11th International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2015)

Proceedings

edited by | Fernando Bobillo
Rommel N. Carvalho
Davide Ceolin
Paulo C. G. da Costa
Claudia d'Amato
Nicola Fanizzi
Kathryn B. Laskey
Kenneth J. Laskey
Thomas Lukasiewicz
Trevor Martin
Matthias Nickles
Michael Pool

Bethlehem, USA, October 12, 2015

collocated with
the 14th International Semantic Web Conference
(ISWC 2015)

Foreword

This volume contains the papers presented at the 11th International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2015), held as a part of the 14th International Semantic Web Conference (ISWC 2015) at Bethlehem, USA, October 12, 2015. 4 technical papers and 2 short papers were accepted at URSW 2015. All the papers were selected in a rigorous reviewing process, where each paper was reviewed by three program committee members.

The International Semantic Web Conference is a major international forum for presenting visionary research on all aspects of the Semantic Web. The International Workshop on Uncertainty Reasoning for the Semantic Web provides an opportunity for collaboration and cross-fertilization between the uncertainty reasoning community and the Semantic Web community.

We wish to thank all authors who submitted papers and all workshops participants for fruitful discussions. We would like to thank the program committee members for their timely expertise in carefully reviewing the submissions.

October 2015

Fernando Bobillo
Rommel N. Carvalho
Davide Ceolin
Paulo C. G. da Costa
Claudia d'Amato
Nicola Fanizzi
Kathryn B. Laskey
Kenneth J. Laskey
Thomas Lukasiewicz
Trevor Martin
Matthias Nickles
Michael Pool

URSW 2015
Workshop Organization

Organizing Committee

Fernando Bobillo (University of Zaragoza, Spain)
Rommel N. Carvalho (Universidade de Brasília, Brazil)
Paulo C. G. da Costa (George Mason University, USA)
Davide Ceolin (VU University Amsterdam, The Netherlands)
Claudia d'Amato (University of Bari, Italy)
Nicola Fanizzi (University of Bari, Italy)
Kathryn B. Laskey (George Mason University, USA)
Kenneth J. Laskey (MITRE Corporation, USA)
Thomas Lukasiewicz (University of Oxford, UK)
Trevor Martin (University of Bristol, UK)
Matthias Nickles (National University of Ireland, Ireland)
Michael Pool (Goldman Sachs, USA)

Program Committee

Fernando Bobillo (University of Zaragoza, Spain)
Rommel N. Carvalho (Universidade de Brasília, Brazil)
Davide Ceolin (VU University Amsterdam, The Netherlands)
Paulo C. G. da Costa (George Mason University, USA)
Fabio Gagliardi Cozman (Universidade de São Paulo, Brazil)
Claudia d'Amato (University of Bari, Italy)
Nicola Fanizzi (University of Bari, Italy)
Marcelo Ladeira (Universidade de Brasília, Brazil)
Kathryn B. Laskey (George Mason University, USA)
Kenneth J. Laskey (MITRE Corporation, USA)
Thomas Lukasiewicz (University of Oxford, UK)
Trevor Martin (University of Bristol, UK)
Alessandra Mileo (DERI Galway, Ireland)
Matthias Nickles (National University of Ireland, Ireland)
Jeff Z. Pan (University of Aberdeen, UK)
Rafael Peñaloza (TU Dresden, Germany)
Michael Pool (Goldman Sachs, USA)
Livia Predoiu (University of Mannheim, Germany)
Guilin Qi (Southeast University, China)
David Robertson (University of Edinburgh, UK)
Daniel Sánchez (University of Granada, Spain)

Giorgos Stoilos (National Technical University of Athens, Greece)
Umberto Straccia (ISTI-CNR, Italy)
Matthias Thimm (Universität Koblenz-Landau, Germany)
Peter Vojtáš (Charles University Prague, Czech Republic)

Table of Contents

URSW 2015 Technical Papers

URSW 2015 Short Papers

Evaluating Uncertainty in Textual Document

Fadhela Kerdjoudj[1,2] and Olivier Curé[1,3]

[1] University of Paris-Est Marne-la-vallée, LIGM, CNRS UMR 8049, France,
{fadhela.kerdjoudj, ocure}@univ-mlv.fr
[2] GeolSemantics, 12 rue Raspail 74250, Gentilly, France.
[3] Sorbonne Universités, UPMC Univ Paris 06, LIP6, CNRS UMR 7606, France

Abstract. In this work, we consider that a close collaboration between the research fields of Natural Language Processing and Knowledge Representation becomes essential to fulfill the vision of the Semantic Web. This will permit to retrieve information from vast amount of textual documents present on the Web and to represent these extractions in an amenable manner for querying and reasoning purposes. In such a context, uncertain, incomplete and ambiguous information must be handled properly. In the following, we present a solution that enables to qualify and quantify the uncertainty of extracted information from linguistic treatment.

1 Introduction

Textual documents abound on the World Wide Web but efficiently retrieving information from them is hard due to their natural language expression and unstructured characteristics. Indeed, the ability to represent, characterize and manage uncertainty is considered as a key factor for the success of the Semantic Web [12]. The accurate and exhaustive extraction of information and knowledge is nevertheless needed in many application domains, *e.g.*, in medicine to comprehend the meaning of clinical reports or in finance to analyze the trends of markets. We consider that together with techniques from Natural Language Processing (NLP), best practices encountered in the Semantic Web have the potential to provide a solution to this problem. For instance, NLP can support the extraction of named entities as well as temporal and spatial aspects, while the Semantic Web is able to provide an agreed upon representation as well as some querying and reasoning facilities. Moreover, by consulting datasets form Linked Open Data (LOD), *e.g.*, DBpedia, Geonames, we can enrich the extracted knowledge and integrate it to the rest of the LOD.

The information contained in Web documents can present some imperfection, it can be incomplete, uncertain and ambiguous. Therefore, the texts content can be called into question, it becomes necessary to qualify and possibly quantify these imperfections to present to the end user a trusted extraction. However, qualification or quantification is a difficult task for any software application. In this paper, we focus on the uncertainty aspect and trustworthiness of the provided information in the text. A special attention of our work has been devoted

to representing such information within the Resource Description Framework (RDF) graph model. The main motivation being to benefit from querying facilities, *i.e.*, using SPARQL.
Usually, uncertainty is represented using reification, but this representation failed in representing uncertainty on triple property. Indeed, the reification does not identify which part of the triple (subject, predicate or the object) is uncertain. Here, we intend to manage these cases of uncertainties, as expressed in Example 1, while in the first sentence, the uncertainty concerns all the moving action (including, the agent, the destination and the date), in the second, the author expressed an uncertainty only on the date of the moving.

Example 1. 1. The US president *probably visited* Cuba this year.
2. The US president visited Cuba, *probably this year.*

We based our approach on an existing system developed at GEOLSemantics[4], a french startup with expertise in NLP. This framework mainly consists of a deep morphosyntactic analysis and an RDF triple creation using trigger's detection. Triggers are composed of one or several words (nouns, verbs, *etc.*) that represent a semantic unit denoting an entity to extract. For instance, the verb "go" denotes a Displacement. The RDF graph obtained complies with an ontology built manually to support different domains such as Security and Economics. Actually, our framework consists of a set of existing vocabularies (such as Schema.org[5], FOAF[6], Prov[7]) to enrich our own main ontology, denoted `geol`. This ontology contains the general classes which are common to many domains:

- *Document* : Text, Sentence, Source, DateIssue, PlaceIssue, etc.
- *Named entities* : Person, Organization, Location, etc.
- *Actions* : LegalProceeding, Displacing, etc.
- *Events* : SocialEvent, SportEvent, FamilialEvent, etc.

The contributions of this paper are two-fold: (1) We present a fine-grained approach to quantify and qualify the uncertainty in the text based on uncertainty markers; (2) We present an ontology which handles this uncertainty both at the resource and property level. This representation of uncertainty can be interrogated with a rewriting of SPARQL query.

The paper is organized as follows. Section 2 describes related work to uncertainty handling in Semantic Web. In Section 3, we present how to spot uncertain information in the text using specific markers. In Section 4, we propose an RDF-based representation of uncertainty in knowledge extraction. In Section 5, a use case is depicted with some SPARQL queries. Finally, we conclude in Section 6.

[4] http://www.geolsemantics.com/
[5] `http://schema.org/docs/schemaorg.owl`
[6] `http://xmlns.com/foaf/spec/`
[7] http://www.w3.org/TR/prov-o/

2 Related work

Integration of imprecise and uncertain concepts to ontologies has been studied for a long time by the Semantic Web community [13]. To tackle this problem, different frameworks have been introduced: Text2Onto [4] for learning ontologies and handling imprecise and uncertain data, BayesOWL [7] based on Bayesian Networks for ontologies mapping. In [6], the authors propose a probabilistic extension for OWL with a Bayesian Network layer for reasoning. Actually, fuzzy OWL [2, 20] was proposed to manage, in addition to uncertainty, some other text imperfection (such as imprecision and vagueness) with the help of fuzzy logics. Moreover, W3C Uncertainty Reasoning for the World Wide Web Incubator Group (URW3-XG) [12] describes an ontology to annotate uncertain and imprecise data. This ontology focuses on the representation of the nature, the model, the type and the derivation of uncertainty. This representation is really interesting but unfortunately does not show how to link the uncertainty to the concerned knowledge described in the text.

However, in all these works, the uncertainty was considered as a metadata. The ontologies which handle uncertainty are proposed to either create a fuzzy knowledge base (fuzzy ABox, fuzzy TBox, fuzzy Rbox) or to associate each class of the ontology to a super class which denotes the uncertain or fuzzy concept. To each axiom is associated a truth degree in [0,1]. Therefore, the user is required to handle two knowledge bases in parallel. The first one is dedicated to certain knowledge whereas the second is dedicated to uncertain knowledge. This representation could induce some inconsistencies between the knowledge bases. From a querying perspective this representation is also not appealing since it forces the user to query both bases and then combine the results. In order to avoid these drawbacks, we propose in this paper, a solution to integrate uncertain knowledge to the rest of the extraction. The idea is to ensure that all extracted knowledge, either be it certain or uncertain, is managed within the same knowledge base. This approach aims at ensuring the consistency of the extracted knowledge and eases its querying.

Moreover, it is worth noting that linguistic processing carried out on uncertainty management notably, Saurì[19] and Rubin [17][18] works, they payed attention to different modalities and polarity to characterize uncertainty/certainty.

The first one [19], considers two dimensions. Each event is associated to a factual value represented as a tuple $< mod, pol >$ where *mod* denotes modality and distinguishes among: *certain, probable, possible* and *unknown*, *pol* denotes polarity values which are *positive, negative* and *unknown.*

In [17][18] four dimensions have been considered:

- *certainty level:* absolute, high, moderate or low.
- *author perspective:* if it is his/her point of view or a reported speech.
- *focus:* if it is an abstract information (opinion, belief, judgment...) or a factual one (event, state, fact...).
- *time:* past, present, future.

This model is more complete even if it does not handle negation. However, the authors do not explain how to combine all these dimensions to get a final interpretation to a given uncertainty. In this paper, we explain how to detect uncertainty in textual document and how to quantify it to get a global interpretation.

3 Uncertainty detection in the text

The Web contains a huge number of documents from heterogeneous sources like forums, blogs, tweets, newspaper or Wikipedia articles. However, these documents cannot be exploited directly by programs because they are mainly intended for humans. Before the emergence of the Semantic Web, only human beings could access the necessary background knowledge to interpret these documents. In order to get a full interpretation of the text content, it is necessary to consider the different aspects of the given information. Some piece of information can be considered as “perfect” only if it contains precise and certain data. This is rarely the case even for a human reader with some context knowledge. Indeed, the reliability of the data available on the Web often needs to be reconsidered, uncertainty, inconsistency, vagueness, ambiguity, imprecision, incompleteness and others are recurrent problems encountered in data mining. According to [9] the information can be classified into two categories : *subjective* and *objective*. An information is objective or quantitative if it indicates an observable, *i.e.*, something which is able to be counted for example. The other category is the subjective (qualitative) information. It can describe the opinion of the author, he may express his own belief, judgment, assumption, etc. Therefore, the second one is subject to contain imperfect data. Then, it becomes necessary to incorporate these imperfections within the representation of the extracted information.

In this paper, we are interested in the uncertainty aspect. In domains such as information theory, knowledge extraction and information retrieval, the term uncertainty refers to the concept of being unsure about something or someone. It denotes a lack of conviction. Uncertainty is a well studied form of data imperfection, but it is rarely considered at the knowledge level during extraction processing. Our approach consists in considering the life cycle of the knowledge from the data acquisition to the final RDF representation steps, *i.e.*, generating and persisting the knowledge as triples.

Evaluating uncertainties in text

As previously explained, the text may contain several imperfections which can affect the trustworthiness of an extracted action or event. So, during the linguistic processing, we need to pay attention to the modalities of the verb which indicate how the action or the event had happened, or how it will. Actually, the text provides information about the epistemic stance of the author, that he often commits according to his knowledge, singular observation or beliefs [16]. Moreover, natural languages offer several ways to express uncertainty, usually expressed using linguistic qualifiers. According to [14, 8, 1] uncertainty qualifiers can be classified as follows:

- verbal phrases *e.g.*, *as likely as, chances are, close to certain, likely, few, high probability, it could be, it seems, quite possible.*
- expression of uncertainty with quantification *all, most, many, some, etc.*,
- modal verbs *e.g.*, *can, may, should.*
- adverbs, *e.g.*, *roughly, somewhat, mostly, essentially, especially, exceptionally, often, almost, practically, actually, really.*
- speculation verbs *e.g.*, *suggest, suppose, suspect, presume.*
- nouns *e.g.*, *speculation, doubt, proposals.*
- expressions *e.g.*, *raise the question of, to the best of our knowledge, as far as I know.*

All these markers help to detect and identify the uncertainty with different intensities. This helps in evaluating the confidence degree associated to the given information. For example : *it* ***may*** *happen* is less certain that *it will* ***probably*** *happen.* It is also necessary to consider modifiers such as *less, more, very.* Depending on the polarity of each modifier we add or subtract a predefined real number α, set to 0.15 in our experiment, to the given marker's degree. We base our approach on a natural language processing. This processing indicates syntactic and semantic dependencies between words. From these dependencies we can identify the scope of each identifier in the text. Once these qualifiers are identified, the uncertainty of the knowledge can be specified and then quantified. By quantifying, we mean attributing a confidence degree which indicates how much we can trust the described entity. To this end, we associate to each marker a probabilistic degree. We defined three levels of certainty: (i) high=0.75, (ii) moderate=0.50, (iii) low=0.25. Moreover, we also base this uncertainty quantification on previous works in this field such as [3, 11] which define a mapping between the confidence degree and each uncertainty marker. This mapping is called Kent's Chart and Table 1 provides an extract of it.

Table 1. Table of Kent's Chart for expressions of degrees of uncertainty

Expression	Probability Degree
certain	100
almost certain, believe, evident, little doubt	85-99
fairly certain, likely, should be, appear to be	60-84
have chances	40-59
probably not, fairly uncertain, is not expected	15-39
not believe, doubtful, not evident	1-14

However, uncertainty markers are not the only way to generate uncertainty. Reported speech and future timeline are also considered as uncertainty sources.

These will be taken into account when the final uncertainty weight will be calculated. We notice that the trust of the reported speech depends of different parameters which affect the trust granted to its content:

- *the author of the declaration*: if the author name is cited, if the author has an official role (prosecutor, president...).
- *the nature of the declaration*: if it is an official declaration, a personal opinion, a rumor...

Example 2. A crook who burglarized homes and crashed a stolen car remains on the loose, but he ***probably*** left Old Saybrook by now, ***police said*** Thursday.

In Example 2, we can identify two forms of uncertainty. First, the author explicitly expresses, using the term *(probably)*, an uncertainty about the fact that the crook left the city. The second one is related to the reported speech which comes from the police and is not assumed to be a known fact.
Therefore, for a given information described in the text, many sources of uncertainty can occur, then, it is necessary to combine all these uncertainties in order to get a final confidence degree to be attributed to the extracted information. With regard to this issue, we chose a Bayesian approach to combine all uncertainties to the concerned information. Indeed Bayesian network are well suited to our knowledge graph which is a directed acyclic graph. This choice is also motivated by the dependency that exists between children of uncertainty nodes. Indeed, to calculate the final degree of uncertain information, we need to consider its parents, if they contain uncertainty, then the conditional probabitlity related to this parent is reverberated on the child.

4 RDF representation of uncertainty

In order to extract complete and relevant knowledge, we consider the uncertainty as an integral part of the knowledge instead of integrating it as an annotation. Usually, uncertainty is added as assertions to triples (the uncertainty assigned to each extracted knowledge). So, we represent it with some reification as recommended by [5]. Nevertheless, we encountered some difficulties to represent uncertainty on triples' predicates, as opposed to the whole triple. In the second sentence of Example 1, the uncertainty does not concern the whole moving but only its date. Only one part of the event is uncertain and the RDF representation has to take this into account. In fact, we cannot indicate using reification which part of the triple is uncertain, as shown in Figure 1, with reification, we give the same representation to both sentences in Example 1 even if they express different information. Indeed, reified statements cannot be used in semantic inferences, and are not asserted as part of the underlying knowledge base [21]. The reified statement and the triple itself are considered as different statements. So, due to its particular syntax (rdf:Statement) the reified triple can hardly be related to other triples in the knowledge base [15]. Moreover, using blank node to identify the uncertain statement prevents from obtaining good performance [10]. Indeed, writing queries over RDF data sets involving reification becomes inconvenient. Especially, for one to refer to a reified triple, we need to use four additional triples linked by a blank node.

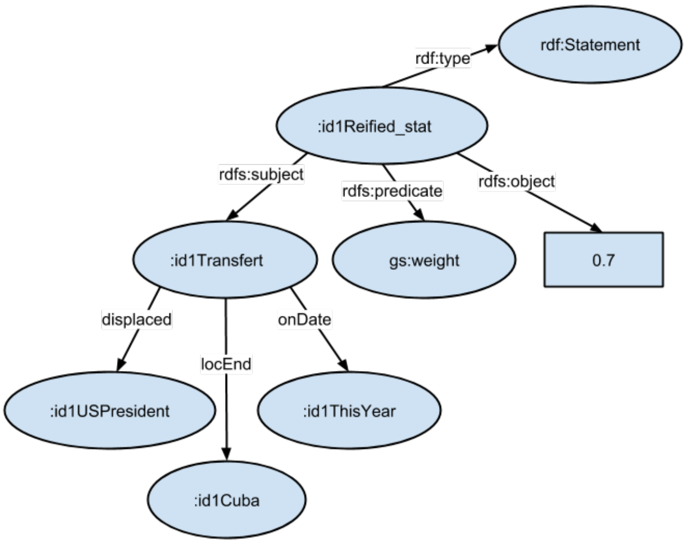


Fig. 1. RDF representation of uncertainty using reification

To deal with previous issues, we propose the `UncertaintyOntology` ontology which contains a concept (`Uncertainty`), a datatype property (***weight*** which have `Uncertainty` as its domain and real values as range) and object properties (***isUncertain*** and ***hasUncertainProp*** which respectively denote an uncertain individual (`Uncertainty` as domain and `owl:Thing` as range) and an uncertain property of a given individual (`Uncertainty` as Domain and `owl:Thing` as Range). This ontology can easily be integrated with our `geol` ontology or with any other ontology requiring some support for uncertainty.

This ontology (*UncertaintyOntology*) handles uncertainty occurring on each level of the triple. If the uncertainty concerns the resource, which denotes a subject or an object triple, so the property *isUncertain* is employed. If the triple's predicate is uncertain then we use *hasUncertainProp* to indicate the uncertainty. UncertainOntology is domain independent, it can be added to any other ontology since we assume that uncertainty occurs on each part of the sentence in a text.

To illustrate this representation, we provide in Figure 2, the RDF representation of Example 1's sentences. In the first sentence (on the left side), the uncertainty concerns the following triples :
:id1Transfer, displaced, :id1USPresident.
:id1Transfer, locEnd, :id1Cuba.
:id1Transfer, onDate, :id1ThisYear.

As we based on Bayesian approach, all these triples have an uncertainty of 0.7, expressed using the uncertainty marker *probably*.
Whereas, in the second sentence, the uncertainty concerns only the property onDate, so, the triple *:id1Transfer, onDate, :id1ThisYear.* is uncertain.

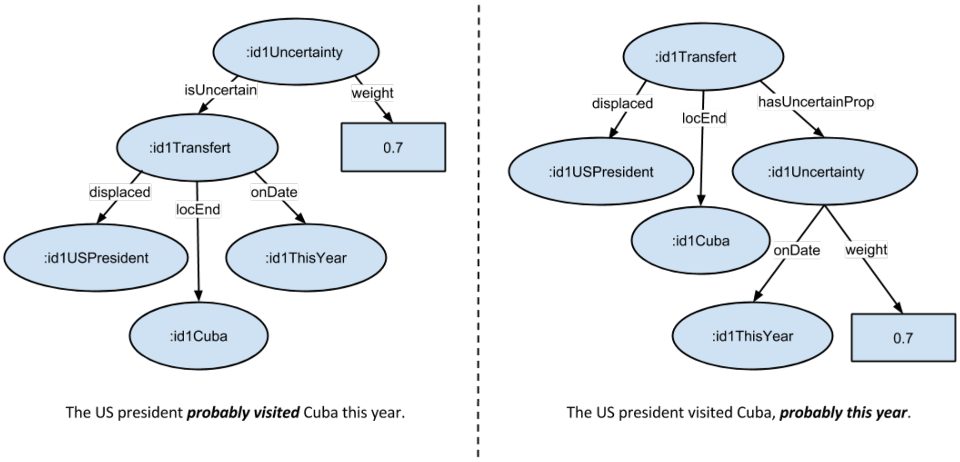


Fig. 2. RDF Knowledge representation of uncertainty in Example 1

Finally, we conclude that using this RDF representation, we identify three different cases of triple uncertainty. Figure 4 shows the representation of different patterns of uncertainty in RDF triples. Pattern 1 describes uncertainty on the object of the triple. Pattern 2 describes uncertainty on the subject and finally, pattern 3, uncertainty on the property.

