (VU, Amsterdam, Netherlands) and **Wouter Beek** (VU, Amsterdam, Netherlands).

Main Publications of the Chapter

[108] *Joe Raad, Wouter Beek, Frank van Harmelen, Nathalie Pernelle and* ***Fatiha Saïs***. Detecting Erroneous Identity Links on the Web Using Network Metrics. The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Proceedings, Part I, pages: 391–407, October 8-12, 2018.

[111] *Joe Raad, Nathalie Pernelle,* ***Fatiha Saïs****, Juliette Dibie-Barthélemy, Liliana Ibanescu and Stéphane Dervaux*. Comment représenter et découvrir des liens d'identités contextuels dans une base de connaissances: applications à des données expérimentales en science du vivant (2018). In Revue d'Intelligence Artificielle - Numéro Spécial d'Ingénierie des Connaissances 2019[to appear].

[110] *Joe Raad, Nathalie Pernelle and* ***Fatiha Saïs***. Detection of Contextual Identity Links in a Knowledge Base. In proceedings of the The Ninth International Conference on Knowledge Capture K-CAP 2017: 8:1-8:8, December 4th-6th, 2017.

[99] *Laura Papaleo, Nathalie Pernelle,* ***Fatiha Saïs****, Cyril Dumont*. Logical Detection of Invalid SameAs Statements in RDF Data (2014). In the proceedings of 19th Knowledge Acquisition, Modeling and Management (EKAW), pages: 33-49, LNCS 8876, Springer 2014, pages 373-384, December, 2014.

CHAPTER 4

DATA ENRICHMENT

4.1 State of the Art

Despite the apparent large size of knowledge graphs, their completeness and consistency are far from being achieved (e.g. Freebase is missing statements for 63.8%, Wikidata for 60.9% and DBpedia for 49.8% of all entities considering a selected set of properties used to describe books, such as language, publisher or number of pages [149]), making their efficient use problematic. Hence, to efficiently exploit information from different knowledge graphs, applications need to address two crucial problems: data quality and data enrichment. The two are tightly interrelated: enriching a KG with data coming from different sources aims at improving the quality of the KG content, as it is expected to be more complete and consistent. On the other hand, data enrichment strategies rely on data quality criteria (e.g. data frequency, source reliability) to solve conflicting values, caused by the variety of data quality across data sources. That is, data quality problems come from data publishing errors, data being outdated or inaccurate. Moreover, usually KGs are built by either using (semi-) automatic extraction methods from existing structured knowledge sources like wikipedia or by relying on the contribution of humans, like domain experts or the crowd. Even though if the reliability of these sources can be rather good, they do not always give complete descriptions about the entities and the information extraction processes may introduce errors due to the lack of accuracy of such tools. This has led to KGs in which, many entities (not all the entities of the world are described), types, properties and values are erroneous or missing.

To overcome the KG incompleteness problem, several approaches have been proposed in the literature. First, information extraction, using patterns, from external sources that can be textual, like wikipedia pages [77] or structured such as wikipedia tables, HTML tables [149] are

applied. Second, rule mining methods such as AMIE [56] are used to make implicit knowledge explicit in knowledge graphs. Third, other approaches like [54, 88] exploit entity links that exist between several knowledge graphs and apply a data fusion process to fill the gaps in the entity descriptions.

Data enrichment may concern different information targets. It may concern either predicting missing entities, missing types for entities and missing property values (objects or literal values). In what follows, we focus our purpose on works dealing with missing property value prediction (for a survey on approaches dealing with other types of missing information prediction, see [102]).

Bellow we describe the related work by distinguishing between the approaches that enrich knowledge graph data by applying a data fusion process, and approaches that apply information prediction mechanisms to determine the missing property values.

4.1.1 Data Fusion for property values enrichment

Research on the data fusion problem begun over two decades ago in the field of relational databases. The survey of Bleiholder and Naumann [21] outlines the state of the art in this direction. However, these approaches defined for the relational model are not applicable as they are in the context of RDF knowledge graphs. This is due to several issues inherent to the RDF data model itself and to the quality of the RDF data published on the LOD. First, unlike the relational model, in RDF data model the properties (i.e. attributes) may be not functional (may have several values for the same URI) and properties can be object properties and take not only literal values but also URIs. Second, data quality problems namely, data incompleteness, errors, imprecision and data redundancy are very frequent in LOD datasets and thus may make data fusion more complex and less effective. Third, the big volume of RDF datasets (millions of RDF triples and thousands of properties in some ontologies) makes the scalability a fundamental aspect that should be considered by data fusion methods in RDF data model.

There are few approaches that dealt with data fusion in knowledge graphs. They can be characterized whether they are *instance based* referring to approaches where the decision is made according to properties of the value itself (e.g., frequency, homogeneity) as opposed to approaches *source metadata based*, where the choice is based on information of the data sources (e.g., reliability, freshness, information extractors confidence) Overall existing approaches considering RDF knowledge graphs attempt to evaluate the quality of each property value, by taking into account various measures based on the value itself and/or on data source metadata. In [54] the authors propose an approach which consists in establishing a confidence degree for each value, by calculating a number of quality criteria for each property. A combination of quality metrics is used to select the most appropriate value. A single value is chosen, the one with the highest quality score. In [88], the framework Sieve is introduced as part of the Linked Data Integration Framework (LDIF), to deal with data quality assessment and fusion. Concerning the fusion phase, Sieve handles conflicts with three strategies, based on the idea described

in [21]: (i) *conflict-ignoring* strategies, where conflicts are left to be resolved by the users; (ii) *conflict-avoiding* strategies, where one decision is uniformly applied to all attributes (example: always trust a specific source); and (iii) *conflict-resolution* strategies, where one of the existing values is *decided* or a new value is *mediated* (e.g., the average, or the maximum of all the given values). In [25], the authors developed an approach that learns conflict resolution strategies on a ground truth dataset. Strategies that minimize the distance to the ground truth values are selected to be used in Sieve framework. A similar framework named ODCS-FusionTool [89] is proposed to deal with the fusion of RDF data. Unlike other approaches that consider the data as already loaded (e.g. in an ETL system) ODCS-FusionTool, is able to operate when data is loaded as well as at query time. Finally, in [39], the authors presented a data fusion approach that is used to build the knwoledge vault knowledge graph. It exploits different extractors from the Web (e.g. text, HTML tables, HTML trees) to augment existing prior sources (i.e. Freebase). The fusion method computes a probability of a triple being true based on agreement between different extractors and prior source.

It is worth mentioning that, except from [39] approach where the uncertainty is modelled as probabilities, none of the existing frameworks have addressed the problem of uncertainty modelling inherent to the data fusion results. Moreover, none of the existing approaches have proposed an elaborated provenance model for the data fusion process.

4.1.2 Prediction-based Data Enrichment

This family of approaches can be referred to as link prediction, relation prediction or missing values prediction approach. They all aim to enrich the knwoledge graphs with new property instances, i.e. given a binary relation, predict couples of objects or couples of objects and literal values that may instantiate those relations. The existing approaches can be distinguished on whether they use an external source to extract/learn new property instances or use only the content of knowledge graph itself to predict and infer new property instance. In [102], the former category is called *external approaches* while the later is called *internal approaches*.

As internal approaches we can cite classification-based methods that, although initially designed for entity type prediction, they are also used to predict (with a high probability) missing property instances. In [128] the authors train a tensor neural network to predict relations by relying on chains of other relations, e.g. if a person is born in a city of *France*, then the approach can predict that the person is of citizenship of *France*. The approach is applied to FreeBase and WordNet. To enrich Yago, AMIE [56], a Horn rule mining approach is applied to predict (with a certain confidence degree) missing property instances. Finally, approaches like [154], use association rule mining methods to find relations or chains of meaningful relations between entities. Such approaches have been applied on Freebase and DBpedia.

As external approaches, methods like [76, 77] learn patterns in Wikipedia pages (abstract or entire articles) using Conditional Random Fields. A very common approach for predicting

relations using external corpora is based on a workflow starting from an entity linking step where the KG entities are linked to the text corpus using named entity recognition. Then, using the entity description in the KG, theses approaches look for text patterns that fit with existing relations and then apply them to find new relations. Such methods have been applied for Freebase by [90], for DBpedia by [8]. An analogous approach [145] considers the Web as an external corpus and uses web search to find new relations. Finally, because of the high level of noise that can be found in free texts corpora, some recent works [92, 103, 114] attempted to use the structured parts of texts, like tables or lists to extract relations for enriching knowledge graphs with new knowledge.

Even though, a significant progress has been made for missing value prediction to fill the gaps for knowledge graph enrichment, some important shortcomings are worth to be highlighted. None of both internal or external approaches for value prediction is able to deal with both symbolic and numeric value prediction. Additionally, none of them are aware of ontological knowledge. Finally all the existing approaches, are based on a learning phase which requires either manually labelled samples or big size of data available.

4.2 Contributions

I studied the problem of data enrichment in knowledge graphs in two different contexts.

First, in a context of data integration where data come from different sources and interlinked through identity links, I developed a data fusion approach [122] that determines a single representation for sets of interlinked resources. It is based on different data quality criteria for the selection of best quality values. More precisely, it establishes a confidence degree for each property value, by measuring a number of quality criteria (e.g. data homogeneity, source reliability) for each property, and combining them. Then, the values with higher confidence degrees are ranked higher. In [58], we presented an extension of this approach to be able to exploit ontology knowledge in the confidence degree computation and provide an elaborated provenance model. As the result of data fusion is derived from a combination of uncertain criteria (e.g. homogeneity, reliability) the result is inevitably uncertain. Thus, we used three different models for uncertainty modelling, namely, fuzzy sets in [122], possibility theory in [124] and belief functions in [36]. We conducted experimental evaluations on real datasets which showed the potential of the proposed approach.

Secondly, I studied the problem of knowledge graph enrichment when one considers a single knowledge graph in which several property values are missing. To overcome this incompleteness, in a context of decision support issues, I proposed an approach in [123] which enriches an existing knowledge graph by constructing new predictions for missing property values. More precisely, it relies on knowledge expressed by numerous existing causal rules that are extracted from various scientific publications in a specific scientific domain. The premisses of these rules express

experimental conditions in the form of a conjunction of binary predicates (e.g. *ProductQuantity*) represented by properties in the domain ontology. The conclusion of these rules is a binary predicate (e.g. *VitaminRate*) that is also a property in the ontology. Our predictive approach allows to predict values for symbolic (i.e. strings) and numeric properties while considering ontological knowledge. It is performed in two steps: a reconciliation step which identifies groups of similar rules expressing a common experimental tendency, and a prediction step which generates new rules, using both descriptions coming from experimental conditions and similar rules obtained in the reconciliation step. The method has been evaluated over a case study related to food science and has been compared to decision trees. It obtained better results in terms of accuracy, completeness and error rate which showed the relevance of such a data reconciliation-based prediction approach.

4.3 Multi-criteria Data Fusion

In this section, I present our data fusion approach presented in [36, 58, 122, 124]. I first present the problem statement and an illustrative example. Then, I describe the multi-criteria data fusion approach which relies on the use of possibility theory for uncertainty modelling. We also used other uncertainty modelling, namely fuzzy sets in [122] and belief theory in [36] which I will not detail here. Finally, I give some experimental results of the developed data fusion approach when applied on real datasets.

4.3.1 Problem Statement and Illustrative Example

We consider N different instances $i_1,\ldots,i_N$ representing ontology class instances, coming from M different sources $S_1,\ldots,S_M$, with $M \leq N$. Note that all sources represent knowledge graphs that share the same ontology. The considered instances of a given ontology class C and have common descriptions represented by a set $\mathbb{P} = \{P_1,\ldots,P_K\}$ of K properties. Figure 4.2 shows instances of the class $Experiment$, where the property set is $\mathbb{P}$ = {cookingTime, usedWater, hasComponent, concentrationVar}.

In the sequel we distinguish two kinds of properties: (i) *literal properties* that take as values literal values that can be numeric (e.g. $cookickingTime$ in Figure 4.1) or symbolic (e.g. $usedWater$ in Figure 4.1), and (ii) *hierarchized properties* that take as values concepts in the ontology (e.g. $hasComponent$ in Figure 4.1).

Example 4.1. *A small part of the class hierarchy of $\mathscr{C}$ is given as an example in Figure 4.1. The properties are pictured by dashed Arrows from the domain class to the range class (or data type). The partial order $\sqsubseteq$ is pictured by $\rightarrow$ and describes the subsomption relation. The equivalence relation $\equiv$ is pictured by the relation $equivTo$.*

We consider two knowledge graphs describing experiments on vitamin rate variation during the cooking of food products. Data is summarized in Figure 4.2.

We denote by $\mathscr{V}_k$ the set of possible values of the property P_k (numerical values, non-hierarchical values or hierarchical values). A given instance i_n can be described by a set $\{v_{n1},\ldots,v_{nK}\}$ of K values, where v_{nK} is the value of the property P_k for the instance i_n.

We consider a set of `owl:sameAs` links between the instances $i_1,\ldots,i_N$ of a given class of the ontology $\mathscr{O}$. From the set of linked pairs, groups of duplicated instances are then built by transitive closure. The obtained reconciled groups[1] provide a partition of $\{i_1,\ldots,i_N\}$. In Example 4.1, the pairs $\{idE11, idE22\}$, $\{idE22, idE23\}$ are considered as identical. The reconciled group built from these pairs is $\{idE11, idE22, idE23\}$. In the sequel, we consider that L groups denoted by $Rec_1,\ldots,Rec_L$ are obtained by data linking techniques (see Chapter 3); and the set of values

[1] A instance that is not duplicated forms a group by itself.

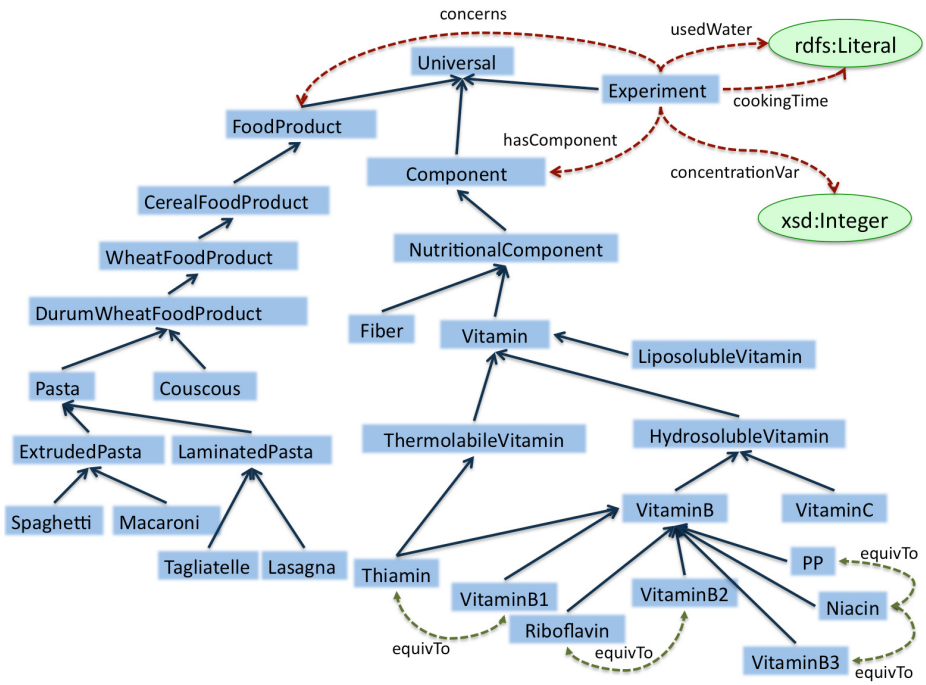


Figure 4.1: A part of the domain ontology

The set of identity links :

`{owl:sameAs(idE11, idE22), owl:sameAs(idE22, idE23)}`

KG_1:

Ref.	cookingTimeMin	usedWater	hasComponent	concentrationVar	exprimentDurationMin
idE11	12	Distilled water	Thiamin	-53.3	17
idE12	12	Tap water	Niacin	-45.6	15

KG_2:

Ref.	cookingTimeMin	usedWater	hasComponent	concentrationVar	exprimentDurationMin
idE21	13	Water	VitaminB6	-46	7
idE22	10	Deionized water	Thiamine	-52.9	14
idE23	10	Deionized water	VitaminB	-51.8	14

Figure 4.2: *Two interlinked knowledge graphs KG_1 and KG_2 containing data on the impact of cooking on vitamin level in pasta*

taken by a property P_k among a group Rec_l will be denoted by V_{lk}, with $k = 1,\ldots,K$ and $l = 1,\ldots,L$.

Data fusion problem consists in merging instances within each group, so that a unique instance is associated to each group (ending up with L instances). For uncertainty modelling, we propose to base this fusion on possibility theory. The method handles ontological knowledge whenever a property takes as value a concept of the ontology $\mathcal{O}$. For a given group Rec_l, data fusion consists in:

- first, build for each instance $i_n \in Rec_l$, a possibility distribution $\pi_{n,k}$ defined on $\mathcal{V}_{lk}$ and describing the uncertainty concerning the real value of property P_k. This step builds a possibility distribution defined over $\mathcal{V}_{lk}$ for each instance, ending up with $|Rec_l|$ distributions for each property;

- the $|Rec_l|$ distributions are then merged in a single one, so that to each property P_k inside a group Rec_l is associated with a unique possibility distribution.

I developed a data fusion method that is based on a set of criteria. These criteria correspond to information that is always available and that appears suitable to evaluate the relevance of a given property value. This allows the method to be general and applicable to the great majority of situations and problems where redundant instances can exist. However, in specific situations or problems, there could be additional criteria that should be considered. In what follows, I describe the approach when possibility theory is used to model the uncertainty of the results.

4.3.2 Data Fusion Method

The data fusion approach is performed in several steps:

Implausible values filtering. A filtering step is performed in order to determine and filter-out the property values that do not follow some of known domain constraints and some property typing constraints. For instance, if the property *cookingTimeMin* is typed as "*xsd:nonNegativeInteger*" then a negative value of it should be considered as implausible.

More-precise relation. This step consists in pairwise comparing the set of values of a given property in a given equality set and determines if there is a more-precise relation. To do so, we exploit: (i) syntactic comparisons between strings, (ii) subsumption relations and (ii) the mereology (*part-of*) relation. For the two former ones, we reasonably assume that we may assign to some properties whose values can be hierarchically organised, available hierarchies such as geographical classifications and concept hierarchies. Thus, the hierarchical structure is used to check whether a value is more precise than an other. For example, the value *"VitaminB"* is more precise than *"NutritionalComponent"*.

Synonymy relations. This step aims at determining synonymy relations between pairs of values of a given property in a given equality set. To do this, when it is not available in the domain ontology, we use available dictionaries like synsets of WordNet [2]. For example the value *"Thiamin"* is synonym of *"VitaminB1"*.

Incompatibility relations. This step aims at identifying the property values that violate the compatibility rules provided by a domain expert. These rules which may involve one or several properties. For example, in the description of $idE21$ the value *13* of the property

[2] https://wordnet.princeton.edu/

ob:cookingTimeMin is incompatible with the value 7 of the property *ob:exprimentDurationMin*.

Quality score computation. This step aims at computing the quality score for each plausible value. This score is obtained using the uncertainty model that follows either possibility theory as detailed bellow, fuzzy sets as we proposed in [122] or belief functions theory as presented in [36].

4.3.2.1 Data Quality Criteria

Several features contribute to the evaluation of the relevance of the property values: variability of encountered values, lack of data, abstract or concrete property, commonness of a given value, incorrect input, etc. Therefore, several criteria are used to build the uncertainty model.

Consider a given group Rec_l and a fixed property P_k. Let v be the value taken by P_k in the considered instance of the group Rec_l. The criteria are:

- *Conceptual Similarity* ***(CS)***: measures the semantic similarity between two classes. Here, we use the Wu & Palmer measure [146]. Let c_1, c_2 be two classes, N_1,N_2 the path lengths between $lcs(c_1,c_2)$ and respectively c_1 and c_2, and N_3 the path length between $lcs(c_1,c_2)$ and the class Universal. Then, $CS(c_1,c_2)$ reads[3]:

$$CS(c_1,c_2) = \frac{2\,N_3}{N_1+N_2+2\,N_3}.$$

  This criterion will be used to compare the values taken by two abstract properties, whose ranges are classes. Indeed, if one hierarchical value would have to be replaced by another one, the best replacement candidates are the one that are semantically closer to it.

- *Homogeneity* ***(Hom)***: measures the frequency of occurrence of a given value v inside a group of reconciled instances $i_n \in Rec_l$. This criterion is chosen for the reason that the more often a value appears in a group, the more likely it is to be the right one. Homogeneity reads:

$$Hom(v) = |\{v_{nk}=v | i_n \in Rec_l\}| / |Rec_l|.$$

- *Syntactic similarity* ***(Sim)***: we will denote by $Sim(v,v')$ a syntactic similarity measure (e.g. Jaccard, Levenstein) between two values v and v' taken by the property P_k in a group of reconciled instances. There are many such measures [30], and choosing a particular measure is often dependant of the nature of the data.

- *Data source reliability* ***(α_m)***: we consider that a reliability value α_m is associated with each source S_m, $m = 1,\ldots,M$, measuring the confidence we have in the information coming from this source. We consider that information coming from a highly reliable source should have more impact than the one coming from a poorly reliable one, without discarding completely any of these information. This reliability can be, for instance, a function of the last update date of the source [122].

[3]Note that Conceptual Similarity between two equivalent classes is 1.

- *Global frequency (**f**)*: measures the frequency of a value v among all the instances i_n, $n = 1,\ldots,N$. Indeed, a value appearing numerous times is less likely to contain typographic errors, and is more reliable. It is as follows:

$$f(v) = {}^{|\{v_{nk}=v | n=1,\ldots,N\}|}/_{N}$$

These criteria form a basis from which uncertainty can be estimated. They are significant and general enough so as to be accessible in most situations. Other criteria, more problem specific, can then be added.

4.3.2.2 Uncertainty modelling

Three cases can occur: P_k takes hierarchical symbolic values (it is an abstract property), non-hierarchical symbolic values or numerical values (in the last two cases it is a concrete property). We mainly concentrate on the first case, the two other cases being simpler to deal with.

Symbolic hierarchical values (hierarchized property). We assume that all values in $\mathcal{V}_{lk}$ are present in the ontology as classes. The order relation induced by CS values is a pre-order, since multiple values can have the same Wu & Palmer measure with respect to v. For $j = 1,\ldots,|\mathcal{V}_{lk}|$, a first possibility distribution $\pi'_{n,k}$ is built as follows:

(4.1) $$\pi'_{n,k}(v^{j,v}) =$$

$$\begin{cases} 1 & \text{if } j = 1 \\ \left(1 - \frac{f(v)}{\sum\limits_{v \in \mathcal{V}_{lk}} f(v)}\right)\left(1 - \frac{\sum\limits_{i<j} CS(v,v^{i,v})}{\sum\limits_{j=1}^{|\mathcal{V}_{lk}|} CS(v,v^{j,v})}\right) & \text{if } j>1 \text{ and } CS(v,v^{j,v})<CS(v,v^{j-1,v}) \\ \pi'_{n,k}(v^{j-1,v}) & \text{if } j>1 \text{ and } CS(v,v^{j,v})=CS(v,v^{j-1,v}) \end{cases}$$

In this distribution, the observed value is the most plausible. When the global frequency of this value v is high, other values are made less plausible (their possibility degree being inversely proportional to $f(v)$). The plausibility degree of other values than v are also made lower when their conceptual similarities with v are lower (note that equivalences and equalities of conceptual similarities are treated by the last case). $\pi'_{n,k}$ thus takes account of both conceptual similarity and global frequency.

The reliability α_m of the source S_m from which the instance comes is then used in a classical discounting operation, which consists in transforming, for all $v \in \mathcal{V}_{lk}$, the distribution $\pi'_{n,k}$ into:

$$\pi_{n,k}(v) = \max(1 - \alpha_m, \pi'_{n,k}(v)).$$

This is equivalent to make the information more imprecise when it is less reliable, thus reducing its impact on the final model (original information is kept if $\alpha_m = 1$ and has no impact at all if $\alpha_m = 0$).

Non hierarchical symbolic values. In this case, no hierarchical proximity has to be integrated to the uncertainty model, and we consider that $\mathcal{V}_{lk}^{v} = \{v^{1,v}, \ldots, v^{|\mathcal{V}_{lk}|,v}\}$ is indexed and ordered according to syntactic similarity of values with v, i.e., $i < j \Rightarrow Sim(v, v^{i,v}) \leq Sim(v, v^{j,v})$. The first distribution $\pi'_{n,k}$ is then computed by the same equation as Eq. (4.1), except that $CS(v, v^{j,v})$ is replaced by $Sim(v, v^{j,v})$. The discounting operation is then applied as in the hierarchical case.