This representation of uncertainty is more compact than reification and improves user understanding regarding the RDF graph.

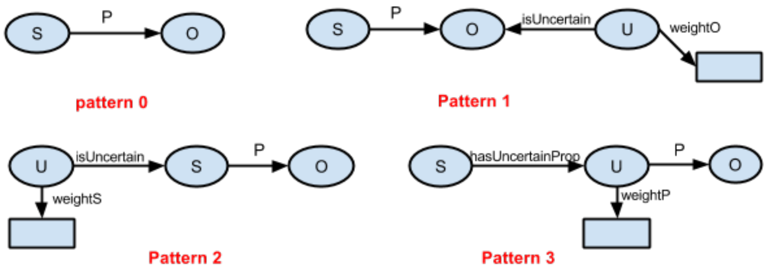


Fig. 3. RDF representation of Uncertainty patterns.

5 SPARQL Querying with uncertainty

The goal of our system is to enable end-users to query the extracted information. These queries take into account the presence of uncertainties by going through a rewriting. Our system discovers if such a rewriting is necessary by executing the following queries. First, we list all uncertain properties, using the query in Listing 1.1. The result is a set of triples (s,p,o) where *p* is an uncertain property.

```
PREFIX gs:<http://www.geolsemantics.com/onto#>
Select ?s ?prop ?o
Where {
        ?s gs:hasUncertainProp ?u.
        ?u gs:weight ?weight.
        ?u ?prop ?o.
}
```

Listing 1.1. SPARQL query Select uncertain properties

Then, we check if the predicates of each triple in the entry query appear in the result set. If so, we rewrite the query by adding the uncertainty on the given predicate using the pattern query in Listing1.2. Finally, we inspect the query result set of the rewritten query, in order to check if an uncertainty occurs on each resource (subject and/or object) extracted.

```
PREFIX gs:<http://www.geolsemantics.com/onto#>
Select ?p ?weight
Where {...
        ?u gs:isUncertain ?p.
        ?u gs:weight ?weight.
...}
```

Listing 1.2. SPARQL query Select uncertain resources

Furthermore, if a user wants to know the list of uncertainties in a given text, the query in Listing 1.3 is used to extract all uncertain data explicitly expressed. We consider that each linguistic extraction is represented according to the schema presented in Section 4. Our goal is now to provide a query interface to the end-user and to qualify the uncertainty associated to each query answer. Of course, the uncertain values that we are associating with the different distinguished variables of a query are directly emerging from the ones we are representing in our graph and which has been described in Section 4. Our system accepts any SPARQL 1.0 queries from the end-user. For testing reasons, we also have defined a set of relevant predefined queries, *e.g.*, the query in Example 3.

```
PREFIX gs:<http://www.geolsemantics.com/onto#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX v:<http://www.w3.org/2006/vcard/ns#>
SELECT distinct ?concept_uncertain ?obj ?weight
WHERE {
{       ?u a gs:Uncertainty.
        ?u gs:isUncertain ?concept_uncertain.
        ?u gs:weight ?weight
}UNION {
        ?u2 a gs:Uncertainty.
        ?u2 gs:weight ?weight.
        ?s ?hasUncertainProp ?u2.
        ?u2 ?prop ?obj.}
}
```

Listing 1.3. SPARQL query : Select all uncertainties in the text

Example 3. Let us consider the query in Listing 1.4.

```
PREFIX gs:<http://www.geolsemantics.com/onto#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX v:<http://www.w3.org/2006/vcard/ns#>
Select ?date
Where {
        ?t gs:displaced ?p.
        ?p gs:role "president".
        ?t gs:locEnd ?l.
        ?l v:location-name "Cuba".
        ?t gs:onDate ?date.
}
```

Listing 1.4. SPARQL query : When did the president go to Cuba?

In order to make query submission easier for the end-user, we do not impose the definition of the triple patterns associated to uncertainty handling. Hence, the end-user just submits a SPARQL query without caring where the uncertainties are. Considering query processing, this implies to reformulate the query before its execution, *i.e.*, to complete the query such that its basic graph pattern is satisfiable in the face of triples using elements of our uncertain ontology.

We can easily understand that a naive reformulation implies a combinatorial explosion. This has direct impact on the efficiency of the query result set computation. This can be prevented by rapidly identifying the triple patterns of a query that are subject to some uncertainty. In fact, since our graphs can only represent uncertainty using one of the three patterns presented in Figure 4, we

```
PREFIX gs:<http://www.geolsemantics.com/onto#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX v:<http://www.w3.org/2006/vcard/ns#>
Select ?date ?w
Where {{
        ?t gs:displaced ?p.
        ?p gs:role "president".
        ?t gs:locEnd ?l.
        ?l v:location-name "Cuba".
        ?t gs:onDate ?date.
}UNION{
        ?t gs:displaced ?p.
        ?p gs:role "president".
        ?t gs:locEnd ?l.
        ?l v:location-name "Cuba".
        ?t gs;hasUncertainProp ?u.
        ?u gs:onDate ?date.
        ?u gs:weight ?w.
}}
```

Listing 1.5. Uncertainty query : When did the president go to Cuba?

can go through a pre-processing step that indexes these triples. To do so, we use a set of SPARQL queries (see Listing 1.1 and 1.2 which respectively retrieve the properties and subject with their weights). These values are stored in hash tables for fast access.

Therefore, Listing 1.5 corresponds to the rewriting of Listing 1.4. We introduced the uncertainty option and obtained the following results :

Sentence	Result	Uncertainty	Uncertainty Detail
(1)	?date = “20150101-20151231”	0.7	On the subject
(2)	?date = “20150101-20151231”	0.7	On the predicate

6 Conclusion and Perspectives

In this article, we addressed the quantification and qualification of uncertain and ambiguous information extracted from textual documents. Our approach is based on a collaboration between Natural Language Processing and Semantic Web technologies. The output of our different processing units takes the form of a compact RDF graph which can be queried with SPARQL queries and reasoned over using ontology based inferences. However, some issues are still unresolved, even for the linguistic community, such as: distinguish between deontic and epistemic meaning. Example: “He can practice sport.” One can interpret this information as a permission and an other as an ability or a certainty.

This work mainly concerns the uncertainty expressed in the text, for future work we intend to consider the trust guaranteed to the source of the text. Indeed, the source can influence the trustworthiness and the reliability of the declared information. Moreover, we plan to consider additional aspects of the information, such as polarity.

References

1. A. Auger and J. Roy. Expression of uncertainty in linguistic data. In *Information Fusion, 2008 11th International Conference on*, pages 1–8. IEEE, 2008.
2. F. Bobillo and U. Straccia. Fuzzy ontology representation using OWL 2. *International Journal of Approximate Reasoning*, 52(7):1073–1094, 2011.
3. T. A. Brown and E. H. Shuford. Quantifying uncertainty into numerical probabilities for the reporting of intelligence. Technical report, 1973.
4. P. Cimiano and J. Vlker. Text2Onto. In *Natural Language Processing and Information Systems*, volume 3513, pages 257–271. Springer Berlin, 2005.
5. W. W. W. Consortium et al. Rdf 1.1 semantics. 2014.
6. Z. Ding and Y. Peng. A probabilistic extension to ontology language OWL. In *System Sciences, 2004. Proceedings of the 37th Annual Hawaii international conference on*, pages 10–pp. IEEE, 2004.
7. Z. Ding, Y. Peng, and R. Pan. BayesOWL: Uncertainty modeling in semantic web ontologies. In *Soft Computing in Ontologies and Semantic Web*, pages 3–29. Springer, 2006.
8. M. J. Druzdzel. Verbal uncertainty expressions: Literature review. 1989.
9. D. Dubois and H. Prade. Formal representations of uncertainty. *Decision-Making Process: Concepts and Methods*, pages 85–156, 2009.
10. O. Hartig and B. Thompson. Foundations of an alternative approach to reification in RDF. *arXiv preprint arXiv:1406.3399*, 2014.
11. E. M. Johnson. Numerical encoding of qualitative expressions of uncertainty. Technical report, DTIC Document, 1973.
12. K. J. Laskey and K. B. Laskey. Uncertainty reasoning for the World Wide Web: Report on the URW3-XG incubator group. In *URSW*. Citeseer, 2008.
13. T. Lukasiewicz and U. Straccia. Managing uncertainty and vagueness in Description Logics for the Semantic Web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(4):291–308, 2008.
14. E. Marshman. Expressions of uncertainty in candidate knowledge-rich contexts: A comparison in english and french specialized texts. *Terminology*, 14(1):124–151, 2008.
15. B. McBride. Jena: Implementing the RDF model and syntax specification. In *SemWeb*, 2001.
16. A. Papafragou. Epistemic modality and truth conditions. *Lingua*, 116(10):1688–1702, 2006.
17. V. L. Rubin, N. Kando, and E. D. Liddy. Certainty categorization model. In *AAAI spring symposium: Exploring attitude and affect in text: Theories and applications, Stanford, CA*, 2004.
18. V. L. Rubin, E. D. Liddy, and N. Kando. Certainty identification in texts: Categorization model and manual tagging results. In *Computing attitude and affect in text: Theory and applications*, pages 61–76. Springer, 2006.
19. R. Saurı and J. Pustejovsky. From structure to interpretation: A double-layered annotation for event factuality. In *The Workshop Programme*, 2008.

20. G. Stoilos, G. B. Stamou, V. Tzouvaras, J. Z. Pan, and I. Horrocks. Fuzzy OWL: Uncertainty and the semantic web. In *OWLED*, 2005.
21. E. R. Watkins and D. A. Nicole. Named graphs as a mechanism for reasoning about provenance. In *Frontiers of WWW Research and Development-APWeb 2006*, pages 943–948. Springer, 2006.

PR-OWL 2 RL - A Language for Scalable Uncertainty Reasoning on the Semantic Web

Laécio L. dos Santos[1], Rommel N. Carvalho[1,2], Marcelo Ladeira[1], Li Weigang[1], and Gilson L. Mendes[2]

[1] Department of Computer Science, University of Brasília (UnB)
Brasília, DF, Brazil
{laecio,mladeira,weigang}@cic.unb.br
[2] Department of Research and Strategic Information, Brazilian Office of the Comptroller General (CGU), Brasília, DF, Brazil
{rommel.carvalho,liborio}@cgu.gov.br

Abstract. Probabilistic OWL (PR-OWL) improves the Web Ontology Language (OWL) with the ability to treat uncertainty using Multi-Entity Bayesian Networks (MEBN). PR-OWL 2 presents a better integration with OWL and its underlying logic, allowing the creation of ontologies with probabilistic and deterministic parts. However, there are scalability problems since PR-OWL 2 is built upon OWL 2 DL which is a version of OWL based on description logic SROIQ(D) and with high complexity. To address this issue, this paper proposes PR-OWL 2 RL, a scalable version of PR-OWL based on OWL 2 RL profile and triplestores (databases based on RDF triples). OWL 2 RL allows reasoning in polynomial time for the main reasoning tasks. This paper also presents First-Order expressions accepted by this new language and analyzes its expressive power. A comparison with the previous language presents which kinds of problems are more suitable for each version of PR-OWL.

1 Introduction

Web Ontology Language (OWL) is the main language in the Semantic Web for creating ontologies. It lacks the capacity for treating uncertainty, limiting its application in several kinds of domains. Various approaches have been proposed to solve this issue using different formalisms, such as Bayesian networks, fuzzy logic, and Dempster-Shaffer theory. One of these approaches, Probabilistic OWL (PR-OWL) [7] adds uncertainty treatment capacity to OWL using Multi-Entity Bayesian Networks (MEBN) [11], which is a very expressive First-Order Probabilistic Logic. PR-OWL has been implemented in UnBBayes [3], which is an open source framework for probabilistic graphical models. PR-OWL 2 [4] extends the previous language adding a tight and better integration between OWL existing concepts and properties and PR-OWL new ones. A PR-OWL 2 implementation also was developed in UnBBayes, using Protégé [4] and its HermiT [13] default

[3] http://unbbayes.sourceforge.net/
[4] http://protege.stanford.edu/

OWL DL reasoner for modeling and reasoning with the deterministic part of the ontology.

PR-OWL 2 implementation, however, has some scalability problems due to the time complexity of OWL 2 DL reasoners to solve complex expressions. This hinders the ability to work with domains that have large assertive databases. One example is the public procurement fraud detection domain developed in Brazil, for which a probabilistic ontology was created using PR-OWL 2 [5]. Although the probabilistic ontology has been successfully tested with simple cases, in a real situation, using government databases, millions of triples will be needed, making the reasoning intractable with PR-OWL 2 and its current implementation.

The solution proposed for overcoming this limitation is to use triplestores together with the OWL 2 RL profile to create a new version of PR-OWL 2, named PR-OWL 2 RL. The OWL 2 RL [17] profile allows implementations with reasoning in polynomial time for the main reasoning tasks in systems based on rules. The reasoning is mainly processed by materialization, where the rule set is evaluated when new statements are included in the base, as well as the new knowledge that is derived by them.

The proposal of this new language requires: 1) to review the PR-OWL language according the OWL 2 RL syntax restrictions; 2) a new algorithm to evaluate the MEBN first-order formulas using triplestores; and 3) to design a scalable algorithm for generating Situation Specific Bayesian Networks (SSBN). This paper discusses the first two issues.

This paper is organized as follows. Section 2 describes some relevant concepts for the understanding of this work: MEBN, OWL and PR-OWL. Section 3 presents PR-OWL 2 bottlenecks that motivated this work. Section 4 introduces the language proposed and shows how the first-order formulas can be evaluated using the SPARQL language. Finally, Section 5 presents some conclusions and possible future work.

2 Fundamentals

This section presents some concepts necessary for the understanding of this paper. Section 2.1 presents Multi-Entity Bayesian Networks, the formalism used by PR-OWL to deal with uncertainty in the OWL language. Section 2.2 presents the OWL language and its versions, including the OWL 2 RL profile, and triplestores. Section 2.3 presents PR-OWL, its extension, PR-OWL 2, and its implementation in the UnBBayes Framework.

2.1 Multi-Entity Bayesian Networks

Multi-Entity Bayesian Networks (MEBN) is a formalism for representing first-order probabilistic knowledge bases [11]. MEBN joins Bayesian networks with First-Order Logic (FOL), augmenting the expressive power of the first by allowing uncertainty representation and reasoning in situations where the quantity of random variables is unknown.

MEBN models the domain using a MEBN Theory (MTheory), which is composed of random variables that together have a unique joint probability distribution. The knowledge is divided into MEBN Fragments (MFrags). Each MFrag is composed by resident nodes, input nodes, and context nodes. Resident nodes are random variables for which the Local Probability Distribution (LPD) is defined in the MFrag where they are. Input nodes are references to resident nodes defined in a different MFrag. Context nodes contain restrictions that need to be satisfied in order to correctly instantiate the corresponding MFrag. The nodes represent entity attributes and relationships between entities. Each node is parameterized with ordinary variables (OV), placeholders filled with entity instances available in the knowledge base during the instantiation of the model.

Figure 1 shows the MFrag `Front Of Enterprise` of the Procurement Fraud ontology [5]. This probabilistic ontology was designed to identify frauds in public procurements in Brazil using the data available in the Brazilian Office of the Comptroller General (CGU). In this MFrag, the resident node `isFrontFor` (node 9) refers to the probability that a person is a front for an enterprise. It is influenced by the input nodes `hasValue`, `hasAnnualIncome`, and `hasEducationLevel` (nodes 6–8). The context nodes (nodes 1–5) show which restrictions need to be satisfied in order to instantiate this MFrag. Nodes 4 and 5, for example, say that the procurement has to be finished and the person of interest has to be responsible for the enterprise that won the procurement.

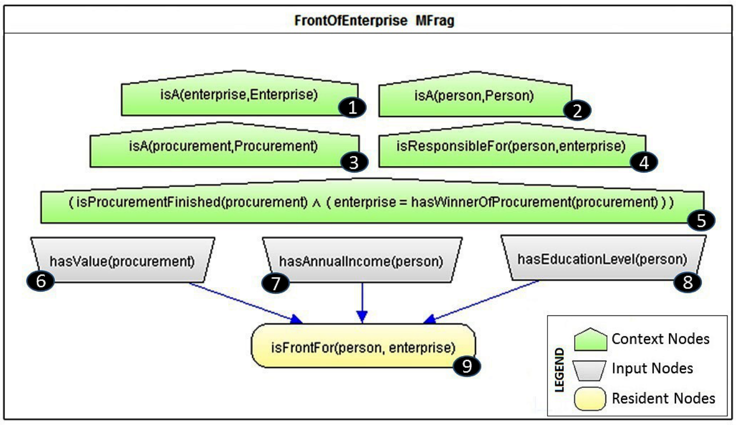

FrontOfEnterprise MFrag
isA(enterprise,Enterprise)
1
isA(person,Person)
2
isA(procurement,Procurement)
3
isResponsibleFor(person,enterprise)
4
(isProcurementFinished(procurement) ∧ (enterprise = hasWinnerOfProcurement(procurement)))
5
hasValue(procurement)
6
hasAnnualIncome(person)
7
hasEducationLevel(person)
8
isFrontFor(person, enterprise)
9
LEGEND
Context Nodes
Input Nodes
Resident Nodes

Fig. 1. MFrag `Front Of Enterprise` for the Procurement Fraud domain

An MTheory works like a template, which is instantiated giving the query nodes and the evidence to build a Situation-Specific Bayesian Network (SSBN), a Bayesian Network with all nodes computationally relevant to answer the queries. Laskey presents in [11] an algorithm for generating a SSBN that expands the network from both the queries and findings in order to build a grand BN which is pruned by removing barren, nuisance, and d-separated nodes.

2.2 OWL

OWL is the main Semantic Web language for building ontologies. OWL 2, its current version, became a W3C recommendation in 2009 [16].

The direct model-theoretic semantics of OWL 2 is called Direct Semantics, which is strongly related to the semantics of description logics [16]. The Direct Semantics assigns meaning directly to ontology structures, in a way compatible with the semantics of the SROIQ description logic [16]. Description Logics are subsets of FOL that model the domain based on its classes and properties. It represents the knowledge by first defining the relevant concepts of the domain (TBox), and then using these concepts to specify properties of objects and individuals occurring in the domain (ABox) [1]. Different description logics have been created, trying to get a favorable trade-off between expressiveness and complexity. OWL 2 DL is based on $\mathcal{SROIQ}(\mathrm{D})$. Several reasoners based on tableau algorithms were created for OWL 2 DL. Some examples are HermiT [13], Pellet [12] and FaCT++ [15].

OWL 2 has three different profiles (syntactic subsets): OWL 2 EL, OWL 2 QL, and OWL 2 RL. All of them are more restrictive than OWL 2 DL and trades off OWL 2's expressive power for computational or implementation benefits. In these profiles, most types of reasoning can be made in polynomial time. OWL 2 EL is suitable for ontologies with a very large but simple TBox. OWL 2 QL is suitable to work with conjunctive queries, permitting the use of ontological reasoning in systems like relational databases through a query rewriting approach. OWL 2 RL, based on Datalog and in R-entailment [14], is suitable to allow an easy implementation in systems based on rules.

W3C proposes a set of rules called OWL 2 RL/RDF that implements the OWL 2 RL profile. This set of rules is based on RDF Semantic, where the knowledge is organized in graphs, composed by RDF triples. Each RDF triple is composed by a subject linked to an object by a property. The reasoning is made through rules, where given a set of specific triples and a rule, we can get another expression that follows logically from the rule. The Theorem PR1 [17] states some conditions that guarantee that the ontology O_2 entailed from O_1 under the Direct Semantics is the same entailed under the first-order axiomatization of RDF semantics using the OWL 2 RL/RDF rules:

- neither O_1 nor O_2 contains an IRI (International Resource Identifier) that is used for more than one type of entity;
- O_1 does not contain the following axioms:
  - `SubAnnotationPropertyOf`,
  - `AnnotationPropertyDomain`,
  - `AnnotationPropertyRange`; and
- each axiom in O_2 is an assertion of the form as specified below, for a_1, a_2, . . . , a_n a named individual:
  - `ClassAssertion`(C a) where C is a class,
  - `ObjectPropertyAssertion`(OP a_1 a_2) where OP is an object property,
  - `DataPropertyAssertion`(DP a v) where DP is a data property, or

- `SameIndividual`($a_1 \dots a_n$).

The OWL 2 RL profile is implemented by some triplestores. Triplestores are databases that organize the knowledge in graphs composed by RDF triples. They are becoming very useful and there are a lot of commercial (*e.g.*, GraphDB, Oracle Spatial and Graph, and AllegroGraph) and free implementations (*e.g.*, Sesame). They normally implement the RDF/RDFS entailment rules, using materialization for expanding the rules when new declarations are added to the base. SPARQL is the main language used for querying RDF databases. It is very similar to SQL, acting over generalized RDF graphs. Most triplestores accept, in addiction to RDFS, inference with some constructions of OWL. Implementations of the OWL 2 RL profile are common.

2.3 PR-OWL

Probabilistic OWL (PR-OWL) is an extension of the OWL language that permits the creation of probabilistic ontologies [7]. It works as an upper-ontology, consisting of a set of classes, subclasses, and properties that allow modeling the uncertainty part of the ontology using MEBN. Figure 2 shows the main concepts involved in a PR-OWL ontology. The probabilistic ontology is modeled using the MTheory class, composed by a set of MFrags. These MFrags must collectively form a consistent MTheory. The MFrags are built from random variables, which have a probabilistic distribution and an exhaustive set of possible states.

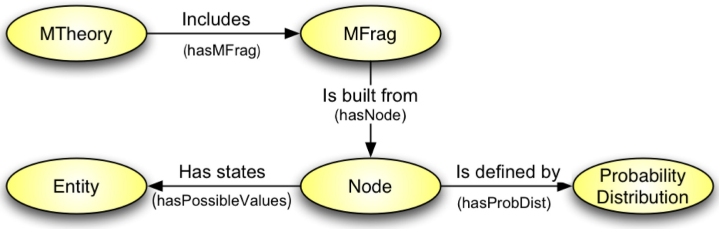


Fig. 2. PR-OWL Main Concepts

UnBBayes has an implementation of PR-OWL and MEBN [3, 6] that allows the design of an MTheory using a graphical user interface (GUI). A pseudocode can be used for defining the LPDs. The MTheory is stored in a knowledge base, supported by the PowerLoom Knowledge Representation and Reasoning (KR&R) System[5]. The algorithm for generating SSBNs is based on the one proposed in [11].

PR-OWL 2 [4] extends PR-OWL by having a better built-in integration between OWL and MEBN. The main concept used for this is the property

[5] http://www.isi.edu/isd/LOOM/PowerLoom/

`definesUncertaintyOf` that links a random variable to an OWL property. The additional properties `isObjectIn` and `isSubObjectIn` allow the mapping of both domain and range of the OWL property to its corresponding concept in MEBN. PR-OWL 2 also has other improvements, like the support to polymorphism and the use of OWL datatypes. A plug-in for PR-OWL 2 was developed in UnBBayes, using Protégé for modeling the deterministic parts of the ontology. Protégé is a popular open source framework for editing ontologies. Moreover, HermiT is used for evaluating the context nodes and for getting information about findings.

3 Description of the Problem

PR-OWL 2 and its implementation in UnBBayes have some scalability and expressibility problems. OWL 2 DL, which is used in PR-OWL 2 definition/implementation, is based on description logic $\mathcal{SROIQ}$(D) that has complexity N2EXPTIME-complete [10] for the main reasoning problems: ontology consistency, class expression satisfiability, class expression subsumption, and instance checking. This class of complexity comprises the problems solvable by nondeterministic algorithm in time at most double exponential in the size of the input [17]. OWL 2 DL reasoners are normally based on tableau algorithms. Donini [8] states two different sources of complexity in tableau calculi: the AND-Branching, responsible for the exponential size of a single candidate model, and the OR-Branching, responsible for the exponential number of different candidate models. This exponential complexity of OWL 2 DL reasoners makes the queries more time/space consuming, the larger/more complex the knowledge base is. Thus, making it inviable for several cases. Furthermore, most OWL reasoners are limited to the available memory of the computational resource used, since the database needs to be loaded into memory to allow inference. This clearly does not scale to real and large databases.

We also have scalability problems because of the use of Protégé's GUI and API in UnBBayes' PR-OWL 2 implementation. We made tests using LUBM ontologies to verify the performance of this implementation. LUBM (Lehigh University Benchmark) [9] is a very popular benchmark for reasoners and triplestores. Using an i5 machine with 3GB of memory dedicated to run Protégé, we could not load nor initialize the reasoner with the LUBM 100, an assertive base containing 2,779,262 instances of classes and 11,096,694 instances of properties. We used the HermiT reasoner, where the initialization consists of building the class hierarchy, classifying object and data properties, computing instances of all classes and object properties, and calculating same as individual. This initialization is necessary to solve the queries. LUBM 100 base has 1,06 GB when stored in an OWL file in XML format, making it clear that the structure used by Protégé adds a great overhead to PR-OWL 2 implementation.

This scalability problems limit the use of PR-OWL in domains with large assertive bases. In the domain of procurement fraud detection [5], for example, the assertive base can easily have millions of assertions. This makes it unsuitable to using an OWL reasoner for the necessary deterministic reasoning, which comprises of evaluating the FOL expressions and searching for findings.

Since PR-OWL 2 is written in OWL 2 DL, one possibility is to use an OWL 2 DL reasoner for solving the FOL formulas during MEBN reasoning. PR-OWL 2 current implementation in UnBBayes does that.

Evaluating MEBN FOL formulas using an OWL DL reasoner requires some workarounds. The Table 1 presents the formulas that are allowed in the current implementation of UnBBayes, where `ov` are ordinary variables, `CONST` are constants, and `booleanRV` are Boolean random variables. Expressions with connectives and quantifiers are not allowed in this version.

Table 1. Types of context node formulas accepted in the PR-OWL 2 implementation

Formula	Negation
ov$_1$ = ov$_2$	NOT (ov$_1$ = ov$_2$)
booleanRV(ov$_1$ [, ov$_2$, . . .])	NOT booleanRV(ov$_1$ [, ov$_2$, . . .])
ov$_0$ = nonBooleanRV(ov$_1$)	NOT (ov$_0$ = nonBooleanRV(ov$_1$))
ov$_0$ = nonBooleanRV(ov$_1$ [, ov$_2$, . . .])	
CONST = nonBooleanRV(ov$_1$ [, ov$_2$, . . .])	
nonBooleanRV(ov$_1$ [, ov$_2$, . . .]) = CONST	
nonBooleanRV(ov$_1$) = ov$_0$	NOT (nonBooleanRV (ov $_1$) = ov$_0$)
nonBooleanRV(ov$_1$ [, ov$_2$, . . .]) = ov$_0$	

4 PR-OWL based on OWL 2 RL profile

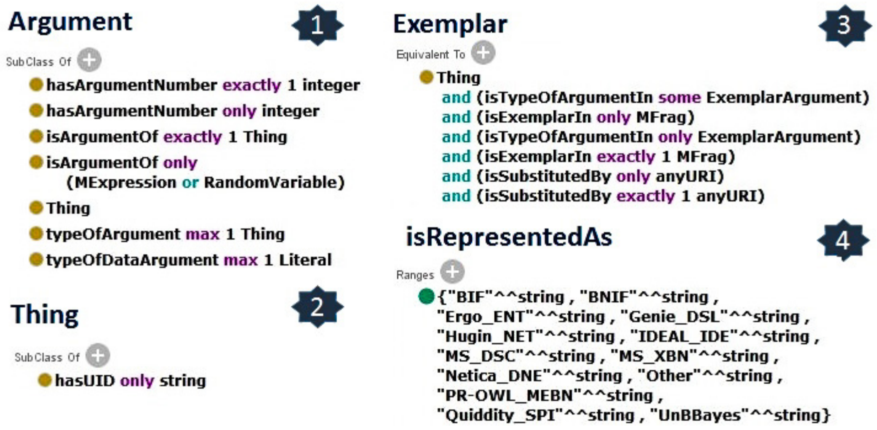
In order to overcome the limitations presented, we propose PR-OWL 2 RL, a more scalable version of PR-OWL based in the OWL 2 RL profile. The purpose is to use an RDF triplestore database for both the storage and reasoning with very large ontologies represented as RDF triples. This is possible because OWL 2 RL allows reasoning in polynomial time for the main reasoning tasks. SPARQL is the common query language used with RDF triples.

Since PR-OWL 2 is written in OWL 2 DL, some adjustments are necessary to adapt it for OWL 2 RL. This is due to the fact that OWL 2 RL imposes several syntactic restrictions on the OWL expressions. Running a validator developed by the Manchester University [6] we found the following unsupported features:

1. Use of non-superclass expression where superclass expression is required;
2. Use of non-subclass expression where subclass expression is required;
3. Use of non-equivalent-class expression where equivalent-class expression is required; and
4. Use of unsupported data range.

[6] http://mowl-power.cs.man.ac.uk:8080/validator/

Figure 3 shows examples for each kind of unsupported feature. The first case occurs for several reasons, such as the use of existential quantifier and disjunction on the right side of a `subClass` expression or in range/domain expressions, the use of `owl:Thing` as superclass or in range/domain expressions, and the use of the qualified restriction `exactly`. The second occurs in the class `owl:Thing`, that is setted as a subclass of the restriction `'hasUID only String'`. The property `hasUID` is used to guarantee that every instance in PR-OWL has a unique identifier (a requisite necessary to work with MEBN, where each possible state has to be unique). The third occurs in all `equivalent` expressions of PR-OWL 2, which includes conjunctions, min/max/exactly cardinality expressions, and universal/existential quantifiers. OWL 2 RL is very restrictive in relation to equivalent expressions, allowing only `Class`, `intersection`, and `hasValue` expressions. Finally, the fourth occurs in the `isRepresentedAs` range expression, where all possible formats to represent the probabilistic distributions are listed.



```
Argument
SubClass Of
  ● hasArgumentNumber exactly 1 integer
  ● hasArgumentNumber only integer
  ● isArgumentOf exactly 1 Thing
  ● isArgumentOf only
      (MExpression or RandomVariable)
  ● Thing
  ● typeOfArgument max 1 Thing
  ● typeOfDataArgument max 1 Literal

Thing
SubClass Of
  ● hasUID only string

Exemplar
Equivalent To
  ● Thing
    and (isTypeOfArgumentIn some ExemplarArgument)
    and (isExemplarIn only MFrag)
    and (isTypeOfArgumentIn only ExemplarArgument)
    and (isExemplarIn exactly 1 MFrag)
    and (isSubstitutedBy only anyURI)
    and (isSubstitutedBy exactly 1 anyURI)

isRepresentedAs
Ranges
  ● {"BIF"^^string , "BNIF"^^string ,
    "Ergo_ENT"^^string , "Genie_DSL"^^string ,
    "Hugin_NET"^^string , "IDEAL_IDE"^^string ,
    "MS_DSC"^^string , "MS_XBN"^^string ,
    "Netica_DNE"^^string , "Other"^^string ,
    "PR-OWL_MEBN"^^string ,
    "Quiddity_SPI"^^string , "UnBBayes"^^string}
```

Fig. 3. Examples of disallows in PR-OWL 2

Since the syntax of OWL 2 RL/RDF is based on RDF, it permits generalized RDF graphs, not having the several restrictions of OWL 2 DL. The pure OWL 2 RL profile, however, has restrictions to allow reasoning also with the Direct Semantics. We choose to adapt PR-OWL 2 RL with the OWL 2 RL restrictions for keeping the compatibility with both semantics. In order to make this adaptation it is necessary to fix the unsupported features listed above.

To solve the unsupported features, we analyzed three alternatives. The first consists in rewriting all expressions of PR-OWL 2 from OWL 2 DL to OWL 2 RL. However, this is not possible due the less expressive power of OWL 2 RL language. Expressions `subClass`, for instance, with existential quantifier on the right side cannot be expressed in this profile. The second alternative consists in rewriting to OWL 2 RL the expressions that can be rewritten, removing the others, and passing the responsibility of validating them forward to the PR-OWL reasoner. The problem with this alternative is that the resulting ontology is hard to understand due to the rewriting of the expressions to less intuitive axioms and because the restrictions are only partially explicit. The last alternative consists in turning PR-OWL into a lightweight ontology, containing only the class hierarchy and the object and data properties (with its domain and range restrictions). The full validation of the probabilistic model consistency is left to the PR-OWL reasoner. This was the chosen alternative because it results in a simpler ontology, sufficient for expressing the MEBN elements in OWL language.

The FOL formulas in PR-OWL 2 RL are evaluated in a different way than they are in the previous versions of PR-OWL. Using materialization, all implications of an added expression is calculated in load time. In the triplestores implementations, we do not have posterior reasoning: the queries are solved with searches on the database, using the SPARQL language, in a similar way to the SQL language in relational databases. This means that in the knowledge base we already have for example the hierarchy of an instance explicitly. For example, if an instance `a` is of the class `A`, and A is subclass of `B`, then, we will also have both `ClassAssertion(A,a)` and `ClassAssertion(B,a)` information on the base, where the second one was derived from the rule showed below (extracted from [17]).

```
IF T(?c1, rdfs:subClassOf, ?c2) AND T(?x, rdf:type, ?c1)
THEN T(?x, rdf:type, ?c2)
```

If we ask if `a` is instance of class `B`, the result will be `TRUE` because we will get the information `ClassAssertion(B,a)`. The advantage of this approach is that queries are very fast. The disadvantage is that complex reasoning cannot be handled, justifying the OWL 2 RL language restrictions.

The BNF grammar below shows the restrictions on the types of context nodes formulas accepted in PR-OWL 2 RL. The evaluation of these context nodes will be handled using the SPARQL language.

Listing 1.1. BNF Grammar for FOL Expressions in PR-OWL 2 RL

```
<atom>        ::= ov1 == ov2 |
                  booleanRV(ov1, [,ov2 ...])  |
                  nonBooleanRV(ov1, [,ov2 ...])  = ov0 |
                  ov0 = nonBooleanRV(ov1, [,ov2 ...]) |
                  nonBooleanRV(ov1, [,ov2 ...]) = CONST |
                  CONST = nonBooleanRV(ov1, [,ov2 ...])
<negation>    ::= NOT <atom>
<conjunction>::= <atom> [AND <atom>]+
<disjunction>::= <atom> [OR <atom>]+
<formula>     ::= <atom> | <negation> |
                  <conjunction> | <disjunction>
```

Table 2 shows how to evaluate these formulas using SPARQL. To solve the `EQUAL TO` operator between two ordinary variables, we can use the SPARQL `FILTER` construction, limiting the result of a query where the terms are equal.

The evaluation of AND and OR connectives is possible using period and UNION constructions. The negation can be implemented by the PR-OWL 2 reasoner in one of three ways depending on each case: for a not equal expression a FILTER can be used with the operator != (different); for a boolean RV it is sufficient to ask if it is equal FALSE; and finally, for a not boolean RV, we can use the operator NOT EXISTS inside a FILTER.

Table 2. Implementing PR-OWL 2 RL FOL expressions using SPARQL

MEBN Expression	SPARQL Expression
AND	. (period)
OR	UNION
EQUAL TO	Use of = inside a FILTER
NOT	It depends on the case

To evaluate expressions where we do not know the value of some ordinary variable, we use the SPARQL SELECT construction. If we already know all values, a command ASK is used. This command evaluates the expression and returns TRUE or FALSE. The evaluation of the context nodes is made one by one and the implementation is responsible for keeping the consistency between the ordinary variable values of each node. The following code shows a SELECT to get which procurements ENTERPRISE_1 won (node 5 in Figure 1).

```
SELECT ?procurement
WHERE  { ?procurement rdf:type Procurement .
       ENTERPRISE_1 hasWinnerOfProcurement ?procurement}
```

Finally, for the new language to be useful, it is also necessary to propose a new algorithm for generating a SSBN. The previous SSBN algorithm implemented in PR-OWL 2 starts from the queries set as well as the findings set. Since we can have a large assertive base in PR-OWL 2 RL, making the findings set very large, the previous SSBN construction algorithm might be hindered. We plan to extend the algorithm previously proposed in [6], by starting only from the queries set and removing known issues with it. For instance, the version proposed in [6] does not evaluate the parent nodes of a query, even if they are not d-separated from the evidence.

Using the new language proposed, together with a triplestore and the materialization approach, it is possible to solve the scalability problems presented. The BNF grammar proposed is sufficient to evaluate all context node formulas used in the Procurement Fraud probabilistic ontology.

The Theorem PR1 [17] limits the entailed statements to assertions. The reasoning in PR OWL 2 RL is mainly over the assertive base (ABox), but, based on the use cases already developed for PR-OWL, this does not seem to be a problem.

It is important to note that Costa, the author of PR-OWL, already visualized the possibility of creating more restrictive versions of the language to guarantee tractability [7]. The objective of PR-OWL 2 RL is not to substitute the previous version (in the way that PR-OWL 2 intends to substitute PR-OWL). Both PR-OWL 2 and PR-OWL 2 RL have characteristics that make them suitable for different kind of domains. While PR-OWL 2 is recommended for heavyweight ontologies, with complex expressions, but limited assertive bases, PR-OWL 2 RL is ideal for lightweight ontologies, with simple expressions and a very large knowledge base. This last one, for example, is the case of the ontologies in Linked Data projects.

5 Conclusion and Future Work

Using a less expressive version of OWL for reasoning in polynomial time, PR-OWL 2 RL is developed to work with ontologies containing millions of triples. When used together with RDF triplestores, it can solve the scalability problem of the previous PR-OWL versions. Using a commercial database it is possible to work with billions of triples, making it suitable even for working with Big Data. The restrictions on the expressiveness of OWL 2 RL do not allow it to express some complex statements, but it is sufficient for a lot of domains, such as the Procurement Fraud, and Linked Open Data projects. This paper presented limitations on the first-order expressions used in context node formulas, restricting the use of MEBN logic, but allowing at the same time the same constructs which are implemented and allowed in UnBBayes' PR-OWL 2 plug-in.

A future work that is already under way is the implementation of a plug-in for PR-OWL 2 RL in UnBBayes. In this plug-in we plan to use the triplestore GraphDB Lite, a free version of GraphDB [2]. GraphDB, previously OWLIM, partially implements the OWL 2 RL profile (it does not implement the rules related to datatypes), using the OWL 2 RL/RDF rules and a materialization approach. The UnBBayes' PR-OWL 2 RL plug-in will allow the user to model a probabilistic ontology using the language, to put it into the triplestore, to fill the assertive base, and to build a SSBN from the queries set. Other future work is create new study cases to validate the solution. We also plan to make an extension of the LUBM ontology by adding uncertainty concepts to it, making it possible to construct a benchmark for large probabilistic ontologies.

References

1. Franz Baader and Werner Nutt. Basic description logics. In *The Description Logic Handbook: Theory, Implementation, and Applications*, pages 43–95, 2003.
2. Barry Bishop and Spas Bojanov. Implementing OWL 2 RL and OWL 2 QL rule-sets for OWLIM. In Michel Dumontier and Mlanie Courtot, editors, *OWLED*, volume 796 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.
3. Rommel N. Carvalho, Marcelo Ladeira, Laécio L. Santos, Shou Matsumoto, and Paulo C. G. Costa. A GUI tool for plausible reasoning in the semantic web using MEBN. In *Innovative Applications in Data Mining*, pages 17–45. 2009.

4. Rommel N. Carvalho, Kathryn B. Laskey, and Paulo C. G. da Costa. PR-OWL 2.0 - bridging the gap to OWL semantics. In *Proceedings of the 6th International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2010), collocated with the 9th International Semantic Web Conference (ISWC-2010), Shanghai, China, November 7, 2010*, pages 73–84, 2010.
5. Rommel N. Carvalho, Shou Matsumoto, Kathryn B. Laskey, Paulo C. G. da Costa, Marcelo Ladeira, and Laécio L. Santos. Probabilistic ontology and knowledge fusion for procurement fraud detection in brazil. In *Uncertainty Reasoning for the Semantic Web II, International Workshops URSW 2008-2010 Held at ISWC and UniDL 2010 Held at FLoC, Revised Selected Papers*, pages 19–40, 2013.
6. Paulo C. G. Costa, Marcelo Ladeira, Rommel N. Carvalho, Kathryn B. Laskey, Laecio L. Santos, and Shou Matsumoto. A first-order Bayesian tool for probabilistic ontologies. In David Wilson and H. Chad Lane, editors, *FLAIRS Conference*, pages 631–636. AAAI Press, 2008.
7. Paulo Cesar G. da Costa, Kathryn B. Laskey, and Kenneth J. Laskey. PR-OWL: A bayesian ontology language for the semantic web. In *International Semantic Web Conference, ISWC 2005, Galway, Ireland, Workshop 3: Uncertainty Reasoning for the Semantic Web, 7 November 2005*, pages 23–33, 2005.
8. Francesco M. Donini. Complexity of reasoning. In *The Description Logic Handbook: Theory, Implementation, and Applications*, pages 96–136, 2003.
9. Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. LUBM: A benchmark for OWL knowledge base systems. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(2):158–182, 2005.
10. Yevgeny Kazakov. RIQ and SROIQ are harder than SHOIQ. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference, KR 2008, Sydney, Australia, September 16-19, 2008*, pages 274–284, 2008.
11. Kathryn B. Laskey. MEBN: a language for First-Order Bayesian knowledge bases. *Artificial Intelligence*, 172(2-3):140–178, 2008.
12. Bijan Parsia and Evren Sirin. Pellet: An OWL DL reasoner. In *Third International Semantic Web Conference-Poster*, volume 18, 2004.
13. Rob Shearer, Boris Motik, and Ian Horrocks. HermiT: A highly-efficient OWL reasoner. In Catherine Dolbear, Alan Ruttenberg, and Ulrike Sattler, editors, *OWLED*, volume 432 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.
14. Herman J. ter Horst. Combining RDF and part of OWL with rules: Semantics, decidability, complexity. In *The Semantic Web - ISWC 2005, 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005, Proceedings*, pages 668–684, 2005.
15. Dmitry Tsarkov and Ian Horrocks. FaCT++ description logic reasoner: System description. In *Automated Reasoning, Third International Joint Conference, IJCAR 2006, Seattle, WA, USA, August 17-20, 2006, Proceedings*, pages 292–297, 2006.
16. W3C. OWL 2 Web Ontology Language document overview. http://www.w3.org/TR/owl2-overview/, 2012.
17. W3C. OWL 2 Web Ontology Language profiles. http://www.w3.org/TR/owl2-profiles/, 2012.