Numerical values. Properties that take numerical values can possibly be subject to small variations between instances. They can be, for example, physical measurements coming out from experiments. In general, such numerical values concern physical parameters. In these cases, assume $[v^-, v^+]$ is the interval given by the source (precise values are retrieved when $v^- = v^+$). The possibility distribution $\pi_{n,k}$ modeling the uncertainty for this property and instance is then

$$\pi_{n,p}(v) = \left\{ \begin{array}{cc} 1 & \text{if } v \in [v^-, v^+] \\ 1 - \alpha_m & \text{if } v \notin [v^-, v^+] \end{array} \right\}. \tag{4.2}$$

Other numerical values such as postal code, customer number, ID number, *etc.* are treated as symbolic values without hierarchical structure.

Missing data. The treatment of missing data is a well-known problem. In the present method, modelling the ignorance about a property value P_k for i_n can be easily done, using the so-called vacuous (or non-informative) possibility distribution, that is the distribution $\pi_{n,k}$ such that, for each $v \in \mathcal{V}_{lk}$, $\pi_{n,k}(v) = 1$. This distribution can then be merged with the others, with the effect of increasing the final imprecision. Note that no additional assumptions has to be made about missing data in this method.

4.3.2.3 Fusion method using possibility theory uncertainty model

Given a group of reconciled instances Rec_l, we denote by i_{Σ_l} the single fused instance resulting from the fusion process. This fused instance will consist of K possibility distributions $\pi_{\Sigma_l,k}$ defined over spaces $\mathcal{V}_{lk}$, $k = 1, \ldots, K$ and obtained from the distributions described in Section 4.3.2.2.

There exists many rules to merge possibility distributions [43]. Here, using a simple arithmetic mean operator is a relevant choice, as it corresponds to a statistical counting and presents a natural way to integrate the homogeneity criterion in the final representation: a value will have all the more weight as it appears more frequently in the group of reconciled instances. For a property P_k and a group Rec_l, the final representation $\pi_{\Sigma_l,k}$ is computed, for all $v \in \mathcal{V}_{lk}$, as

follows:

$$\pi_{\Sigma_l,k}(v) = \sum_{i_n \in Rec_l} \frac{1}{|Rec_l|} \pi_{n,k}(v) \tag{4.3}$$

which is then made consistent by applying the following transformation to all $v \in \mathcal{V}_{lk}$: $\pi_{\Sigma_l,k}(v) = \pi'_{\Sigma_l,k}(v) / \max_{v \in \mathcal{V}_{lk}} \pi'_{\Sigma_l,k}(v)$. Once this fusion step is achieved, we end up with L final representations, where each property value is described by a possibility distribution reflecting our uncertainty about the real value.

4.3.3 Data Fusion Implementation

Since the calculation of the quality score and the selection of the appropriate value is more complex, we realize that offering explanations on the provenance of the fusion decisions is a useful feature. To achieve this, we found the reification mechanism, also used in [122], more flexible and suitable for our representation needs. The reification mechanism allows to enrich the RDF declarations by adding new elements. Substantially, it offers the possibility to create a metadata ontology describing existing RDF triples.

The data fusion metadata ontology. We introduce a data fusion metadata ontology in order to used the RDF reification mechanism[4] to annotate fused data by the quality information that are exploited to achieve the fusion decisions (see Figure 4.3). The rdf:$Statement$ class is enriched by the object property $hasQuality$. The main class $Quality$ has three object properties which organize the different quality aspects. The $hasCriteria$ property groups up all the measures used to calculate the quality score, the $hasRelations$ brings together the relations of the value with other values, and the $hasIndicators$ contains the remaining information.

A metadata document conforming to this ontology is produced by the system as a result of the data fusion algorithm. It provides descriptions for the quality measures of the value and the reasons why it was chosen or excluded (e.g., which rule it violates, the values that are more precise than it, etc.). However, the initial purpose of the metadata document is to be automatically queried. Depending on the quality scale of a value ($excellent, medium, poor$), different queries are intended to provide a "story" explaining the aspects of the value's quality.

An excerpt of data fusion metadata file. We give below an excerpt of the metadata file obtained when data fusion is applied on INA dataset (see details in section 4.3.4). We used the namespace *dfa*, standing for *data fusion annotation*. In this example, the two values for the property dfa:$first\_name$ are presented and annotated with all the useful information concerning their quality. Specifically, the implausible value *Jacues* contains only the information about its homogeneity, which is very low (0.015) and, thus, accounts for its exclusion from the list of plausible values. On the other side, for the plausible value *Jacques*, all the information

[4]Other contextual knowledge graphs models can be used in place of reification such as named graphs or singleton property

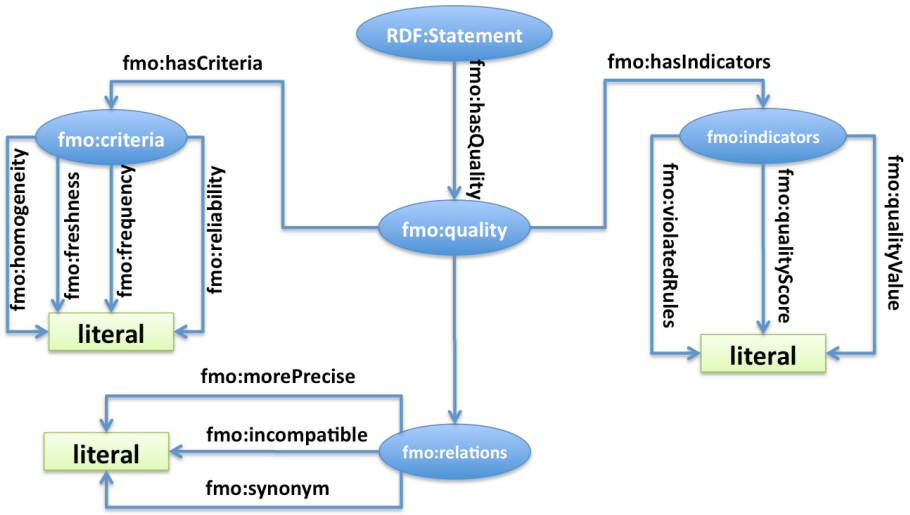


Figure 4.3: The Data Fusion Metadata Ontology

concerning the calculation of its quality score are contained in the file. For this example, if the system queries the metadata document for the value *Jacues*, the answer would be *"The value is implausible due to very low appearance in the data sources. Possible reason: misspelling."*, while for the value *Jacques* it will be *"The value is the only plausible one. It has a combination of high appearance in the data source and high level of trust on its data source"*.

```
PERSON-17123430093 rdf:type PhysicalPerson
v1 rdf:type Value
q1 rdf:type Quality
c1 rdf:type Criteria
PERSON-17123430093 dfa:first_name v1
v1 dfa:hasValue Jacques
v1 dfa:isImplausible false
...
PERSON-17123430093 dfa:first_name v2
v2 rdf:type Value
q2 rdf:type Quality
c2 rdf:type Criteria
v2 dfa:hasValue Jacues
v2 dfa:isImplausible true
v2 fmo:hasQuality q2
q2 fmo:hasCriteria c2
c2 fmo:hasHomogeneity 0.015
```

4.3.4 Data Fusion Evaluation

In this section I will present some experimental results of the fusion method when applied on real datasets. The first dataset was provided by our partner INA (National Institute of Audiovisual in France) in the Qualinca ANR project which concerns descriptions of TV shows. The second dataset concerns descriptions of scientific publications in computer science.

4.3.4.1 Experiments on *INA* dataset

This dataset contains descriptions of TV shows archived by INA. We considered a set of groups of 10819 reconciled instances (pairwise linked using $owl{:}sameAs$ links) that represent French famous persons and content notices where they are involved. In Table 4.1.(a) we show the distribution of the reconciled instances groups. These instances are described using different properties, as $aPourNom$, $aPourTitreCollection$, $aPourDateDiffusion$, and so on.

Table (a)

Person	# instances
Jacques Martin	10288
Philippe Bouvard	264
Daniel Prevost	214
Frederic Martin	26
Emmanuel Petit	12
Luis Fernandez	7
Michel Leclerc	6
Virginie Lemoine	2

Table (b)

	#values	%
#distinct values	14588	–
#isImplausible = “true”	9370	64.23 %
#isImplausible = “false”	5218	34.76 %
#qualityValue = “excellent”	2	0.04 %
#qualityValue = “medium”	3233	61.95 %
#qualityValue = “poor”	1983	38 %

Table 4.1: Table (a) Groups of reconciled instances; Table (b) First data fusion results

In Table 4.1.(b) we show the first results obtained by our data fusion approach. From the annotation file we extracted the number of distinct values, the number of values that are detected as implausible and the ones that are plausible thanks to the frequency computation. The three-valued scale (*excellent, medium, poor*) was validated bt one domain expert (a researcher in computer science), since it captures the intuitive idea of neutral / positive / negative score. It could be more precise (with 5 values for instance) but this would still be a refinement of the three-valued scale. For the quality value that is computed we used three values of thresholds to determine them:

- if qualityScore ≥ 0.67 then qualityValue = “excellent”.
- if 0.33 <qualityScore > 0.67 then qualityValue = “medium”.
- if qualityScore ≤ 0.33 then qualityValue = “poor”.

What we can observe from these results is that more than 64% of values are detected as implausible and considered as inappropriate for the corresponding properties. Furthermore,

	Article	**Conference**	**Person**
#Groups	124	134	68
#References	1295	1292	3521
#distinct-values-per-group	[1..5]	[1..28]	[1..37]
Avg(#disctinct values-per-group)	3	8	9

Table 4.2: *Cora dataset description.*

from the plausible values almost 62% of the values have a quality value that is *medium* what consolidates the results of the first step concerning the selection of the plausible values. More qualitative experiments are needed to better qualify the reasons why the 38% of values having a *poor* quality value appear as plausible.

4.3.4.2 Experiments on *Cora* dataset

The fusion method has been implemented and evaluated on the *Cora* data set related to the scientific publication domain. It has been used as a benchmark by several data linking approaches [40, 119]. Cora dataset is a collection of 1295 citations of 124 different research papers in computer science. These citations have been collected from the research engine Cora specialized on scientific publications search. We associate an instance to each article, conference and author (person). An article is described by several properties: *title*, *year*, *pageFrom*, *pageTo* and *type* which takes values in {proceedings, journal, book, ...}. A person is described by his *name* and a conference is described by three properties: *confName*, *confYear* and a *city*. There are two relations (objectProperties) which respectively link each article to its authors and to the conference where it is published.

We have applied the fusion method on the gold-standard of Cora dataset. It is organized as a set of 328 groups of pairwise reconciled instances for the three classes: article, conference and person. In table 4.2, we present some statistics of the characteristics of the gold-standard: the number of linked instance groups, the number of instances, the interval bounded by the minimum and maximum number of distinct values per group and the average of distinct values per reconciled group.

In order to assess the quality of the method, we compared, for a set of selected properties, the ranking of their values according to the confidence degree obtained by the fusion method with the ranking given by a human expert.

In some application domains, the identification of the right value can be purely subjective. For example, choosing between the two painting names "*La joconde*" and "*Mona-lisa*" is not obvious, as the two names are acceptable. Nevertheless, there are some obvious criteria that allow differentiating a right from a wrong value, which mainly consists in features that contribute to the syntactic integrity of the values:

– Typographical errors, like "*Cr**i**ptographic*" instead of "*Cr**y**ptographic*".

– Syntactic errors that are due to the data extraction processing, like, "*for - -mulae*" instead of

"*formulae*" or "***Bart (1993).*** *Reasoning with characteristic models*" instead of "*Reasoning with characteristic models*".

– Missing words, like "... *free probabilistic concepts*" instead of "... *free probabilistic* ***learning*** *concepts*"

– Additionnal words, like "***some*** *experiments with a new ...*" instead of "*experiments with a new*"

When the previous criteria do not help the expert to classify the values, the DBLP[5] browser is used to determine the right value and the wrong ones.

The second evaluation step consists in reviewing the list of ranked values of each property and classifying them, according to the previous criteria, into two classes: the right values and the wrong values. In the case of the Cora dataset, there is only one right value which satisfies the defined criteria. However, in some application domains they can be several values which can correspond to the right value in case of synonymies. The expert gives a ranking of the values by putting the right value in the top rank, i.e., before all the wrong values. The third step consists in comparing the ranked lists of values obtained by the fusion method with those given by the expert. In this step we count:

1. #*well-ranked-RV*: the number of well-ranked right values, that is the number of right values that appear in the top rank.

2. #*misranked-RV*: the number of misranked right values, that is the number of cases where the right value appears after one or several wrong values (it has a lower confidence degree).

A less strict evaluation protocol could be used: instead of considering the top rank of the value list, we can consider the top-k list of values and check if the right value belongs to this top-k list of values or not.

Qualitative results on *Cora* dataset. In Table 4.3, we give the results for the three properties which contain most of syntactic variations: *Title*, *ConfName* and person *Name*. We compute the precision for the right values as the proportion of the number of well-ranked values in reconciled groups [6].

We note that the recall value is equal to the precision because of the strict evaluation protocol. Indeed, as we consider that the fusion method fails when the right value does not appear in the top position, the recall value corresponds also to the proportion of the well-ranked right values in the reconciled groups.

The results of Table 4.3 show that the fusion method has obtained a precision of 93.9% for the ranking of the right values of article title. It obtains also a precision of 74.2 % for the ranking

[5]The DBLP Computer Science Bibliography which provides bibliographic information on major computer science journals and proceedings.

[6]We have considered the reconciled groups where the size of value list is (≥ 2).

	Article-Title	**Conference-Name**	**Person-Name**
#reconciled-groups	66	66	44
#well-ranked-RV	62	49	33
#misranked-RV	4	17	11
Precision=Recall	93.9%	74.2%	75%

Table 4.3: Fusion results in terms of precision for the values of: Title, ConfName, Name

of conference name and of 75% for the person names. We can notice that the precision for the conference names and for the person names are lower than the precision for article titles. This can be due to the important rate of syntactic variation in their corresponding possible values.

As it is shown in Table 4.2, the number of distinct values of the conference names variates between 1 to 28 values and between 1 to 37 for the person names. For the conference names, the variations are mostly caused by abbreviations (e.g. proc./proceedings, symp./symposium), a variety of codifications (e.g. 9th/ninth) and extraction problems (e.g. net-works/networks). For the person names, even when only considering the English-speaking world, a name can have several different spelling forms for a variety of reasons. In the Anglo-Saxon region and most other Western countries, a personal name is usually made of a given name, an optional middle name, and a surname or family name. Hispanic names can contain two surnames. For example, in the dataset we have 11 variations for the person name *Umesh Virkumar Vazirani*: {*Umesh Vazirani*; *U. Vazirani*; *Umesh V. Vazirani*; *Vazirani U.V.*; etc.}. Hence the main difficulties arise from what we could consider as abbreviations or synonyms. We could therefore improve our results by declaring such values as synonyms in the ontology. However, the results about article title show a very good recognition rate in case of a strict evaluation protocol. We can guarantee that the results can only be better for a top-k evaluation.

Thanks to these experiments we have shown the relevance of a such multi-criteria data fusion method. However, a more extensive evaluation, involving more human experts and using crowdsourcing frameworks, is needed to evaluate the method performances in a large scale and heterogeneous settings.

4.4 Data Reconciliation-based Missing Property Values Prediction

Experimental conditions sometimes only differ by a small variation of one experimental parameter, which may be fundamental in the case of a highly discriminant parameter, but negligible for others. Hence, knowledge that both (i) concern close experimental conditions and (ii) show similar results may be identified, which is a reconciliation problem. Such reconciled knowledge have a semantics, since they express a common experimental tendency. In addition to the experimental knowledge, general domain knowledge is available, and it has been modeled in an ontology. The ontology includes a vocabulary organized by subsumption, disjunction and synonymy relations. Moreover, it provides less common information concerning the status of concepts, such as functional dependencies and discriminance of concepts for prediction.

In this section, I present the approach that I developed for generating predictions by relying on case-based and reconciliation methods, using an ontology. The approach has been tested within a food science application concerning food quality management in the cereal agri-food chain and it has been compared to a classic predictive technique.

4.4.1 Causality Rule Prediction Problem

We consider the ontology depicted in Figure 4.1. It gives a small part of the set of concepts $\mathscr{C}$, partially ordered by the subsomption relation (pictured by '$\rightarrow$'). We consider also a set of disjunctions axioms such as: ($FoodProduct \perp Component$), ($Spaghetti \perp Macaroni$), ($Fiber \perp Vitamin$), ($Vitamin\ C \perp Vitamin\ B$), ...

Note that the considered ontologies are not restricted to trees, they are general graphs. This is an important feature of our work with respect to previous approaches, such as [153], where only trees are considered.

Relationship between ontology concepts and experimental variables. We consider a set of experimental descriptions containing K variables. Each variable X_k, $k = 1,\ldots,K$, is associated with a concept $c \in \mathscr{C}$ of the ontology $\mathscr{O}$. Each variable can be instantiated by a value that belongs to the definition domain of concept c.

Variable discriminance. For each variable X_k, a discriminance score, denoted by λ_k, is declared. It is a real value in the interval $[0;1]$. It is obtained through an iterative approach performed by domain experts, as explained in detail in [138].

Domain Rules. Each domain rule expresses the relation cause-to-effect between a set of parameters (e.g. *KindOfWater*, *CookingTimeMin*) of a unit operation belonging to the transformation

Id.	CookingTimeMin	KindOfWater	Component		ConcentrationVariation (%)
R1:	12	Tap water	Riboflavin	⇒	-53.3
R2:	12	Tap water	Niacin	⇒	-45.6
R3:	12	Tap Water	Vitamin B	⇒	-45.6
R4:	24	Tap Water	Fiber	⇒	-45.6
R5:	13	Water	Vitamin B6	⇒	-46
R6:	10	Deionized water	Thiamin	⇒	-52.9
R7:	10	Deionized water	Vitamin B1	⇒	-51.8
R8:	15	Distilled water	Vitamin B2	⇒	-45.5

Table 4.4: *A set of causality rules*

process, and the variation of a given product property (e.g. variation of *vitamin content*). The set of domain rules is denoted by $\mathscr{R}$.

Definition 4.1. (Causality rule). A causality rule $R \in \mathscr{R}$ is defined by a pair (H, C) where:

- $H = \{(X_1, v_1), \ldots, (X_h, v_h)\}$ corresponds to the rule hypothesis. It is a conjunction of variable/value criteria describing a set of experimental conditions in the form $(X_i = v_i)$. The value v_i may take numeric or symbolic (flat or hierarchized) values.
- $C = (X_c, v_c)$ corresponds to the rule conclusion that is composed of a single variable/value effect describing the resulting impact on the considered property.

It is interpreted by:

$$(X_1 = v_1) \wedge (X_2 = v_2) \wedge \ldots \wedge (X_h = v_h) \Rightarrow (X_c = v_c)$$

Table 4.4 shows a part of the set of rules of the considered case study. The experimental variables are given in the first line and the values of these variables are given in the other lines of the table. The last column represents the conclusion part of the rules (i.e., the variable X_c and its values). For example, the rule $R1$ given in the second line is interpreted as follows:

(CookingTimeMin = 12) ∧ (KindOfWater = Tap Water) ∧ (Component = Riboflavin)) ⇒ (ConcentrationVariation = -53.3).

In the following, domain rules will be compared on the basis of the set of variables they have in common, called common description and defined as follows.

Definition 4.2. (Common description). A common description between two causality rules R_1 and R_2, denoted by *CommonDesc(R_1, R_2)*, consists of the set of variables appearing at the same time in the set of variables/values of R_1 and R_2. Let H_1 and H_2 be the hypotheses of the rules R_1 and R_2 respectively. Let C_1 and C_2 be their conclusions.

$CommonDesc(R_1, R_2) = \{X \mid \exists v_1, v_2,\ [(X = v_1) \in H_1 \ \ and (X = v_2) \in H_2] \ \ or\ [(X = v_1) = C_1\ and\ (X = v_2) = C_2]\}$.

We denote as $CommonDesc_H(R_1,R_2)$ the common description of R_1, R_2 reduced to their hypotheses, i.e. $CommonDesc_H(R_1,R_2) = \{X \mid \exists v_1,v_2, [(X = v_1) \in H_1 \; and \; (X = v_2) \in H_2]\}$.

4.4.1.1 Domain Rule Partitioning

This section presents a method for domain rule partitionning that partitions the set of rules by using techniques adapted from data reconciliation. We first give the general principle of rule partitioning approach. Then, we present how rules are filtered, compared and finally grouped into several groups of similar rules.

Principle. The general principle behind rule reconciliation consists in determining which rules can be considered as similar, thus providing a partition of the given set of rules. The partition is computed according to a specific similarity measure. Some rules are considered as similar if their similarity value is above a certain threshold, otherwise they are called dissimilar. Similar rules are positioned in the same reconciliation group (partition subset).

Obviously, the key problem is related to the definition of an appropriate similarity measure which must take into account several features. In the literature, classic similarity measures are used for basic parameter values as in [30]. The choice of these similarity measures is done according to the features of the values, e.g. symbolic/numeric, length of values, and so on. It is also possible to consider a semantic similarity measure that exploits the hierarchy distance of the values in a given domain ontology (see Wu and Palmer measure [146]). In a complementary approach, additional ontological knowledge, as synonymy relations between concepts, can be exploited.

Ontology-based filtering step A pre-processing step relies on knowledge declared in the ontology. In this step, we exploit the semantics of the disjunction relations between concepts. For example, the concepts *ExtrudedPasta* and *LaminatedPasta* of Figure 4.1 are declared as disjoint. In this step we claim that two rules containing disjoint values for hypothesis variables cannot belong to the same group, and they are defined as “disjoint”.

4.4.1.2 Variable relevance in rule similarity computation

In order to associate each variable X_k with a score λ_k reflecting its discriminance power, there are two possible strategies: (i) ask a domain expert to specify such knowledge or (ii) obtain it automatically by using a learning method. In this work, we adopted both approaches in a complementary way. Thus the computation of the score λ_k is adapted from the N2R method proposed in [119], but it is determined using both declared expert knowledge and supervised learning, in the following way.

– Functional dependencies are declared knowledge, obtained through a collaboration with domain experts. For instance, in the considered application, a subset of the variables describing the experimental conditions has been identified by the experts as determining the *ConcentrationVariation* variable, i.e. the one to be predicted. These variables are: *Temperature*, *SaltPercentage*, *CookingTimeMin*, *KindOfWater*, *Component*, *AdditionOfIngredients*. The *FoodProduct*, *hasMethod*, and *ValueBefore* variables do not belong to this list. We proposed to take into account this knowledge by associating a low discriminance score with the latter variables (the chosen weight is 0 in the practical evaluation of section 4.4.3, i.e. the variables are ignored).

– In order to distinguish between the variable importance among those variables that belong to functional dependencies an iterative method in interaction with experts is applied (see [138] for details).

Similarity measures In order to compute the similarity of two causality rules, a prerequisite is to measure the similarity of the pairs of values, in an arbitrary order, belonging to their common descriptions.

Since there is no universal measure which can be qualified as the most efficient for every kind of value, the choice of a suitable similarity measure depends on the features of the considered basic values, as for example, symbolic/numeric values or length of the values.

Below, I outline that three cases may be distinguished, depending on the kind of the definition domain of the considered variables.

Hierarchized symbolic variables: We retained the Wu and Palmer measure, which is more intuitive and easy to implement. Its principle is based on the length of the path between two concepts in the hierarchy.

'Flat' (non hierarchized) symbolic variables: We use the Braun & Banquet similarity measure (see [80] for more details).

Numeric variables: Let X be a numeric variable, v_1 and v_2 two values of $Range(X)$. The similarity measure we use is defined in a classical way, as the complement to 1 of a normalised distance:

$$Sim(v_1, v_2) = 1 - \frac{|v_2 - v_1|}{|Range(X)|}.$$

Similarity measures between two causality rules In this work, the similarity between two rules is computed as a weighted average of the similarity scores of the values taken by their common variables. We denote as $Similarity(R_1, R_2)$ the real function which measures the similarity between two causality rules R_1 and R_2. It is computed as follows:

Definition 4.3. (Similarity of two causality rules).

$$Similarity(R_1,R_2) = \sum_k \lambda_k * Sim(v_{k_1},v_{k_2}),$$

where $k \in [1\ ;\ |CommonDesc(R_1,R_2)|\ ]$, λ_k is the discriminance power of the variable $X_k \in CommonDesc(R_1,R_2)$, and v_{k_1},v_{k_2} are the values taken by X_k in R_1 and R_2 respectively.

4.4.1.3 Partitioning the set of rules into reconciliation groups

To decide which rules must be reconciled, we use a similarity threshold [7] and assign to the same group each pair of rules having a similarity score beyond this given threshold. Rules having a similarity score beyond the threshold express a common experimental tendency.