Efficient Learning of Entity and Predicate Embeddings for Link Prediction in Knowledge Graphs

Pasquale Minervini, Claudia d'Amato, Nicola Fanizzi, Floriana Esposito

LACAM – Department of Computer Science – Università degli Studi di Bari Aldo Moro, Italy
{firstname.lastname}@uniba.it

Abstract. *Knowledge Graphs* are a widely used formalism for representing knowledge in the Web of Data. We focus on the problem of predicting missing links in large knowledge graphs, so to discover new facts about the world. Recently, *representation learning* models that embed entities and predicates in continuous vector spaces achieved new state-of-the-art results on this problem. A major limitation in these models is that the *training process*, which consists in learning the optimal entity and predicate embeddings for a given knowledge graph, can be very computationally expensive: it may even require days of computations for large knowledge graphs. In this work, by leveraging *adaptive learning rates*, we propose a principled method for reducing the training time by an order of magnitude, while learning more accurate link prediction models. Furthermore, we employ the proposed training method for evaluating a set of novel and scalable models. Our evaluations show significant improvements over state-of-the-art link prediction methods on the WordNet and Freebase datasets.

1 Introduction

Knowledge Graphs (KGs) are graph-structured Knowledge Bases (KBs), where factual knowledge about the world is represented in the form of relationships between entities. They are widely used for representing relational knowledge in a variety of domains, such as citation networks and protein interaction networks. An example of their widespread adoption is the *Linked Open Data* (LOD) Cloud, a set of interlinked KGs such as Freebase [1] and WordNet [2]. As of April 2014, the LOD Cloud was composed by 1,091 interlinked KBs, describing over 8×10^6 entities, and 188×10^6 relationships holding between them [1].

Despite their large size, many KGs are still largely incomplete. For example consider Freebase [2], a core element in the Google Knowledge Vault project [3]: 71% of the persons described in Freebase have no known place of birth, 75% of them have no known nationality, and the coverage for less frequent predicates can be even lower [3].

In this work we focus on the problem of *completing missing links* in large KGs, so to discover new facts about the world. In the literature, this problem is referred to as *link prediction* or *knowledge graph completion*, and has received a considerable attention over the last years [4,5].

[1] State of the LOD Cloud 2014: `http://lod-cloud.net/`

[2] Publicly available at `https://developers.google.com/freebase/data`

Recently, *representation learning* [6] models such as the *Translating Embeddings* model (TransE) [7] achieved new state-of-the-art link prediction results on large and Web-scale KGs [4,5]. Such models learn a unique *distributed representation*, or *embedding*, for each entity and predicate in the KG: each entity is represented by a low-dimensional continuous *embedding vector*, and each predicate is represented by an *operation* in the embedding vector space, such as a *translation* (as in [7]) or an *affine transformation* (as in [8]). We refer to these models as *embedding models*, and to the learned distributed representations as *embeddings*.

The embeddings of all entities and predicates in the KG are learned jointly: the learning process consists in minimizing a global loss functional considering the whole KG, by back-propagating the loss to the embeddings [3]. As a consequence, the learned entity and predicate embeddings retain global, structural information about the whole KG, and can be used to serve several kinds of applications. In *link prediction*, the confidence of each candidate edge can be measured as a function of the embeddings of its source entity, its target entity, and its predicate.

A major limitation in embedding models proposed so far, however, is that the learning procedure (i.e. learning the optimal embeddings of all entities and predicates in the KG) can be very time-consuming: it is based on an incremental optimization algorithm that may require days of computation to converge for large KGs [10].

In this work, we propose a novel principled method for significantly reducing the learning time in embedding models, based on *adaptive per-parameter learning rates*. Furthermore, we employ the proposed training method for evaluating a variety of novel embedding models: our evaluations achieves new state-of-the-art link prediction results on the WordNet and Freebase datasets.

2 Basics

RDF Graphs The most widely used formalism for representing knowledge graphs is the W3C *Resource Description Framework* (RDF) [4], a recommended standard for representing knowledge on the Web. An RDF KB, also referred to as *RDF graph*, is a set of *RDF triples* in the form $\langle s, p, o \rangle$, where s, p and o respectively denote the *subject*, the *predicate* and the *object* of the triple: s and o are *entities*, and p is a relation type. Each triple $\langle s, p, o \rangle$ describes a statement, which is interpreted as "*A relationship p holds between entities s and o*".

Example 2.1 (Shakespeare). The statement "*William Shakespeare is an author who wrote Othello and the tragedy Hamlet*" can be expressed by the following RDF triples:

⟨Shakespeare,	profession,	Author⟩
⟨Shakespeare,	author,	Hamlet⟩
⟨Shakespeare,	author,	Othello⟩
⟨Hamlet,	genre,	Tragedy⟩

[3] In natural language processing, a similar procedure is used by the word2vec model [9] for learning an unique distributed representation for each word in a corpus of documents.

[4] http://www.w3.org/TR/rdf11-concepts/

An RDF graph can be viewed as a *labeled directed multigraph*, where each entity is a vertex, and each RDF triple is represented by a directed edge whose label is a predicate, and emanating from its subject vertex to its object vertex. In RDF KBs, the *Open-World Assumption* holds: a missing triple does not mean that the corresponding statement is false, but rather that its truth value is unknown (it cannot be observed). In the following, given an RDF graph G, we denote as $\mathcal{E}_G$ the set of all entities occurring as subjects or objects in G, and as $\mathcal{R}_G$ the set of all predicates occurring in G:

$$\begin{aligned}\mathcal{E}_G &= \{s \mid \exists \langle s, p, o \rangle \in G\} \cup \{o \mid \exists \langle s, p, o \rangle \in G\},\\ \mathcal{R}_G &= \{p \mid \exists \langle s, p, o \rangle \in G\}.\end{aligned}$$

For instance, in the case of the RDF graph shown in Ex. 2.1, the sets $\mathcal{E}_G$ and $\mathcal{R}_G$ are the following: $\mathcal{E}_G = \{\texttt{Author}, \texttt{Shakespeare}, \texttt{Hamlet}, \texttt{Othello}, \texttt{Tragedy}\}$ and $\mathcal{R}_G = \{\texttt{profession}, \texttt{author}, \texttt{genre}\}$.

Furthermore, we denote as $\mathcal{S}_G = \mathcal{E}_G \times \mathcal{R}_G \times \mathcal{E}_G$ the space of *possible triples* of G, i.e. the set of all triples that can be created by using the entities and predicates in G (note that $G \subseteq \mathcal{S}_G$). We refer to all triples in G as *observed triples*, and to all triples in $\mathcal{S}_G \setminus G$ as *unobserved triples*.

Energy-Based Models Embedding models for KGs can be described in terms of *Energy-Based Models* (EBMs) [11]: EBMs are a versatile and flexible framework for modeling dependencies between variables. In the fields of *representation learning* and *deep learning* [6], EBMs are employed as building blocks for constructing hierarchical models that achieve ground-breaking results in several learning tasks.

A fundamental component in an EBM is a scalar-valued *energy function* (or *scoring function*) $E_\theta(\cdot)$, parametrized by θ, which associates a scalar *energy value* to the configuration of a set of variables. The energy of a configuration of a set of variables is inversely proportional to its probability: *more likely configurations* are associated with *lower energy values*, while *less likely configurations* are associated with *higher energy values*. Several tractable methods have been proposed for learning the parameters of an energy function [11,6]. In particular, the problem of learning the optimal parameters $\hat{\theta}$ can be cast as solving the following optimization problem [11]:

$$\hat{\theta} = \arg\min_{\theta \in \Theta} \mathcal{L}(E_\theta, \mathcal{D}),$$

where Θ is the *parameter space*, and $\mathcal{L}(\cdot)$ is a *loss functional* which measures the quality of the energy function $E_\theta(\cdot)$ on the data $\mathcal{D}$. Intuitively, the loss functional $\mathcal{L}(\cdot)$ assigns a lower loss to energy functions that associate a lower energy (corresponding to a higher probability) to correct answers, and higher energy (corresponding to a lower probability value) to all other incorrect answers.

Energy-Based Models for RDF Graphs As discussed in [12], embedding models for KGs define an *energy distribution* $E_\theta : \mathcal{S}_G \to \mathbb{R}$ over the space of possible triples $\mathcal{S}_G$. For instance, the models proposed in [8,7,13,12] are used for assigning a score $E(\langle s, p, o \rangle)$ to each triple $\langle s, p, o \rangle$ in $\mathcal{S}_G$. In a *link prediction* setting, such models are

Table 1: Energy functions used by several link prediction models, and the corresponding number of parameters: $n_e = |\mathcal{E}_G|$ and $n_r = |\mathcal{R}_G|$ respectively denote the number of entities and predicates in G, and $k, d \in \mathbb{N}$ are user-defined hyper-parameters.

Model	Energy function $E(\langle s, p, o \rangle)$	Parameters Complexity
Unstructured [12]	$\|\mathbf{e}_s - \mathbf{e}_o\|_2^2, \qquad \mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^k$	$\mathcal{O}\left(n_e k\right)$
TransE [7]	$\|(\mathbf{e}_s + \mathbf{e}_p) - \mathbf{e}_o\|_{\{1,2\}}, \qquad \mathbf{e}_p \in \mathbb{R}^k$	$\mathcal{O}\left(n_e k + n_r k\right)$
SE [8]	$\|\mathbf{W}_{p,1}\mathbf{e}_s - \mathbf{W}_{p,2}\mathbf{e}_o\|_1, \qquad \mathbf{W}_{p,i} \in \mathbb{R}^{k \times k}$	$\mathcal{O}\left(n_e k + n_r k^2\right)$
SME lin. [12]	$(\mathbf{W}_1\mathbf{e}_s + \mathbf{W}_2\mathbf{e}_p + \mathbf{b}_1)^T(\mathbf{W}_3\mathbf{e}_o + \mathbf{W}_4\mathbf{e}_p + \mathbf{b}_2)$ $\mathbf{e}_p \in \mathbb{R}^k, \mathbf{W}_i \in \mathbb{R}^{d \times k}, \mathbf{b}_j \in \mathbb{R}^d$	$\mathcal{O}\left(n_e k + n_r k + dk\right)$
SME bil. [12]	$\left[(\mathbf{W}_1 \times_3 \mathbf{e}_p)\mathbf{e}_s\right]^T \left[(\mathbf{W}_2 \times_3 \mathbf{e}_p)\mathbf{e}_o\right]$ $\mathbf{e}_p \in \mathbb{R}^k, \mathbf{W}_i \in \mathbb{R}^{d \times k \times k}, \mathbf{b}_j \in \mathbb{R}^d$	$\mathcal{O}\left(n_e k + n_r k + dk^2\right)$
NTN [13]	$\mathbf{u}_p^T f\left(\mathbf{e}_s^T \mathbf{W}_p \mathbf{e}_o + \mathbf{W}_{p,1}\mathbf{e}_s + \mathbf{W}_{p,2}\mathbf{e}_o + \mathbf{b}_p\right)$ $\mathbf{W}_p \in \mathbb{R}^{k \times k \times d}, \mathbf{W}_{p,i} \in \mathbb{R}^{d \times k}, \mathbf{u}_p, \mathbf{b}_p \in \mathbb{R}^d$	$\mathcal{O}\left(n_e k + n_r dk^2\right)$

used as follows. First, the optimal parameters $\hat{\theta}$ of the energy function are learned: the parameters are composed by the embeddings of all entities and predicates in the KG. Then, the energy function $E_{\hat{\theta}}(\cdot)$ is used for ranking unobserved triples: those with lower energy values have a higher probability of representing true statements, and are considered more likely candidates for a completion of the KG.

Consider the RDF graph shown in Ex. 2.1. In such a graph, we prefer learning an energy-based model that assigns a lower energy (a higher probability value) to the triple $\langle$`Othello`, `genre`, `Tragedy`$\rangle$, which is unobserved but represents the true statement "*Othello is a Tragedy*", and a higher energy (a lower probability value) to other unobserved triples, for example $\langle$`Hamlet`, `genre`, `Author`$\rangle$.

3 Energy-Based Embedding Models

Several EBMs have been proposed in the literature for addressing the problem of link prediction in KGs [8,14,7,13,12,15]. These models share a fundamental characteristic: they can be used for learning a *distributed representation* (or *embedding*) for each entity and predicate in the KG. We refer to such models as *embedding models*, and denote the distributed representation of an entity or predicate z by adding a subscript to the corresponding vector or matrix representation, as in $\mathbf{e}_z \in \mathbb{R}^k$.

Formally, let G be an RDF graph. For each entity $x \in \mathcal{E}_G$, embedding models learn a continuous vector representation $\mathbf{e}_x \in \mathbb{R}^k$, with $k \in \mathbb{N}$, called the *embedding vector* of x. Similarly, for each predicate $p \in \mathcal{R}_G$, embedding models learn an *operation* on the embedding vector space, characterized by a set of *embedding parameters*. This can be an empty set of parameters, as in the *Unstructured* model proposed in [12]; a translation vector $\mathbf{e}_p \in \mathbb{R}^k$, as in the *Translating Embeddings* model proposed in [7]; or a more complex set of parameters.

The distributed representations of all entities and predicates in G are then used for defining an *energy distribution* $E : \mathcal{S}_G \to \mathbb{R}$ over the space of possible triples of G. In particular, the energy $E(\langle s, p, o \rangle)$ of a triple $\langle s, p, o \rangle$ is defined as a function of the distributed representations of its subject s, its predicate p and its object o.

In Tab. 1, we report the energy functions adopted by several models proposed in the literature. For each model, we report the number of parameters needed for storing the distributed representations of all entities and predicates: $n_e = |\mathcal{E}_G|$ denotes the number of entities in the KG, $n_r = |\mathcal{R}_G|$ denotes the number of predicates, and $k, d \in \mathbb{N}$ are user-defined hyper-parameters. In general, if the number of parameters in a model grows *super-linearly* with the number of entities and predicates in the KG, it becomes increasingly harder for the model to scale to very large and Web-scale KGs.

The Translating Embeddings model Among the models outlined in Tab. 1, the recently proposed *Translating Embeddings* model (TransE) [7] has interesting characteristics, and recently received a considerable attention [4]:

- It achieves more accurate link prediction results than other state-of-the-art methods on several datasets.
- The number of parameters in TransE scales *linearly* in the number of entities n_e and predicates n_r in the KG: this allows TransE to potentially scale to large KGs.

The TransE model is very simple. In TransE, each entity $x \in \mathcal{E}_G$ is represented by its embedding vector $\mathbf{e}_x \in \mathbb{R}^k$, and each predicate $p \in \mathcal{R}_G$ is represented by a (vector) *translation operation* $\mathbf{e}_p \in \mathbb{R}^k$. The energy of a triple $\langle s, p, o \rangle$ is given by the L_1 or L_2 distance between $(\mathbf{e}_s + \mathbf{e}_p)$ and $\mathbf{e}_o$:

$$E(\langle s, p, o \rangle) = \|(\mathbf{e}_s + \mathbf{e}_p) - \mathbf{e}_o\|_{\{1,2\}}.$$

In TransE, all the embedding and translation vectors are learned jointly from the KG by using *Stochastic Gradient Descent*, as discussed in Sect. 4. The number of parameters needed by the TransE model for storing all the embedding and translation vectors is $(n_e k + n_r k)$, a quantity that grows *linearly* with n_e and n_r. For such a reason, TransE can potentially scale to very large and highly-relational KGs [7].

A New Set of Embedding Models In the following, we propose a set of variants of the TransE model, which preserve its scalability properties. Let $d(\mathbf{x}, \mathbf{y})$ be a *dissimilarity* function, from the following set: $d(\mathbf{x}, \mathbf{y}) \in \{\|\mathbf{x} - \mathbf{y}\|_1, \|\mathbf{x} - \mathbf{y}\|_2, -\mathbf{x}^T\mathbf{y}\}$, i.e. chosen from the L_1 and L_2 distance, and the negative inner product. We propose the following embedding models, where each is defined by the corresponding energy function $E(\cdot)$:

- TransE : $E(\langle s, p, o \rangle) = d(\mathbf{e}_s + \mathbf{e}_p, \mathbf{e}_o)$,
- TransE$^+$: $E(\langle s, p, o \rangle) = d(\mathbf{e}_s + \mathbf{e}_{p,1}, \mathbf{e}_o + \mathbf{e}_{p,2})$,
- ScalE : $E(\langle s, p, o \rangle) = d(\mathbf{e}_s \circ \mathbf{e}_p, \mathbf{e}_o)$,
- ScalE$^+$: $E(\langle s, p, o \rangle) = d(\mathbf{e}_s \circ \mathbf{e}_{p,1}, \mathbf{e}_o \circ \mathbf{e}_{p,2})$,

where $\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^k$ are the embedding vectors of the entities appearing as the subject s and the object o; $\mathbf{e}_{p,\cdot} \in \mathbb{R}^k$ are the embedding parameters of the predicate p, denoting either a *translation* or a *scaling* vector; and $\circ$ denotes the Hadamard (element-wise) product, corresponding to the vector *scaling* operation. The energy function in TransE is the same used in [7], but also allows using the negative inner product as a dissimilarity measure between the (translated) subject and object embedding vectors, if it shows to

Algorithm 1 Learning the model parameters via SGD

```
Require: Learning rate $\eta$, Batch size $n$, Iterations $\tau$
Ensure: Optimal model parameters (embeddings) $\hat{\theta}$
Initialize the model parameters $\theta_0$
for $t \in \langle 1, \ldots, \tau \rangle$ do
    $\mathbf{e}_x \leftarrow \mathbf{e}_x / \|\mathbf{e}_x\|, \; \forall x \in \mathcal{E}_G$    {Normalize all entity embedding vectors}
    $T \leftarrow \textsc{SampleBatch}(G, n)$    {Sample a batch of observed and corrupted triples}
    $g_t \leftarrow \nabla \sum_{(y, \tilde{y}) \in T} \left[ \gamma + E_\theta(y) - E_\theta(\tilde{y}) \right]_+$    {Compute the gradient of the loss}
    $\varDelta_t \leftarrow -\eta g_t$    {Compute the update to model parameters (embeddings)}
    $\theta_t \leftarrow \theta_{t-1} + \varDelta_t$    {Update the model parameters}
end for
return $\theta_\tau$
```

improve the performance on the validation set. The TransE^+ model generalizes TransE by also translating the object embedding vector $\mathbf{e}_o$.

The ScalE and ScalE^+ models are similar to the previous two models, but replace the vector *translation* with a *scaling* operation. The rationale behind ScalE and ScalE^+ is the following: scaling the embedding vector of an entity can be seen as *weighting* the (latent) features of such an entity in the embedding vector space.

All proposed models share the same advantages as the TransE model: (i) the required number of parameters is $\mathcal{O}(n_e k + n_r k)$, which grows *linearly* with n_e and n_r, and (ii) the energy function and its gradient w.r.t. the embedding of entities and predicates can be computed very efficiently, using element-wise vector operations.

4 Improving the Efficiency of the Embeddings Learning Process

In [8,7,12], authors propose a method for jointly learning the embeddings of all entities and predicates in a KG G. The method relies on a *stochastic optimization process*, that iteratively updates the embeddings by reducing the energy of triples in G (observed triples) while increasing the energy of triples in $\mathcal{S}_G \setminus G$ (unobserved triples).

During the learning process, unobserved triples are randomly generated by means of a *corruption process*, which replaces either the subject or the object of each observed triple with another entity in G. More formally, given an observed triple $y \in G$, let $\mathcal{C}_G(y)$ denote the set of all corrupted triples obtained by replacing either its subject or object with another entity in G:

$$\mathcal{C}_G(\langle s, p, o \rangle) = \{ \langle \tilde{s}, p, o \rangle \mid \tilde{s} \in \mathcal{E}_G \} \cup \{ \langle s, p, \tilde{o} \rangle \mid \tilde{o} \in \mathcal{E}_G \}.$$

The embeddings of all entities and predicates in the KG, which compose the *model parameters*, can be learned by minimizing a *margin-based ranking loss*. More formally, learning the optimal model parameters $\hat{\theta}$, corresponding to all the entity and predicate embeddings, is equivalent to solving the following constrained minimization problem:

$$\begin{aligned} \underset{\theta \in \Theta}{\text{minimize}} \quad & \sum_{y \in G} \; \sum_{\tilde{y} \in \mathcal{C}_G(y)} \left[\gamma + E_\theta(y) - E_\theta(\tilde{y}) \right]_+ \\ \text{subject to} \quad & \forall x \in \mathcal{E}_G : \; \|\mathbf{e}_x\| = 1, \end{aligned} \tag{1}$$

where $[x]_+ = \max\{0, x\}$, and $\gamma \geq 0$ is a hyper-parameter referred to as *margin*. The objective function in Eq. 1 (corresponding to the loss functional $\mathcal{L}(\cdot)$ discussed in Sect. 2) enforces the energy of observed triples to be lower than the energy of unobserved triples. The constraints in the optimization problem prevent the training process to trivially solve the problem by increasing the entity embedding norms.

Stochastic Gradient Descent In the literature, the constrained loss minimization problem in Eq. 1 is solved using *Stochastic Gradient Descent* (SGD) in mini-batch mode, as summarized in Alg. 1. On each iteration, the algorithm samples a batch of triples from the knowledge graph G. Batches are obtained by first randomly permuting all triples in G, partitioning them into n_b batches of the same size, and then iterating over such batches. A single pass over all triples in G is called an *epoch*. Then, for each triple y in the batch, the algorithm generates a *corrupted* triple $\tilde{y}$ uniformly sampled from $\mathcal{C}_G(y)$: this leads to a set T of observed/corrupted pairs of triples $\langle y, \tilde{y} \rangle$. The observed/corrupted triple pairs are used for computing the gradient of the objective (loss) function in Eq. 1 w.r.t. the current model parameters θ. Finally, θ is updated in the steepest descent direction of the objective function. This procedure is repeated until convergence.

The main drawback of SGD is that it requires an initial, careful tuning of the global learning rate η, which is then used for updating all model parameters, regardless of their peculiarities. However, if an entity $x \in \mathcal{E}_G$ occurs in a limited number of triples in G, the corresponding embedding vector $\mathbf{e}_x \in \mathbb{R}^k$ will be updated less often, and it will require a much longer time to be learned. For such a reason, SGD may be very time-consuming and slow to converge. For instance, it was reported in [10] that learning the optimal embeddings in TransE may require days of computation for large KGs.