We denote as $Reconcile(R_1,R_2)$ the predicate which expresses that the causality rules R_1 and R_2 express the same experimental tendency and must be reconciled.

Finally, to obtain disjoint groups of rules, we compute a transitive closure for the set of reconciliation decisions previously computed.

4.4.2 Prediction of New Causality Rules

Principle. We consider G and R. G is the set of reconciliation groups computed as presented in the previous section, and R is a new rule whose conclusion value is still unknown, and which has a set of variable/value criteria corresponding to its hypothesis. The new rule R is called an "unknown output rule", denoted by "UO-rule", and it is in the following form:

$R = (H,C)$ with $H = \{(X_1 = v_1),\ldots,(X_H = v_H)\}$ and $C = (X_c = x)$, where x is unknown.

The objective of this step is to predict the conclusion value x of the UO-rule R, that is, the expected variable/value effect, by comparing it to the existing known rules.

The method consists in selecting the closest reconciliation group and then using to predict the conclusion value x.

To perform this stage, R is compared to representative member(s) – called the "kernel" – of each reconciliation group. R is then assumed to be part of the group g whose kernel contains the rule the most similar to R. The value of the conclusion of R is predicted as a combination of the conclusion values of the rules of g.

Kernel Rule(s) of a Group Both for combinatory and for semantic reasons, a *kernel* is computed for each reconciliation group, which corresponds to the set of rules that are most representative of the group. From a combinatory point of view, comparing the UO-rule R to the kernel of each group is of course simpler than comparing it to the whole set of rules. From a semantic point of view, the kernel can be seen as a characteristic sample of a reconciliation group. Aggregating

[7] We note that the threshold is fixed experimentally by exploiting the size and the homogeneity nature of the rule set (see Section 4.4.3).

multiple knowledge sources by retaining a well-chosen piece of them is a common feature in knowledge fusion [44].

Definition 4.4. (Kernel). Let g be a group of rules and $R_i \in g$. The representativity of R_i is measured by:

$$Repr(R_i) = \sum_{j=1\ (j\neq i)}^{j=|g|} Similarity(R_i, R_j).$$

The kernel of g is then defined by:

$$Kernel(g) = \{r \in g \mid Repr(r) = \max_{R_i \in g} Repr(R_i)\}.$$

The most representative rule is thus chosen (or several ones in case of *ex aequo*). Note that the set $Kernel(g)$ is often reduced to a unique rule.

Allocation of R to the closest reconciliation group To compare R with the rule(s) of $Kernel(g)$, a similarity score is computed as in the Definition 4.3. The difference is that only the variable/value criteria appearing in the rule hypotheses are taken into account, since the conclusion value is unknown for R.
R is then allocated to the group g whose kernel contains the rule the most similar to R.

Definition 4.5. (Group allocated to a UO-rule). Let G be a set of reconciliation groups and R a UO-rule. R is allocated to a group $g \in G$, which satisfies the following condition:

$$\exists R' \in Kernel(g), Similarity(R, R') = \max_{r \in Kernel(g_i), i \in [1;|G|]} Similarity(R, r).$$

4.4.2.1 Prediction Method

To predict the conclusion value x of R, two cases are distinguished, according to the nature – symbolic or numeric – of the conclusion variable X_c.

Symbolic case. It is the case where the definition domain of the considered variable is hierarchized or flat symbolic. The conclusion variable X_c may take several values in the reconciled rules of group g. We make the assumption that the value x to be predicted figures among those already appearing in the group g, i.e. that X_c does not take a new value in the conclusion of R.

For each distinct value v_i taken by X_c in the group g, a confidence degree $conf_{v_i}$ (a real value in [0;1]) is computed. It can be interpreted as the confidence in the prediction that "the value taken by X_c in R is v_i". It is defined as the ratio between the sum of similarity scores between R and the rules where X_c takes the value v_i, and the total sum of similarity scores between R and the rules of g, that is:

$$conf_{v_i} = \frac{\sum_{\{r \in g | X_c = v_i\}} Similarity(R, r)}{\sum_{\{r \in g\}} Similarity(R, r)}.$$

The value with the highest confidence provides the prediction of the conclusion value x:

$$(\tilde{x} = v)\ such\ that\ conf_v = \max_i\ conf_{v_i},$$

where $\tilde{x}$ denotes the value predicted for x.

Numeric case. The prediction of the value x taken by X_c in R is computed as a weighted mean of the values taken by X_c in all the rules of the group g. The weight associated with each rule r of g is the similarity score between R and r. Let v_r be the value taken by X_c in r, the value x is predicted by:

$$\tilde{x} = \frac{\sum_{r \in g}(Similarity(R,r) \times v_r)}{\sum_{r \in g} Similarity(R,r)}.$$

A confidence degree $conf$ is attached to the prediction. This confidence degree is the weighted mean of the similarity scores between R and the rules of g:

$$conf = \frac{\sum_{r \in g}(Similarity(R,r))^2}{\sum_{r \in g} Similarity(R,r)}.$$

The algorithms are detailed in [123].

4.4.3 Evaluation: Application in Food Science

In this section, the application domain and the adopted evaluation protocol are described. Then the results are presented, providing a qualitative and quantitative evaluation.

4.4.3.1 Context of the Case Study

We considered knowledge on the domain of transformation and properties of cereal-based food that are available through the international literature of the domain concerning the impact of the transformation process on food properties. More precisely, the following elements are described: – The “technical” information concerns the unit operations which are involved in transformation from the wheat grains to the ready-to-use food (e.g. grinding, storage, drying, baking);

– The “property” information define the criteria which are used to represent the properties of products, according to three aspects: organoleptic, nutritional and safety properties (e.g. colors, vitamins contents, pesticides contents);

– The “result” information provide the impact of a unit operation on a property (i.e. what happens at the “intersection” between an element of the technical information and an element of the property information).

For each unit operation composing the transformation process, and for each family of product properties, these pieces of knowledge have been expressed as causality rules [137]. Approximately 600 rules are available in the application.

Evaluation Protocol. The evaluation was made in collaboration with food science experts. The proposed predictive method was compared with a classic decision tree prediction approach. We used C4.5 recursive dichotomous partitioning [24] in the R software [74] *rpart* implementation. The methodology used to evaluate the proposed method followed six steps:

1. definition of requirements on the quantity and form of the test queries (i.e. UO-rules), so that the results are significant;
2. definition of requirements on the rule base, so that the results are interpretable;
3. definition of a set of test queries that satisfy the previous requirements;
4. definition of the parameters used by the methods;
5. execution of the set of test queries;
6. analysis of the results.

Twenty test queries are defined as UO-rule hypotheses. The objective is to predict their conclusion values (see [123] for more details on the test queries).

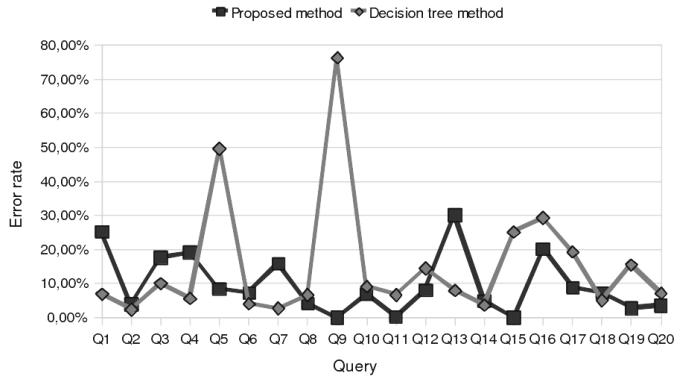
Method Parameters. The predictive method we developed was used with two different similarity thresholds for partitioning, respectively 0.8 and 0.9. The implementation used for decision trees is the R software with the *rpart* package for CART trees. The parameters of the *rpart* algorithm are: cross validation = 100, minimum instances per leaf = 6 (default value). Each of these methods was executed on two different rule bases. Rule base 1 contains 109 rules. Rule base 2 contains 117 rules. All of them concern the "cooking in water" unit operation and the "vitamin content" and "mineral content" properties. The queries were executed both using the reconciliation method presented in this section, and using the decision tree (DT) method.

Rule base 1			**Rule base 2**		
Proposed method		**Decision trees**	**Proposed method**		**Decision trees**
Threshold 0.8	**Threshold 0.9**		**Threshold 0.8**	**Threshold 0.9**	
10.53 %	9.68 %	15.37 %	17.87 %	13.19 %	21.39 %

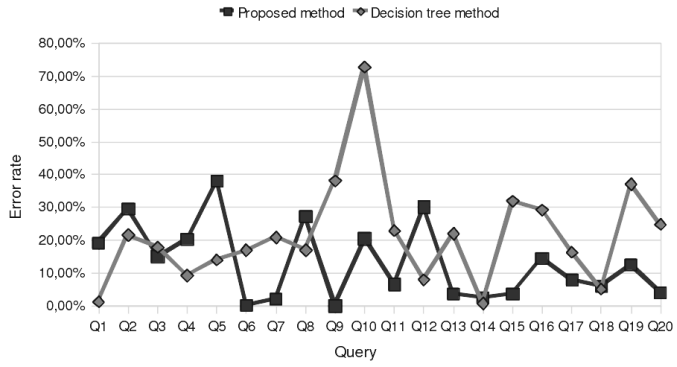
Table 4.5: Average error rates, for the proposed method vs decision trees (DT)

Result Analysis. In Table 4.5, the error rates obtained with each method are computed. These results clearly show a lower error rate obtained with the proposed predictive method, in all cases. The two tested thresholds were chosen experimentally. The obtained results tend to show that there is an optimum threshold of 0.9. The choice of the threshold impacts the number and size of the groups obtained in the reconciliation step of the method. They must neither be too numerous and small (as in the case of a high threshold) nor too few and large (as in the case of a low threshold).

With rule base 1, the number of obtained groups was: 14 for threshold 0.8, and 45 for threshold 0.9. With rule base 2, the number of obtained groups was: 5 for threshold 0.8, and 25 for threshold 0.9. When the obtained groups are few and large (threshold 0.8), we can notice that the obtained predictions are more homogeneous among the tested queries. On the contrary, when the obtained groups are numerous and small (threshold 0.9), the obtained predictions are more various among the tested queries. Figures 4.4 (a) and (b) present the error rates obtained query by query, with the proposed method for the optimum threshold 0.9, and with the decision tree method, respectively for rule base 1 and rule base 2.



(a)



(b)

Figure 4.4: Error rates obtained for each query with (a) rule base 1 and (b) rule base 2

We can make the following observation. The queries that obtained the most different results, if we compare both methods, are those for which exact or close answers were present in the rule base (such as for query $Q9$ in rule base 1), or those for which the closest answers, even if not so

close, are quite different from the rest of the base and show different trends (such as for query $Q10$ in rule base 2).

The latter result is not very surprising since sensitivity to outliers is a well-known drawback of decision trees, and a strong point of our method which relies on the identification of common tendencies. Let us recall that our interest in (i) the case-based and (ii) the decision tree approaches is motivated, as previously mentioned, by specific features that are not handled by other methods (or not simultaneously), namely (i) missing values, (ii) both numerical and symbolic values and (iii) a limited number of cases (here rules). The proposed method thus takes the best from case-based and reconciliation approaches, moreover it is aware of ontological knowledge, and an improvement may legitimely be expected. Here we can note that the decision tree strategy, which processes step by step by considering each variable separately, turns out to be less relevant than the proposed method, which considers the rules globally, involving all the variables.

4.5 Lessons Learned

In this chapter I presented my contributions to knowledge graph enrichment that consists in two approaches. The first is a data fusion approach which improves the data completeness by merging the descriptions of the resources that are determined as identical. The second approach exploits sets of grouped similar descriptions to predict values for numerical and symbolic properties, in a case of a knowledge base representing causality rules. This last has shown good results when compared with a baseline method based on decision trees. The quality of the results were evaluated through the error rate of the predicted values. For the first method we succeeded to conduct a partial evaluation of the results on terms of the quality of the provided value rankings. Indeed, validating the content of the enriched KG requires involvement of human experts whom, additionally, may have different judgements on the right values e.g., who is the US president *Barack Obama* or *Donald Trump*? Hence, as mentioned in [102] there is a need for approaches of KG enrichment which simultaneously deal with the correction and anomaly detection while enriching the knowledge graph content. That is, we may insure the simultaneous improvement of knowledge graphs for both completeness and correction. Furthermore, in addition to the subjectivity aspect on the choice of the right value, some dynamic properties (i.e., that evolve over time) are dependant on the context (e.g., temporal, spatial, social). Designing methods which make explicit the contexts of the values validity will allow to increase the accuracy and the interpretablility of the knowledge graph content. In an ongoing work [85], we proposed a first approach for temporal meta-facts generation in a knowledge graph, that can be represented through a reification mechanism or using more RDF suitable models such as named graphs [27] or singleton properties [96].

Main Collaborations and projects of the Chapter

The research work I presented in this chapter has been achieved thanks to the collaboration with **Rallou Thomopoulos (INRA Montpellier)** on the prediction of missing property values [121, 123] as well as data fusion problem [36, 58, 124] (partly in the setting of Qualinca ANR project (2012-2016)), and with **Sébstien Destercke (CNRS)** on data fusion problem [36, 124] with a focus on the uncertainty modelling of the data fusion results.

Main Publications of the Chapter

[121] ***Fatiha Saïs*** *and Rallou Thomopoulos*. A reconciliation-driven approach of case-based prediction: State of the art, method overview and application in food science. Book chapter in *Case-Based Reasoning: Strategies, Developments and Applications*, in Ed. Nova Science Publishers, Tao LIN (Ed.), 2015.

[58] *Ioanna Giannopoulou,* ***Fatiha Saïs*** *and Rallou Thomopoulos*. Linked Data Annotation and Fusion driven by Data Quality Evaluation. In Actes of the 15th days of "Extraction et Gestion des Connaissances", (EGC'2015), Revue des Technologies de l'Information - RNTI, du 27 au 30 Janvier 2015, Luxembourg.

[123] ***Fatiha Saïs****, Rallou Thomopoulos*. Ontology-aware prediction from rules: A reconciliation-based approach. In *Journal of Knowledge Based Systems* (2014), Elsevier. Volume 67, pages 117-130. April 2014.

[124] ***Fatiha Saïs****, Rallou Thomopoulos and Sébastien Destercke.* Ontology-Driven Possibilistic Reference Fusion. In the 9th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE 2010), Springer LNCS 6427, pages : 1079-1096. October 2010, Crete (Greece).

[36] *Sebastien Destercke,* ***Fatiha Saïs*** *and Rallou Thomopoulos*. Evidential reference fusion and flexible querying. Rencontres francophones sur la Logique Floue et ses Applications. November 2009, Annecy (France).

CHAPTER 5

KEY DISCOVERY

Ontology axioms like property functionality, class disjointness and keys express valuable knowledge and relevant data characteristics, thereby enabling reasoning and different data management tasks. Indeed, in the Web of data, (inverse) functionality and key axioms have shown their importance and relevance for data linking [6, 95, 118, 140], data profiling [1, 46], knowledge base fusion [41] and enrichment [106]. For data linking, it has been raised that when keys are used the precision of the results is significantly improved and the more keys we use the more identity links can be discovered.

Key axioms can be declared in OWL2 ontologies, through the *owl:hasKey* construct [113]. For a given class of an ontology, a key axiom allows one to represent the set of properties that uniquely identifies each instance of this class. However, in contexts like Web of Data containing large knowledge graphs with billions of triples (e.g. DBpedia[1] contains 1.5 billion triples in 2018) and hundreds of thousands of concepts (SNOMED CT[2] contains 311,000 of concepts in 2018), it is impractical for human experts to specify keys manually. Moreover, except for obvious IDs like ISBN for a book, a SSN for a person and a SIREN for a company, specifying the set of all minimal composite keys for a given dataset is a challenging task even for a human expert. One way to overcome this issue is to develop methods that discover automatically keys from available datasets.

In this chapter, I will present the state of the art on key discovery approaches and describe my contributions for this problem.

[1] https://wiki.dbpedia.org/
[2] http://www.snomed.org/

5.1 State of the Art and Contributions

Below, we first give some important definitions, semantics and properties of keys. Then we present the state of the art of key discovery in both relational and semantic Web fields. Finally, we present a summary of our contributions that we present in the remaining sections of this chapter.

5.1.1 Key Semantics and Properties

Declaring key axioms in an ontology is possible, since OWL2 has became a W3C recommendation. It can be declared through *owl:hasKey* construct as: *owl:hasKey* $(c)(dp_1,\ldots,dp_n)(op_1,\ldots,op_m)$ to express that the set of properties $P=\{dp_1,\ldots,dp_n,op_1,\ldots,op_m\}$ is a key for the class c. For sake of simplicity, in the sequel, we consider a set of properties $P=\{p_1,\ldots,p_n\}$ without distinguishing between data properties and object properties.

Two different key semantics are adopted by key discovery approaches in knowledge graphs. They depend on whether the approaches assume the Open World Assumption (OWA) or the Closed World Assumption (CWA) while dealing with multi-valued properties. OWA states if a piece of knowledge is not explicitly given or cannot be inferred then it is not false, while it is considered as necessary false under the CWA. An OWA-based key discovery approach considers that two instances are different if they disagree on any value for all the properties of the key. A CWA-based key discovery approach considers that two instances are different when they disagree on at least one value of the properties of the key. As we proposed in [11] we call the keys that are discovered by an OWA-based approach *Some-keys* and *Forall-keys* for the one discovered by CWA-based approaches.

In the following, we define two notions of keys: *Some*-keys that corresponds to the `hasKey` axiom of OWL2, and *Forall*-keys which has been adopted by approaches like [14, 129].

Definition 5.1 (*Some-key Semantics*). The semantics of a *Some*-key $\{p_1,\ldots,p_n\}$ for a class C is defined by the following rule:

$$\forall x \forall y \forall z_1 ... z_n (C(x) \wedge C(y) \wedge \bigwedge_{i=1}^{n} (p_i(x,z_i) \wedge p_i(y,z_i)) \rightarrow x = y)$$

Declaring that the set $\{p_1,\ldots,p_n\}$ is a *Some*-key for an atomic class C is denoted by *Some*-key$(C,(p_1,\ldots,p_n))$.

n in OWL 2 (c.f. Section 9.5 of [112] and Section 2.3.5 of [113]) enforces the considered instances to be named (*i.e.* they have to be URIs or literals, but not blank nodes). Hence, in the above definition the variables x and y can be unified by only contants and not blank nodes.

Some-keys do not require that the two instances x and y of C coincide on all values of the key properties to be equal: it suffices to have at least one pair of values that coincide for all p_i to decide that x and y refer to the same entity. However, in case of not functional properties that

represent a full list of items (e.g., a list of authors of a given paper, a list of actors of a given movie) it can be more meaningful to consider the fact that the instances should coincide for all the property values.

Definition 5.2 (*Forall-key*). The *Forall*-key for a class C is the rule defined as follows:

$$\forall x \forall y (C(x) \wedge C(y)) \wedge \bigwedge_{i=1}^{n} (\forall z_i (p_i(y,z_i) \rightarrow p_i(x,z_i)) \wedge$$
$$(\forall w_i (p_i(x,w_i) \rightarrow p_i(y,w_i))) \rightarrow (x = y))$$

Declaring that the set $\{p_1,\ldots,p_n\}$ is a *Forall*-key for an atomic class C is denoted by *Forall*-key$(C,(p_1,\ldots,p_n))$.

When there is no need to distinguish between the two semantics of keys, we denote $hasKey(C,P)$ to express that the set of properties P is declared as a key for the class C.

According to these semantics, one would expect that approaches that discover *Some*-keys being more conservative and then discover less rules but with better quality. While the approaches that discover *Forall*-keys find more keys but their use for data linking may lead to erroneous links.

Definition 5.3 (*Minimal Key*). A set of properties P of a given key is minimal if there is no subset P' of P, $P' \subseteq P$ that is a key.

Definition 5.4. (Non-keys). A set of properties $P = \{p_1,\ldots,p_n\}$ is a non key for the class C if:

$$\exists X \, \exists Y \, \exists Z_1,\ldots,\exists Z_n \, (p_1(X,Z_1) \wedge p_1(Y,Z_1) \wedge \ldots \wedge$$
$$p_n(X,Z_n) \wedge p_n(Y,Z_n) \wedge (X \neq Y) \wedge C(X) \wedge C(Y))$$

It is denoted by $nonKey(C,P)$.

Definition 5.5 (*Maximal non-Key*). A set of properties P of a given non-key is maximal if there is no super set P' of P, $P \subseteq P'$ that is a non-key.

5.1.1.1 Key Properties

Key axioms regardless of their semantics fulfil two properties. The first is key monotonicity which intuitively expresses that all super-sets of a key is also a key. It is formalized in the following rule:

$$hasKey(C,P) \Rightarrow \forall P \subseteq P' : hasKey(C,P') \tag{5.1}$$

The second property concerns the anti-monotonicity of the non-keys, i.e., the set of properties that does not form a key, and which intuitively expresses that every subset of a non-key is also a non-key. It is formalized in the following rule:

$$nonKey(C,P) \Rightarrow \forall P' \subseteq P : nonKey(C,P') \tag{5.2}$$

5.1.2 State of the Art

The problem of discovering keys has been intensively studied in the relational databases field and more recently in semantic Web field. Several approaches have been developed for automatic key discovery and more intensively for functional dependencies discovery, especially in relational data bases. In relational databases numerous and different methods [20, 66, 127] have been developed. They can be roughly classified into three main families: *schema-based* approaches, like [66], which enumerate possible combinations of attributes and check if they are valid keys, i.e. all the tuples agree with the key constraint; *data-driven* approaches, like [127] where the keys are built from the data and very recently, *hybrid* approaches [20] which combine schema- and data driven discovery techniques. However, these approaches defined for the relational model are not applicable in the context of RDF knowledge graphs. This is due to several issues inherent to the RDF data model itself and to the quality of the RDF data published on the LOD. First, unlike the relational model, in RDF data model the properties (i.e. attributes) may be not functional (may have several values for the same URI) and properties can be object properties and take not only literal values but also URIs. Second, data quality problems namely, data incompleteness, errors, imprecision and data redundancy are very frequent in LOD datasets and thus may make discovering keys more complex and in some cases impractical. Third, the big volume of RDF datasets (millions of RDF triples and thousands of properties in some ontologies) makes the scalability a fundamental aspect that should be considered by key discovery methods in RDF data model.

For the aforementioned reasons, several methods have recently been developed for discovering keys in RDF KGs. In 2011, we developed the first approach named KD2R [104, 135] that is able to discover keys from RDF KGs (see Section 5.2 for more details). Since then and thanks to the proliferation of key- and rule-based data linking approaches, other methods [12, 15, 129] have been developed for automatic key discovery from RDF datasets. Some are *schema-based* like [12, 15] and others use a refinement-based-operator to discover minimal keys like [129]. All these approaches consider datasets that conform to the same ontology (or mapped ontologies) except Linkkey approach [12].

In the following, I present the state of the art on key discovery approaches by classifying them according to the search strategies that are used during the key exploration phase, the pruning strategies, the key validity checking functions as well as the semantics of the discovered keys. I first give an overview of existing approaches of key and related dependencies in relation databases. Then, I will present the different methods of key discovery in knowledge graphs.

5.1.2.1 Keys and Dependencies Discovery in Relational Databases

The key discovery problem can be viewed as a sub-problem of functional dependency (FD) discovery (see [82]). Indeed, a FD ($X \rightarrow A$) states that of all pairs of tuples t_1, t_2 in a relation r

fulfil this condition $\forall B \in X$, $t_1[B] = t_2[B]$ then $t_1[A] = t_2[A]$. X is called *left-hand-side (lhs)* and A *right-hand-side (rhs)* of the FD.

FD and key discovery problem can be commonly expressed as enumerating all attribute combinations and checking their validity in the data. The search space is exponential and the complexity of a naïve discovery approach is of $\mathcal{O}(n^2(\frac{m}{2})^2 2^m)$ [82, 100] where n is the number of tuples in r and m is the number of attributes of r. The search space is usually modelled (but not built) as a power lattice where each node contains a unique set of attributes and is connected to nodes representing either its direct supersets or its direct subsets. Each level i in this lattice represents all the attribute sets of size i. In order to reduce this complexity, dependency and key discovery algorithms apply aggressive pruning strategies and sophisticated validation methods. In such setting, a popular data structure that is used to check the validity of FD or a key is the tuple partition π where each equivalent class represents a set of tuples sharing values fo a subset of attributes X. This structure is used to check if a FD $X \rightarrow A$ is valid when each equivalent classes of π is a subset of an equivalent class of the partition π' of an attribute A.