A possible solution to this problem consists in associating *smaller learning rates* to parameters updated more often, such as the embedding vectors of entities appearing more frequently, and *larger learning rates* to parameters updated less often.

Adaptive Learning Rates In order to reduce the time required for learning all entity and predicate embeddings, in this work we propose leveraging *Adaptive Per-Parameter Learning Rates*. While SGD uses a global, fixed learning rate η for updating all parameters, we propose relying on methods for estimating the *optimal* learning rate for each parameter, while still being tractable for learning very large models.

We consider two highly-scalable criteria for selecting the optimal learning rates, namely the *Momentum method* [16] and *AdaGrad* [17]: they specify alternate ways of computing the parameters update $\varDelta_t$, defined in Alg. 1 on line 6.

Momentum Method The idea behind this method is to accelerate the progress along dimensions where the sign of the gradient does not change, while slowing the progress along dimensions where the sign of the gradient continues to change. The new update rule is defined as follows:

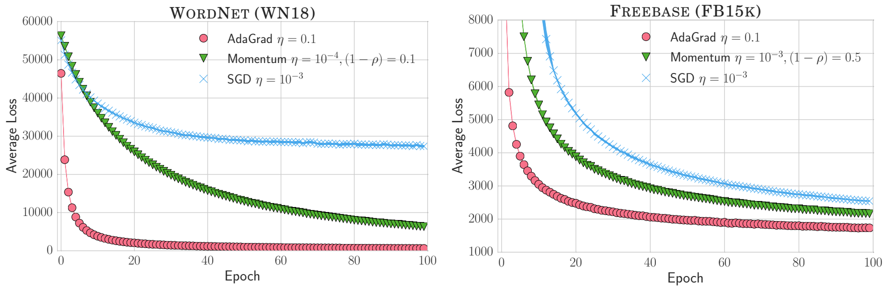
$$\varDelta_t \leftarrow \rho \varDelta_{t-1} - \eta_m g_t,$$

where $\eta_m \in \mathbb{R}$ is a user-defined hyper-parameter.

Table 2: Statistics for the datasets used in the **Link Prediction** and **Triple Classification** tasks.

Dataset	**Entities**	**Predicates**	**Train. Triples**	**Valid. Triples**	**Test Triples**
FREEBASE (FB15K) [7]	14,951	1,345	483,142	50,000	59,071
WORDNET (WN18) [7]	40,943	18	141,442	5,000	5,000

Fig. 1: Average loss (the lower, the better) across 10 TransE parameters learning tasks on the WORDNET (WN18) and FREEBASE (FB15K) datasets, using the optimal TransE settings reported in [7]. For each optimization method, we report the hyper-parameter values that achieve the lowest average loss after 100 epochs, and the corresponding average loss values.



AdaGrad This method is based on the idea that the learning rate of each parameter should grow with the inverse of gradient magnitudes. The update rule in AdaGrad is:

$$\varDelta_t \leftarrow -\frac{\eta_a}{\sqrt{\sum_{j=1}^{t} g_j^2}} g_t,$$

where $\eta_a \in \mathbb{R}$ is a user-defined hyper-parameter. AdaGrad adds nearly no complexity, it has very strong convergence guarantees [17], and it has shown remarkable results on large scale learning tasks in distributed environments [18].

5 Empirical Evaluations

This section is organized as follows. In Sect. 5.1 we describe experimental settings, datasets and evaluation metrics. In Sect. 5.2, we show that *adaptive learning rates* sensibly improve both the efficiency of the learning process, and the predictive accuracy of embedding models. In Sect. 5.3, we empirically evaluate the novel embedding models proposed in Sect. 3, by training them using adaptive learning rates.

5.1 Experimental Settings

Link Prediction In these experiments, we used the metrics proposed in [7] for evaluating the *rank* of each test triple. In particular, for each test triple $\langle s, p, o \rangle$, its object

Table 3: **Link Prediction** Results: Test performance of several link prediction methods on the WordNet and Freebase datasets. Results show the Mean Rank (the lower, the better) and Hits@10 (the higher, the better) for both the Raw and the Filtered settings [7].

Dataset	WordNet (WN18)				Freebase (FB15k)			
Metric	Mean Rank		Hits@10 (%)		Mean Rank		Hits@10 (%)	
	Raw	Filt.	Raw	Filt.	Raw	Filt.	Raw	Filt.
Unstructured [12]	315	304	35.3	38.2	1,074	979	4.5	6.3
RESCAL [19]	1,180	1,163	37.2	52.8	828	683	28.4	44.1
SE [8]	1,011	985	68.5	80.5	273	162	28.8	39.8
SME lin. [12]	545	533	65.1	74.1	274	154	30.7	40.8
SME bil. [12]	526	509	54.7	61.3	284	158	31.3	41.3
LFM [14]	469	456	71.4	81.6	283	164	26.0	33.1
TransE [7]	263	251	75.4	89.2	243	125	34.9	47.1
TransE^A	**169**	**158**	**80.5**	**93.5**	**189**	**73**	**44.0**	**60.1**

o is replaced with every entity $\tilde{o} \in \mathcal{E}_G$ in the KG G in turn, generating a set of *corrupted* triples in the form $\langle s, p, \tilde{o} \rangle$. The energies of corrupted triples are first computed by the model, then sorted in ascending order, and used to compute the rank of the correct triple. This procedure is repeated by corrupting the subject. Aggregated over all the test triples, this procedure leads to the following two metrics: the *averaged rank*, denoted by Mean Rank, and the *proportion of ranks not larger than* 10, denoted by Hits@10. This is referred to as the Raw setting.

In the Filtered setting, corrupted triples that exist in either the training, validation or test set were removed before computing the rank of each triple. In both settings, a lower Mean Rank is better, while a higher Hits@10 is better.

5.2 Evaluation of Adaptive Learning Rates

Learning Time For comparing Momentum and AdaGrad with SGD on the task of solving the optimization problem in Eq. 1, we empirically evaluated such methods on the task of learning the parameters in TransE on WN18 and FB15k, using the optimal hyper-parameter settings reported in [7]: $k = 20$, $\gamma = 2$, $d = L_1$ for WN18, and $k = 50$, $\gamma = 1$, $d = L_1$ for FB15k. Following the evaluation protocol in [20], we compared the optimization methods by using a large grid of hyper-parameters. Let $\mathcal{G}_\eta = \{10^{-6}, 10^{-5}, \ldots, 10^{1}\}$ and $\mathcal{G}_\rho = \{1-10^{-4}, 1-10^{-3}, \ldots, 1-10^{-1}, 0.5\}$. The grids of hyper-parameters considered for each of the optimization methods were the following:

- **SGD** and **AdaGrad:** rate $\eta, \eta_a \in \mathcal{G}_\eta$.
- **Momentum:** rate $\eta_m \in \mathcal{G}_\eta$, decay rate $\rho \in \mathcal{G}_\rho$.

For each possible combination of optimization method and hyper-parameter values, we performed an evaluation consisting in 10 learning tasks, each using a different random initialization of the model parameters.

Fig. 1 shows the behavior of the loss functional for each of the optimization methods, using the best hyper-parameter settings selected after 100 training epochs. We can

Table 4: **Link Prediction** Results: Test performance of the embedding models proposed in Sect. 3 on the WordNet and Freebase datasets. Results show the Mean Rank (the lower, the better) and Hits@10 (the higher, the better) [7].

Dataset	WordNet (WN18)				Freebase (FB15k)			
Metric	Mean Rank		Hits@10 (%)		Mean Rank		Hits@10 (%)	
	Raw	Filt.	Raw	Filt.	Raw	Filt.	Raw	Filt.
TransE, from [7] SGD	263	251	75.4	89.2	243	125	34.9	47.1
TransE AdaGrad	161	150	80.5	93.5	183	63	47.9	68.2
TransE$^+$ (Sect. 3) AdaGrad	**159**	**148**	79.6	92.6	196	78	44.9	62.4
ScalE (Sect. 3) AdaGrad	187	174	82.7	94.5	194	62	**49.8**	**73.0**
ScalE$^+$ (Sect. 3) AdaGrad	298	287	**83.7**	**95.5**	**185**	**59**	50.0	71.5

immediately observe that, for both WordNet (WN18) and Freebase (FB15k), AdaGrad (with rate $\eta_a = 0.1$) yields sensibly lower values of the loss functional than SGD and Momentum, even after very few iterations ($<$ 10 epochs). The duration of each epoch was similar in all methods: each epoch took approx. **1.6 seconds** in WordNet (WN18), and approx. **4.6 seconds** in Freebase (FB15k) on a single i7 CPU.

Quality of Learned Models We also measured the *quality* of models learned by AdaGrad, in terms of the Mean Rank and Hits@10 metrics, in comparison with SGD. For this purpose, we trained TransE using AdaGrad (instead of SGD) with $\eta_a = 0.1$ for 100 epochs, denoted as TransE^A, and compared it with results obtained with TransE from the literature on *Link Prediction* tasks on the WordNet and Freebase datasets. Hyper-parameters were selected according to the performance on the validation set, using the same grids of hyper-parameters used for TransE in [7] for the *Link Prediction* tasks. The results obtained by TransE^A, in comparison with state-of-the-art results reported in [7], are shown in Tab. 3. Despite the sensibly lower number of training iterations (we trained the model using AdaGrad for only 100 epochs, while in [7] TransE was trained using SGD for 1,000 epochs), TransE^A yields more accurate link prediction models (i.e. with lower Mean Rank and higher Hits@10 values) than every other prediction model in the comparison.

5.3 Evaluation of the Proposed Embedding Models

In this section, we evaluate the embedding models inspired by TransE and proposed in Sect. 3: ScalE, TransE$^+$ and ScalE$^+$. Model hyper-parameters were selected according to the performance on the validation set. In the following experiments, we considered a wider grid of hyper-parameters: in particular, we selected the embedding vector dimension k in $\{20, 50, 100, 200, 300\}$, the *dissimilarity* function $d(\mathbf{x}, \mathbf{y})$ in $\{\|\mathbf{x} - \mathbf{y}\|_1, \|\mathbf{x} - \mathbf{y}\|_2, -\mathbf{x}^T\mathbf{y}\}$, and the margin γ in $\{1, 2, 5, 10\}$. All models were trained using AdaGrad, with $\eta_a = 0.1$, for only 100 epochs. The reduced training time enabled us to experiment with a wider range of hyper-parameters in comparison with related works in literature [7].

Results are summarized in Tab. 4. We can see that, despite their very different geometric interpretations, all of the embedding models proposed in Sect. 3 achieve sensibly higher results in terms of HITS@10 in comparison with every other link prediction models outlined in Sect. 3. An explanation is that both TransE [7] and the proposed models TransE$^+$, ScalE and ScalE$^+$ have a *limited model capacity* (or complexity) in comparison with other models. For such a reason, they are *less prone to underfitting* for lack of training instances than other more expressive link prediction models, such as RESCAL [19], SME [12] and NTN [13].

In each experiment, the proposed models ScalE and ScalE$^+$ always improve over TransE in terms of HITS@10. We can clearly see that, by leveraging: (i) *Adaptive* learning rates, and (ii) The proposed embedding models ScalE and ScalE$^+$, we were able to achieve a record **95.5%** HITS@10 on WORDNET, and a **73.0%** HITS@10 on FREEBASE. These results are sensibly higher than state-of-the-art results reported in [7]. It is also remarkable that, during learning, the proposed method required a much lower learning time (100 epochs, approx. **30 minutes** on FREEBASE, on a single CPU) in comparison with [7] (1,000 epochs, and careful learning rate tuning).

A significantly lower training time – from days, as reported by [10], to minutes – can sensibly improve the applicability of embedding models for knowledge graphs in the Web of Data.

6 Conclusions

We focused on the problem of link prediction in Knowledge Graphs. Recently, *embedding models* like the TransE [7] model achieved new state-of-the-art link prediction results, while showing the potential to scale to very large KGs.

In this paper, we proposed a method for sensibly reducing the learning time in embedding models based on *adaptive learning rate selection*, and proposed a set of new models with interesting scalability properties. We extensively evaluated the proposed methods in several experiments on real world large datasets. Our results show a significant improvement over state-of-the-art link prediction methods, while significantly reducing the required training time by an order of magnitude.

The contributions in this paper sensibly improve both the effectiveness and applicability of embedding models on large and Web-scale KGs. Source code and datasets for reproducing the empirical evaluations discussed in this paper are available on-line [5].

References

1. K. D. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, “Freebase: a collaboratively created graph database for structuring human knowledge,” in *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008*, J. T. Wang, Ed. ACM, 2008, pp. 1247–1250.
2. G. A. Miller, “Wordnet: A lexical database for english,” *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.
3. 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*, S. A. Macskassy *et al.*, Eds. ACM, 2014, pp. 601–610.

[5] Source code and datasets: `https://github.com/pminervini/DeepKGC`

4. A. Bordes and E. Gabrilovich, "Constructing and mining web-scale knowledge graphs: KDD 2014 tutorial," in *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14*, 2014, p. 1967.
5. ——, "Constructing and mining web-scale knowledge graphs: WWW 2015 Tutorial," in *Proceedings of the 24th International Conference on World Wide Web, WWW 2015*, 2015, p. 1523.
6. Y. Bengio, A. C. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.
7. A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in Neural Information Processing Systems 26*, C. Burges *et al.*, Eds. Curran Associates, Inc., 2013, pp. 2787–2795.
8. A. Bordes, J. Weston, R. Collobert, and Y. Bengio, "Learning structured embeddings of knowledge bases," in *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011*, W. Burgard *et al.*, Eds. AAAI Press, 2011.
9. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, Eds., 2013, pp. 3111–3119.
10. K. Chang, W. Yih, B. Yang, and C. Meek, "Typed tensor decomposition of knowledge bases for relation extraction," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, A. Moschitti *et al.*, Eds. ACL, 2014, pp. 1568–1579.
11. Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," in *Predicting Structured Data*, G. Bakir *et al.*, Eds. MIT Press, 2006.
12. A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "A semantic matching energy function for learning with multi-relational data - application to word-sense disambiguation," *Machine Learning*, vol. 94, no. 2, pp. 233–259, 2014.
13. R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Advances in Neural Information Processing Systems 26*, C. Burges *et al.*, Eds. Curran Associates, Inc., 2013, pp. 926–934.
14. R. Jenatton, N. L. Roux, A. Bordes, and G. R. Obozinski, "A latent factor model for highly multi-relational data," in *Advances in Neural Information Processing Systems 25*, F. Pereira *et al.*, Eds. Curran Associates, Inc., 2012, pp. 3167–3175.
15. Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, C. E. Brodley *et al.*, Eds. AAAI Press, 2014, pp. 1112–1119.
16. D. E. Rumelhart, G. E. Hinton, and R. J. Wilson, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.
17. J. C. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.
18. J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. A. Ranzato, A. Senior, P. Tucker, K. Yang, Q. V. Le, and A. Y. Ng, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira *et al.*, Eds. Curran Associates, Inc., 2012, pp. 1223–1231.
19. M. Nickel, V. Tresp, and H. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, L. Getoor *et al.*, Eds. Omnipress, 2011, pp. 809–816.
20. T. Schaul, I. Antonoglou, and D. Silver, "Unit tests for stochastic optimization," in *International Conference on Learning Representations*, 2014.

Probabilistic Ontological Data Exchange with Bayesian Networks

Thomas Lukasiewicz[1], Maria Vanina Martinez[3],
Livia Predoiu[1 2], and Gerardo I. Simari[3]

[1] Department of Computer Science, University of Oxford, UK
[2] Department of Computer Science, Otto-von-Guericke Universität Magdeburg, Germany
[3] Dept. of Comp. Sci. and Eng., Univ. Nacional del Sur and CONICET, Argentina

Abstract. We study the problem of exchanging probabilistic data between ontology-based probabilistic databases. The probabilities of the probabilistic source databases are compactly encoded via Boolean formulas with the variables adhering to the dependencies imposed by a Bayesian network, which are closely related to the management of provenance. For the ontologies and the ontology mappings, we consider different kinds of existential rules from the Datalog+/– family. We provide a complete picture of the computational complexity of the problem of deciding whether there exists a probabilistic (universal) solution for a given probabilistic source database relative to a (probabilistic) ontological data exchange problem. We also analyze the complexity of answering UCQs (unions of conjunctive queries) in this framework.

1 Introduction

Large volumes of uncertain data are best modeled, stored, and processed in probabilistic databases [22]. Enriching databases with terminological knowledge encoded in ontologies has recently gained increasing importance in the form of ontology-based data access (OBDA) [21]. A crucial problem in OBDA is to integrate and exchange knowledge. Not only in the context of OBDA, but also in the area of the Semantic Web, there are distributed ontologies that we may have to map and integrate to enable query answering over them. Here, apart from the uncertainty attached to source databases, there may also be uncertainty regarding the ontology mappings establishing the proper correspondence between items in the source ontology and items in the target ontology. This especially happens when the mappings are created automatically.

Data exchange [11] is an important theoretical framework used for studying data-interoperability tasks that require data to be transferred from existing databases to a target database that comes with its own (independently created) schema and schema constraints. The expressivity of the data exchange framework goes beyond the classical data integration framework [17]. For the translation, schema mappings are used, which are declarative specifications that describe the relationship between two database schemas. In classical data exchange, we have a source database, a target database, a deterministic mapping, and deterministic target dependencies. Recently, a framework for probabilistic data exchange [10] has been proposed where the classical data exchange framework based on weakly acyclic existential rules has been extended to consider a probabilistic source database and a probabilistic source-to-target mapping.

In this paper, we study an expressive extension of the probabilistic data exchange framework in [10], where the source and the target are ontological knowledge bases, each consisting of a probabilistic database and a deterministic ontology describing terminological knowledge about the data stored in the database. The two ontologies and the mapping between them are expressed via existential rules. Our extension of the data exchange framework is strongly related to exchanging data between incomplete databases, as proposed in [3], which considers an incomplete deterministic source database in the data exchange problem. However, in that work, the databases are deterministic, and the mappings and the target database constraints are full existential rules only. In our complexity analysis in this paper, we consider a host of different classes of existential rules, including some subclasses of full existential rules. In addition, our source is a probabilistic database relative to an underlying ontology.

Our work in this paper is also related to the recently proposed knowledge base exchange framework [2, 1], which allows knowledge to be exchanged between deterministic *DL-Lite*$_{RDFS}$ and *DL-Lite*$_{\mathcal{R}}$ ontologies. In this paper, besides considering probabilistic source databases, we are also using more expressive ontology languages, since already linear existential rules from the Datalog+/– family are strictly more expressive than the description logics (DLs) *DL-Lite*$_X$ of the *DL-Lite* family [9] as well as their extensions with n-ary relations *DLR-Lite*$_X$. Guarded existential rules are sufficiently expressive to model the tractable DL $\mathcal{EL}$ [4, 5] (and $\mathcal{ELI}^f$ [16]). Note that existential rules are also known as tuple-generating dependencies (TGDs) and Datalog+/– rules [7].

The main contributions of this paper are summarized as follows.

– We introduce deterministic and probabilistic ontological data exchange problems, where probabilistic knowledge is exchanged between two Bayesian network-based probabilistic databases relative to their underlying deterministic ontologies, and the deterministic and probabilistic mapping between the two ontologies is defined via deterministic and probabilistic existential mapping rules, respectively.

– We provide an in-depth analysis of the data and combined complexity of deciding the existence of probabilistic (universal) solutions and obtain a (fairly) complete picture of the data complexity, general combined complexity, bounded-arity (*ba*) combined, and fixed-program combined (*fp*) complexity for the main sublanguages of the Datalog+/– family. We also delineate some tractable special cases, and provide complexity results for exact UCQ (union of conjunctive queries) answering.

– For the complexity analysis, we consider a compact encoding of probabilistic source databases and mappings, which is used in the area of both incomplete and probabilistic databases, and also known as data provenance or data lineage [14, 12, 13, 22]. Here, we consider data provenance for probabilistic data that is structured according to an underlying Bayesian network.

2 Preliminaries

We assume infinite sets of *constants* $\mathbf{C}$, *(labeled) nulls* $\mathbf{N}$, and regular *variables* $\mathbf{V}$. A *term* t is a constant, null, or variable. An *atom* has the form $p(t_1, \ldots, t_n)$, where p is an n-ary predicate, and $t_1, \ldots, t_n$ are terms. Conjunctions of atoms are often identified with the sets of their atoms. An *instance* I is a (possibly infinite) set of atoms $p(\mathbf{t})$, where $\mathbf{t}$ is a tuple of constants and nulls. A *database* D is a finite instance that contains only constants. A *homomorphism* is a substitution $h : \mathbf{C} \cup \mathbf{N} \cup \mathbf{V} \rightarrow \mathbf{C} \cup \mathbf{N} \cup \mathbf{V}$ that is the identity on $\mathbf{C}$. We assume familiarity with *conjunctive queries (CQs)*. The answer to a CQ q over an instance I is denoted $q(I)$. A Boolean CQ (BCQ) q evaluates to *true* over I, denoted $I \models q$, if $q(I) \neq \varnothing$.

A *tuple-generating dependency (TGD)* σ is a first-order formula $\forall \mathbf{X}\, \varphi(\mathbf{X}) \rightarrow \exists \mathbf{Y}\, p(\mathbf{X}, \mathbf{Y})$, where $\mathbf{X} \cup \mathbf{Y} \subseteq \mathbf{V}$, $\varphi(\mathbf{X})$ is a conjunction of atoms, and $p(\mathbf{X}, \mathbf{Y})$ is an atom. We call $\varphi(\mathbf{X})$ the *body* of σ, denoted $body(\sigma)$, and $p(\mathbf{X}, \mathbf{Y})$ the *head* of σ, denoted $head(\sigma)$. We consider only TGDs with a single atom in the head, but our results can be extended to TGDs with a conjunction of atoms in the head. An instance I *satisfies* σ, written $I \models \sigma$, if the following holds: whenever there exists a homomorphism h such that $h(\varphi(\mathbf{X})) \subseteq I$, then there exists $h' \supseteq h|_{\mathbf{X}}$, where $h|_{\mathbf{X}}$ is the restriction of h to $\mathbf{X}$, such that $h'(p(\mathbf{X}, \mathbf{Y})) \in I$. A *negative constraint (NC)* ν is a first-order formula $\forall \mathbf{X}\, \varphi(\mathbf{X}) \rightarrow \bot$, where $\mathbf{X} \subseteq \mathbf{V}$, $\varphi(\mathbf{X})$ is a conjunction of atoms, called the *body* of ν, denoted $body(\nu)$, and $\bot$ denotes the truth constant *false*. An instance I *satisfies* ν, denoted $I \models \nu$, if there is no homomorphism h such that $h(\varphi(\mathbf{X})) \subseteq I$. Given a set Σ of TGDs and NCs, I *satisfies* Σ, denoted $I \models \Sigma$, if I satisfies each TGD and NC of Σ. For brevity, we omit the universal quantifiers in front of TGDs and NCs.

Given a database D and a set Σ of TGDs and NCs, the answers that we consider are those that are true in *all* models of D and Σ. Formally, the *models* of D and Σ, denoted $mods(D, \Sigma)$, is the set of instances $\{I \mid I \supseteq D \text{ and } I \models \Sigma\}$. The *answer* to a CQ q relative to D and Σ is defined as the set of tuples $ans(q, D, \Sigma) = \bigcap_{I \in mods(D,\Sigma)} \{\mathbf{t} \mid \mathbf{t} \in q(I)\}$. The answer to a BCQ q is *true*, denoted $D \cup \Sigma \models q$, if $ans(q, D, \Sigma) \neq \varnothing$. The problem of *CQ answering* is defined as follows: given a database D, a set Σ of TGDs and NCs, a CQ q, and a tuple of constants $\mathbf{t}$, decide whether $\mathbf{t} \in ans(q, D, \Sigma)$. Following Vardi's taxonomy [23], the *combined complexity* of BCQ answering is calculated by considering all the components, i.e., the database, the set of dependencies, and the query, as part of the input. The *bounded-arity combined complexity* (or simply *ba-combined complexity*) is calculated by assuming that the arity of the underlying schema is bounded by an integer constant. Notice that in the context of description logics (DLs), whenever we refer to the combined complexity in fact we refer to the *ba*-combined complexity since, by definition, the arity of the underlying schema is at most two. The *fixed-program combined complexity* (or simply *fp-combined complexity*) is calculated by considering the set of TGDs and NCs as fixed.

3 Ontological Data Exchange

In this section, we define the notions of *deterministic* and *probabilistic ontological data exchange*. The source (resp., target) of the deterministic/probabilistic ontological data exchange problems that we consider in this paper is a probabilistic database (resp., probabilistic instance), each relative to a deterministic ontology. Here, a *probabilistic database* (resp., *probabilistic instance*) over a schema $\mathbf{S}$ is a probability space $Pr = (\mathcal{I}, \mu)$ such that $\mathcal{I}$ is the set of all (possibly infinitely many) databases (resp., instances) over $\mathbf{S}$, and $\mu \colon \mathcal{I} \rightarrow [0, 1]$ is a function that satisfies $\sum_{I \in \mathcal{I}} \mu(I) = 1$.

3.1 Deterministic Ontological Data Exchange

Ontological data exchange formalizes data exchange from a probabilistic database relative to a source ontology Σ_s (consisting of TGDs and NCs) over a schema $\mathbf{S}$ to a probabilistic target instance Pr_t relative to a target ontology Σ_t (consisting of a set of TGDs and NCs) over a schema $\mathbf{T}$ via a (source-to-target) mapping (also consisting of a set of TGDs and NCs). More specifically, an *ontological data exchange (ODE) problem* $\mathcal{M}$= ($\mathbf{S}$, $\mathbf{T}$, Σ_s, Σ_t, Σ_{st}) consists of (i) a source schema $\mathbf{S}$, (ii) a target schema $\mathbf{T}$ disjoint from $\mathbf{S}$, (iii) a finite set Σ_s of TGDs and NCs over $\mathbf{S}$ (called *source ontology*), (iv) a finite set Σ_t of TGDs and NCs over $\mathbf{T}$ (called *target ontology*), and (v) a finite set Σ_{st} of TGDs and NCs σ over $\mathbf{S} \cup \mathbf{T}$ (called *(source-to-target) mapping*) such that *body*(σ) and *head*(σ) are defined over $\mathbf{S} \cup \mathbf{T}$ and $\mathbf{T}$, respectively.

Ontological data exchange with deterministic databases is based on defining a target instance J over $\mathbf{T}$ as being a *solution* for a deterministic source database I over $\mathbf{S}$ relative to an ODE problem $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma_s, \Sigma_t, \Sigma_{st})$, if $(I \cup J) \models \Sigma_s \cup \Sigma_t \cup \Sigma_{st}$. We denote by $Sol_{\mathcal{M}}$ the set of all such pairs (I, J). Among the possible deterministic solutions J to a deterministic source database I relative to $\mathcal{M}$ in $Sol_{\mathcal{M}}$, we prefer *universal* solutions, which are the most general ones carrying only the necessary information for data exchange, i.e., those that transfer only the source database along with the relevant implicit derivations via Σ_s to the target ontology. A universal solution can be homomorphically mapped to all other solutions leaving the constants unchanged. Hence, a deterministic target instance J over $\mathbf{S}$ is a *universal solution* for a deterministic source database I over $\mathbf{T}$ relative to a schema mapping $\mathcal{M}$, if (i) J is a solution, and (ii) for each solution J' for I relative to $\mathcal{M}$, there is a homomorphism $h\colon J \rightarrow J'$. We denote by $USol_{\mathcal{M}}$ $(\subseteq Sol_{\mathcal{M}})$ the set of all pairs (I, J) of deterministic source databases I and target instances J such that J is a universal solution for I relative to $\mathcal{M}$.

When considering probabilistic databases and instances, a joint probability space *Pr* over the solution relation $Sol_{\mathcal{M}}$ and the universal solution relation $USol_{\mathcal{M}}$ must exist. More specifically, a probabilistic target instance $Pr_t = (\mathcal{J}, \mu_t)$ is a *probabilistic solution* (resp., *probabilistic universal solution*) for a probabilistic source database $Pr_s = (\mathcal{I}, \mu_s)$ relative to an ODE problem $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma_s, \Sigma_t, \Sigma_{st})$, if there exists a probability space $Pr = (\mathcal{I} \times \mathcal{J}, \mu)$ such that (i) the left and right marginals of *Pr* are Pr_s and Pr_t, respectively, i.e., (i.a) $\mu_s(I) = \sum_{J \in \mathcal{J}} \mu(I, J)$ for all $I \in \mathcal{I}$, (i.b) $\mu_t(J) = \sum_{I \in \mathcal{I}} \mu(I, J)$ for all $J \in \mathcal{J}$; and (ii) $\mu(I, J) = 0$ for all $(I, J) \notin Sol_{\mathcal{M}}$ (resp., $(I, J) \notin USol_{\mathcal{M}}$). Note that this intuitively says that all non-solutions (I, J) have probability zero and the existence of a solution does not exclude that some source databases with probability zero have no corresponding target instance.

Example 1. An ontological data exchange (ODE) problem $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma_s, \Sigma_t, \Sigma_{st})$ is given by the source schema $\mathbf{S} = \{Researcher/2, ResearchArea/2, Publication/3\}$ (the number after each predicate denotes its arity), the target schema $\mathbf{T} = \{UResearchArea/3, Lecturer/2\}$, the source ontology $\Sigma_s = \{\sigma_s, \nu_s\}$, the target ontology $\Sigma_t = \{\sigma_t, \nu_t\}$, and the mapping $\Sigma_{st} = \{\sigma_{st}, \nu_m\}$, where:

$$\begin{aligned}
\sigma_s &: \textit{Publication}(\mathrm{X}, \mathrm{Y}, \mathrm{Z}) \rightarrow \textit{ResearchArea}(\mathrm{X}, \mathrm{Y}),\\
\nu_s &: \textit{Researcher}(\mathrm{X}, \mathrm{Y}) \wedge \textit{ResearchArea}(\mathrm{X}, \mathrm{Y}) \rightarrow \perp,\\
\sigma_t &: \textit{UResearchArea}(\mathrm{U}, \mathrm{D}, \mathrm{T}) \rightarrow \exists \mathrm{Z}\, \textit{Lecturer}(\mathrm{T}, \mathrm{Z}),\\
\nu_t &: \textit{Lecturer}(\mathrm{X}, \mathrm{Y}) \wedge \textit{Lecturer}(\mathrm{Y}, \mathrm{X}) \rightarrow \perp,
\end{aligned}$$

Possible source database facts	
r$_a$	*Researcher*(Alice, UnivOx)
r$_p$	*Researcher*(Paul, UnivOx)
p$_{aml}$	*Publication*(Alice, ML, JMLR)
p$_{adb}$	*Publication*(Alice, DB, TODS)
p$_{pdb}$	*Publication*(Paul, DB, TODS)
p$_{pai}$	*Publication*(Paul, AI, AIJ)

Derived source database facts	
a$_{aml}$	*ResearchArea*(Alice, ML)
a$_{adb}$	*ResearchArea*(Alice, DB)
a$_{pdb}$	*ResearchArea*(Paul, DB)
a$_{pai}$	*ResearchArea*(Paul, AI)

Probabilistic source database $Pr_s = (I, \mu_s)$	
I_1 = {r$_a$,r$_p$,p$_{aml}$,p$_{pdb}$,a$_{aml}$,a$_{pdb}$}	0.5
I_2 = {r$_a$,r$_p$,p$_{aml}$,p$_{pai}$,a$_{aml}$,a$_{pai}$}	0.2
I_3 = {r$_a$,r$_p$,p$_{adb}$,p$_{pai}$,a$_{adb}$,a$_{pai}$}	0.15
I_4 = {r$_a$,r$_p$,p$_{adb}$,p$_{pdb}$,a$_{adb}$,a$_{pdb}$}	0.075
I_5 = {r$_a$,p$_{adb}$,a$_{adb}$}	0.075

Possible target instance facts	
u$_{ml}$	*UResearchArea*(UnivOx, N_1, ML)
u$_{ai}$	*UResearchArea*(UnivOx, N_2, AI)
u$_{db}$	*UResearchArea*(UnivOx, N_3, DB)
l$_{ml}$	*Lecturer*(ML, N_4)
l$_{ai}$	*Lecturer*(AI, N_5)
l$_{db}$	*Lecturer*(DB, N_6)

Probabilistic target instance $Pr_{t_1} = (J_1, \mu_{t_1})$	
J_1 = {u$_{ml}$,u$_{db}$,l$_{ml}$,l$_{db}$}	0.5
J_2 = {u$_{ml}$,u$_{ai}$,l$_{ml}$,l$_{ai}$}	0.2
J_3 = {u$_{ai}$,u$_{db}$,l$_{ai}$,l$_{db}$}	0.15
J_4 = {u$_{db}$,l$_{db}$}	0.15

Probabilistic target instance $Pr_{t_2} = (J_2, \mu_{t_2})$	
J_5 = {u$_{ml}$,u$_{db}$,l$_{ml}$,l$_{db}$}	0.55
J_6 = {u$_{ml}$,u$_{ai}$,l$_{ml}$,l$_{ai}$}	0.1
J_7 = {u$_{ml}$,u$_{ai}$,u$_{db}$,l$_{ml}$,l$_{ai}$,l$_{db}$}	0.35

Table 1. Probabilistic source database and two probabilistic target instances for Example 1 ($N_1, \ldots, N_6$ are nulls); both are probabilistic solutions, but only Pr_{t_1} is universal.

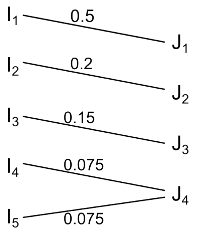


Fig. 1. Probabilistic universal solution Pr_{t_1}.

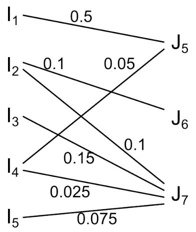


Fig. 2. Probabilistic solution Pr_{t_2}.

$$\sigma_{st} : \textit{ResearchArea}(\mathrm{N}, \mathrm{T}) \wedge \textit{Researcher}(\mathrm{N}, \mathrm{U}) \rightarrow \exists \mathrm{D}\ \textit{UResearchArea}(\mathrm{U}, \mathrm{D}, \mathrm{T}),$$
$$\nu_{st} : \textit{ResearchArea}(\mathrm{N}, \mathrm{T}) \wedge \textit{UResearchArea}(\mathrm{U}, \mathrm{T}, \mathrm{N}) \rightarrow \bot.$$

Given the probabilistic source database in Table 1, two probabilistic instances $Pr_{t_1} = (\mathcal{J}_1, \mu_{t_1})$ and $Pr_{t_2} = (\mathcal{J}_2, \mu_{t_2})$ that are probabilistic solutions are shown in Table 1. Note that only Pr_{t_1} is also a probabilistic universal solution. Note also that Figures 1 and 2 show the probability spaces over Pr_{t_1} and Pr_{t_2}, respectively. ■

Query answering in ontological data exchange is performed over the target ontology and is generalized from deterministic data exchange. A *union of conjunctive queries* (or *UCQ*) has the form $q(\mathbf{X}) = \bigvee_{i=1}^{k} \exists \mathbf{Y}_i\ \Phi_i(\mathbf{X}, \mathbf{Y}_i, \mathbf{C}_i)$, where each $\exists \mathbf{Y}_i\ \Phi_i(\mathbf{X}, \mathbf{Y}_i, \mathbf{C}_i)$ with $i \in \{1, \ldots, k\}$ is a CQ with exactly the variables $\mathbf{X}$ and $\mathbf{Y}_i$, and the constants $\mathbf{C}_i$. Given an ODE problem $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma_s, \Sigma_t, \Sigma_{st})$, probabilistic source database $Pr_s = (\mathcal{I}, \mu_s)$, UCQ $q(\mathbf{X}) = \bigvee_{i=1}^{k} \exists \mathbf{Y}_i\ \Phi_i(\mathbf{X}, \mathbf{Y}_i, \mathbf{C}_i)$, and tuple $\mathbf{t}$ (a ground instance of $\mathbf{X}$ in q) over $\mathbf{C}$, the *confidence* of $\mathbf{t}$ relative to q, denoted $\mathit{conf}_q(\mathbf{t})$, in Pr_s relative to $\mathcal{M}$ is the infimum of $Pr_t(q(\mathbf{t}))$ subject to all probabilistic solutions Pr_t for Pr_s relative to $\mathcal{M}$. Here, $Pr_t(q(\mathbf{t}))$ for $Pr_t = (\mathcal{J}, \mu_t)$ is the sum of all $\mu_t(J)$ such that $q(\mathbf{t})$ evaluates to true in the instance $J \in \mathcal{J}$ (i.e., some BCQ $\exists \mathbf{Y}_i\, \Phi_i(\mathbf{t}, \mathbf{Y}_i, \mathbf{C}_i)$ with $i \in \{1, \ldots, k\}$ evaluates to true in J).

Example 2. Consider again the setting of Example 1, and let q be a UCQ of a student who wants to know whether she can study either machine learning or artificial intelligence at the University of Oxford: $q() = \exists \mathrm{X}, \mathrm{Z}(\mathit{Lecturer}(\mathit{AI}, \mathrm{X}) \wedge \mathit{UResearchArea}(\mathit{UnivOx}, \mathrm{Z}, \mathrm{AI})) \vee \exists \mathrm{X}, \mathrm{Z}(\mathit{Lecturer}(\mathit{ML}, \mathrm{X}) \wedge \mathit{UResearchArea}(\mathit{UnivOx}, \mathrm{Z}, \mathit{ML}))$. Then, q yields the probabilities 0.85 and 1 on Pr_{t_1} and Pr_{t_2}, respectively. ■

3.2 Probabilistic Ontological Data Exchange

Probabilistic ontological data exchange extends deterministic ontological data exchange by turning the deterministic source-to-target mapping into a probabilistic source-to-target mapping, i.e., we have a probability distribution over the set of all subsets of Σ_{st}. More specifically, a *probabilistic ontological data exchange (PODE) problem* $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma_s, \Sigma_t, \Sigma_{st}, \mu_{st})$ consists of (i) a source schema $\mathbf{S}$, (ii) a target schema $\mathbf{T}$ disjoint from $\mathbf{S}$, (iii) a finite set Σ_s of TGDs and NCs over $\mathbf{S}$ (called *source ontology*), (iv) a finite set Σ_t of TGDs and NCs over $\mathbf{T}$ (called *target ontology*), (v) a finite set Σ_{st} of TGDs and NCs σ over $\mathbf{S} \cup \mathbf{T}$, and (vi) a function $\mu_{st}\colon 2^{\Sigma_{st}} \to [0, 1]$ such that $\sum_{\Sigma' \subseteq \Sigma_{st}} \mu_{st}(\Sigma') = 1$ (called *probabilistic (source-to-target) mapping*).

A probabilistic target instance $\mathit{Pr}_t = (\mathcal{J}, \mu_t)$ is a *probabilistic solution* (resp., *probabilistic universal solution*) for a probabilistic source database $\mathit{Pr}_s = (\mathcal{I}, \mu_s)$ relative to a PODE problem $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma_s, \Sigma_t, \Sigma_{st}, \mu_{st})$, if there exists a probability space $\mathit{Pr} =$ $= (\mathcal{I} \times \mathcal{J} \times 2^{\Sigma_{st}}, \mu)$ such that: (i) the three marginals of μ are μ_s, μ_t, and μ_{st}, such that: (i.a) $\mu_s(I) = \sum_{J \in \mathcal{J},\, \Sigma' \subseteq \Sigma_{st}} \mu(I, J, \Sigma')$ for all $I \in \mathcal{I}$, (i.b) $\mu_t(J) = \sum_{I \in \mathcal{I},\, \Sigma' \subseteq \Sigma_{st}} \mu(I, J, \Sigma')$ for all $J \in \mathcal{J}$, and (i.c) $\mu_{st}(\Sigma') = \sum_{I \in \mathcal{I},\, J \in \mathcal{J}} \mu(I, J, \Sigma')$ for all $\Sigma' \subseteq \Sigma_{st}$; and (ii) $\mu(I, J, \Sigma') = 0$ for all $(I, J) \notin \mathit{Sol}_{(\mathbf{S}, \mathbf{T}, \Sigma')}$ (resp., $(I, J) \notin \mathit{USol}_{(\mathbf{S}, \mathbf{T}, \Sigma')}$).

Using probabilistic (universal) solutions for probabilistic source databases relative to PODE problems, the semantics of UCQs is lifted to PODE problems as follows. Given a PODE problem $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma_s, \Sigma_t, \Sigma_{st}, \mu_{st})$, a probabilistic source database $\mathit{Pr}_s = (\mathcal{I}, \mu_s)$, a UCQ $q(\mathbf{X}) = \bigvee_{i=1}^{k} \exists \mathbf{Y}_i\, \Phi_i(\mathbf{X}, \mathbf{Y}_i, \mathbf{C}_i)$, and a tuple $\mathbf{t}$ (a ground instance of $\mathbf{X}$ in q) over $\mathbf{C}$, the *confidence* of $\mathbf{t}$ relative to q, denoted $\mathit{conf}_q(\mathbf{t})$, in Pr_s relative to $\mathcal{M}$ is the infimum of $\mathit{Pr}_t(q(\mathbf{t}))$ subject to all probabilistic solutions Pr_t for Pr_s relative to $\mathcal{M}$. Here, $\mathit{Pr}_t(q(\mathbf{t}))$ for $\mathit{Pr}_t = (\mathcal{J}, \mu_t)$ is the sum of all $\mu_t(J)$ such that $q(\mathbf{t})$ evaluates to true in the instance $J \in \mathcal{J}$.

3.3 Compact Encoding

We use a compact encoding of both probabilistic databases and probabilistic mappings, which is based on annotating facts, TGDs, and NCs by probabilistic events in a Bayesian network, rather than explicitly specifying the whole probability space.

We first define annotations and annotated atoms. Let $e_1, \ldots, e_n$ be $n \geq 1$ *elementary events*. A *world* w is a conjunction $\ell_1 \wedge \cdots \wedge \ell_n$, where each ℓ_i, $i \in \{1, \ldots, n\}$, is either the elementary event e_i or its negation $\neg e_i$. An *annotation* λ is any Boolean combination of elementary events (i.e., all elementary events are annotations, and if λ_1 and λ_2

	Possible source database facts	Annotation
r_a	*Researcher*(Alice, UnivOx)	true
r_p	*Researcher*(Paul, UnivOx)	$e_1 \vee e_2 \vee e_3 \vee e_4$
p_{aml}	*Publication*(Alice, ML, JMLR)	$e_1 \vee e_2$
p_{adb}	*Publication*(Alice, DB, TODS)	$\neg e_1 \wedge \neg e_2$
p_{pdb}	*Publication*(Paul, DB, TODS)	$e_1 \vee (\neg e_2 \wedge \neg e_3 \wedge e_4)$
p_{pai}	*Publication*(Paul, AI, AIJ)	$(\neg e_1 \wedge e_2) \vee (\neg e_1 \wedge e_3)$

Table 2. Annotation encoding of the probabilistic source database in Table 1.