FD Discovery. TANE [71] is the first approach that has been developed for Functional dependency discovery. It is *schema-based* approach, applies a bottom-up level-wise lattice traversal and uses different pruning strategies: minimality pruning, key pruning and *rhs* attribute pruning (i.e., no more FD can be discovered from *rhs* attributes). It discovers exact FDs and approximate FDs (i.e., FDs that almost hold). Each approximate FD is associated to an error measure which is the minimal fraction of tuples to remove for the FD to hold in the dataset (when this measure gets 0 this means that the set X is a key). TANE has several variants where the authors attempted to optimize either the exploration phase or the validity checking function. For instance, DFD [2] proposes a depth-first and random walk traversal of the lattice.

Another family of approaches is the *data-driven* approaches that are based on the computation of *free-sets* [83, 147] that represent the sets of attributes where a FD cannot holds. These free-sets are then used to derive the set of attributes that can be involved in a FD. Finally, FDep [53] proposes another kind of approaches that is based on FD induction process. It starts with a general FD, i.e. all attributes functionally determine all others, and then specialize it by removing attributes and checking its validity.

Conditional FD Discovery. A conditional functional dependency (CFD) expresses a functional dependency between two sets of attributes that holds on a subset of tuples [28]. For example, a CFD could state that when two customers are based in the UK, the zipcode uniquely determines the city. A conditional key is a particular type of CFD, where the second set of attributes is a unique identifier for a record in the database. CFD discovery has been addressed in [28, 49, 61]. The work of [28] uses a breadth-first strategy inspired by the schema-based approach TANE [71]. FastCFD [49] finds a canonical cover of all minimal CFDs that satisfy a given support using a

depth-first strategy. Compared to [28] (which works well when the number of tuples is large), FastCFD [49] is efficient when the number of attributes is large.

Key Discovery. Different types of key discovery approaches have been proposed for relational databases: data-driven [127] and schema-based [66]. Gordian is a data-driven method [127] that allows discovering exact composite keys from relational data represented in a prefix-tree. To avoid scanning all the data, this method discovers first the maximal non keys and use them to derive the minimal keys. In [66], the authors propose DUCC, a hybrid approach where property combinations of a certain size are generated and then checked on the tuples. To this end, it exploits both the monotonic characteristic of keys and the anti-monotonic characteristic of non keys to prune the search space. To improve the efficiency of the approach, DUCC uses parallelization techniques to test different sets of attributes simultaneously.

Denial Constraints Discovery. Knowing that discovering FDs, keys and CFDs is still a complex and time consuming task, two recent holistic approaches FastDC [29] and Hydra [20] have proposed to discover a generalized form of these constraints called *denial constraints* (DC). A denial constraint is a logical formula that is expressive enough to capture the aforementioned constraints, i.e. FDs, keys, CFDs and ordering relations. It is expressed as a conjunction of clauses involving predicates, variables, constants and comparison operators (e.g. $=, \neq, \leq, \geq$, etc.) . That is, all these different constraints can be discovered in one pass, i.e. one exploration phase of the search space. FastDC [29] performs a pairwise comparison of set of tuples and form the evidence sets (i.e. the set of clauses that are satisfied by tuple pairs). Finally, it computes the minimal cover of the evidence set. Hydra [20] is more efficient approach which, instead of n^2 tuple comparisons, proceeds by sampling (random or focused sampling) and proposes a more efficient algorithm for deriving DCs from evidence sets.

These holistic approaches are promising for data profiling and data linking area, since they ensure good performances for DC discovery. However, they need to be extended to capture data quality characteristics (e.g. errors, redundancies and incompleteness). Furthermore, like the non holistic approaches, i.e. FD, key and CFD discovery approaches, this kind of approaches cannot be applied for data profiling and axiom discovery in knowledge graphs. This is due to the reasons that we mentioned before: open world assumption, graph-data (in contrast to single relation) and multi-valued properties.

5.1.2.2 Key Discovery in Knowledge Graphs

Thanks to the proliferation of data linking approaches and to the acknowledgement of key based approaches [5, 6, 95, 118, 140], several methods have been developed for the automatic discovery of keys from RDF knowledge graphs. We present bellow our existing competitors for

key discovery problem in RDF datasets, namely Atencia et al. [15], Rocker [129] and Linkkey [13]. These approaches can be roughly classified into two families depending on how they deal with multivalued properties [10]: *forall-key* approaches and *some-key* approaches.

Forall-key discovery approaches. *Forall-key* approaches, similarly to relational data bases approaches, discover keys by considering the Closed World Assumption (CWA), but only for multi-valued properties. They discover keys that fire when two entities share *all* values for each property. Inspired from TANE [71], Atencia et al. [14] developed a level-wise schema-based approach, that discovers minimal pseudo-keys (keys with exceptions) from RDF KGs. As in TANE, it considers a power lattice of all possible combinations of properties which it explores through level-wise bottom-up strategy. To check if a set of properties is a key, it builds a partition where each equivalent class represents the set of instances sharing the values for this set of properties. For multi-valued properties, an instance is added to an equivalent class if it has the same set of values than the other instances of this class. When the size of all equivalent classes of the partition is of size one then the considered set of properties is returned as a key. To capture possible errors and exceptions in data, Atencia et al. have defined two quality measures, key support and key discriminability. Thresholds are fixed to control the number of allowed exceptions, that is, if the support and the discriminability degree of a key is greater than these thresholds, then a pseudo-key is generated.
Rocker [129] is a refinement-operator-based approach that efficiently discovers minimal almost keys. This operator is defined over the space of property combinations of a given class. The combinations of properties that are ordered in an inverted directed tree, where the root is the empty set and the leaves are the minimal almost keys. The refinement operator is defined as a combination of two measures, namely the coverage and the discriminability of the properties. When the value of the refinement operator is equal to 1, for a given set of properties, then this last is considered as a key. To explore the candidate keys, Rocker applies a bottom-up traversal, i.e. starting from an empty set of properties and applying the refinement operator at each node then when the operator is of 1 (or greater than a threshold for almost-keys) all the sub-nodes are pruned thanks to the key monotonicity property.

Some-key discovery approaches. The *some-key* approaches discover keys that fire as soon as two entities share *at least one* value for each property. Some-key semantics suits with *owl:hasKey* semantics of OWL2. This kind of keys can be particularly useful under the Open World Assumption (OWA), where the KG may not contain all relevant facts. Thus, it is for example sufficient that two researchers share their last name, their first name, and one of their publications in order for them to be linked – even if the KG does not know all of their publications. Up to our knowledge, besides our key discovery approaches, namely KD2R [104], SAKey [131] and VICKEY [134] that we will describe in the following sections, Linkkey [13] is the only approach [13] that discovers some-keys. Linkkey generalizes the notion of key that is defined for one single ontology into keys

that are defined in terms of property mappings between two different ontologies. That is, a *key* (also called a linkkey by the authors) is a combination of a set of aligned properties defined across two not disjoint classes of two KGs that conform to two different ontologies. Linkkey approach proceeds, first, by generating a set of candidate keys. Then, to check the validity of a key, Linkkey computes different quality measures depending on the context: (i) supervised context, i.e. there is a sample of identity links, then an approximation of precision and recall is used to evaluate the ability of the key to generate a large number of identity links and, (ii) unsupervised context, the approach uses the coverage and the discriminability measures of the key to assess its quality. The originality of this approach resides in the fact that it is the only one that considers ontology alignments and that does not generate minimal keys. Indeed, the size of the keys can be arbitrary and the generated keys depend on their quality degree and not on their size.

5.1.3 Contributions

My contributions to the problem of automatic key discovery from RDF knowledge graphs have been materialized in three methods: KD2R [104], SAKey [131] and VICKEY [134]. These methods discover different kinds of keys and apply different strategies to explore the search space of possible keys. They can also be distinguished according to the assumptions on the data they consider, meaning, data incompleteness, errors, exceptions and redundancies in the data. Nevertheless they are based on several common fundamental characteristics: use of some-key semantics, non-key discovery first and Open World Assumption.

KD2R [104, 135], inspired from GORDIAN [127], is a method that allows to discover minimal exact some-keys that are valid for every instance of the KG. The algorithm, based on a prefix-tree, discovers first the set of maximal non-keys and maximal undermined keys. A maximal non-key is a combination of properties, which for at least two instances share values for these properties. The maximal undetermined keys are the combination of properties, because of data incompleteness, it does non fulfil the non-key definition neither the key definition. To deal with absent property values, two heuristics have been proposed, a pessimistic one, where absent values are considered to be likely to belong to the set of already provided property values, while the optimistic heuristic considers that the absent values must be different from the existing ones. Therefore, to derive the keys, since a key cannot belong to the set of maximal non-keys nor to the set of undetermined keys, KD2R computes the complement sets of the union of the maximal non-keys and undetermined keys and then applies a Cartesian product from which it extracts minimal keys. Thanks to the experiments on real datasets, we observed that KD2R does not scale to datasets with millions of triples especially when the pessimistic heuristic is used.

In order to gain in scalability and allow exceptions, that can capture data imperfections (e.g. errors, redundancies), we developed SAKey [131] that is a method that discovers almost-keys that are valid in the whole dataset except for n exceptions. SAKey discovers first the set of maximal $(n+1)$ non-keys that are defined as the set of properties that share values for at least $(n+1)$

instances. From this set of $(n+1)$ non-keys, it derives the set of n-almost-keys. To reduce the non-key search space, SAKey applies different pruning rules, such as, non-key anti-monotonicity and some semantic dependencies that can be discovered during the exploration phase. Then, an efficient key derivation algorithm is applied to determine the n-almost-keys from the $(n+1)$ non-keys.
Finally, we developed VICKEY [134] that is able to discover conditional keys that are valid on only a part of the data when no or few keys can be discovered. The discovered conditional keys can be declared in OWL2 as non-atomic classes (i.e. class expressions) using predicates like *owl:DataHasValue* or *owl:ObjectHasValue* that allow one to express conditions on the property values. In order to evaluate the quality of the conditional-keys, we use the *support* and the coverage. To avoid to scan the whole dataset for each possible conditional key, analogously to KD2R and to SAKey, VICKEY discovers first the set of maximal non-keys using SAKey. Then, for each given non-key, two subsets of properties are generated: the conditional part and the key part. If the condition part has a support that is greater than a fixed threshold then a Conditional Key Graph is created (CKG). Finally, from these CKGs VICKEY discovers all the minimal conditional keys, by starting from conditions of size one, of size two and repeats the process until all the minimal conditional keys are discovered. The monotonicity property of keys is applied to prune the search space.

With regard to the related work, it is worth to mention that KD2R, in 2011, was the first approach that has been developed for key discovery in knowledge graphs. Furthermore, up to know both SAKey and KD2R remain the only approaches that discover minimal some-keys in knowledge graphs. Besides this originality, both methods use new prunings to reduce the search space such as semantic dependencies in SAKey and key inheritance in KD2R. Finally, VICKEY is the first and the only approach that is able to discover conditional keys in both KGs.
The three methods, KD2R, SAKey and VICKEY have been intensively tested and evaluated on real datasets (e.g., nine datasets for KD2R) including datasets from the LOD like DBpedia and Yago with classes of millions of triples. They were evaluated according to their scalability, the impact of different prunings as well as the quality of the data linking results when the discovered keys are used.

5.2 KD2R: Discovery of Minimal Composite Keys

5.2.1 KD2R Overview

The most naive automatic way to discover keys is to check all the possible property combinations of a class. Assume that we have a class that is described by 15 properties. In this case, the number of candidate keys is $2^{15} - 1$. In order to minimize the number of computations, KD2R retrieves the set of maximal non keys (i.e. combinations of properties that share the same values for at least two instances) and then computes the set of minimal keys, based on the set of the discovered non keys. Indeed, to make sure that a set of properties is a key, we have to scan the whole set of instances of a given class. On the other hand, finding two instances that share the same values for the considered set of properties would suffice to be sure that this set is a non key.

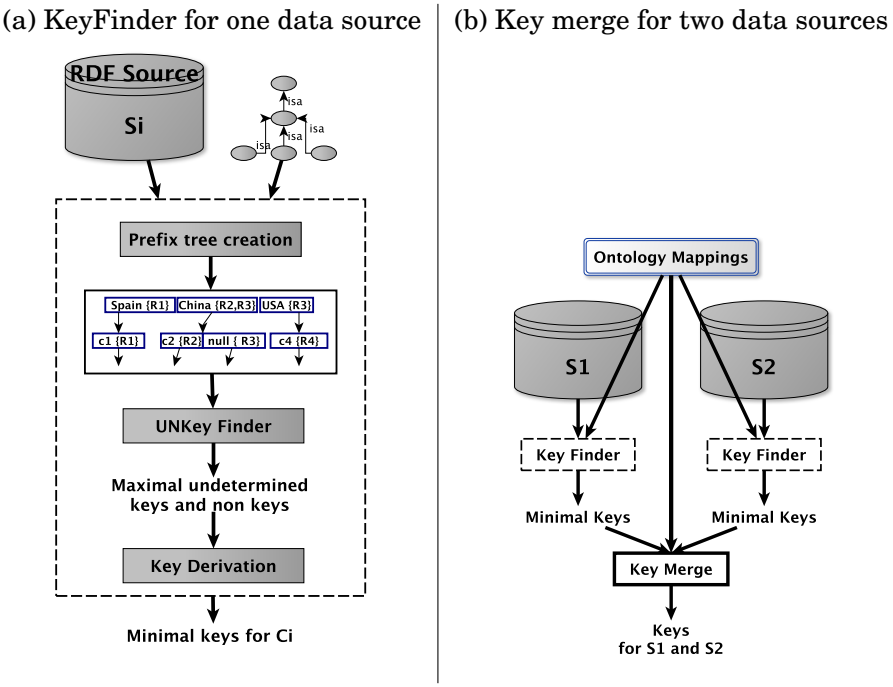


Figure 5.1: Key Discovery for two data sources

Since real RDF data sources might contain descriptions that are incomplete, we have defined the notion of undetermined keys which represent sets of property expressions that cannot be considered neither as keys nor as non keys.

In KD2R, distinguishing undetermined keys from keys and non keys, depends of the heuristic that is used to interpret not given property values:

1. *Pessimistic heuristic*: consider that the not given property values are all the values that appear in the data source for this property. Thus, KD2R provides non-keys, undetermined

keys and keys. Then, the combinations of properties that are not included in non-keys and neither in keys are undetermined keys.

2. *Optimistic heuristic*: consider the closed world assumption and assign to the not given property values an artificial null value that is different from all the values that appear in the data source for this property and from all the other artificial null values that have been already assigned for this property. KD2R, considers here the undetermined keys as keys, this is why it is called optimistic heuristic.

In Figure 5.1 we show the main steps of KD2R approach. It discovers the keys for each RDF data source independently. In each data source, KD2R is applied on the classes in a topologically sorted order. In this way, the keys that are discovered in the super-classes are exploited in the processing of their subclasses. For a given data source s_i and a given class c we apply $KeyFinder$ Algorithm which aims at finding keys for the class c that are valid in the data source s_i. KeyFinder method (see Figure 5.1(a)) allows one to retrieve in three main steps the minimal keys that can be added to the ontology, for each RDF data source conforming to an OWL ontology: (1) prefix-tree creation, (2) undetermined and non key discovery and (3) key derivation from the set of maximal undetermined keys and maximal non keys. The obtained keys are then merged in order to compute the set of keys that are valid for both data sources (see Figure 5.1(b)).

5.2.2 KD2R KEY DISCOVERY APPROACH

KD2R is a data-driven approach for key discovery. It discovers first non-keys (see Definition 5.4) and then derive keys (see Definition 5.2). Indeed, it is much faster to check if a set of properties is non key rather that checking if it is a key, which needs to scan the whole dataset.

Since real RDF data sources might contain descriptions that are incomplete, some combinations of properties are neither keys nor non keys. More precisely, a set of properties is called an *undetermined key* (*c.f.* definition 5.6) for a class if it is not a non key and there exist two instances of the class such that the instances share the same values for a subset of the properties. The remaining properties are unknown for at least one of the two instances. The undetermined keys can be validated by a human expert. Indeed, it allows a system to propose to the expert all the candidate keys that can be valid regarding to the dataset(s).

Definition 5.6. **(Undetermined Keys).** Let NK_C be the set of non-keys of the class C. A set of properties $uk_C = \{pe_1, \ldots, pe_n\}$ is an undetermined key for the class C:

– (i) $uk_C \notin NK_C$ and

– (ii) $\exists X \, \exists Y \, (C(X) \wedge C(Y) \wedge (X \neq Y) \wedge$

$$\bigwedge_{j=1}^{j=n} (((\exists Z \, (pe_j(X,Z) \wedge pe_j(Y,Z)) \vee \nexists W \, (pe_j(X,W) \vee \nexists W \, pe_j(Y,W)))))$$

We denote UK_C the set of undetermined keys of the class C. An undetermined key uk_C is maximal if it does not exist an undetermined key uk'_C such that $uk_C \subset uk'_C$.

KD2R algorithms. KeyFinder algorithm (see Figure 5.1), starts by computing the topological order of the classes by exploiting the subsumption relation between them. It exploits a possible set of inherited keys to compute the complete set of minimal keys for each class. For each class, KeyFinder builds an intermediate prefix tree which is a compact representation of the class instances in the data source. The intermediate prefix tree considers the cases where property values are not given in the dataset. Thus, an artificial empty value is created for those properties. We use the URI list to store the URIs for which the property value was unknown. This information is used by the undetermined- and non-key finder algorithm to distinguish non keys from undetermined keys. Then a final prefix tree is generated in order to take into account the possible unknown property values. Using the final prefix tree the $UNKFinder$ method is called to retrieve the maximal non keys and the maximal undetermined keys using inherited keys if there is any. Finally, KeyFinder computes the complete set of minimal keys for each class.

To ensure the scalability of the undetermined and non key discovery, UNKFinder performs three kinds of pruning:

(A) The subsumption relation between classes is exploited to prune the prefix tree traversal. Indeed, when a key is already discovered for a class using one data source, then this key is also valid for all the subclasses in this data source. Thus, parts of the prefix tree are not explored.

(B) When all the further new combinations of properties in a given path cannot lead to new maximal non keys then the exploration of this path stops.

(C) The monotonic characteristic of keys: when a node describes only one instance we are sure that adding more properties in the current path will not lead to a non key.

Finally, KeyFinder algorithm applies a key derivation step which allows to determine the set of minimal keys from the set of maximal undetermined- and non-keys. The main idea is that a key is a set of properties that is not included or is not equal to any maximal non key or undetermined key. Thus, for each maximal non key and undetermined key, we compute the complement sets. Then, the obtained properties are combined using a Cartesian product and only the minimal sets are kept.

When keys are discovered from two data sources that conform to two different ontologies, we compute the keys that are valid in both data sources. The keys are expressed using the common vocabulary. First, for each data source and class we delete from $K_{s.c}$ all the keys that contain properties that do not belong to P_{ic}, i.e. the set of mapped properties. Then, for each pair of equivalent classes we compute the Cartesian product between their set of minimal keys. Finally, we select only the minimal ones. This way we guarantee that the obtained keys are valid in both data sources.
For example, consider two data sources $D = \{s1, s2\}$,
if $K_{s1.db:Restaurant} = \{\{db:address\}$,

$\{db:name\}\}$ and
$K_{s2.db:Restaurant} = \{\{db:telephone,$
$db:city\},\{db:name\}\}$

then the multi-source keys will be:
$K_{D:Restaurant} = \{\{db:telephone, db:address, db:city\},\{db:name\}\}$.

Optimistic variant of KD2R. In the optimistic variant of KD2R we do not build the intermediate prefix tree and all the undetermined keys that could be discovered by the pessimistic approach are considered as keys in the optimistic approach. Thus we discover more keys.

5.2.3 KD2R Experiments and Evaluation

We applied KD2R on more than eight datasets (available at [3]) of different application domains (e.g., bibliographical catalogues, restaurants, films) and different sizes varying from several thousands of triples to several millions like DBpedia with more than 5.6 millions of triples. Each dataset contains two RDF data sources and two OWL ontologies. UNA is declared for each RDF data source. For each dataset, we discovered the keys using KD2R with both pessimistic and optimistic heuristics. Thus, we observed the changes in the obtained sets of keys, evaluated KD2R scalability and for some datasets evaluated the quality of the inferred identity links when the keys are used.

To evaluate the quality of the obtained keys we applied KD2R on datasets from OAEI2010 PR track contest for which the gold-standard is available. Then we provided the obtained keys to a data linking tool and evaluated the recall, precision and F-Measure scores.

We also compared links found using KD2R keys with links found using expert keys and without using keys. Then, we show that when we use the obtained keys in a data linking task, we obtain results that are better than those obtained without keys and comparable to those obtained using expert keys.

Scalability Evaluation. In order to show the scalability of KD2R, we applied KD2R on two datasets extracted from DBpedia[4]: the first dataset contains descriptions of persons and the second one concerns natural places. One of the characteristics of DBpedia is that the UNA is not fulfilled. We have observed that some persons are represented several times using distinct URIs, but in different contexts (e.g. one soccer-player is represented using several URIs, but for each URI the description concerns its transfer into a new club). Therefore, in such cases keys can be discovered.

[3] `http://www.lri.fr/~sais/KD2R-DataSets`
[4] http://dbpedia.org/Downloads37

On small data sources such as OAEI or GFT[5] (less than 10 000 triples), KD2R can be applied using the pessimistic or the optimistic heuristic. Nevertheless, on large datasets such as DBpedia persons (more than 5.6 millions of triples) or DBpedia natural places (more than 1.6 millions of triples), the pessimistic approach cannot by used (a time-out is reached). Indeed, such datasets contain a lot of properties that are rarely instantiated which leads to a final prefix tree that contains too many nodes (i.e. assignation of all the possible values to the artificial "null" values in the prefix tree). Hence, in such cases only the optimistic heuristic can be applied. Thus, in our experiments we have considered only the properties that are instantiated for at least T distinct instances.

Moreover, the pruning techniques enable KD2R to be more efficient and scalable in big datasets. Hence, we have experimentally evaluated on all considered datasets the gain in terms of efficiency brought by the different kinds of pruning that are used during the prefix tree exploration: key inheritance, non key antimonotonicity and key monotonicity. The pruning techniques enable KD2R to be more efficient and scalable in big datasets. The results showed that on five rather small data sources, the execution times of keyFinder (using pessimistic or optimistic) is less than 8 seconds. For the two DBpedia data sources, the execution times is less than 441 seconds. Thanks to the different kinds of pruning, less than 50% of the nodes of the prefix tree are explored for all datasets.

Evaluation of the key quality on OAEI2010 PR Track. To evaluate the quality of the KD2R keys we used our data linking tool LN2R [118] which allowed us to show the benefits of using discovered keys in the data linking process. More precisely, we have compared the results that are obtained by LN2R tool in four cases: (i) without keys, (ii) KD2R-O keys (with optimistic heuristic), (iii) KD2R-P keys (with pessimistic heuristic) and (iv) when keys are manually specified by an expert.

The results on Person dataset of OAEI 2010 (composed of two data sources) in terms of recall, precision and F-Measure are presented in Table 5.1 by varying the linking threshold $TRec$[6] from 1 to 0.8. We can notice that, for all values of $TRec$, the results obtained for the Person dataset are better when we use keys obtained by either KD2R-O or KD2R-P than when the keys are not used. When the threshold is bigger than 0.95 the F-Measure of LN2R using KD2R-O keys is 100%. This is an example that shows that the results using keys found with the optimistic heuristic can be better than the ones found with the pessimistic heuristic. In the restaurant dataset (that we do not show here), when $TRec \geq 0.9$, the F-measure is almost three times higher than the F-measure obtained when keys are not declared. This big difference is due to the fact that the recall is much higher when KD2R keys are added. Indeed, even when some property values are syntactically different, it suffices that it exists one key for which the property values are similar,

[5]Dataset from Google Fusion Tables `https://support.google.com/fusiontables/answer/2571232`)

[6]The instance pairs for which the similarity is greater than a given threshold $TRec$ are linked by an identity link.

to infer an identity link. Furthermore, our results are very close to the ones obtained using expert keys.

TRec	Keys	Recall	Precision	F-Measure
1	without	0%	- %	- %
	KD2R-O	100%	100%	100%
	KD2R-P	95.00%	100%	97.44%
	expert	98.40%	100%	99.19%
0.95	without	61.20%	100%	75.93%
	KD2R-O	100%	100%	100%
	KD2R-P	95.00%	100%	97.44%
	expert	98.60%	100%	99.30%
0.9	without	64.2%	100%	78.20%
	KD2R-O	100%	98.04%	99.01%
	KD2R-P	95.00%	100%	97.44%
	expert	98.60%	100%	99.30%
0.85	without	65.20%	100%	78.93%
	KD2R-O	100%	81.30%	89.68%
	KD2R-P	99.80%	100%	99.90%
	expert	99.80%	100%	99.90%
0.8	without	90.20%	100%	94.85%
	KD2R-O	100%	35.71%	52.63%
	KD2R-P	99.80%	100%	99.90%
	expert	100%	100%	100%

Table 5.1: KD2R: Recall, Precision and F-measure for Person@OAEI2010 data source

In table 5.2 we give a comparison between the results obtained by LN2R using KD2R keys with other tools that have used the Person-Restaurant (PR) dataset of OAEI 2010–Instance Matching track. We can notice that the obtained results in terms of F-measure are comparable to those obtained by semi-supervised approaches like ObjectCoref [70]. It is nevertheless less efficient than approaches that learn linkage rules that are specific to the dataset like KoFuss+GA.

Discussion. These experiments showed the usefulness of KD2R keys for data linking. They also allowed one to observe that the optimistic approach is more efficient and obtains keys with better quality. the pessimistic approach of KD2R is not applicable when data is sparse.

Dataset	LN2R+KD2R-P	LN2R+KD2R-O	ASMOV	LN2R	CODI	ObjectCoref	RIMOM	KnoFuss+GA
Person 1	0.99	1.00	1.00	1.00	0.91	1.00	1.00	1.00
Restaurant	0.728	–	0.70	0.75	0.72	0.73	0.81	0.78

Table 5.2: Comparison of F-Measure with other tools on PR dataset of OAEI 2010 benchmark

The optimistic approach has shown that relevant keys can be discovered. Nevertheless, when data is sparse and big KD2R is not applicable. Furthermore, when datasets contain duplicates or erroneous data, no keys can be discovered. Hence, in case of datasets with a large number of properties (e.g. DBPedia) the approach is not scalable. This is why we have defined a new algorithm named SAKey which is more scalable and able to discover keys with exceptions (see Section 5.3).

5.3 SAKEY: Scalable almost-Key Discovery

5.3.1 SAKey Overview

In the Web of data, it frequently occurs that RDF datasets contain erroneous data and duplicates. Thus, discovering keys in RDF datasets without taking into account these data characteristics may lead to loose keys or in the worst case to do not discover keys. Furthermore, there exist sets of properties that even if they are not keys, due to a small number of shared values, can be useful for data linking or data cleaning. These sets of properties are particularly needed when a class has no keys. This is why we defined a new notion of keys, *keys with exceptions* called *n-almost keys*. A set of properties is a n-almost key if there exist at most n instances that share values for this set of properties.

In [132] we presented an approach called SAKey that allows to automatically discover n-almost keys from RDF datasets. To check if a set of properties is a n-almost key for a class c in a dataset D, a naive approach would scan all the instances of a class c to verify if at most n instances share values for these properties. Even when a class is described by few properties, the number of candidate n-almost keys can be huge. An efficient way to obtain n-almost keys, as already proposed in [104, 127], is to discover first all the maximal sets of properties that are not n-almost keys and then use them to derive the minimal n-almost keys. Indeed, to show that a set of properties is not a n-almost key, it is sufficient to find only (n+1) instances that share values for this set. We call the sets that are not n-almost keys, *n-non keys*. We developed several filtering and pruning strategies to both reduce the in-memory size and to avoid unnecessary computations.

5.3.2 n-almost keys Discovery Approach

SAKey allows one to discover n-almost keys from an RDF dataset. It introduces a new notion of keys with exceptions called n-almost keys. A set of properties is a n-almost key if there exist at most n instances that share values for this set of properties. The dataset D1 in Figure 5.2 contains descriptions of films. Each film can be described by its name, the release date, the language in which it was filmed, the actors and the directors of the movie.

In D1 (see Figure 5.2) the property $db{:}hasActor$ is not a key for the class $Film$ since there exists at least one actor that plays in several films. Indeed, we notice that "$G.Clooney$" plays in

Dataset D1:
db:Film(f1), $db{:}hasActor(f1,''B.Pitt'')$, $db{:}hasActor(f1,''J.Roberts'')$,
$db{:}director(f1,''S.Soderbergh'')$, $db{:}releaseDate(f1,''3/4/01'')$, $db{:}name(f1,''Ocean's\ 11'')$,
db:Film(f2), $db{:}hasActor(f2,''G.Clooney'')$, $db{:}hasActor(f2,''B.Pitt'')$,
$db{:}hasActor(f2,''J.Roberts'')$, $db{:}director(f2,''S.Soderbergh'')$, $db{:}director(f2,''P.Greengrass'')$,
$db{:}director(f2,''R.Howard'')$, $db{:}releaseDate(f2,''2/5/04'')$, $db{:}name(f2,''Ocean's\ 12'')$
db:Film(f3), $db{:}hasActor(f3,''G.Clooney'')$, $db{:}hasActor(f3,''B.Pitt'')$
$db{:}director(f3,''S.Soderbergh'')$, $db{:}director(f3,''P.Greengrass'')$, $db{:}director(f3,''R.Howard'')$,
$db{:}releaseDate(f3,''30/6/07'')$, $db{:}name(f3,''Ocean's\ 13'')$,
db:Film(f4), $db{:}hasActor(f4,''G.Clooney'')$, $db{:}hasActor(f4,''N.Krause'')$,
$db{:}director(f4,''A.Payne'')$, $db{:}releaseDate(f4,''15/9/11'')$, $db{:}name(f4,''The\ descendants'')$,
$db{:}language(f4,''english'')$
db:Film(f5), $db{:}hasActor(f5,''F.Potente'')$, $db{:}director(f5,''P.Greengrass'')$,
$db{:}releaseDate(f5,''2002'')$, $db{:}name(f5,''The\ bourne\ Identity'')$, $db{:}language(f5,''english'')$
db:Film(f6), $db{:}director(f6,''R.Howard'')$, $db{:}releaseDate(f6,''2/5/04'')$,
$db{:}name(f6,''Ocean's\ twelve'')$

Figure 5.2: SAKey: Example of RDF data

films $f2$, $f3$ and $f4$ while "$M.Daemon$" in $f1$, $f2$ and $f3$. Thus, there exist in total four films sharing actors. Considering each film that share actors with other films as an exception, there exist 4 exceptions for the property $db:hasActor$. We consider the property $db:hasActor$ as a 4-almost key since it contains at most 4 exceptions.

The set of exceptions E_P contains the set of instances that share values with at least one instance, for a given set of properties P.

Definition 5.7. (Exception set). Let c be a class and $P = \{p_1, ..., p_n\}$ be as et of properties where $P \in \mathscr{P}$. The exception set E_P is defined as:

$$E_P = \{X \mid \exists Y (X \not\equiv Y) \wedge c(X) \wedge c(Y) \wedge (\bigwedge_{p \in P} \exists U \exists V\, p(X,U) \wedge p(Y,V) \wedge (U \equiv V))\}^7$$

For example, in D1 of Figure 5.2 we have: $E_{\{db:hasActor\}} = \{f1, f2, f3, f4\}$, $E_{\{db:hasActor, db:director\}} = \{f1, f2, f3\}$.
Using the exception set E_P we give the following definition of a n-almost key.

Definition 5.8. (n-almost key). Let $P \in \mathscr{P}$ be set of properties and n an integer. P is a *n-almost key* for the class c if $|E_P| \leq n$

This means that a set of properties is considered as a n-almost key, if the dataset contains from 1 to n exceptions in the dataset. For example, in D1 $\{db:hasActor, db:director\}$ is a 3-almost key and also a n-almost key for each $n \geq 3$.

By definition, if a set of properties P is a n-almost key, every superset of P is also a n-almost key. We are interested in discovering only minimal n-almost keys, i.e. n-almost keys that do not contain subsets of properties that are n-almost keys for a fixed n.

[7] $U \equiv V$ represents the semantic equivalence of resources or literals. In the example, we have considered all the syntactically distinct literals or resources as semantically distinct.

To discover n-almost keys SAKey proceeds in three main phases: (1) the pre-processing step that allows avoiding useless computations (2) the discovery of maximal (n+1)-non keys and finally (3) the derivation of n-almost keys from the set of (n+1)-non keys.

In what follows, we consider a knowledge graph $\mathcal{K} = (\mathcal{O}, \mathcal{D})$ (see Definition 2.1) which represents the RDF dataset on which we aim to discover n-almost keys.

Phase 1: Preprocessing. Initially we represent the data in a structure called *initial map* in the form of a set of $(map-key, map-values)$. The $map-key$ is a property and the $map-values$ represent the sets of instances that share one value for a given property. Table 5.3 shows the initial map of the dataset D1 presented in Fig. 5.2. For example, the set $\{f2, f3, f4\}$ of $d1$:$hasActor$ represents the films that "G. Clooney" has played in.

$d1$:$hasActor$	$\{\{f1, f2, f3\}, \{f2, f3, f4\}, \{f1, f2\}, \{f4\}, \{f5\}, \{f6\}\}$
$d1$:$director$	$\{\{f1,f2,f3\}, \{f2, f3, f5\}, \{f2, f3, f6\}, \{f4\}\}$
$d1$:$releaseDate$	$\{\{f1\}, \{f2, f6\}, \{f3\}, \{f4\}, \{f5\}\}$
$d1$:$language$	$\{\{f4, f5\}\}$
$d1$:$name$	$\{\{f1\}, \{f2\}, \{f3\}, \{f4\}, \{f5\}, \{f6\}\}$

Table 5.3: SAKey: Initial map of D1

Data Filtering. To improve the scalability of our approach, we introduced the two following techniques to filter the data of the initial map.

a. Singleton Sets Filtering. Sets of size 1 in the map represent instances that do not share values with other instances for a given property. These sets cannot lead to the discovery of a n-non key. Thus, only sets of instances with size bigger than 1 are kept. Such sets are called *v-exception sets*.

Definition 5.9. (v-Exception set E_p^v). A set of instances $\{i_1, \ldots, i_k\}$ of the class c is a E_p^v for the property $p \in \mathcal{P}$ of $\mathcal{O}$ and the value v iff $\{p(i_1, v), \ldots, p(i_k, v)\} \subseteq \mathcal{D}$ and $|\{i_1, \ldots, i_k\}| > 1$.

We denote by $\mathfrak{E}_p$ the *collection* of all the v-exception sets of the property p. $\mathfrak{E}_p = \{E_p^v\}$

For example, in D1, the set $\{f1, f2, f3\}$ is a v-exception set of the property $d1$:$director$.

Singleton sets filtering (i.e., $\mathfrak{E}_p = \emptyset$) allows the discovery of single keys (i.e., keys composed of only one property).

b. v-Exception Sets Filtering. To compute all the maximal n-non keys of a dataset, only the maximal v-exception sets are necessary. Thus, all the non maximal v-exception sets are removed. Table 5.4 presents the data after applying the two filtering techniques on the data of table 5.3. This structure is called *final map*.

c. Elimination of Irrelevant Sets of Properties.

Table 5.4: SAKey: Final map of D1

$d1$:$hasActor$	{{$f1$, $f2$, $f3$}, {$f2$, $f3$, $f4$}}
$d1$:$director$	{{$f1$,$f2$,$f3$}, {$f2$, $f3$, $f5$}, {$f2$, $f3$, $f6$}}
$d1$:$releaseDate$	{{$f2$, $f6$}}
$d1$:$language$	{{$f4$, $f5$}}

If two properties have less than n instances in common, these two properties will never participate together to a n-non key. We denote by *potential n-non key* a set of properties sharing two by two, at least n instances.

Definition 5.10. (Potential n-non key). A set of properties $pnk_n = \{p_1, ..., p_m\}$ is a *potential n-non key* for a class c iff: $\forall \{p_i, p_j\} \in (pnk_n \times pnk_n) \mid I(p_i) \cap I(p_j)| \geq n$ where $I(p)$ is the set of instances that are subject of p.

Phase 2: Not n-almost keys Discovery. To discover all the maximal n-non keys (Definition 5.5) in a given dataset it suffices to find the n-non keys contained in the set of maximal potential n-non keys (PNK). For this purpose, we build a graph where each node represents a property and each edge between two nodes denotes the existence of at least n shared instances between these properties.

In D1, PNK = {{$d1$:$hasActor$, $d1$:$director$, $d1$:$releaseDate$},{$d1$:$language$}} corresponds to the set of maximal potential n-non keys when n=2. By construction, all the subsets of properties that are not included in these maximal potential n-non keys are not n-non keys.

To find all the maximal not n-almost keys, SAKey uses the *intersect operator* $\otimes$ to compute the intersection of collections of exception sets while keeping only sets greater than one.

Definition 5.11. (Intersect operator $\otimes$). Given two collections of v-exception sets $\mathfrak{E}_p$ and $\mathfrak{E}_{p'}$, we define the intersect $\otimes$ as follow:

$$\mathfrak{E}_{p_i} \otimes \mathfrak{E}_{p_j} = \{E^v_{p_i} \cap E^v_{p_j} \mid E^v_{p_i} \in \mathfrak{E}_{p_i}, E^v_{p_j} \in \mathfrak{E}_{p_j}, and\ |E^v_{p_i} \cap E^v_{p_j}| > 1\}$$

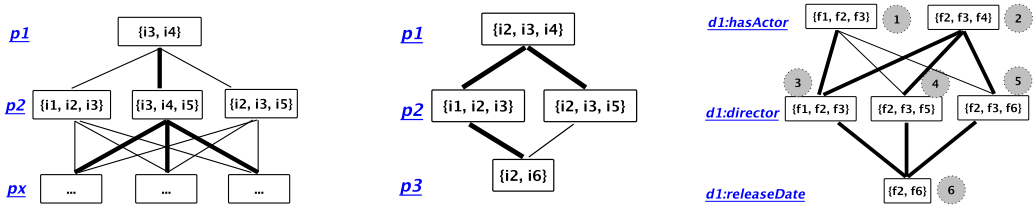
Given a set properties P, the set of exceptions E_P can be computed by applying the intersect operator to all the collections $\mathfrak{E}_p$ such that $p \in P$.

$$E_P = \bigcup \bigotimes_{p \in P} \mathfrak{E}_p$$

For example, for the set of properties P = {$d1$:$hasActor$, $d1$:$hasDirector$}, $\mathfrak{E}_P$={{f_1, f_2, f_3}, {f_2, f_3}} while E_P = {f_1, f_2, f_3} represents films having a same actor and same director.

Pruning Strategies. Computing the intersection of all the collections of v-exception sets represents the worst case scenario of finding maximal n-non keys within a potential n-non key. We have defined several strategies to avoid useless computations. We illustrate the pruning

strategies in Figure 5.3 where each level corresponds to the collection $\mathfrak{E}_p$ of a property p and the edges express the intersections that should be computed in the worst case scenario. Thanks to the prunings, only the intersections appearing as highlighted edges are computed.



(a) Inclusion Pruning (b) Seen Intersection Pruning (c) example of nNonKeyFinder

Figure 5.3: SAKey: nNonKeyFinder prunings and execution

a. Anti-monotonicity Pruning. This strategy exploits the anti-monotonicity characteristic of a n-non key, i.e., if a set of properties is a n-non key, all its subsets are by definition n-non keys. Thus, no subset of an already discovered n-non key will be explored.

b. Inclusion Pruning. Given a set of properties $P = \{p_1, \ldots, p_{j-1}, p_j, \ldots, p_n\}$, when the intersection of $p_1, \ldots, p_{j-1}$ is included in any v-exception set of p_j only this subpath is explored (example in Figure 5.3(a)).

c. Seen Intersection Pruning. In Figure 5.3(b), we observe that starting from the v-exception set of the property p_1, the intersection between $\{i_2, i_3, i_4\}$ and $\{i_1, i_2, i_3\}$ or $\{i_2, i_3, i_5\}$ will be in both cases $\{i_2, i_3\}$. Thus, the discovery using the one or the other v-exception set of p_2 will lead to the same n-almost keys. More generally, when a new intersection is included in an already computed intersection, this exploration stops.

To discover the maximal n-non keys, the v-exception sets, obtained after filtering, are explored in a depth-first way (see Figure 5.3(c)). The set of Potential n-non keys is explored while applying the aforementioned pruning strategies. Since the condition for a set of properties P to be a n-non key is $E_P \geq n$ the exploration stops as soon as n exceptions are found (for a more detailed description of the algorithm see [132]).

Phase 3: n-almost keys Derivation. A set of properties is a n-almost key, if it is not equal or included to any maximal (n+1)-non key. Indeed, when all the (n+1)-non keys are discovered, all the sets not found as (n+1)-non keys will have at most n exceptions (n-almost keys).

Both [127] and [104] derive the set of keys by iterating two steps: (1) computing the Cartesian product of complement sets of the discovered non keys and (2) selecting only the minimal sets. Deriving keys using this algorithm is very time consuming when the number of properties is big. To avoid useless computations, we proposed a new algorithm that derives fast minimal n-almost keys. In this algorithm, the properties are ordered by their frequencies in the complement sets. At each iteration, the most frequent property is selected and all the n-almost keys involving this property are discovered recursively. For each selected property p, we combine p with the

properties of the selected complement sets that do not contain p. Indeed, only complement sets that do not contain this property can lead to the construction of minimal n-almost keys. When all the n-almost keys containing p are discovered, this property is eliminated from every complement set. When at least one complement set is empty, all the n-almost keys have been discovered. If every property has a different frequency in the complement sets, all the n-almost keys found are minimal n-almost keys. In the case where two properties have the same frequency, additional heuristics should be taken into account to avoid computations of non minimal n-almost keys.

5.3.3 SAKey Experiments and Evaluation

We evaluated SAKey using 3 groups of experiments. In the first group, we demonstrate the scalability of SAKey thanks to its filtering and pruning techniques. In the second group we compare the performances of SAKey against KD2R, the only approach that discovers composite OWL2 keys, when the experiments were conducted. The two approaches are compared in two steps. Finally, we show how n-almost keys can improve the quality of data linking. The experiments are executed on 3 different datasets, DBpedia[8], YAGO[9] and OAEI 2013[10].

The execution time of each experiment corresponds to the average time of 10 repetitions. In all experiments, the data is stored in a dictionary-encoded map, where each distinct string appearing in a triple is represented by an integer. The experiments have been executed on a single machine with 12GB RAM, processor 2x2.4Ghz, 6-Core Intel Xeon and runs Mac OS X 10.8.

Scalability of SAKey Here, we present the scalability on the classes $DB{:}NaturalPlace$, $DB{:}BodyOfWater$ and $DB{:}Lake$ of DBpedia (see Figure **??**(b) for more details) when n = 1. We first compare the size of data before and after the filtering steps (see Table 5.5), and then we run SAKey on the filtered data with and without applying the prunings (see Table 5.6).

As shown in Table 5.5, thanks to the filtering steps, the complete set of n-non keys can be discovered using only a part of the data. We observe that in all the three datasets more than 88% of the sets of instances of the initial map are filtered applying both the singleton filtering and the v-exception set filtering. Note that more than 50% of the properties are suppressed since they are single 1-almost keys (singleton filtering).

In Table 5.6, we show that the number of calls of nNonKeyFinder decreases significantly using the prunings. Indeed, in the class $DB{:}Lake$ the number of calls decreases to half. Subsequently, the runtime of SAKey is significantly improved. For example, in the class $DB{:}NaturalPlace$ the time decreases by 23%.

To evaluate the scalability of SAKey when n increases, nNonKeyFinder has been executed with different n values. This experiment has shown that nNonKeyFinder is not strongly affected

[8] http://wiki.dbpedia.org/Downloads39
[9] http://www.mpi-inf.mpg.de/yago-naga/yago/downloads.html
[10] http://oaei.ontologymatching.org/2013

Table 5.5: SAKey: Data filtering results on different DBpedia classes

class	# Initial sets	# Final sets	# Singleton sets	# E_p^v filtered	Suppressed Prop.
$DB{:}Lake$	57964	4856(8.3%)	50807	2301	78 (54%)
$DB{:}BodyOfWater$	139944	14833(10.5%)	120949	4162	120 (60%)
$DB{:}NaturalPlace$	206323	22584(11%)	177278	6461	131 (60%)

Table 5.6: Pruning results of SAKey on different DBpedia classes

class	without prunings		with prunings	
	Calls	Runtime	Calls	Runtime
$DB{:}Lake$	52337	13s	25289 (48%)	9s
$DB{:}BodyOfWater$	443263	4min28s	153348 (34%)	40s
$DB{:}NaturalPlace$	1286558	5min29s	257056 (20%)	1min15s

by the increase of n. Indeed, allowing 300 exceptions (n=300) for the class $DB{:}NaturalPlace$, the execution time increases only by 2 seconds.

Scalability Results: KD2R vs. SAKey. To compare the efficiency of SAKey against KD2R we conducted two experiments: the non key discovery and the n-almost keys derivation process. Note that, to obtain the same results from both KD2R and SAKey, the value of n is set to 1.

n-non key Discovery. In Figure 5.4, we compare the runtimes of the non key discovery of both KD2R and SAKey for the class $DB{:}NaturalPlace$, starting from the 10 most frequent properties until the whole set of properties is explored. We observe that KD2R is not resistant to the number of properties and its runtime increases exponentially. For example, when the 50 most frequent properties are selected, KD2R takes more than five hours to discover the non keys while SAKey takes only two minutes. Moreover, we notice that SAKey is linear in the beginning and almost constant after a certain size of properties. This happens since the class $DB{:}NaturalPlace$ contains many single keys and unlike KD2R, SAKey is able to discover them directly using the singleton sets pruning. In Table. 5.7, we observe that SAKey is orders of magnitude faster than KD2R in classes of DBpedia and YAGO. Moreover, KD2R runs out of memory in classes containing many properties and triples.

n-almost key Derivation. We compared the runtimes of the key derivation of KD2R and SAKey on several sets of non keys. In Figure 5.5, we present how the time evolves when the number of non keys of the class $DB{:}BodyOfWater$ increases. SAKey scales almost linearly to the number of non keys while the time of KD2R increases significantly. For example, when the number of non keys is 180, KD2R needs more than 1 day to compute the set of minimal keys while SAKey less than 1 minute. Additionally, to show the efficiency of SAKey over KD2R, we compare their runtimes on several datasets (see Table 5.8). In every case, SAKey outperforms KD2R since it discovers fast the set of minimal keys.

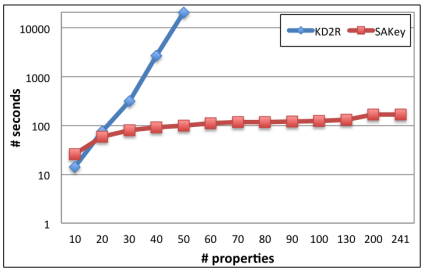


Figure 5.4: SAKey: nNonKeyFinder on $DB{:}NaturalPlace$–KD2R and SAKey runtime

Table 5.7: SAKey: nNonKeyFinder on different classes – runtime comparison with KD2R

class	#triples	#instances	#prop.	KD2R runtime	SAKey runtime
$DB{:}Lake$	409016	9438	111	outOfMem	8s
$DB{:}BodyOfWater$	1068428	34000	200	outOfMem	37s
$DB{:}NaturalPlace$	1604348	49913	243	outOfMem	1min10s
$YA{:}Building$	114783	54384	17	26s	9s
$YA{:}SportsSeason$	83944	17839	35	2min	9s
$DB{:}Website$	8506	2870	66	13min	1s
$DB{:}Mountain$	115796	12912	124	191min	11s

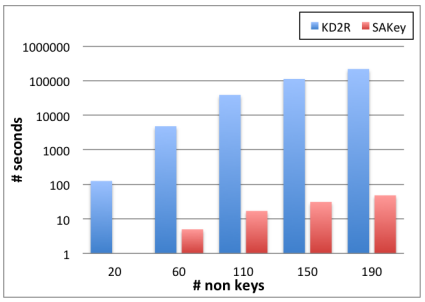


Figure 5.5: SAKey: KeyDerivation on $DB{:}BodyOfWater$

In the biggest class of DBpedia, $DB{:}Person$ (more than 8 million triples, 9 hundred thousand instances and 508 properties), SAKey takes 19 hours to compute the n-non keys while KD2R cannot even be applied.

Data Linking with n-almost keys Here, we evaluate the quality of identity links that can be found using n-almost keys. We have exploited one of the datasets provided by the OAEI'13. The benchmark contains one original file and five test cases. The second file is taken from the first test case. Both files contain DBpedia descriptions of persons and locations (1744 triples, 430 instances, 11 properties). Table 5.9 shows the results when n varies from 0 to 18. In Table 5.9(a), strict equality is used to compare literal values while in Table 5.9(b), the Jaro-Winkler similarity measure is used. The recall, precision and F-measure of our linking results has been computed using the gold-standard provided by OAEI'13.