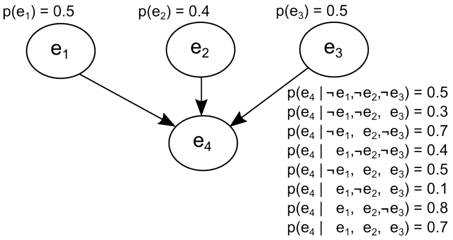


Table 3. Bayesian network over the elementary events.

are annotations, then also $\neg\lambda_1$ and $\lambda_1 \wedge \lambda_2$). An *annotated atom* has the form $a\colon \lambda$, where a is an atom, and λ is an annotation.

The compact encoding of probabilistic databases can then be defined as follows. Note that this encoding is also underlying our complexity analysis in Section 4. A set $\mathbf{A}$ of annotated atoms along with a probability $\mu(w) \in [0,1]$ for every world w *compactly encodes a probabilistic database* $Pr = (\mathcal{I}, \mu)$ whenever: (i) the probability μ of every annotation λ is the sum of the probabilities of all worlds in which λ is true, and (ii) the probability μ of every subset-maximal database $\{a_1, \ldots, a_m\} \in \mathcal{I}$ [4] such that $\{a_1\colon \lambda_1, \ldots, a_m\colon \lambda_m\} \subseteq \mathbf{A}$ for some annotations $\lambda_1, \ldots, \lambda_m$ is the probability μ of $\lambda_1 \wedge \cdots \wedge \lambda_m$ (and the probability μ of every other database in $\mathcal{I}$ is 0).

We assume that the probability distributions for the underlying events are given by a Bayesian network, which is usually used for compactly specifying a joint probability space, encoding also a certain causal structure between the variables. The following example in Tables 2 and 3 illustrates the compact encoding of probabilistic source databases via Boolean annotations relative to an underlying Bayesian network.

If the mapping is probabilistic as well, then we use two disjoint sets of elementary events, one for encoding the probabilistic source database and the other one for the mapping. In this way, the probabilistic source database is independent from the probabilistic mapping. We now define the compact encoding of probabilistic mappings. An *annotated* TGD (resp., NC) has the form $\sigma\colon \lambda$, where σ is a TGD (resp., NC), and λ is an annotation. A set Σ of annotated TGDs and NCs $\sigma\colon \lambda$ with $\sigma \in \Sigma_{st}$ along with a probability $\mu(w) \in [0,1]$ for every world w *compactly encodes a probabilistic mappings* $\mu_{st}\colon 2^{\Sigma_{st}} \to [0,1]$ whenever (i) the probability μ of every annotation λ is the sum of the probabilities of all worlds in which λ is true, and (ii) the probability μ_{st} of every

[4] That is, we do not consider subsets of the databases here.

subset-maximal $\{\sigma_1, \ldots, \sigma_k\} \subseteq \Sigma_{st}$ such that $\{\sigma_1 \colon \lambda_1, \ldots, \sigma_k \colon \lambda_k\} \subseteq \Sigma$ for some annotations $\lambda_1, \ldots, \lambda_k$ is the probability μ of $\lambda_1 \wedge \cdots \wedge \lambda_k$ (and the probability μ_{st} of every other subset of Σ_{st} is 0).

3.4 Computational Problems

We consider the following computational problems:

Existence of a solution (resp., universal solution): Given an ODE or a PODE problem $\mathcal{M}$ and a probabilistic source database Pr_s, decide whether there exists a probabilistic (resp., probabilistic universal) solution for Pr_s relative to $\mathcal{M}$.

Answering UCQs: Given an ODE or a PODE problem $\mathcal{M}$, a probabilistic source database Pr_s, a UCQ $q(\mathbf{X})$, and a tuple $\mathbf{t}$ over $\mathbf{C}$, compute $conf_Q(\mathbf{t})$ in Pr_s w.r.t. $\mathcal{M}$.

4 Computational Complexity

We now analyze the computational complexity of deciding the existence of a (universal) probabilistic solution for deterministic and probabilistic ontological data exchange problems. We also delineate some tractable special cases, and we provide some complexity results for exact UCQ answering for ODE and PODE problems.

We assume some elementary background in complexity theory [15, 20]. We now briefly recall the complexity classes that we encounter in our complexity results. The complexity classes PSPACE (resp., P, EXP, 2EXP) contain all decision problems that can be solved in polynomial space (resp., polynomial, exponential, double exponential time) on a deterministic Turing machine, while the complexity classes NP and NEXP contain all decision problems that can be solved in polynomial and exponential time on a nondeterministic Turing machine, respectively; coNP and coNEXP are their complementary classes, where "Yes" and "No" instances are interchanged. The complexity class AC^0 is the class of all languages that are decidable by uniform families of Boolean circuits of polynomial size and constant depth. The inclusion relationships among the above (decision) complexity classes (all currently believed to be strict) are as follows:

$$\mathrm{AC}^0 \subseteq \mathrm{P} \subseteq \mathrm{NP}, \mathrm{coNP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP} \subseteq \mathrm{NEXP}, \mathrm{coNEXP} \subseteq 2\mathrm{EXP}$$

The (function) complexity class $\#\mathrm{P}$ is the set of all functions that are computable by a polynomial-time nondeterministic Turing machine whose output for a given input string I is the number of accepting computations for I.

4.1 Decidability Paradigms

The main (syntactic) conditions on TGDs that guarantee the decidability of CQ answering are guardedness [6], stickiness [8], and acyclicity. Each one of these conditions has its "weak" counterpart: weak guardedness [6], weak stickiness [8], and weak acyclicity [11], respectively.

A TGD σ is *guarded* if there exists an atom in its body that contains (or "guards") all the body variables of σ. The class of guarded TGDs, denoted G, is defined as the

	Data	**Comb.**	ba**-comb.**	fp**-comb.**
L, LF, AF	in AC^0	PSPACE	NP	NP
G	P	2EXP	EXP	NP
WG	EXP	2EXP	EXP	EXP
S, SF	in AC^0	EXP	NP	NP
F, GF	P	EXP	NP	NP
A	in AC^0	NEXP	NEXP	NP
WS, WA	P	2EXP	2EXP	NP

Fig. 3. Complexity of BCQ answering [18]. All entries except for "in AC^0" are completeness ones, where hardness in all cases holds even for ground atomic BCQs.

	Data	**Comb.**	ba**-comb.**	fp**-comb.**
L, LF, AF	coNP	PSPACE	coNP	coNP
G	coNP	2EXP	EXP	coNP
WG	EXP	2EXP	EXP	EXP
S, SF	coNP	EXP	coNP	coNP
F, GF	coNP	EXP	coNP	coNP
A	coNP	coNEXP	coNEXP	coNP
WS, WA	coNP	2EXP	2EXP	coNP

Fig. 4. Complexity of existence of a probabilistic (universal) solution (for both deterministic and probabilistic ODE). All entries are completeness results.

family of all possible sets of guarded TGDs. A key subclass of guarded TGDs are the so-called linear TGDs with just one body atom (which is automatically a guard), and the corresponding class is denoted L. *Weakly guarded* TGDs extend guarded TGDs by requiring only "harmful" body variables to appear in the guard, and the associated class is denoted WG. It is easy to verify that L $\subset$ G $\subset$ WG.

Stickiness is inherently different from guardedness, and its central property can be described as follows: variables that appear more than once in a body (i.e., join variables) are always propagated (or "stick") to the inferred atoms. A set of TGDs that enjoys the above property is called *sticky*, and the corresponding class is denoted S. Weak stickiness is a relaxation of stickiness where only "harmful" variables are taken into account. A set of TGDs which enjoys weak stickiness is *weakly sticky*, and the associated class is denoted WS. Observe that S $\subset$ WS.

A set Σ of TGDs is *acyclic* if its predicate graph is acyclic, and the underlying class is denoted A. In fact, an acyclic set of TGDs can be seen as a nonrecursive set of TGDs. We say Σ is *weakly acyclic* if its dependency graph enjoys a certain acyclicity condition, which actually guarantees the existence of a finite canonical model; the associated class is denoted WA. Clearly, A $\subset$ WA.

Another key fragment of TGDs, which deserves our attention, are the so-called *full* TGDs, i.e., TGDs without existentially quantified variables, and the corresponding class is denoted F. If we further assume that full TGDs enjoy linearity, guardedness, stickiness, or acyclicity, then we obtain the classes LF, GF, SF, and AF, respectively.

4.2 Overview of Complexity Results

Our complexity results for deciding the existence of a probabilistic (universal) solution for both ODE and PODE problems with annotations over events relative to an underlying Bayesian network are summarized in Fig. 4 for all classes of existential rules discussed above in the data, combined, *ba*-combined, and *fp*-combined complexity (all entries are completeness results). For L, LF, AF, S, SF, and A in the data complexity, we obtain tractability when the underlying Bayesian network is a polytree. For all other cases, hardness holds even when the underlying Bayesian network is a polytree. Finally, for all classes of existential rules discussed above except for WG, answering UCQs for both ODE and PODE problems is in $\#\text{P}$ in the data complexity.

4.3 Deterministic Ontological Data Exchange

The first result shows that deciding whether there exists a probabilistic (or probabilistic universal) solution for a probabilistic source database relative to an ODE problem is complete for $\mathcal{C}$ (resp., co$\mathcal{C}$), if BCQ answering for the involved sets of TGDs and NCs is complete for a deterministic (resp., nondeterministic) complexity class $\mathcal{C} \supseteq$ PSPACE (resp., $\mathcal{C} \supseteq$ NP), and hardness holds even for ground atomic BCQs. As a corollary, by the complexity of BCQ answering with TGDs and NCs in Figure 3 [18], we immediately obtain the complexity results shown in Figure 4 for deciding the existence of a probabilistic (universal) solution (in deterministic ontological data exchange) in the combined, ba-combined, and *fp*-combined complexity, and for the class WG of TGDs and NCs in the data complexity. The hardness results hold even when the underlying Bayesian network is a polytree.

Theorem 1. *Given a probabilistic source database Pr_s relative to a source ontology Σ_s and an ODE problem $\mathcal{M} = (\boldsymbol{S}, \boldsymbol{T}, \Sigma_s, \Sigma_t, \Sigma_{st})$ such that $\Sigma_s \cup \Sigma_t \cup \Sigma_{st}$ belongs to a class of TGDs and NCs for which BCQ answering is complete for a deterministic (resp., nondeterministic) complexity class $\mathcal{C} \supseteq$ PSPACE (resp., $\mathcal{C} \supseteq$ NP), and hardness holds even for ground atomic BCQs, deciding the existence of a probabilistic (universal) solution for Pr_s relative to Σ_s and $\mathcal{M}$ is complete for $\mathcal{C}$ (resp., co$\mathcal{C}$). Hardness holds even when the underlying Bayesian network is a polytree.*

The following result shows that deciding whether there exists a probabilistic (universal) solution for a probabilistic source database relative to an ODE problem is complete for coNP in the data complexity, for all classes of sets of TGDs and NCs considered in this paper, except for WG. Hardness for coNP for the classes G, F, GF, WS, and WA holds even when the underlying Bayesian network is a polytree.

Theorem 2. *Given a probabilistic source database Pr_s relative to a source ontology Σ_s and an ODE problem $\mathcal{M} = (\boldsymbol{S}, \boldsymbol{T}, \Sigma_s, \Sigma_t, \Sigma_{st})$ such that $\Sigma_s \cup \Sigma_t \cup \Sigma_{st}$ belongs to a class among* L, LF, AF, G, S, SF, F, GF, A, WS, *and* WA, *deciding whether there exists a probabilistic (or probabilistic universal) solution for Pr_s relative to Σ_s and $\mathcal{M}$ is* coNP*-complete in the data complexity. Hardness for* coNP *for the classes* G, F, GF, WS, *and* WA *holds even when the underlying Bayesian network is a polytree.*

The following result shows that deciding whether there exists a probabilistic (or probabilistic universal) solution for a probabilistic source database relative to an ODE problem is in P in the data complexity, if BCQ answering for the involved sets of TGDs and NCs is first-order rewritable as a Boolean UCQ, and the underlying Bayesian network is a polytree. As a corollary, by the complexity of BCQ answering with TGDs and NCs, deciding the existence of a solution is in P for the classes L, LF, AF, S, SF, and A in the data complexity, if the underlying Bayesian network is a polytree.

Theorem 3. *Given a probabilistic source database Pr_s relative to a source ontology Σ_s, with a polytree as Bayesian network, and an ODE problem $\mathcal{M} = (\boldsymbol{S}, \boldsymbol{T}, \Sigma_s, \Sigma_t, \Sigma_{st})$ such that $\Sigma_s \cup \Sigma_t \cup \Sigma_{st}$ belongs to a class of TGDs and NCs for which BCQ answering is first-order rewritable as a Boolean UCQ, deciding whether there exists a probabilistic (universal) solution for Pr_s relative to Σ_s and $\mathcal{M}$ is in* P *in the data complexity.*

Finally, the following theorem shows that answering UCQs for probabilistic source databases relative to an ODE problem is complete for #P in the data complexity for all above classes of existential rules except for WG.

Theorem 4. *Given (i) an ODE problem $\mathcal{M} = (\boldsymbol{S}, \boldsymbol{T}, \Sigma_t, \Sigma_s, \Sigma_{st})$ such that $\Sigma_s \cup \Sigma_{st} \cup \Sigma_t$ belongs to a class among* L, LF, AF, G, S, SF, F, GF, A, WS, *and* WA, *and (ii) a probabilistic source database Pr_s relative to Σ_s such that there exists a solution for Pr_s relative to $\mathcal{M}$, (iii) a UCQ $Q = q(\boldsymbol{X})$ over $\boldsymbol{T}$, and (iv) a tuple $\mathbf{a}$, computing $conf_Q(\mathbf{a})$ is #P-complete in the data complexity.*

4.4 Probabilistic Ontological Data Exchange

All the results of Section 4.3 in Theorems 1 and 4 carry over to the case of probabilistic ontological data exchange. Clearly, the hardness results carry over immediately, since deterministic ontological data exchange is a special case of probabilistic ontological data exchange. As for the membership results, we additionally consider the worlds for the probabilistic mapping, which are iterated through in the data complexity and guessed in the combined, the *ba*-combined, and the *fp*-combined complexity.

5 Summary and Outlook

We have defined deterministic and probabilistic ontological data exchange problems, where probabilistic knowledge is exchanged between two ontologies. The two ontologies and the mapping between them are defined via existential rules, where the rules for the mapping are deterministic and probabilistic, respectively. We have given a precise analysis of the computational complexity of deciding the existence of a probabilistic (universal) solution for different classes of existential rules in both deterministic and probabilistic ontological data exchange. We also have delineated some tractable special cases, and we have provided some complexity results for exact UCQ answering.

An interesting topic for future research is to further explore the tractable cases of probabilistic solution existence and whether they can be extended, e.g., by slightly generalizing the type of the mapping rules. Another issue for future work is to further analyze the complexity of answering UCQs for different classes of existential rules in deterministic and probabilistic ontological data exchange.

Acknowledgments. This work was supported by an EU (FP7/2007-2013) Marie-Curie Intra-European Fellowship ("PRODIMA"), the UK EPSRC grant EP/J008346/1 ("PrOQAW"), the ERC grant 246858 ("DIADEM"), a Yahoo! Research Fellowship, and funds from Universidad Nacional del Sur and CONICET, Argentina. This paper is a short version of a paper that appeared in Proc. RuleML 2015 [19].

References

1. Arenas, M., Botoeva, E., Calvanese, D., Ryzhikov, V.: Exchanging OWL2 QL knowledge bases. In: Proc. IJCAI. pp. 703–710 (2013)
2. Arenas, M., Botoeva, E., Calvanese, D., Ryzhikov, V., Sherkhonov, E.: Exchanging description logic knowledge bases. In: Proc. KR. pp. 563–567 (2012)

3. Arenas, M., Pérez, J., Reutter, J.L.: Data exchange beyond complete data. J. ACM 60(4), 28:1–28:59 (2013)
4. Baader, F.: Least common subsumers and most specific concepts in a description logic with existential restrictions and terminological cycles. In: Proc. IJCAI. pp. 364–369 (2003)
5. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: Proc. IJCAI. pp. 364–369 (2005)
6. Calì, A., Gottlob, G., Kifer, M.: Taming the infinite chase: Query answering under expressive relational constraints. J. Artif. Intell. Res. 48, 115–174 (2013)
7. Cali, A., Gottlob, G., Lukasiewicz, T., Marnette, B., Pieris, A.: Datalog+/–: A family of logical knowledge representation and query languages for new applications. In: Proc. LICS. pp. 228–242 (2010)
8. Calì, A., Gottlob, G., Pieris, A.: Towards more expressive ontology languages: The query answering problem. Artif. Intell. 193, 87–128 (2012)
9. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. J. Autom. Reasoning 39(3), 385–429 (2007)
10. Fagin, R., Kimelfeld, B., Kolaitis, P.G.: Probabilistic data exchange. J. ACM 58(4), 15:1–15:55 (2011)
11. Fagin, R., Kolaitis, P.G., Miller, R.J., Popa, L.: Data exchange: Semantics and query answering. Theor. Comput. Sci. 336(1), 89–124 (2005)
12. Fuhr, N., Rölleke, T.: A probabilistic relational algebra for the integration of information retrieval and database systems. ACM Trans. Inf. Sys. 15(1), 32–66 (1997)
13. Green, T.J., Karvounarakis, G., Tannen, V.: Provenance semirings. In: Proc. PODS. pp. 31–40 (2007)
14. Imielinski, T., Witold Lipski, J.: Incomplete information in relational databases. J. ACM 31(4), 761–791 (1984)
15. Johnson, D.S.: A catalog of complexity classes. In: van Leeuwen, J. (ed.) Handbook of Theoretical Computer Science, vol. A, chap. 2, pp. 67–161. MIT Press (1990)
16. Krisnadhi, A., Lutz, C.: Data complexity in the $\mathcal{EL}$ family of description logics. In: Proc. LPAR, LNCS, vol. 4790, pp. 333–347. Springer (2007)
17. Lenzerini, M.: Data integration: A theoretical perspective. In: Proc. PODS. pp. 233–246 (2002)
18. Lukasiewicz, T., Martinez, M.V., Pieris, A., Simari, G.I.: From classical to consistent query answering under existential rules. In: Proc. AAAI. pp. 1546–1552 (2015)
19. Lukasiewicz, T., Martinez, M.V., Predoiu, L., Simari, G.I.: Existential rules and Bayesian networks for probabilistic ontological data exchange. In: Proc. RuleML. LNCS, vol. 9202, pp. 294–310. Springer (2015)
20. Papadimitriou, C.H.: Computational Complexity. Addison-Wesley (1994)
21. Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Linking data to ontologies. J. Data Sem. 10, 133–173 (2008)
22. Suciu, D., Olteanu, D., Ré, C., Koch, C.: Probabilistic Databases. M & C (2011)
23. Vardi, M.Y.: The complexity of relational query languages (extended abstract). In: Proc. STOC. pp. 137–146 (1982)

Refining Software Quality Prediction with LOD

Davide Ceolin[1], Till Döhmen[1], and Joost Visser[2]

[1] VU University Amsterdam
de Boelelaan 1081
1081HV Amsterdam, The Netherlands
d.ceolin@vu.nl
[2] Software Improvement Group
Rembrandt Toren, 15th floor, Amstelplein 1
1096 HA Amsterdam, The Netherlands

Abstract. The complexity of software systems is growing and the computation of several software quality metrics is challenging. Therefore, being able to use the already estimated quality metrics to predict their evolution is a crucial task. In this paper, we outline our idea to use Linked Open Data to enrich the information available for such prediction. We report our experience so far, and we outline the preliminary results obtained.

1 Introduction

Software size and complexity is growing, thus being able to estimate and predict software quality is crucial to monitor the process of software development and promptly steer it. In fact, a quality metric provides a value summarizing one relevant aspect of the software that can be consulted to identify issues or risks in the development process or in the software itself. Therefore, several different quality dimensions have been defined, as described, for instance, by Kan [8].

Estimating software quality is then a crucial but challenging task, for several reasons including the complexity of the software to be measured and the fact that these measures are often hard to quantify: some of them depend on run-time software behavior, some on static software properties. The estimation of the values of these measures is possible, as demonstrated, for instance, by Alves and Visser [2] and Bouwers [4]. However, given the complexity of this task, we propose to use such estimates to predict the temporal evolution of these values.

Preliminary analyses on a dataset from the Software Improvement Group[3] show encouraging results on the use of these estimates as starting point for the prediction of the evolution over time of software quality ratings.[4] We hypothesize that, by using Linked Open Data (LOD) we can improve and refine the accuracy of our predictions. In particular, by enriching the information available about the projects analyzed, we can categorize these projects (e.g., by industry sector or programming language), thus increasing the possibility to group

[3] http://www.sig.eu

[4] For confidentiality reasons, we could not make the dataset publicly available.

together projects showing similar quality evolution over time. We present here some preliminary encouraging results obtained in this direction, and we discuss a series of open issues that we need to address in order to extend this research.

The rest of this paper is structured as follows: Section 2 introduces related work. Section 3 describes the enrichment of software projects data. Section 4 provides preliminary results, that are discussed in Section 5.

2 Related Work

Software quality prediction is an important issue, that has been tackled from different points of view. As Al-Jamini and Ahmed [1] describe in their review, several relevant approaches to this problem make use of machine learning.

We have also employed machine learning techniques (in particular, Markov chains [12]) to predict software quality based on the starting rating of a project [5]. The results are promising and we will aim at perfecting them with additional features, properly selected from external sources, like LOD. The future quality value of systems shows a strong correlation with the current quality rating, due to the fact that the rating usually changes very slowly over time. Moreover, a second trend was discovered which revealed that higher quality systems tend to deteriorate in quality and low-quality systems tend to improve, both with the tendency towards the medium quality level. This could be explained as a case of regression towards the mean [6], i.e., could be due to noise in the extreme quality ratings that disappears as more accurate estimates are provided. However, this possible explanation still needs to be evaluated and, anyway, could explain only the second trend. These two trends, for very high or very low-quality systems, yield a high uncertainty in the prediction. Using LOD, we expect to obtain more tailored predictions (e.g., by identifying software quality trends associated to the programming language adopted) to reduce prediction uncertainty.