Table 5.8: SAKey: KeyDerivation runtime for DBpedia and YAGO classes – comparison with KD2R

class	#non-keys	#keys	KD2R runtime	SAKey runtime
DB:Website	9	44	1s	1s
YA:Building	15	34	1s	1s
YA:SportsSeason	22	175	2s	1s
DB:Lake	50	480	1min10s	1s
DB:Mountain	49	821	8min	1s
DB:BodyOfWater	220	3846	> 1 day	66s
DB:NaturalPlace	302	7011	> 2days	5min

Table 5.9: Data Linking in OAEI 2013

# exceptions	Recall	Precision	F-Measure
0, 1, 2	25.6%	100%	41%
3, 4	47.6%	98.1%	64.2%
5, 6	47.9%	96.3%	63.9%
7, ..., 17	48.1%	96.3%	64.1%
18	49.3%	82.8%	61.8%

(a) Data Linking using strict equality

# exceptions	Recall	Precision	F-Measure
0, 1, 2	64.4%	92.3%	75.8%
3, 4	73.7%	90.8%	81.3%
5, 6	73.7%	90.8%	81.3%
7, ..., 17	73.7%	90.8%	81.3%
18	74.4%	82.4%	78.2%

(b) Data Linking using similarity measures

In both tables, we observe that the quality of the data linking improves when few exceptions are allowed. As expected, when simple similarity measures are used, the recall increases while the precision decreases, but overall, better F-measure results are obtained.

5.4 VICKEY: Discovery of Minimal Composite Conditional Keys

5.4.1 VICKEY Overview

To deal with the problem of imperfect data in the LOD, we developed first SAKey which allows one to discover n-almost keys allowing n exceptions that can capture some redundancies and errors. However, because of coarse structuration of some datasets allowing exceptions is not always sufficient to discover keys. To deal with the problem of lack or absence of keys, we developed VICKEY [134] approach to be able to discover conditional keys which express explicitly the parts of the data where keys are valid. A conditional key is an axiom saying that under particular conditions, no two distinct entities can have the same values on a particular set of properties. For example, we can say that in a German university, no two professors can advise the same doctoral student. The situation might be different at a French or American university – hence the key is "conditional" to German universities. In VICKEY approach, we distinguish conditional keys from classical keys, which hold for an atomic class (or for every tuple of a table in a relational database). Conditional keys can express constraints on entities and are strictly more expressive

Table 5.10: VICKEY: Example dataset

	FirstName	**LastName**	**Gender**	**Lab**	**Nationality**
r1	Claude	Dupont	Female	Paris-Sud	France
r2	Claude	Dupont	Male	Paris-Sud	Belgium
r3	Juan	Rodríguez	Male	INRA	Spain, Italy
r4	Juan	Salvez	Male	INRA	Spain
r5	Anna	Georgiou	Female	INRA	Greece, France
r6	Pavlos	Georgiou	Male	Paris-Sud	Greece
r7	Marie	Legendre	Female	INRA	France

than classical keys. Therefore, they can be more productive in tasks such as entity linking – as we showed in the experiments. Apart from this, conditional keys carry knowledge in themselves. For example, it is interesting to know that France allows several advisors, while Germany does not.

Mining all the conditional keys automatically from the data is highly combinatorial, since it needs to go through all the combinations of properties as for classical and n-almost keys, but in addition, it also needs to explore all the possible property values to find the conditions on which a key is valid. Thus KGs with millions of statements would require billions of possible conditions and property combinations that could define a conditional key. Our proposal is to combine key discovery techniques [131] with techniques from rule mining [55] to filter-out irrelevant combinations. More precisely, VICKEY discovers first the set of maximal non-keys from which the conditional keys can be computed. Thus, the search space can be significantly reduced while avoiding to scan all the data. Secondly, VICKEY applies a breadth-first strategy to discover frequent candidate conditional keys and efficiently check their validity.

5.4.2 Conditional Key Discovery Approach

To learn conditional keys on RDF datasets, we assume that all instances in a dataset refer to distinct real world objects, and that all unknown values are different from the existing ones in the dataset as in KD2R and SAKey.

The discovery of simple keys alone already requires checking a large number of property combinations (of which there are $2^{|\mathscr{P}|}$ in total, where $\mathscr{P}$ is the set of properties). Discovering conditional keys is even more complex, since the search space is in the order of $O(|\mathscr{V}|^{|\mathscr{P}|})$, where $\mathscr{V}$ is the set of objects in the dataset. That is, we developed VICKEY approach that is able to discover conditional keys efficiently in spite of this large search space.

The example in Table 5.10 shows two researchers with the last name Dupont. Therefore the property *lastName* is not a key. The combination *{firstName, lastName}* is not a key either, because there are two researchers with the same first and last names. However, when we restrict our set of researchers to those working at INRA, the property *lastName* identifies researchers

uniquely. In contrast, this is not true for the researchers in Paris-Sud. Thus, *{lastName, lab}* is not a key in general. We say that *lastName* is a *conditional key* for people working at INRA.

In this approach we focused on conditions that can be expressed using constraints on property values. More formally, a condition is a pair composed of a property p and an object o, written $p = o$. A condition cd with property p and object o holds for a subject x, written $cd(x)$, if $p(x,o)$. In the example, the condition $(lab = INRA)$ holds for $r3$, $r4$, $r5$ and $r7$.

Definition 5.12. (Conditional key.) A conditional key for a dataset $\mathscr{D}$ is a non-empty set of conditions $\{cd_1,...,cd_n\}$ and a non-empty set of properties $\{p_1,...,p_m\}$ of $\mathscr{D}$ (disjoint from the properties in the conditions), such that:

$$\forall x, y, u_1,\ldots,u_m \left(\bigwedge_{i=1..n} (cd_i(x) \wedge cd_i(y)) \wedge \bigwedge_{i=1..m} (p_i(x,u_i) \wedge p_i(y,u_i)) \Rightarrow x = y \right)$$

Definition 5.13. (Minimal conditional key) A conditional key with conditions CD and properties P is minimal, if the removal of a condition in CD, the removal of a property from P, or the transfer of a property p from CD to P (with the corresponding removal of the condition), result in something that is neither a conditional key nor a key.

Conditional keys can be defined in the ontology language OWL2 [142]. OWL2 allows defining keys not just on atomic classes (such as *researcher*), but also on more complex class expressions. For example, we can define, the class "Researchers who work at INRA" as $c = Researcher \sqcap \exists lab.\{INRA\}$. Then, $\{lastName\}$ is a key on the dataset of c according to Definition 5.1.

In our example, *lastName* is a conditional key with condition $nationality = Spanish \wedge lab = INRA$, but this conditional key is not minimal, because there exists a simpler version of the key with fewer conditions, namely $nationality = Spanish$. In the same vein, $\{lastName\}$ with the condition *gender=male* is not a minimal conditional key, because $\{lastName, gender\}$ is a key.

VICKEY takes as input a dataset $\mathscr{D}$ and a threshold θ for the minimal support (i.e. the number of instances for with the key is valid) of the discovered keys and returns all the minimal conditional keys agreeing with θ.

Definition 5.14. (Conditional Key Support). The *support* of a conditional key with properties $\{p_1,...,p_m\}$ and conditions $\{cd_1,...,cd_n\}$ is the number of subjects x such that $\bigwedge_{i=1..m} \exists u_i : p_i(x,u_i)$ and $\bigwedge_{i=1..n} cd_i(x)$.

A proportional version of the support, which we call the *coverage*, measures the ratio of subjects in the dataset identified by the conditional key. In our example, the support of the key $\{lastName\}$ under the condition *lab=INRA* is 4, and the coverage is $\frac{4}{7}$, since there are 7 subjects.

The naive method to mine conditional keys explores all possible combinations of properties and conditions in the input KB and verifies whether they fulfill Definition 5.12. Such an approach

is infeasible on large datasets. Our main idea (the key insight, so to speak) is the following (see [133] for more details):

Observation 5.1. ***(Conditional Keys and Non-Keys)*** *Given a minimal conditional key for a dataset $\mathcal{D}$ with properties P and conditions $\{p_1 = o_1, ..., p_n = o_n\}$, the set of properties $P \cup \{p_1, ..., p_n\}$ must be a non-key for $\mathcal{D}$.*

This follows from Definition 5.13. In our example from Table 5.10, *{firstName}* is a minimal conditional key with condition *gender=Female*, and *{gender, firstName}* is a non-key. Thus, if we want to mine the complete set of minimal conditional keys, it suffices to consider only the property combinations given by non-keys. Since maximal non-keys are super-sets of all other non-keys (Definition **??**), it is sufficient to explore only property combinations given by maximal non-keys.
To discover minimal conditional keys, VICKEY proceeds in three phases:
1) Discovery of non-keys: Instead of exploring the whole set of combinations of properties, we focus our search on those combinations that are not keys.
2) Generation of Conditional Key Graphs: We use the non-keys to generate candidate keys, which we store in conditional key graphs.
3) Exploration of Conditional Key Graphs: The conditional key graphs are then explored for minimal conditional keys discovery.

Phase 1: Discovery of non-keys. The maximal non-keys in the input dataset can be mined efficiently with the SAKey algorithm [131] (Section 5.3). Thus, we concentrate in the following on mining the conditional keys from these maximal non-keys. As a running example, consider again the dataset in Table 5.10. It contains two maximal non-keys: {*firstName*, *lastName*, *lab*} and {*firstName*, *gender*, *lab*, *nationality*}. As a running example, consider again the dataset in Table 5.10. It contains two maximal non-keys: {*firstName*, *lastName*, *lab*} and {*firstName*, *gender*, *lab*, *nationality*}.

Phase 2: Generation of Conditional Key Graphs. Our method for discovering conditional keys from non-keys relies on a modifiable data structure that we call a *Conditional Key Graph (CKG)*. Such a graph is a tuple $\langle P^k, P^c, cond, G \rangle$ with the following components:

- P^k and P^c are disjoint sets of properties, called *key properties* and *condition properties*, respectively.
- *cond* is a set of conditions on P^c.
- G is a directed graph. Each node v is associated to a set $v.p \subseteq P^k$ and to a boolean flag $v.explore$ set by default to true. There is a directed edge from u to v if $u.p \subset v.p$ and $|u.p| = |v.p| - 1$.

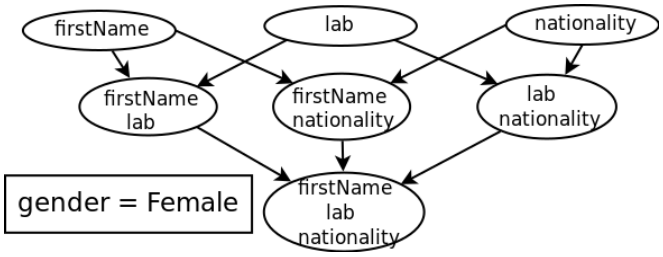


Figure 5.6: Example of a conditional key graph with P^k = *{firstName, lab, nationality}*, P^c =*{gender}*, $cond = \{gender = Female\}$.

To construct the initial conditional key graphs we apply an algorithm which takes as input the dataset, the support threshold θ (Definition 5.14), and the non-keys discovered in Section 5.4.2. The algorithm first constructs all possible conditions $p = a$ that combine a property p from the non-keys with an instance or literal a from the dataset. Conditions with support less than θ are not considered. It then looks at all non-keys N in which p appears. The conditional key graph for the condition $p = a$ contains as nodes all subsets of $N \setminus \{p\}$ except the empty set. As an example, let us consider again the dataset of Table 5.10 and its two maximal non-keys {*firstName*, *lastName*, *lab*} and {*firstName*, *gender*, *lab*, *nationality*}. Figure 5.6 depicts the conditional key graph associated to the condition *gender* = *Female* constructed by the algorithm.

Phase 3: Exploration of Conditional Key Graphs. We explore the conditional key graphs for finding keys with an algorithm that takes as input a dataset, a support threshold, and the set of conditional key graphs constructed in the previous phase.

The algorithm proceeds in levels, looking first at the nodes that contain one property, then two properties, etc. For each level, we consider every node *cand*. If the node is still marked for exploration, we construct a candidate conditional key, with the input conditions as condition part, and the properties in *cand.p* as the key part. We then verify if the candidate key (a) meets the definition of a conditional key and (b) is minimal with respect to the other keys that have already been discovered.

If that is the case, the conditional key is added to the output (Line 7). If the key is a minimal key, then any extension of the key with more properties in the key part must be non-minimal and can safely be abandoned. Likewise, if the support of the candidate key is below the given threshold, so are its refinements. In both cases, we can prune the node and all descendants.
As an example, let us consider again the data from Table 5.10, with the condition *gender=Female* and the maximal non-key {*firstName*, *gender*, *lab*, *nationality*}. The corresponding conditional key graph after scanning the first level is shown in Figure 5.7(a). Nodes with the *explore* flag set to false are greyed out. At the end of this step, only the property *firstName* is discovered as a key, since first names are unique among female researchers. It follows that nodes containing this property in the next levels of the graph define non-minimal keys. They are therefore discarded for further exploration (the *explore* flag is set to false). The search for conditional keys is then

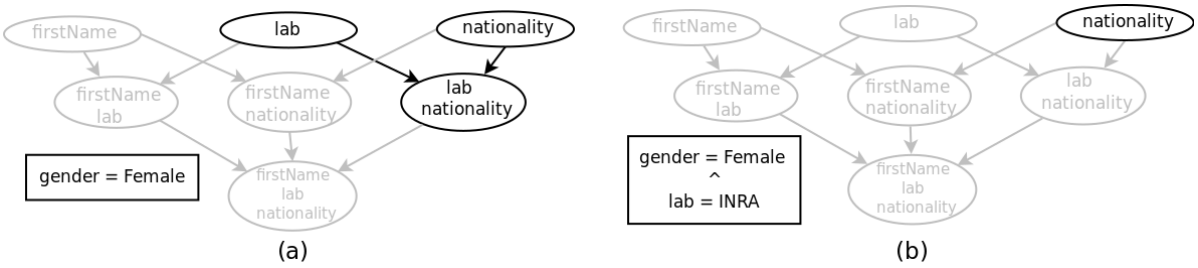


Figure 5.7: (a) Keys of size 1 explored for the condition $gender = Female$. (b) Example of a merged graph with condition $\{gender = Female, lab = INRA\}$

applied to the nodes on levels 2 and 3, for which the *explore* flag is still true.

Merging conditions. The process of generating more complex conditions is done by an algorithm that takes as input a set of conditional key graphs, a support threshold, a dataset, and a size parameter. It looks at all conditional key graphs that have a condition set of the given size and then applies a merge operation.
The *merge* operation between two conditional key graphs $\langle P_1^k,\ P_1^c,\ cond_1,\ (V_1,E_1)\rangle$ and $\langle P_2^k,P_2^c,cond_2,(V_2,E_2)\rangle$ with $P_1^c \cap P_2^c = \emptyset$, produces a new conditional graph $\langle P^k,P^c,cond,(V,E)\rangle$ with:

- $P^k = P_1^k \cap P_2^k$ and $P^c = P_1^c \cup P_2^c$.
- $cond = cond_1 \cup cond_2$
- $V = \{\langle v.p, v.explore\rangle : \exists v_1 \in V_1, v_2 \in V_2 : v_1.p = v_2.p = v.p \quad \wedge \quad v.explore = (v_1.explore \wedge v_2.explore)\}$
- $E = \{u, v \in V : u.p \subset v.p \wedge |u.p| = |v.p| - 1\}$

As an example, Figure 5.7(b) shows the conditional graph with the set of conditions $\{gender = Female, lab = INRA\}$ produced by the algorithm from the conditional graphs with conditions $gender = Female$ and $lab = INRA$. This graph is a clone of the graph with the condition $gender = Female$. A node is marked to be explored only if it was marked to be explored in both of the original graphs.

5.4.3 VICKEY Experiments and Evaluation

We evaluate VICKEY in two series of experiments. First, we show the ability of VICKEY to discover conditional keys in large datasets with millions of triples. We compare the runtime of VICKEY to a generic rule mining approach, AMIE [55]. Then, we evaluate the utility of conditional keys for the task of data linking. We compare the conditional keys discovered by VICKEY to the classical keys discovered by SAKey [131].

Table 5.11: VICKEY vs AMIE on DBpedia

Class	Triples	Inst.	#Pro	#NKs	VICKEY	AMIE	#CKs
Actor	57.2k	5.8k	71	137	4.52m	12.58h	311
Album	786.1k	85.3k	39	68	1.53h	3.90h	304
Book	258.4k	30.0k	51	95	11.84h	> 1d	419
Film	832.1k	82.6k	74	132	1.37h	3.64h	185
Mount.	127.8k	16.4k	58	47	2.86m	23.57m	257
Museum	12.9k	1.9k	65	17	1.46s	6.45s	58
Organiz.	1.82M	178.7k	553	3221	26.32h	> 36h	28
Scientist	258.5k	19.7k	73	309	27.67m	> 1d	582
Univ.	85.8k	8.7k	89	140	14.45h	> 1d	941

Table 5.12: Linked classes stats

Class	#Pro	#Ks	#NKs	#CKs
Actor	16	93	22	748
Album	5	1	2	5864
Book	7	5	2	538
Film	9	14	13	26750
Mount.	5	3	2	775
Museum	7	14	5	80
Organiz.	17	149	3	9737
Scientist	10	22	8	407
Univ.	9	5	5	449

5.4.3.1 Runtime Experiments

Setting. To evaluate the performance of VICKEY and AMIE [55], we adapt AMIE to mine rules of the form: $P_c \wedge P_k \Rightarrow x = y$. Here, $P_c = \bigwedge_{1..n} pc_i(x, A_i) \wedge pc_i(y, A_i)$ corresponds to the condition part of a key expression, and $P_k = \bigwedge_{1..m} pk_i(x, u_i) \wedge pk_i(y, u_i)$ represents the key part. Both AMIE and VICKEY take as input a set of maximal non-keys. These non-keys are obtained from the input dataset using SAKey [131]. Like VICKEY, our adapted variant of AMIE uses the non-keys to restrict the search space by pruning the combinations of properties that do not occur in the non-keys. Unlike VICKEY, AMIE searches exhaustively for all rules that define conditional keys in the input dataset, regardless of their minimality. AMIE therefore requires a post-processing phase where all non-minimal conditional keys are removed. Both AMIE and VICKEY are run with a coverage threshold of 1%. We set the confidence threshold of AMIE to 100%, so that VICKEY and the modified AMIE mine exactly the same set of conditional keys. As datasets, we have used nine classes from DBpedia [78], covering different domains such as people, organizations, and locations. All experiments are run on a server with an AMD Opteron 6376 Processor (2.40GHz), 8 cores, and 128GB of RAM under Ubuntu Server 16.04.

Results. Our results are shown in Table 5.11. The first three columns show some statistics about the testing datasets, followed by the number of discovered non-keys (NKs), the runtimes of both VICKEY and AMIE and finally the number of obtained conditional keys (CKs). We observe that a generic rule mining solution cannot handle some of the input datasets in less than 1 day. VICKEY, in contrast, runs on the smaller datasets *Actor*, *Mountain*, *Museum* and *Scientist* in less than 1 hour. This is because VICKEY's strategy prunes the search space much more effectively by avoiding candidate CKs that are not minimal. Other classes, such as *University* and *Organization*, are more challenging because they have many long non-keys (up to 15 properties). The longer the non-keys, the larger the number of property combinations in the search space. For example, for the class *Album*, AMIE explores more than 12.3k rules (including intermediate rules), where 6.4k rules correspond to potential conditional keys. In contrast, VICKEY explores only 4.1k candidates. This shows that VICKEY's strategy indeed prunes the search space much more effectively. It can mine conditional keys on hundreds of thousands of facts in a matter of minutes.

5.4.3.2 Extrinsic Evaluation

Setting. As KBs, we chose DBpedia [78] and YAGO [130], because there is a gold standard available for the entity links on the YAGO Web page. We have used the same set of classes as for the runtime experiments. As this type of entity linking assumes that the properties have been aligned, we mapped the properties of these classes manually, and rewrote the properties of YAGO using its DBpedia counterparts. We ran SAKey [131] and VICKEY on DBpedia to find standard and conditional keys, respectively. Table 5.12 shows the number of common properties, the number of keys (Ks), non-keys (NKs) and conditional keys (CKs) in each DBpedia class. Among others, VICKEY finds that *motto* is a key for universities in Italy and some other countries – but not in all countries; and that the name is a key for organizations in certain places – but not all places. To link the datasets, we use a simple algorithm [131]: For each key, we iterate over the entities in DBpedia that have the key properties. If there is an entity in YAGO that shares at least one value for every of these properties, we link the two. For conditional keys, we also check whether the conditions of the key are fulfilled in both datasets.

Table 5.13: Linking results with classical keys (Ks), conditional keys (CKs), and both.

Class		Recall	Precision	F1	
Actor	Ks [131]	0.27	0.99	**0.43**	
	CKs	0.57	0.99	0.73	× 1.75
	Ks+CKs	0.60	0.99	**0.75**	
Album	Ks [131]	0.00	1	**0.00**	
	CKs	0.15	0.99	0.26	× 869
	Ks+CKs	0.15	0.99	**0.26**	
Book	Ks [131]	0.03	1	**0.06**	
	CKs	0.11	0.99	0.20	× 3.48
	Ks+CKs	0.13	0.99	**0.23**	
Film	Ks [131]	0.04	0.99	**0.08**	
	CKs	0.38	0.96	0.54	× 7.1
	Ks+CKs	0.39	0.98	**0.55**	
Mountain	Ks [131]	0.00	1	**0.00**	
	CKs	0.28	0.99	0.44	× 101
	Ks+CKs	0.29	0.99	**0.45**	
Museum	Ks [131]	0.12	1	**0.21**	
	CKs	0.25	1	0.40	× 2.19
	Ks+CKs	0.31	1	**0.47**	
Organization	Ks [131]	0.01	1	**0.02**	
	CKs	0.14	0.98	0.24	× 11
	Ks+CKs	0.14	0.99	**0.24**	
Scientist	Ks [131]	0.05	0.98	**0.11**	
	CKs	0.16	0.99	0.28	× 2.96
	Ks+CKs	0.19	0.99	**0.32**	
University	Ks [131]	0.09	0.99	**0.16**	
	CKs	0.22	0.99	0.36	× 2.44
	Ks+CKs	0.25	0.99	**0.40**	

Results Table 5.13 shows the precision, recall and F1 measure of the entity linking task using a) classical keys mined by SAKey [131], b) conditional keys alone and c) both types of keys (VICKEY). We first observe that the precision is always over 98%. Conversely, the recall is low in some cases. This happens mainly due to our simple linking method, which uses a strict string equality when comparing the values of properties, and also due to the incompleteness of the

data in both YAGO and DBpedia. However, even with this simple method, the use of conditional keys can lead to a significant increase in recall – with a negligible impact on precision. For example, for the class *Film*, recall increases from 4% to 38% when conditional keys are considered. Furthermore, when combining classic keys and conditional keys, the recall improves further. Overall, we observe an average increase of 21 percentage points in recall, and of 29 points in F1 when both standard keys and conditional keys are used to link the data. The average drop in precision is only 0.5 percentage points. This shows that conditional keys can significantly increase the performance of entity linking.

5.5 Lessons Learned

In this chapter, I presented both the existing work on key discovery and my contributions to this problem in the Web of data. My contributions on key discovery were driven by both the scalability and the relevance of keys when used for data linking. My work on this problem showed that following a strategy of discovering non-keys first allows to override the data volume barrier and leads to the ability of handling datasets of several millions and hundreds of properties. Moreover, the use of aggressive filtering and pruning strategies have shown their importance for enabling the discovery of all minimal keys on big datasets in a reasonable time. The discovery of the two new kinds of keys, namely, almost-keys and conditional keys have shown strong quality (in terms of data linking) and efficiency results and proved their relevance and necessity to overcome RDF data incompleteness and redundancy, i.e., when the UNA is relaxed. In our previous work, we have developed different methods for discovering different kinds of keys exact and almost-keys by SAKey [131] and conditional keys by VICKEY [134]. Given the variability of dataset characteristics, a more holistic approach which discovers in one pass all the different kinds of keys could be more suitable to the linked data setting.

Thanks to the work [11] that we have conducted on the comparison of two main different semantics of keys, i.e., some-keys and forall-keys, we came to the conclusion that depending on the data completeness both semantics may be relevant. Hence, a deeper study and analysis need to be conducted to qualify in which cases one semantics of these existing ones should be used. For key selection, for which a preliminary work has been conducted in [50], different quality measures need to be defined to select the keys that are the most relevant to the target task, like data linking, index building, anomaly detection, and so on.

Finally, with regard to data evolution the approaches that we developed are not incremental, in the sense that if data changes, the methods should be relaunched to make updates on the keys. It however can be worth to design incremental key discovery algorithms that prevent from recomputing all the keys each time the data evolves. Consequently, when the set of keys evolves, an incremental propagation of the changes to the identity links that are derived form these keys would be worth to be designed.

Main Collaborations and projects of the Chapter

The research work I presented in this chapter has been achieved thanks to the collaboration with several colleagues:

KD2R [104] and SAKey [10, 132] with **Nathalie Pernelle** (Paris Sud University) during the PhD of **Danai Symeonidou** (2011-2014),

VICKEY [134] with **Nathalie Pernelle** (Paris Sud University), **Danai Symeonidou** (INRA Montpellier), **Luis Galarraga** (Aalborg University, Denmark) and **Fabian Suchanek** (TelecomParisTech),

Key Semantics study [10] with **Manuel Atencia** (Grenoble University), **Jérôme David** (Grenoble University), **Michel Chein** (Montpellier Unievrsity), **Madalina Croitoru** (Montpellier Unievrsity), **Michel Leclère** (Montpellier Unievrsity), **Nathalie Pernelle** (Paris Sud University) and **Danai Symeonidou** (Paris Sud University) in the setting of the Qualinca ANR project.

Main Publications of the Chapter

[134] Danai Symeonidou, Luis Galarraga, Nathalie Pernelle, **Fatiha Saïs** and Fabian Suchanek. VICKEY: Mining Conditional Keys on RDF datasets. In proceedings of the 16th International Semantic Web Conference (ISWC) 2017, pages: 661-677. Vienna, Austria, October 21-25, 2017.

[136] Key Discovery for Numerical Data: Application to Oenological Practices. *Danai Symeonidou, Isabelle Sanchez, Madalina Croitoru, Pascal Neveu, Nathalie Pernelle,* ***Fatiha Saïs****, Aurelie Roland-Vialaret, Patrice Buche, Aunur-Rofiq Muljarto, Remi Schneider*. In the proceedings of the 22nd International Conference on Conceptual Structures (ICCS), pages: 222-236, LNCS 9717, Springer 2016, Annecy (France)

[10] *Manuel Atencia, Michel Chein, Madalina Croitoru, Jérôme David, Michel Leclère, Nathalie Pernelle,* ***Fatiha Saïs****, François Scharffe, Danai Symeonidou*. Defining Key Semantics for the RDF Datasets: Experiments and Evaluations (2014). In the proceedings of the 21th International Conference on Conceptual Structures (ICCS), pages: 65-78, LNCS 8577, Springer 2014, Iasi (Romania).

[132] *Danai Symeonidou, Vincent Armant, Nathalie Pernelle,* ***Fatiha Saïs***. SAKey: Scalable Almost Key Discovery in RDF Data (2014). In the proceedings of the 13th International Semantic Web Conference (ISWC), pages: 33-49, LNCS 8796, Springer 2014, 19-23 October, Riva del Garda (Italy).

[104] *Nathalie Pernelle,* ***Fatiha Saïs****, Danai Symeonidou*. An Automatic Key Discovery Approach for Data Linking. In *Journal of Web Semantics* (2013), Elsevier. Volume 23, pages 16-30. December 2013.

CHAPTER 6

CONCLUSION AND RESEARCH PERSPECTIVES

SUMMARY

Knowledge Graph (KG) completion and correctness, when taken globally is very often considered as an unachievable goal and technically intractable, even though some problems can be addresses individually by several methods. Indeed, the existing approaches do not simultaneously deal with completeness and correctness of knowledge graphs, and usually only address a part of target aspects, such as, type or relation assertions, literal values, or ontology axioms.

Recently emerged knowledge graphs are undeniably of great assets for knwoledge driven systems. However, there are still problems that have to be dealt with to make the knwoledge graph correctness and completeness higher. Indeed, anomalies in the originating information sources, absent values, errors, redundancy and obsolescence of data and knowledge are still present in knowledge graphs. In such a context, the ultimate goal of knowledge graph refinement is to ideally design iterative and conservative refinement workflows which at each iteration improves both completeness and correctness of knowledge graphs, that is, the more the refinement is applied the better KG quality is.

In the last decade, my contributions to Knowledge Graph Refinement Research field have mainly been oriented to the following three research topics.

Identity Management with a special focus on the problem of erroneous identity links represented by `owl:sameAs` predicates in the LOD. We tried to address two research questions: (i) how to automatically detect erroneous `owl:sameAs` links? and (ii) how to detect and represent contextual identity links in case of near identity or similarity?

For the first research question, I developed two different approaches. The first is an inconsistency and content-based link invalidation approach [99] which exploits some ontology axioms and

the descriptions of the resources to infer incorrect links in case of inconsistency. Thanks to the informative aspect of this approach it is able to achieve good accuracy but, it can not scale to big datasets. The second approach is content and assumption agnostic. It exploits only the topology of the identity graph and the symmetry property of `owl:sameAs` predicate to assign for each identity link an erroneousness degree. That is, a ranking function is applied to order the links from the most plausible ones to the erroneous ones. This approach is the first that is applied to the whole LOD content (28 billion triples and more than 500 million `owl:sameAs` links) while obtaining a reasonable accuracy. Both approaches are relevant to be used and applied but in different contexts: the first on small datasets and rather rich ontologies, and the second approach is relevant for very large scale datasets. However, their combination is worth to be investigated in a way to apply the approach presented in [107] first to rank the links and then apply a more informative and costly approach presented in [99] to improve the accuracy.

For the second research question I developed an approach that automatically detects contextual identity links, where the context represents the sub-part of the ontology in which two resources are identical while maintaining identity relation semantics restricted to the context. We defined an order relation of the contexts which allows to infer more contextual identity links. This approach can be used as an alternative for the strict `owl:sameAs` link detection methods. The contexts are additionally relevant for the explanation identity links. This approach can easily be extended to compute also context of difference, an aspect that have not been yet well explored. More generally, detecting difference links will contribute in both completeness of KGs and correctness, since it will allow to exhibit more easily inconsistent cases.

Data Enrichment with a special focus on enriching a knowledge graph by either applying a data fusion approach or a missing values prediction approach. Thus, in one hand I developed a multi criteria data fusion method [122] that exploits data quality features to assign a confidence degree for each possible value. For the inherent uncertainty of the fusion results, I studied three different uncertainty models, namely, fuzzy sets in [122], possibility theory in [124] and belief functions in [36]. In the other hand, I proposed a predictive approach that is based on data reconciliation to predict new values for properties in a knowledge graph. These two approaches have been applied on real datasets and obtained promising results evaluated on samples. However, the evaluation and the validation of such approaches is a challenge in itself, since it requires a lot of manual efforts by human experts or by the crowd. Hence, benchmarks with associated gold standards results are also needed for such important validation phase. Finally, there is a need to develop powerful provenance and explanation models to help experts understanding the results.

Knowledge Discovery with a special focus on key discovery in RDF knowledge graphs that are of great usefulness for identity link detection. We investigated the discovery of three kinds of keys depending on the data characteristics. First, in [104, 135] we proposed KD2R a method that

is able to discover exact keys (valid for every instance in the dataset). It was the first method in 2011 that was developed to this purpose in RDF datasets. Second, in [131] we presented SAKey a new method that is able to discover n-almost-keys that are available in the whole dataset except for n exceptions. Third, in [134] we presented VICKEY method that is able to discover conditional keys that are available in a part of, and non necessary in the whole, RDF dataset. Up to our knowledge, VICKEY is the only method that exists and is able to discover conditional keys in RDF datasets. The search space of key discovery problem being exponential, the three methods that we proposed are all based on the discovery of non-keys first, and then derive minimal keys from the non-keys and all define aggressive filtering and pruning strategies. Thus, the size search space is drastically reduced and the methods are scalable to big datasets, e.g. datasets of up to 28 million triples for SAKey. All these methods discover keys that follow the OWL 2 *hasKey* semantics that is more suitable in case of incompleteness of data (see [11] for a theoretical and experimental comparison of the three different semantics of keys that exist). Some future direction are worth to be investigated to even more improve the scalability and the efficiency of key discovery by developing holistic approaches to discover in one pass all these different kinds of keys as well as other complex and expressive constraints like *key graphs* [48], *referring expressions* and [9].

RESEARCH PERSPECTIVES

Many issues and research perspectives are raised by our proposals and this section exposes the most salient ones.

Knowledge Graph Refinement under Data Evolution

One of the intrinsic features of knowledge graphs is their dynamicity. They are frequently updated because the world is evolving, new knowledge and data about it is continuously generated: "*Every day, we create 2.5 quintillion bytes of data*" [1]. When a KG evolves, the changes may concern the ontological level where changes may involve classes, properties, axioms or mappings to other ontologies. The changes may also concern the instance level where data modifications may affect instance typings, property instances, or identity links between instances. In the literature there are many proposals for dealing with ontology evolution [151] without considering data evolution. My research direction will be focused on the impact of instance-level evolution on the whole workflow of knowledge graph refinement (see Figure 1.1), namely, knowledge discovery, identity management and data enrichment. Indeed, when data changes, the axioms (e.g., keys, functional properties) that are discovered from it may become invalid and new ones may emerge (as a preliminary work [120], where we proposed an approach that infers axiom evolution from a semantic representation of the evolution of RDF datasets); identity links may become invalid and new ones may be discovered; and data fusion and value prediction tasks should be applied

[1] https://www.domo.com/learn/data-never-sleeps-5?aid=ogsm072517_1&sf100871281=1

incrementally to insure synchronisation and consistency of knowledge graphs. To this end, three orthogonal research questions should be addressed: (i) how to characterize and semantically represent data evolution in KGs to make it machine readable?, (ii) how to make explicit temporal information (e.g. *<Obama isPresidentOf USA> [2008-2016]*) and reason on it while dealing with evolving knowledge graph refinement?; and (iii) how to design incremental and efficient algorithms for evolving knowledge refinement without loosing completeness and correction of KGs?

This research perspective is included in a collaboration project proposal with IBM (France), that is currently under review.

Detection of gradual causal rules in transformation processes

The identification, detection and explicitation of causal relations (i.e. if A and B Then C or in other words A and B can explain C) between variables that may represent states or events are challenging questions in life science. Indeed, in such a context the targeted issues of a given application domain (e.g. environmental, agriculture or industrial processes) are complex, multi-scale, multi-objectives (e.g. different studies, different experiments, different scientific targets) usually represented by multiple models describing different aspects with a variety of details, from the very high level (e.g. at the system level) to the very low technical levels (e.g. at the bacteria level). In this direction I will investigate the detection of causal rules, represented in first order logic, in knowledge graphs representing transformation processes. More precisely, I will consider causal rules detection in distinct experiments coming from different sources, representing different domains (e.g. health, nutrition) but that can be considered as similar in a given explicit context to be defined. I will focus on gradual causal rules which means that we want to express the variation effect introduced by the experimental conditions on experiments results represented by observations (e.g. the effect of the temperature increase on the nutritional quality of a given food). A gradual pattern expresses some attribute co-variations: the more A increases, the more B decreases. Several approaches have been defined to discover such patterns in relational databases. In a recent work I studied the extraction of gradual subgraph patterns in RDF data representing transformation processes [126]. In this future direction, I aim to define an approach that discovers gradual causal rules from knowledge graphs representing sequences of events chronologically (partially) ordered. The developed approach should consider objects that are characterized by complex attributes represented by paths in RDF graphs (e.g. to know the temperature of a mixture $m1$ at time t_1, one has to explore a graph representing: an observation $o1$ observes the mixture $m1$ that has for attribute the temperature $t1$, its observation result $r1$ is a sensor output $so1$ that has for measure the simple measure 23 which has for unit of measure the unit °C). Some properties may express changes (e.g. the increase of the property temperature) introduced at some point in time and other the effects of these changes (e.g. the decrease of the water quantity, the occurrence/presence of salt). Such links between the changes (i.e. causes) and

the effects are implicit in the data and need to be discovered and made explicit to the end-user (i.e. the domain expert). For that one may exploit the notion of contextual identity links [110]. In this work I have proposed a semantic approach for contextual identity links detection expressing the set of classes and properties of an ontology for which two objects can be considered as identical. Therefore, I plan to develop an innovative approach which is able to detect gradual causal rules expressing the RDF subgraph of the changes (i.e. increase/decrease of some properties) associated with a given effect, in inter-domains knowledge graphs relying on time relations and contextual identity links.

Explainable Veracity Assessment in the Web

One of the four "*V's*" of big data is the *veracity* which refers to the trustworthiness of facts. Numerous research works have addressed the determination of the veracity of claims, in structured datasets [42, 81], social networks [91, 148] and the Web in general [105]. Some frameworks and APIs have recently began to appear, e.g., [16], Defacto [57]. As opposed to the current approaches, our objective is to combine information from multiple sources (e.g., social media, general-purpose and common-sense knowledge bases) for assessing the veracity of facts. Moreover, most of existing veracity assessment approaches have focused on identifying static facts encoded as binary relations (Fionda, Gutierrez, and Pirro 2016). However, the vast majority of facts are fluents (dynamic relations whose truth is a function of time), only holding during an interval of time. Facts like (<Obama isPresidentOf USA>) loose relevance without a temporal scope (2008–2016 in this case). Thus, these temporal contexts can be provided as explanations to the user. Nevertheless, there are very few sources where temporal information are associated to facts (e.g. Wikidata [141] or Yago2 [68] that contains temporal meta-facts, but for only 15 predicates). That is, in an ongoing work [85], we developed a rule-based approach which addresses the enrichment of existing knowledge graphs by explicit temporal information. Then, we envision to design a new approach that exploits these temporal information from several sources and combine them to provide an explainable veracity assessment of facts. Similarly, we also plan to study how spatial contexts can also be used to assess the veracity of facts. Finally, for the explanation models, we expect that results from argumentation theory [7] can be applied and exploited for providing rich and readable explanations to the user.

Some parts of this research perspective is an ongoing work in the setting of VASTE project (2018-2019) funded by the Labex Digicosme Paris Saclay.

Knowledge-based Privacy Preserving

The phenomenon of open big datasets is acknowledged and welcomed in numerous application domains, such as, life sciences, administration, cultural heritage, marketing, smart cities and so on. The sources are omnipresent and numerous like, IoT, social networks, navigation systems and health sources. In this context, two crucial and antagonistic problems have to be dealt with. The first concerns the opening and the use of data for a variety of tasks, ranging from scientific research to commercial exploitation. The second problem concerns preserving the privacy of individuals and ensuring the not disclosing information that may allow individual identification directly or indirectly through interlinking with other data sources. For this second purpose, efficient tools have to be designed for data anonymization and access control while accompanying the data openness and the GDPR[2] compliance. We envision to develop a flexible approach for data anonymization that is able to exploit domain knowledge on the identification of individuals such as keys, functional dependencies and referring expressions while considering the different cases, namely mono-source or multi-sources anonymization. GDPR constraints, such as the data retention duration (e.g. the list of books rent by a person should be removed after 3 months) and data storage restrictions have to be taken into account.

This research perspective is part of the GDPR work-package in the project DataForYou (2018-2020) with SAAT Paris Saclay and DataForYou Startup [3].

[2] https://eugdpr.org/
[3] https://www.dataforyou.fr/

BIBLIOGRAPHY

[1] Z. ABEDJAN, L. GOLAB, AND F. NAUMANN, *Profiling relational data: a survey*, VLDB J., 24 (2015), pp. 557–581.

[2] Z. ABEDJAN, P. SCHULZE, AND F. NAUMANN, *DFD: efficient functional dependency discovery*, in Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014, 2014, pp. 949–958.

[3] M. ACHICHI, P. LISENA, K. TODOROV, R. TRONCY, AND J. DELAHOUSSE, *DOREMUS: A graph of linked musical works*, in The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part II, 2018, pp. 3–19.

[4] M. ACOSTA, A. ZAVERI, E. SIMPERL, D. KONTOKOSTAS, S. AUER, AND J. LEHMANN, *Crowdsourcing linked data quality assessment*, in The Semantic Web – ISWC 2013, H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. X. Parreira, L. Aroyo, N. Noy, C. Welty, and K. Janowicz, eds., Berlin, Heidelberg, 2013, Springer Berlin Heidelberg, pp. 260–276.

[5] M. AL-BAKRI, M. ATENCIA, J. DAVID, S. LALANDE, AND M. ROUSSET, *Uncertainty-sensitive reasoning for inferring sameas facts in linked data*, in ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016), G. A. Kaminka, M. Fox, P. Bouquet, E. Hüllermeier, V. Dignum, F. Dignum, and F. van Harmelen, eds., vol. 285 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2016, pp. 698–706.

[6] M. AL-BAKRI, M. ATENCIA, S. LALANDE, AND M. ROUSSET, *Inferring same-as facts from linked data: An iterative import-by-query approach*, in Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA., B. Bonet and S. Koenig, eds., AAAI Press, 2015, pp. 9–15.

[7] L. AMGOUD AND H. PRADE, *Using arguments for making and explaining decisions*, Artif. Intell., 173 (2009), pp. 413–436.

[8] A. P. APROSIO, C. GIULIANO, AND A. LAVELLI, *Extending the coverage of dbpedia properties using distant supervision over wikipedia*, in Proceedings of the 2013th International Conference on NLP & DBpedia - Volume 1064, NLP-DBPEDIA'13, Aachen, Germany, Germany, 2013, CEUR-WS.org, pp. 20–31.

[9] C. ARECES, A. KOLLER, AND K. STRIEGNITZ, *Referring expressions as formulas of description logic*, in Proceedings of the Fifth International Natural Language Generation Conference, INLG '08, Stroudsburg, PA, USA, 2008, Association for Computational Linguistics, pp. 42–49.

[10] M. ATENCIA, M. CHEIN, M. CROITORU, J. DAVID, M. LECLÈRE, N. PERNELLE, F. SAÏS, F. SCHARFFE, AND D. SYMEONIDOU, *Defining key semantics for the RDF datasets: Experiments and evaluations*, in ICCS, 2014.

[11] M. ATENCIA, M. CHEIN, M. CROITORU, M. L. JEROME DAVID, N. PERNELLE, F. SAÏS, F. SCHARFFE, AND D. SYMEONIDOU, *Defining key semantics for the rdf datasets: Experiments and evaluations*, in ICCS, 2014.

[12] M. ATENCIA, J. DAVID, AND J. EUZENAT, *Data interlinking through robust linkkey extraction*, in ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014), 2014, pp. 15–20.

[13] M. ATENCIA, J. DAVID, AND J. EUZENAT, *Data interlinking through robust linkkey extraction*, in ECAI, Czech Republic, 2014.

[14] M. ATENCIA, J. DAVID, AND F. SCHARFFE, *Keys and pseudo-keys detection for web datasets cleansing and interlinking*, in EKAW, 2012.

[15] M. ATENCIA, J. DAVID, AND F. SCHARFFE, *Keys and pseudo-keys detection for web datasets cleansing and interlinking*, in Knowledge Engineering and Knowledge Management - 18th International Conference, EKAW 2012, Galway City, Ireland, October 8-12, 2012. Proceedings, 2012, pp. 144–153.

[16] M. L. BA, L. BERTI-EQUILLE, K. SHAH, AND H. M. HAMMADY, *Vera: A Platform for Veracity Estimation Over Web Data*, in WWW, 2016, pp. 159–162.

[17] W. BEEK, J. RAAD, J. WIELEMAKER, AND F. VAN HARMELEN, *sameas.cc: The closure of 500m owl:sameas statements*, in International ESWC Conference, 2018.

[18] W. BEEK, L. RIETVELD, AND S. SCHLOBACH, *Lod laundromat (archival package 2016/06)*. `https://doi.org/10.17026/dans-znh-bcg3`, 2016.

[19] W. BEEK, S. SCHLOBACH, AND F. VAN HARMELEN, *A contextualised semantics for owl: sameas*, in International Semantic Web Conference, Springer, 2016, pp. 405–419.

[20] T. BLEIFUSS, S. KRUSE, AND F. NAUMANN, *Efficient denial constraint discovery with hydra*, PVLDB, 11 (2017), pp. 311–323.

[21] J. BLEIHOLDER AND F. NAUMANN, *Data fusion*, ACM Comput. Surv., 41 (2009), pp. 1:1–1:41.

[22] V. BLONDEL, J.-L. GUILLAUME, R. LAMBIOTTE, AND E. LEFEBVRE, *Fast unfolding of communities in large networks*, J. of statistical mechanics, 2008 (2008), p. P10008.

[23] P. BOUQUET, F. GIUNCHIGLIA, F. VAN HARMELEN, L. SERAFINI, AND H. STUCKENSCHMIDT, *C-owl: Contextualizing ontologies*, in International Semantic Web Conference, Springer, 2003, pp. 164–179.

[24] L. BREIMAN, *Classification and Regression Trees*, Boca Raton: Chapman & Hall/CRC, 1984.

[25] V. BRYL AND C. BIZER, *Learning conflict resolution strategies for cross-language wikipedia data fusion*, in 23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014, Companion Volume, 2014, pp. 1129–1134.

[26] O. CAMPAIGN, *Im@oaei2010 - persons-restaurants (pr) dataset*, April 2014.

[27] J. J. CARROLL, C. BIZER, P. HAYES, AND P. STICKLER, *Named graphs, provenance and trust*, in International conference WWW, ACM, 2005, pp. 613–622.

[28] F. CHIANG AND R. J. MILLER, *Discovering data quality rules*, in VLDB, 2008.

[29] X. CHU, I. F. ILYAS, AND P. PAPOTTI, *Discovering denial constraints*, Proc. VLDB Endow., 6 (2013), pp. 1498–1509.

[30] W. W. COHEN, P. RAVIKUMAR, AND S. E. FIENBERG, *A comparison of string distance metrics for name-matching tasks.*, in Proceedings of IJCAI-03 Workshop on Information Integration, August 2003, pp. 73–78.

[31] P. CUDREMAUROUX, P. HAGHANI, M. JOST, K. ABERER, AND H. DE MEER, *idmesh: graph-based disambiguation of linked data*, in International conference WWW, ACM, 2009, pp. 591–600.

[32] J. CUZZOLA, E. BAGHERI, AND J. JOVANOVIC, *Filtering inaccurate entity co-references on the linked open data*, in International DEXA Conference, Springer, 2015, pp. 128–143.

[33] G. DE MELO, *Not quite the same: Identity constraints for the web of linked data.*, in AAAI, AAAI Press, 2013.

[34] S. DE ROOIJ, W. BEEK, P. BLOEM, F. VAN HARMELEN, AND S. SCHLOBACH, *Are names meaningful? quantifying social meaning on the semantic web*, in ISWC, Springer, 2016, pp. 184–199.

[35] M. DEAN, G. SCHREIBER, S. BECHHOFER, F. VAN HARMELEN, J. HENDLER, I. HORROCKS, D. MCGUINNESS, P. PATEL-SCHNEIDER, AND L. STEIN, *Owl web ontology language reference*, W3C Recommendation February, 10 (2004).

[36] S. DESTERCKE, F. SAIS, AND R. THOMOPOULOS, *Fusion évidentielle de références et interrogation flexible*, in LFA: Logique Floue et ses Applications, LFA '09, 2009, pp. 15–22.

[37] L. DING, J. SHINAVIER, T. FININ, AND D. L. MCGUINNESS, *owl:sameAs and Linked Data: An empirical study*, in International Web Science Conference, 2010.

[38] L. DING, J. SHINAVIER, Z. SHANGGUAN, AND D. MCGUINNESS, *Sameas networks and beyond: Analyzing deployment status and implications of owl: sameas in linked data*, in Intern. Semantic Web Conference, vol. 6496 of LNCS, Springer, 2010, pp. 145–160.

[39] X. DONG, E. GABRILOVICH, G. HEITZ, W. HORN, N. LAO, K. MURPHY, T. STROHMANN, S. SUN, AND W. ZHANG, *Knowledge vault: a web-scale approach to probabilistic knowledge fusion*, in The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014, 2014, pp. 601–610.

[40] X. DONG, A. HALEVY, AND J. MADHAVAN, *Reference reconciliation in complex information spaces*, in ACM SIGMOD, ACM Press, 2005, pp. 85–96.

[41] X. L. DONG, E. GABRILOVICH, G. HEITZ, W. HORN, K. MURPHY, S. SUN, AND W. ZHANG, *From data fusion to knowledge fusion*, in VLDB, 2014.

[42] X. L. DONG, E. GABRILOVICH, K. MURPHY, V. DANG, W. HORN, C. LUGARESI, S. SUN, AND W. ZHANG, *Knowledge-based Trust: Estimating the Trustworthiness of Web Sources*, VLDB, 8 (2015), pp. 938–949.

[43] D. DUBOIS AND H. PRADE, *Possibility theory in information fusion*, in Data fusion and Perception, G. D. Riccia, H. Lenz, and R. Kruse, eds., vol. CISM Courses and Lectures N 431, Springer Verlag, Berlin, 2001, pp. 53–76.

[44] D. DUBOIS AND H. PRADE, *On the use of aggregation operations in information fusion processes*, Fuzzy Sets and Systems, 142 (2004), pp. 143–161.

[45] L. EHRLINGER AND W. WÖSS, *Towards a definition of knowledge graphs*, in Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic

Systems - SEMANTiCS2016 and the 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS'16) co-located with the 12th International Conference on Semantic Systems (SEMANTiCS 2016), Leipzig, Germany, September 12-15, 2016., 2016.

[46] M. B. ELLEFI, Z. BELLAHSENE, J. G. BRESLIN, E. DEMIDOVA, S. DIETZE, J. SZYMANSKI, AND K. TODOROV, *RDF dataset profiling - a survey of features, methods, vocabularies and applications*, Semantic Web, 9 (2018), pp. 677–705.