Misirli et al.[11] propose the use of Bayesian Networks to make software quality predictions. As the number of potentially useful features grows (consequently to LOD enrichment), we will consider this approach in the future. Jing et al. [7] use a dictionary learning-approach that represents a more specialized but limited approach as compared to our use of LOD.

3 Enriching Software Quality Prediction with LOD

Our hypothesis is that by enriching the information about the projects we analyze with LOD, we can obtain features that are useful for improving the software quality prediction. For instance, software quality could vary in different industrial sectors or the programming language used could affect quality evolution.

Our focus is on a dataset provided by the Software Improvement Group, which consists mainly of projects of Dutch companies and of a few additional European customers. We enriched the dataset using mainly DBpedia [3]. In the enrichment process, we encountered the following issues:

Missing information DBpedia contains a description of only 209 companies located in the Netherlands. Additional companies have been identified in the Dutch DBpedia[5], which contains the description of 3.883 companies, but does not provide information about their location.

Disambiguation Some companies have homonyms. To disambiguate resources and identify the right URI for a given company, we expect to employ heuristics based on the company website, its location, and industry sector.

Consistency literals vs. URIs Some classifications are available in an inconsistent manner. For instance, industry can appear both as `http://dbpedia.org/ontology/industry` and `http://dbpedia.org/property/industry`. In some cases, the value of one of these two properties is reported only as a literal value, thus affecting the possibility to perform ontological reasoning.

4 Preliminary Results

We performed a preliminary analysis on a dataset consisting of 1019 snapshots of maintainability of 112 companies. These snapshots already presented a first industry classification provided by SIG. In total, 14 industrial sectors are present.

We computed the semantic similarity between each possible combination of industrial categories using the Wikipedia distance [10] and the WU & Palmer distance [17]. On these data, we performed a series of preliminary analyses:

1. We run a Wilcoxon signed-rank test [16] at 95% confidence level to check if the observations are significantly different when grouped per industrial sector. These results show a weak positive Spearman [15] correlation with both the Wikipedia (0.07) and the Wu & Palmer (0.14) distances.
2. We computed the same procedure as above by using also the Kolmogorov-Smirnov test [9, 14] . This resulted in a slightly higher correlation, 0.16 for the Wikipedia distance and 0.24 for the Wu & Palmer distance.
3. We computed the contrast analysis [13] of the linear combinations of the observations, again grouped per industrial sector. The resulting contrast estimators showed a weak correlation with the Wikipedia distance (0.15) and with the Wu & Palmer distance values (0.12).
4. We grouped a small set of observations aligned with DBpedia by industrial sector of the companies involved (telecommunication and financial services). According to a Wilcoxon signed-rank test at 90% significance, the two groups are significantly different, according to the Kolmogorov-Smirnov test, not.

5 Discussion and Future Work

We present an early stage work about the use of LOD to refine the precision and accuracy of software quality prediction. We performed a series of exploratory and preliminary studies which shows a low correlation between the maintainability

[5] `http://nl.dbpedia.org`

and the industry sector of these projects. These results provide the basis for further exploration because: (1) the existence of a weak correlation is confirmed by more tests, hence it is possible that we can identify a subset of the data analyzed that presents a higher correlation; (2) the different methods for computing semantic similarity and different statistical significance tests provided significantly different results, thus indicating the need for exploring different computational techniques; (3) as shown by the last item of Section 4, the industrial sector seems to be a discriminant for software quality, although this aspect needs to be evaluated on larger datasets; and (5) our analyses focused on a limited set of enrichment features, but several others are utilizable. So, we plan to extend this research to identify the most robust methods to perform these predictions, and we will extend these analyses including additional LOD features and sources.

Acknowledgements This work is funded by Amsterdam Data Science.

References

1. H. Al-Jamimi and M. Ahmed. Machine learning-based software quality prediction models: State of the art. In *ICISA*, pages 1–4, 2013.
2. T. L. Alves and J. Visser. Static estimation of test coverage. In *SCAM*, pages 55–64. IEEE Computer Society, 2009.
3. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A Nucleus for a Web of Open Data. In *ISWC*, volume 4825, pages 722–735. Springer, 2007.
4. E. Bouwers, J. P. Correia, A. van Deursen, and J. Visser. Quantifying the analyzability of software architectures. In *WICSA*, pages 83–92. IEEE, 2011.
5. T. Döhmen, D. Ceolin, and J. Visser. Towards Building a Software Quality Prediction Model. Technical report, Software Improvement Group, 2015.
6. F. Galton. Regression towards mediocrity in hereditary stature. *The Journal of the Anthropological Institute of Great Britain and Ireland*, 15:246–263, 1886.
7. X.-Y. Jing, S. Ying, Z.-W. Zhang, S.-S. Wu, and J. Liu. Dictionary learning based software defect prediction. In *ICSE*, pages 414–423, 2014.
8. S. Kan. *Metrics and Models in Software Quality Engineering.* Pearson, 2002.
9. A. Kolmogorov. Sulla determinazione empirica di una legge di distribuzione. *Giornale dell'Istituto Italiano degli Attuari*, 4:1–11, 1933.
10. D. Milne and I. H. Witten. An open-source toolkit for mining wikipedia. *Artif. Intell.*, 194:222–239, 2013.
11. A. T. Misirli and A. B. Bener. A mapping study on bayesian networks for software quality prediction. In *RAISE*, pages 7–11, 2014.
12. J. R. Norris. *Markov chains.* Cambridge University Press, 1998.
13. R. Rosenthal and R. L. Rosnow. *Contrast analysis : focused comparisons in the analysis of variance.* Cambridge University press, 1985.
14. N. Smirnov. Table for Estimating the Goodness of Fit of Empirical Distributions. *The Annals of Mathematical Statistics*, 19(2):279–281, 1948.
15. C. Spearman. The proof and measurement of association between two things. *Amer. J. Psychol.*, 15:72101, 1904.
16. F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1:80–83, 1945.
17. Wu, Z. and Palmer, M. Verb semantics and lexical selection. In *ACL*. ACL, 1994.

Reducing the Size of the Optimization Problems in Fuzzy Ontology Reasoning

Fernando Bobillo[1] and Umberto Straccia[2]

[1] Dpt. of Computer Science & Systems Engineering, University of Zaragoza, Spain
[2] Istituto di Scienza e Tecnologie dell'Informazione (ISTI - CNR), Pisa, Italy
Email: fbobillo@unizar.es, straccia@isti.cnr.it

Abstract. Fuzzy ontologies allow the representation of imprecise structured knowledge, typical in many real-world application domains. A key factor in the practical success of fuzzy ontologies is the availability of highly optimized reasoners. This short paper discusses a novel optimization technique: a reduction of the size of the optimization problems obtained during the inference by the fuzzy ontology reasoner fuzzyDL.

1 Introduction

In recent years, we have noticed an increase in the number of applications for mobile devices that could benefit from the use of semantic reasoning services [1]. Because of the limited capabilities of mobile devices, it is especially important to develop reasoning algorithms performing efficiently in practice. In order to deal with imprecise knowledge, such applications could use fuzzy ontologies [8]. In fuzzy ontologies, concepts and relations are fuzzy. Consequently, the axioms are not in general either true or false, but they may hold to some degree of truth.

However, little effort has been paid so far to the study and implementation of optimization techniques for fuzzy ontology reasoning, which is essential to reason with real-world scenarios in practice (some exceptions are [3,4,5,6]). This short paper discusses some optimization techniques to improve the performance of the reasoning algorithm by reducing the size of optimization problems obtained during the inference. In particular, we will provide optimized MILP encodings of the restrictions involving n-ary operators and fuzzy membership functions. Such optimizations have been implemented in *fuzzyDL*, arguably the most popular and advanced fuzzy ontology reasoner [2], and proved their usefulness.

2 Background on fuzzyDL reasoning

We assume the reader to be familiar with the syntax and semantics of fuzzy Description Logics (DLs) [8]. The reasoning algorithm implemented in fuzzyDL combines tableaux rules with an optimization problem. After some preprocessing, fuzzyDL applies tableau rules decomposing complex concept expressions into simpler ones, as usual in tableau algorithms, but also generating a system of inequation constraints. These inequations have to hold in order to respect the semantics of the DL constructors. After all rules have been applied, an optimization problem must be solved before obtaining the final solution. The tableau rules are deterministic and the optimization problem is unique.

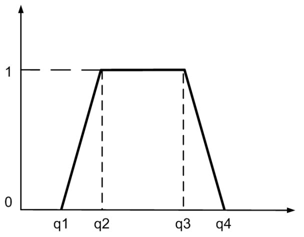
This optimization problem has a solution iff the fuzzy KB is consistent. In fuzzyDL, we obtain a bounded Mixed Integer Linear Programming [7] (MILP) problem, that is, minimising a linear function with respect to a set of constraints that are linear inequations in which rational and integer variables can occur. The problem is bounded, with rational variables ranging over $[0, 1]$ and some integer variables ranging over $\{0, 1\}$. For example, in Łukasiewicz fuzzy DLs, the restriction $x_1 \otimes_{\text{Ł}} x_2 = z$ can be encoded using the set of constraints $\{x_1 + x_2 - 1 \leq z, x_1 + x_2 - 1 \geq z - y, z \leq 1 - y, y \in \{0, 1\}\}$. Observe that the MILP encoding of the restriction has introduced a new variable y: the two possibilities $y = 0$ and $y = 1$ encode the non-deterministic choice implicit in the interpretation of the conjunction under Łukasiewicz fuzzy logic. The complexity of solving a MILP problem is NP-complete and it depends on the number of variables, so it is convenient to reduce the number of new variables.

Let x, z be $[0, 1]$-variables, and x_u be a rational unbounded variable. fuzzyDL has to solve some restrictions involving fuzzy connectives, such as $x_1 = \ominus x_2$, $x_1 \otimes x_2 = z$, $x_1 \oplus x_2 = z$, or $x_1 \Rightarrow x_2 = z$. Furthermore, it also needs to solve some restrictions $\mathbf{d}(x_u) \geq z$ involving fuzzy membership functions $\mathbf{d}$ such as the $trapezoidal(k1, k2, q1, q2, q3, q4)$ (see Table 1 (a)), the $triangular(k1, k2, q1, q2, q3)$, $left(k1, k2, q1, q2)$, or $right(k1, k2, q1, q2)$ [8].

3 Optimizing Łukasiewicz N-ary Operators

Let us start with the case of conjunction concepts in Łukasiewicz fuzzy DLs. An n-ary concept of the form $(C_1 \sqcap C_2 \sqcap \cdots \sqcap C_n)$ can be represented, using associativity, only using binary conjunctions $(C_1 \sqcap (C_2 \sqcap (\cdots \sqcap C_n)) \dots)$. A binary conjunction concept introduces a restriction of the form $x_1 \otimes_{\text{Ł}} x_2 = z$ which, as shown in Section 2, can be encoded adding a new binary variable y. Hence, in order to represent the n-ary conjunction, $n-1$ new variables y_i would be needed. However, it is possible to give a more efficient representation by considering the conjunction as an n-ary operator. Indeed, a restriction of the form $x_1 \otimes_{\text{Ł}} x_2 \otimes \cdots \otimes x_n = z$ can be encoded using only *one* new binary variable and, thus, saves 2^{n-2} possible alternative assignments to the variables y_i.

$$\begin{aligned} \sum_{i=1}^{n} x_i - (n-1) &\leq z, \\ y &\leq 1 - z, \\ \sum_{i=1}^{n} x_i - (n-1) &\geq z - (n-1)y, \\ y &\in \{0, 1\}. \end{aligned}$$



i	y_3	y_2	y_1	condition
1	0	0	0	$x_1 = z$
2	0	0	1	$x_2 = z$
3	0	1	0	$x_3 = z$
4	0	1	1	$x_4 = z$
5	1	0	0	$x_5 = z$

Table 1. (a) Trapezoidal membership function; (b) Encoding of 5 states.

$y = 0$ encodes the case $z = \sum_{i=1}^{n} x_i - (n-1) \geq 0$, and $y = 1$ encodes the case $z = 0$ and $\sum_{i=1}^{n} x_i - (n-1) < 0$. Let us consider now disjunction concepts in Łukasiewicz fuzzy DLs. A binary disjunction can be represented adding a new binary variable y as $\{x_1 + x_2 \leq z + y, y \leq z, x_1 + x_2 \geq z, y \in \{0,1\}\}$. Again, $n-1$ new binary variables would be needed but, similarly as before, considering the disjunction as an n-ary operator we would need only *one* new binary variable:

$$\begin{aligned} \sum_{i=1}^{n} x_i \leq z + (n-1)y, \\ y \leq z, \\ \sum_{i=1}^{n} x_i \geq z, \\ y \in \{0,1\}. \end{aligned}$$

4 Optimizing Göedel N-ary Operators

An n-ary conjunction can be represented using binary conjunctions adding restrictions of the form $x_1 \otimes_G x_2 = z$, which can be encoded as follows:

$$\begin{aligned} z \leq x_1, \\ z \leq x_2, \\ x_1 \leq z + y, \\ x_2 \leq z + (1-y), \\ y \in \{0,1\}. \end{aligned}$$

The idea is that if $y = 0$, $x_1 = z$ is the minimum; whereas if $y = 1$, $x_2 = z$ is the minimum. This adds a new variable y, so in the case of n-ary conjunctions there would be $n-1$ new variables. Treating the conjunction as an n-ary operator, a more efficient representation is possible. An n-ary conjunction introduces a restriction of the form $x_1 \otimes_G x_2 \otimes \cdots \otimes x_n = z$. To represent that the minimum of n variables x_i is equal to z, we can use n binary variables y_i such that if y_i

takes the value 0 then x_i (representing the minimum) is equal to z, and such that the sum of the y_i is 1, so z takes the value of some x_i. Note that the minimum may not be unique. Such a representation is as follows:

$$\begin{aligned}
z &\leq x_i, \text{ for } i \in \{1, \ldots, n\}, \\
x_i &\leq z + y_i, \text{ for } i \in \{1, \ldots, n\}, \\
\sum_{i=1}^{n} y_i &= 1, \\
y_i &\in \{0, 1\}, \text{ for } i \in \{1, \ldots, n\}.
\end{aligned}$$

Now, we will show that it is possible to give a more efficient representation, Essentially, we need to encode n possible states. However, n possible states can be encoded using $m = \lceil \log_2 n \rceil$ new binary variables only. For instance, for $n = 5$, only $\lceil \log_2 5 \rceil = 3$ binary variables are necessary, where we use the encoding of the $n = 5$ states in Table 1 (b).

The main point is now to correctly encode the condition $x_i \leq z + y_i$ of the old encoding. We proceed as follows. Let b_i be a string of length m, representing the value $i - 1$ in base 2 ($1 \leq i \leq n$). For instance, for $i = 4$, $b = 011$, as illustrated in the table above. Let us define the expression e_{ij} ($1 \leq i \leq n$, $1 \leq j \leq m$) as:

$$e_{ij} = \begin{cases} y_j & \text{if the } j\text{th bit of } b_i \text{ is } 0 \\ 1 - y_j & \text{otherwise.} \end{cases}$$

For $i = 4$, we have $b = 011$ and, thus, $e_{41} = 1 - y_1$, $e_{42} = 1 - y_2$, and $e_{43} = y_3$. Now we are ready to provide the whole encoding:

$$\begin{aligned}
z &\leq x_i, \text{ for } i = 1, \ldots, n \\
x_i &\leq z + \sum_{j=1}^{m} e_{ij}, \text{ for } i = 1, \ldots, n \\
\sum_{j=1}^{m} 2^{j-1} y_j &\leq n - 1, \\
y_j &\in \{0, 1\}, \text{ for } j = 1, \ldots, m.
\end{aligned}$$

The first condition is the same as before. The second condition guarantees that $x_i \leq z$ in the state b_i. Finally, the third condition ensures that we are not addressing more than n states. For instance, for $n = 5$ we have:

$$\begin{aligned}
& z \leq x_1, \\
& z \leq x_2, \\
& z \leq x_3, \\
& z \leq x_4, \\
& z \leq x_5, \\
& x_1 \leq z + y_1 + y_2 + y_3,
\end{aligned}$$

$$\begin{aligned}
&x_2 \leq z + (1 - y_1) + y_2 + y_3, \\
&x_3 \leq z + y_1 + (1 - y_2) + y_3, \\
&x_4 \leq z + (1 - y_1) + (1 - y_2) + y_3, \\
&x_5 \leq z + y_1 + y_2 + (1 - y_3), \\
&y_1 + 2y_2 + 4y_3 \leq 4, \\
&y_1 \in \{0, 1\}, \\
&y_2 \in \{0, 1\}, \\
&y_3 \in \{0, 1\}.
\end{aligned}$$

The case of the disjunction in Gödel fuzzy DLs is dual. If an n-ary concept of the form $(C_1 \sqcup C_2 \sqcup \cdots \sqcup C_n)$ is represented using binary disjunctions, $n - 1$ new binary variables are needed. However, if we consider it as an n-ary concept, it is possible to use $\lceil \log_2 n \rceil$ new binary variables only:

$$\begin{aligned}
z \geq x_i, \text{ for } i = 1, \ldots, n \\
x_i + \sum_{j=1}^{m} e_{ij} \geq z \text{ for } i = 1, \ldots, n \\
\sum_{j=1}^{m} 2^{j-1} y_j \leq n - 1, \\
y_j \in \{0, 1\} \text{ for } j = 1, \ldots, m.
\end{aligned}$$

5 Optimizing Fuzzy Membership Functions

Let us start with the case of trapezoidal functions, which introduce a restriction of the form $\mathit{trapezoidal}(k1, k2, q1, q2, q3, q4)(x_u) \geq z$. A restriction of that form can be represented by adding 5 new binary variables y_i as follows:

$$\begin{aligned}
x_u + (k_1 - q1)y_2 &\geq k_1, \\
x_u + (k_1 - q2)y_3 &\geq k_1, \\
x_u + (k_1 - q3)y_4 &\geq k_1, \\
x_u + (k_1 - q4)y_5 &\geq k_1, \\
x_u + (k_2 - q1)y_1 &\leq k_2, \\
x_u + (k_2 - q2)y_2 &\leq k_2, \\
x_u + (k_2 - q3)y_3 &\leq k_2, \\
x_u + (k_2 - q4)y_4 &\leq k_2, \\
x_u &\leq 1 - y_1 - y_5, \\
x_u &\geq y_3, \\
x_u + (q1 - q2)x_u + (k_2 - q1)y2 &\leq k_2, \\
x_u + (q1 - q2)x_u + (k_1 - q2)y2 &\geq k_1 + q1 - q2,
\end{aligned}$$

$$
\begin{aligned}
x_u + (q4 - q3)x_u + (k_2 - q3)y4 \leq k_2 + q4 - q3, \\
x_u + (q4 - q3)x_u + (k_1 - q4)y4 \geq k_1, \\
y_1 + y_2 + y_3 + y_4 + y_5 = 1, \\
y_i \in \{0, 1\}, \text{for } i = 1, \ldots, 5.
\end{aligned}
$$

To reduce now the number of binary variables, the idea is to have 5 binary variables encoding the 5 possible states: $x_u \leq q1$ ($y_1 = 1$), $x_u \in [q1, q2]$ ($y_2 = 1$), $x_u \in [q2, q3]$ ($y_3 = 1$), $x_u \in [q3, q4]$ ($y_4 = 1$), and $x_u \geq q4$ ($y_5 = 1$). However, as shown in Table 1 (b), it is possible to represent 5 states using only 3 variables.

The case of other fuzzy membership functions is similar. In triangular functions, a naïve encoding introduces 4 new variables to represent the 4 possible states, but it is possible to consider only 2. Finally, in left and right shoulder functions, it is necessary to consider 3 states, which can be achieved by adding 2 new binary variables, instead of the 3 ones needed in the non-optimal encoding.

By considering the fact that even for moderate sized ontologies we may easily generate thousands of such constraints, it is evident that the number of saved binary variables n, and hence the number of saved assignments 2^n, is non-negligible.

Acknowledgement This research work has been partially supported by the CICYT project TIN2013-46238-C4-4-R and DGA-FSE.

References

1. C. Bobed, R. Yus, F. Bobillo, and E. Mena. Semantic reasoning on mobile devices: Do androids dream of efficient reasoners? *Journal of Web Semantics*, In press.
2. F. Bobillo and U. Straccia. fuzzyDL: An expressive fuzzy description logic reasoner. In *Proceedings of the 17th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2008)*, pages 923–930, 2008.
3. F. Bobillo and U. Straccia. On partitioning-based optimisations in expressive fuzzy description logics. In *Proceedings of the 24th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2015)*, 2015.
4. F. Bobillo and U. Straccia. Optimising fuzzy description logic reasoners with general concept inclusions absorption. *Fuzzy Sets and Systems*, In press.
5. V. Haarslev, H.-I. Pai, and N. Shiri. Optimizing tableau reasoning in $\mathcal{ALC}$ extended with uncertainty. In *Proceedings of the 20th International Workshop on Description Logics (DL 2007)*, volume 250, pages 307–314. CEUR Workshop Proceedings, 2007.
6. G. S. N. Simou, T. Mailis and G. Stamou. Optimization techniques for fuzzy description logics. In *Proceedings of the 23rd International Workshop on Description Logics (DL 2010)*, volume 573. CEUR Workshop Proceedings, 2010.
7. H. M. Salkin and K. Mathur. *Foundations of Integer Programming.* North-Holland, 1989.
8. U. Straccia. *Foundations of Fuzzy Logic and Semantic Web Languages.* CRC Studies in Informatics Series. Chapman & Hall, 2013.