[47] P. ERNST, C. MENG, A. SIU, AND G. WEIKUM, *Knowlife: A knowledge graph for health and life sciences*, in IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014, 2014, pp. 1254–1257.

[48] W. FAN, Z. FAN, C. TIAN, AND X. L. DONG, *Keys for graphs*, PVLDB, 8 (2015), pp. 1590–1601.

[49] W. FAN, F. GEERTS, J. LI, AND M. XIONG, *Discovering conditional functional dependencies*, IEEE Trans. on Knowl. and Data Eng., (2011).

[50] H. FARAH, D. SYMEONIDOU, AND K. TODOROV, *Keyranker: Automatic RDF key ranking for data linking*, in Proceedings of the Knowledge Capture Conference, K-CAP 2017, Austin, TX, USA, December 4-6, 2017, 2017, pp. 7:1–7:8.

[51] J. D. FERNÁNDEZ, W. BEEK, M. A. MARTÍNEZ-PRIETO, AND M. ARIAS, *LOD-a-lot - A Queryable Dump of the LOD Cloud*, in ISWC, 2017.

[52] A. FERRARA, A. NIKOLOV, AND F. SCHARFFE, *Data linking for the semantic web*, Int. J. Semantic Web Inf. Syst., 7 (2011), pp. 46–76.

[53] P. A. FLACH AND I. SAVNIK, *Database dependency discovery: A machine learning approach*, AI Commun., 12 (1999), pp. 139–160.

[54] G. FLOURIS, Y. R. andMARIA POVEDA-VILLALON andPABLO N. MENDES, AND I. FUNDULAKI, *Using provenance for quality assessment and repair in linked open data*, in In Proceedings of the 2nd Joint Workshop on Knowledge Evolution and Ontology Dynamics (EvoDyn-12), 2012.

[55] L. GALÁRRAGA, C. TEFLIOUDI, K. HOSE, AND F. M. SUCHANEK, *AMIE: association rule mining under incomplete evidence in ontological knowledge bases*, in WWW, 2013.

[56] L. A. GALÁRRAGA, C. TEFLIOUDI, K. HOSE, AND F. M. SUCHANEK, *AMIE: association rule mining under incomplete evidence in ontological knowledge bases*, in 22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013, 2013, pp. 413–422.

[57] D. GERBER, D. ESTEVES, J. LEHMANN, L. BÜHMANN, R. USBECK, A.-C. NGONGA NGOMO, AND R. SPECK, *DeFacto-Temporal and Multilingual Deep Fact Validation*, Web Semant., 35 (2015), pp. 85–101.

[58] I. GIANNOPOULOU, F. SAÏS, AND R. THOMOPOULOS, *Linked data annotation and fusion driven by data quality evaluation*, in 15èmes Journées Francophones sur l'Extraction et Gestion des Connaissances, EGC 2015, 27-30 Janvier 2015, Luxembourg, 2015, pp. 257–262.

[59] J. M. GIMÉNEZ-GARCÍA, A. ZIMMERMANN, AND P. MARET, *Ndfluents: an ontology for annotated statements with inference preservation*, in International ESWC Conference, Springer, 2017, pp. 638–654.

[60] H. GLASER, A. JAFFRI, AND I. MILLARD, *Managing co-reference on the semantic web*, in WWW2009 Workshop: Linked Data on the Web (LDOW2009), April 2009.

[61] L. GOLAB, H. KARLOFF, F. KORN, D. SRIVASTAVA, AND B. YU, *On generating near-optimal tableaux for conditional functional dependencies*, VLDB, (2008).

[62] T. R. GRUBER, *Toward principles for the design of ontologies used for knowledge sharing*, Int. J. Hum.-Comput. Stud., 43 (1995), pp. 907–928.

[63] C. GUÉRET, P. GROTH, C. STADLER, AND J. LEHMANN, *Assessing linked data mappings using network measures*, in Extended Semantic Web Conference, Springer, 2012, pp. 87–102.

[64] H. HALPIN, P. J. HAYES, J. P. MCCUSKER, D. L. MCGUINNESS, AND H. S. THOMPSON, *When owl:sameAs isn't the same: An analysis of identity in Linked Data*, in International Semantic Web Conference, Springer, 2010, pp. 305–320.

[65] J. HEFLIN AND H. MUÑOZ-AVILA, *LCW-based agent planning for the semantic web*, in Ontologies and the Semantic Web Workshop, AAAI Press, 2002, pp. 63–70.

[66] A. HEISE, J. QUIANÉ-RUIZ, Z. ABEDJAN, A. JENTZSCH, AND F. NAUMANN, *Scalable discovery of unique column combinations*, PVLDB, 7 (2013), pp. 301–312.

[67] P. HITZLER, M. KRÖTZSCH, B. PARSIA, P. PATEL-SCHNEIDER, AND S. RUDOLPH, eds., *OWL 2 Web Ontology Language: Primer*, W3C Recommendation, 2009.
Available at `http://www.w3.org/TR/owl2-primer/`.

[68] J. HOFFART, F. M. SUCHANEK, K. BERBERICH, AND G. WEIKUM, *Yago2: A spatially and temporally enhanced knowledge base from wikipedia*, Artif. Intell., 194 (2013), pp. 28–61.

[69] A. HOGAN, A. ZIMMERMANN, J. UMBRICH, A. POLLERES, AND S. DECKER, *Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora*, J. Web Sem., 10 (2012), pp. 76–110.

[70] W. HU, J. CHEN, AND Y. QU, *A self-training approach for resolving object coreference on the semantic web*, in WWW, 2011, pp. 87–96.

[71] Y. HUHTALA, J. KÄRKKÄINEN, P. PORKKA, AND H. TOIVONEN, *Tane: An efficient algorithm for discovering functional and approximate dependencies*, Computer Journal, 42 (1999).

[72] L. IBANESCU, J. DIBIE, S. DERVAUX, E. GUICHARD, AND J. RAAD, *Poˆ 2-a process and observation ontology in food science. application to dairy gels*, in Metadata and Semantics Research: 10th International Conference, MTSR 2016, Göttingen, Germany, November 22-25, 2016, Proceedings, Springer, 2016, pp. 155–165.

[73] A. K. IDRISSOU, R. HOEKSTRA, F. VAN HARMELEN, A. KHALILI, AND P. VAN DEN BESSELAAR, *Is my: sameas the same as your: sameas?: Lenticular lenses for context-specific identity*, in International K-CAP Conference, ACM, 2017, p. 23.

[74] R. IHAKA AND R. GENTLEMAN, *R: A language for data analysis and graphics*, Journal of Computational and Graphical Statistics, 5 (1996), pp. 299–314.

[75] A. JAFFRI, H. GLASER, AND I. MILLARD, *URI disambiguation in the context of linked data*, in WWW Workshop on Linked Data on the Web, LDOW, 2008.

[76] J. D. LAFFERTY, A. MCCALLUM, AND F. C. N. PEREIRA, *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*, in Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01, San Francisco, CA, USA, 2001, Morgan Kaufmann Publishers Inc., pp. 282–289.

[77] D. LANGE, C. BÖHM, AND F. NAUMANN, *Extracting structured information from wikipedia articles to populate infoboxes*, in Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM '10, New York, NY, USA, 2010, ACM, pp. 1661–1664.

[78] J. LEHMANN, R. ISELE, M. JAKOB, A. JENTZSCH, D. KONTOKOSTAS, P. N. MENDES, S. HELLMANN, M. MORSEY, P. VAN KLEEF, S. AUER, AND C. BIZER, *DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia*, Semantic Web J., 6 (2015).

[79] D. B. LENAT AND E. A. FEIGENBAUM, *On the thresholds of knowledge*, Artif. Intell., 47 (1991), pp. 185–250.

[80] M.-J. LESOT, *Similarity, typicality and fuzzy prototypes for numerical data*, Journal Res-Systemica, Special issue on the 6th European Congress on Systems Science, 5 (2005).

[81] Q. LI, Y. LI, J. GAO, L. SU, B. ZHAO, M. DEMIRBAS, W. FAN, AND J. HAN, *A Confidence-aware Approach for Truth Discovery on Long-tail Data*, VLDB, 8 (2014), pp. 425–436.

[82] J. LIU, J. LI, C. LIU, AND Y. CHEN, *Discover dependencies from data—a review*, IEEE Trans. on Knowl. and Data Eng., 24 (2012), pp. 251–264.

[83] S. LOPES, J.-M. PETIT, AND L. LAKHAL, *Efficient discovery of functional dependencies and armstrong relations*, in Proceedings of the 7th International Conference on Extending Database Technology: Advances in Database Technology, EDBT '00, London, UK, UK, 2000, Springer-Verlag, pp. 350–364.

[84] W. LOYOLA, *Comparison of approaches toward formalising context: implementation characteristics and capacities*, Electronic Journal of Knowledge Management, 5 (2007), pp. 203–214.

[85] J. G. MALAVERRI, F. SAÏS, AND G. QUERCINI, *Enriching knowledge bases with temporal meta-facts*, in Atelier VERITA@EGC19, organisé dans le cadre des 19èmes Journées Francophones sur l'Extraction et Gestion des Connaissances, EGC 2019, 21-25 Janvier 2019, Metz, 2019.

[86] F. MANOLA AND E. MILLER, *RDF primer*, W3C recommendation, W3C, Feb. 2004. http://www.w3.org/TR/2004/REC-rdf-primer-20040210/.

[87] F. MANOLA, E. MILLER, AND B. MCBRIDE, *Rdf primer*, W3C recommendation, 10 (2004), pp. 1–107.

[88] P. N. MENDES, H. MÜHLEISEN, AND C. BIZER, *Sieve: linked data quality assessment and fusion*, in Proceedings of the 2012 Joint EDBT/ICDT Workshops, Berlin, Germany, March 30, 2012, 2012, pp. 116–123.

[89] J. MICHELFEIT AND J. MYNARZ, *New directions in linked data fusion*, in Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014., 2014, pp. 397–400.

[90] M. MINTZ, S. BILLS, R. SNOW, AND D. JURAFSKY, *Distant supervision for relation extraction without labeled data*, in Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2, ACL '09, Stroudsburg, PA, USA, 2009, Association for Computational Linguistics, pp. 1003–1011.

[91] S. MUKHERJEE AND G. WEIKUM, *Leveraging Joint Interactions for Credibility Analysis in News Communities*, in CIKM, ACM, 2015, pp. 353–362.

[92] E. MUÑOZ, A. HOGAN, AND A. MILEO, *Triplifying wikipedia's tables*, in Proceedings of the First International Workshop on Linked Data for Information Extraction (LD4IE 2013) co-located with the 12th International Semantic Web Conference (ISWC 2013), Sydney, Australia, October 21, 2013., 2013.

[93] M. NENTWIG, M. HARTUNG, A. N. NGOMO, AND E. RAHM, *A survey of current link discovery frameworks*, Semantic Web, 8 (2017), pp. 419–436.

[94] C. B. NETO, D. KONTOKOSTAS, S. HELLMANN, K. MÜLLER, AND M. BRÜMMER, *Assessing quantity and quality of links between link data datasets*, in Proceedings of the Workshop on Linked Data on the Web, LDOW 2016, co-located with 25th International World Wide Web Conference (WWW 2016), 2016.

[95] A.-C. N. NGOMO AND S. AUER, *Limes a time-efficient approach for large-scale link discovery on the web of data*, in IJCAI, 2011, pp. 2312–2317.

[96] V. NGUYEN, O. BODENREIDER, AND A. SHETH, *Don't like RDF reification?: making statements about statements using singleton property*, in International conference WWW, ACM, 2014, pp. 759–770.

[97] W. OWL WORKING GROUP, *OWL 2 Web Ontology Language: Document Overview*, W3C Recommendation, 27 October 2009.
Available at `http://www.w3.org/TR/owl2-overview/`.

[98] D. C. PAPAGEORGIOU, N. PERNELLE, AND F. SAÏS, *Approche numérique pour l'invalidation de liens d'identité (owl:sameas)*, in IC 2017 : 28es Journées francophones d'Ingénierie des Connaissances (Proceedings of the 28th French Knowledge Engineering Conference), Caen, France, July 3-7, 2017., 2017.

[99] L. PAPALEO, N. PERNELLE, F. SAÏS, AND C. DUMONT, *Logical detection of invalid sameas statements in rdf data*, in International Conference EKAW, Springer, 2014, pp. 373–384.

[100] T. PAPENBROCK, J. EHRLICH, J. MARTEN, T. NEUBERT, J. RUDOLPH, M. SCHÖNBERG, J. ZWIENER, AND F. NAUMANN, *Functional dependency discovery: An experimental evaluation of seven algorithms*, PVLDB, 8 (2015), pp. 1082–1093.

[101] H. PAULHEIM, *Identifying wrong links between datasets by multi-dimensional outlier detection.*, in WoDOOM, 2014, pp. 27–38.

[102] H. PAULHEIM, *Knowledge graph refinement: A survey of approaches and evaluation methods*, Semantic Web, 8 (2017), pp. 489–508.

[103] H. PAULHEIM AND S. P. PONZETTO, *Extending dbpedia with wikipedia list pages*, in Proceedings of the NLP & DBpedia workshop co-located with the 12th International Semantic Web Conference (ISWC 2013), Sydney, Australia, October 22, 2013., 2013.

[104] N. PERNELLE, F. SAÏS, AND D. SYMEONIDOU, *An automatic key discovery approach for data linking*, Journal of Web Semantics, 23 (2013), pp. 16–30.

[105] K. POPAT, S. MUKHERJEE, J. STRÖTGEN, AND G. WEIKUM, *Where the Truth Lies: Explaining the Credibility of Emerging Claims on the Web and Social Media*, in WWW, 2017, pp. 1003–1012.

[106] N. PREDA, G. KASNECI, F. M. SUCHANEK, T. NEUMANN, W. YUAN, AND G. WEIKUM, *Active knowledge: dynamically enriching RDF knowledge bases by web services*, in Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010, Indianapolis, Indiana, USA, June 6-10, 2010, 2010, pp. 399–410.

[107] J. RAAD, W. BEEK, F. VAN HARMELEN, N. PERNELLE, AND F. SAÏS, *Detecting erroneous identity links on the web using network metrics*, in The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, MONTEREY, CALIFORNIA, USA, October 8-12, 2018, 2018.

[108] ——, *Detecting erroneous identity links on the web using network metrics*, in The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I, 2018, pp. 391–407.

[109] J. RAAD, N. PERNELLE, AND F. SAÏS, *Decide - detecting contextual identity*, tech. rep., LRI, Paris-Sud University, 2017.

[110] J. RAAD, N. PERNELLE, AND F. SAÏS, *Detection of contextual identity links in a knowledge base*, in International K-CAP Conference, ACM, 2017, p. 8.

[111] J. RAAD, N. PERNELLE, F. SAÏS, J. DIBIE-BARTHÉLEMY, L. IBANESCU, AND S. DERVAUX, *Comment représenter et découvrir des liens d'identités contextuels dans une base de connaissances: applications à des données expérimentales en science du vivant*, in Revue d'intelligence artificielle, 2018, p. to appear.

[112] W. RECOMMENDATION, *Owl 2 web ontology language: Structural specification and functional-style syntax*, in http://www.w3.org/TR/owl2-syntax/, P. B. Motik B., Patel-Schneider P. F., ed., W3C, 27 October 2009.

[113] ——, *Owl2 web ontology language: Direct semantics*, in http://www.w3.org/TR/owl2-direct-semantics, C. G. B. Motik B., Patel-Schneider P. F., ed., W3C, 27 October 2009.

[114] D. RITZE, O. LEHMBERG, AND C. BIZER, *Matching html tables to dbpedia*, in Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics, WIMS '15, New York, NY, USA, 2015, ACM, pp. 10:1–10:6.

[115] G. ROBINSON AND L. WOS, *Paramodulation and theorem-proving in first-order theories with equality*, in Automation of Reasoning 2: Classical Papers on Computational Logic, Springer, 1969, pp. 298–313.

[116] S. J. RUSSELL AND P. NORVIG, *Artificial intelligence - a modern approach, 2nd Edition*, Prentice Hall series in artificial intelligence, Prentice Hall, 2003.

[117] F. SAÏS, *Semantic Data Integration driven by an Ontology. (Intégration sémantique de données guidée par un ontologie).*, PhD thesis, University of Paris-Sud, Orsay, France, 2007.

[118] F. SAÏS, N. PERNELLE, AND M.-C. ROUSSET, *Combining a logical and a numerical method for data reconciliation*, Journal on Data Semantics, 12 (2009), pp. 66–94.

[119] F. SAÏS, N. PERNELLE, AND M.-C. ROUSSET, *Combining a logical and a numerical method for data reconciliation*, Journal of Data Semantics (JoDS), 12 (2009), pp. 66–94.
LNCS 5480.

[120] F. SAÏS, C. PRUSKI, AND M. D. SILVEIRA, *Inferring the evolution of ontology axioms from RDF data dynamics*, in Proceedings of the Knowledge Capture Conference, K-CAP 2017, Austin, TX, USA, December 4-6, 2017, 2017, pp. 43:1–43:4.

[121] F. SAÏS AND R. THOMOPOULOS, *A reconciliation-driven approach of case-based prediction: State of the art, method overview and application in food science*.

[122] F. SAÏS AND R. THOMOPOULOS, *Reference fusion and flexible querying*, in Proceedings of On the Move to Meaningful Internet Systems: OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008, Part II, 2008, pp. 1541–1549.

[123] ——, *Ontology-aware prediction from rules: A reconciliation-based approach*, Knowledge-Based Systems, 67 (2014), pp. 117–130.

[124] F. SAÏS, R. THOMOPOULOS, AND S. DESTERCKE, *Ontology-driven possibilistic reference fusion*, in Proceedings of On the Move to Meaningful Internet Systems, OTM 2010 - Confederated International Conferences: CoopIS, IS, DOA and ODBASE, Part II, 2010, pp. 1079–1096.

[125] C. SARASUA, S. STAAB, AND M. THIMM, *Methods for intrinsic evaluation of links in the web of data*, in The Semantic Web - 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 - June 1, 2017, Proceedings, Part I, 2017, pp. 68–84.

[126] S. SER, F. SAÏS, AND M. TEISSEIRE, *Découverte de motifs graduels partiellement ordonnés : application aux données d'expériences scientifiques*, in Extraction et Gestion des Connaissances, EGC 2018, Paris, France, January 23-26, 2018, 2018, pp. 227–238.

[127] Y. SISMANIS, P. BROWN, P. J. HAAS, AND B. REINWALD, *Gordian: efficient and scalable discovery of composite keys*, in VLDB, 2006.

[128] R. SOCHER, D. CHEN, C. D. MANNING, AND A. Y. NG, *Reasoning with neural tensor networks for knowledge base completion*, in Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1, NIPS'13, USA, 2013, Curran Associates Inc., pp. 926–934.

[129] T. SORU, E. MARX, AND A. N. NGOMO, *ROCKER: A refinement operator for key discovery*, in WWW, 2015.

[130] F. M. SUCHANEK, G. KASNECI, AND G. WEIKUM, *Yago: a core of semantic knowledge*, in WWW, 2007.

[131] D. SYMEONIDOU, V. ARMANT, N. PERNELLE, AND F. SAÏS, *SAKey: Scalable Almost Key discovery in RDF data*, in Proceedings of the 13th International Semantic Web Conference (ISWC2014), ISWC '14, Springer Verlag, 2014, p. 16.

[132] D. SYMEONIDOU, V. ARMANT, N. PERNELLE, AND F. SAÏS, *Sakey: Scalable almost key discovery in RDF data*, in The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I, 2014, pp. 33–49.

[133] D. SYMEONIDOU, L. GALARRÁGA, N. PERNELLE, F. SAÏS, AND F. SUCHANEK, *VICKEY: Mining Conditional Keys on RDF datasets*, tech. rep., `https://doi.org/10.5281/zenodo.835647`, 2017.

[134] D. SYMEONIDOU, L. GALÁRRAGA, N. PERNELLE, F. SAÏS, AND F. M. SUCHANEK, *VICKEY: mining conditional keys on knowledge bases*, in The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I, 2017, pp. 661–677.

[135] D. SYMEONIDOU, N. PERNELLE, AND F. SAÏS, *KD2R: A key discovery method for semantic reference reconciliation*, in On the Move to Meaningful Internet Systems: OTM 2011 Workshops - Confederated International Workshops and Posters: EI2N+NSF ICE, ICSP+INBAST, ISDE, ORM, OTMA, SWWS+MONET+SeDeS, and VADER 2011, Hersonissos, Crete, Greece, October 17-21, 2011. Proceedings, 2011, pp. 392–401.

[136] D. SYMEONIDOU, I. SANCHEZ, M. CROITORU, P. NEVEU, N. PERNELLE, F. SAÏS, A. ROLAND-VIALARET, P. BUCHE, A. MULJARTO, AND R. SCHNEIDER, *Key discovery for numerical data: Application to oenological practices*, in ICCS, 2016.

[137] R. THOMOPOULOS, J.-F. BAGET, AND O. HAEMMERLÉ, *Conceptual graphs as cooperative formalism to build and validate a domain expertise*, in ICCS, 2007, pp. 112–125.

[138] R. THOMOPOULOS, S. DESTERCKE, B. CHARNOMORDIC, I. JOHNSON, AND J. ABÉCASSIS, *An iterative approach to build relevant ontology-aware data-driven models*, Information Sciences, 221 (2013), pp. 452–472.

[139] A. VALDESTILHAS, T. SORU, AND A.-C. N. NGOMO, *Cedal: time-efficient detection of erroneous links in large-scale link repositories*, in International Conference on Web Intelligence, ACM, 2017, pp. 106–113.

[140] J. VOLZ, C. BIZER, M. GAEDKE, AND G. KOBILAROV, *Discovering and maintaining links on the web of data*, in Proceedings of the 8th International Semantic Web Conference(ISWC), ISWC '09, Berlin, Heidelberg, 2009, Springer-Verlag, pp. 650–665.

[141] D. VRANDEČIĆ AND M. KRÖTZSCH, *Wikidata: a free collaborative knowledgebase*, Comm. of the ACM, 57 (2014).

[142] W3C RECOMMENDATION., *http://www.w3.org/tr/owl-features/*.

[143] ——, *http://www.w3.org/tr/rdf-schema/*.

[144] W3C SUBMISSION., *http://www.w3.org/submission/swrl/*.

[145] R. WEST, E. GABRILOVICH, K. MURPHY, S. SUN, R. GUPTA, AND D. LIN, *Knowledge base completion via search-based question answering*, in Proceedings of the 23rd International Conference on World Wide Web, WWW '14, New York, NY, USA, 2014, ACM, pp. 515–526.

[146] Z. WU AND M. PALMER, *Verbs semantics and lexical selection*, in Proceedings of the 32nd annual meeting on Association for Computational Linguistics, Morristown, NJ, USA, 1994, Association for Computational Linguistics, pp. 133–138.

[147] C. WYSS, C. GIANNELLA, AND E. L. ROBERTSON, *Fastfds: A heuristic-driven, depth-first algorithm for mining functional dependencies from relation instances - extended abstract*, in Proceedings of the Third International Conference on Data Warehousing and Knowledge Discovery, DaWaK '01, London, UK, UK, 2001, Springer-Verlag, pp. 101–110.

[148] A. YATES, N. GOHARIAN, AND O. FRIEDER, *Extracting Adverse Drug Reactions from Social Media*, in AAAI, vol. 15, 2015, pp. 2460–2467.

[149] R. YU, U. GADIRAJU, B. FETAHU, O. LEHMBERG, D. RITZE, AND S. DIETZE, *Knowmore – knowledge base augmentation with structured web markup*, Semantic Web Journal, IOS Press, (2017).

[150] J. R. YVES, E. SHIRONOSHITA, AND M. KABUKA, *Ontology matching with semantic verification*, Web Semantics, 7 (2009), pp. 235–251.

[151] F. ZABLITH, G. ANTONIOU, M. D'AQUIN, G. FLOURIS, H. KONDYLAKIS, E. MOTTA, D. PLEXOUSAKIS, AND M. SABOU, *Ontology evolution: a process-centric survey*, Knowledge Eng. Review, 30 (2015), pp. 45–75.

[152] A. ZAVERI, A. RULA, A. MAURINO, R. PIETROBON, J. LEHMANN, AND S. AUER, *Quality assessment for linked data: A survey*, Semantic Web, 7 (2016), pp. 63–93.

[153] J. ZHANG, A. SILVESCU, AND V. HONAVAR, *Ontology-driven induction of decision trees at multiple levels of abstraction*, Lecture Notes in Computer Science, (2002), pp. 316–323.

[154] Y. ZHAO, S. GAO, P. GALLINARI, AND J. GUO, *Knowledge base completion by learning pairwise-interaction differentiated embeddings*, Data Mining and Knowledge Discovery, 29 (2015), pp. 1486–1504.