Journal Subline

LNCS 5480

Journal on Data Semantics XII

Stefano Spaccapietra
Editor-in-Chief

Springer

Stefano Spaccapietra (Ed.)

Journal on Data Semantics XII

Springer

Volume Editor

Stefano Spaccapietra
École Polytechnique Fédérale de Lausanne
EPFL-IC, Database Laboratory
1015 Lausanne, Switzerland
E-mail: stefano.spaccapietra@epfl.ch

Library of Congress Control Number: Applied for

CR Subject Classification (1998): H.2, H.4, C.2, H.3, I.2

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

ISSN	0302-9743 (Lecture Notes in Computer Science)
ISSN	1861-2032 (Journal on Data Semantics)
ISBN-10	3-642-00684-1 Springer Berlin Heidelberg New York
ISBN-13	978-3-642-00684-5 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12641147 06/3180 5 4 3 2 1 0

The LNCS Journal on Data Semantics

Computerized information handling has changed its focus from centralized data management systems to decentralized data-exchange facilities. Modern distribution channels, such as high-speed Internet networks and wireless communication infrastructure, provide reliable technical support for data distribution and data access, materializing the new, popular idea that data may be available to anybody, anywhere, anytime. However, providing huge amounts of data on request often turns into a counterproductive service, making the data useless because of poor relevance or inappropriate level of detail. Semantic knowledge is the essential missing piece that allows the delivery of information that matches user requirements. Semantic agreement, in particular, is essential to meaningful data exchange.

Semantic issues have long been open issues in data and knowledge management. However, the boom in semantically poor technologies, such as the Web and XML, has boosted renewed interest in semantics. Conferences on the Semantic Web, for instance, attract big crowds of participants, while ontologies on their own have become a hot and popular topic in the database and artificial intelligence communities.

Springer's LNCS *Journal on Data Semantics* aims at providing a highly visible dissemination channel for remarkable work that in one way or another addresses research and development related to the semantics of data. The target domain ranges from theories supporting the formal definition of semantic content to innovative domain-specific application of semantic knowledge. This publication channel should be of the highest interest to researchers and advanced practitioners working on the Semantic Web, interoperability, mobile information services, data warehousing, knowledge representation and reasoning, conceptual database modeling, ontologies, and artificial intelligence.

Topics of relevance to this journal include:

- Semantic interoperability, semantic mediators
- Ontologies
- Ontology, schema and data integration, reconciliation and alignment
- Multiple representations, alternative representations
- Knowledge representation and reasoning
- Conceptualization and representation
- Multimodel and multiparadigm approaches
- Mappings, transformations, reverse engineering
- Metadata
- Conceptual data modeling
- Integrity description and handling
- Evolution and change
- Web semantics and semi-structured data

- Semantic caching
- Data warehousing and semantic data mining
- Spatial, temporal, multimedia and multimodal semantics
- Semantics in data visualization
- Semantic services for mobile users
- Supporting tools
- Applications of semantic-driven approaches

These topics are to be understood as specifically related to semantic issues. Contributions submitted to the journal and dealing with semantics of data will be considered even if they are not from the topics in the list.

While the physical appearance of the journal issues is like the books from the well-known Springer LNCS series, the mode of operation is that of a journal. Contributions can be freely submitted by authors and are reviewed by the Editorial Board. Contributions may also be invited, and nevertheless carefully reviewed, as in the case for issues that contain extended versions of the best papers from major conferences addressing data semantics issues. Special issues, focusing on a specific topic, are coordinated by guest editors once the proposal for a special issue is accepted by the Editorial Board. Finally, it is also possible that a journal issue be devoted to a single text.

The Editorial Board comprises an Editor-in-Chief (with overall responsibility), a Coeditor-in-Chief, and several members. The Editor-in-Chief has a four-year mandate. Members of the board have a three-year mandate. Mandates are renewable and new members may be elected at any time.

We are happy to welcome you to our readership and authorship, and hope we will share this privileged contact for a long time.

Stefano Spaccapietra
Editor-in-Chief
http://lbd.epfl.ch/e/Springer/

Previous Issues

- JoDS I: Special Issue on Extended Papers from 2002 Conferences, LNCS 2800, December 2003
  Coeditors: Sal March and Karl Aberer
- JoDS II: Special Issue on Extended Papers from 2003 Conferences, LNCS 3360, December 2004
  Coeditors: Roger (Buzz) King, Maria Orlowska, Elisa Bertino, Dennis McLeod, Sushil Jajodia, and Leon Strous
- JoDS III: Special Issue on Semantic-Based Geographical Information Systems, LNCS 3534, August 2005
  Guest Editor: Esteban Zimányi
- JoDS IV: Normal Issue, LNCS 3730, December 2005
- JoDS V: Special Issue on Extended Papers from 2004 Conferences, LNCS 3870, February 2006
  Coeditors: Paolo Atzeni, Wesley W. Chu, Tiziana Catarci, and Katia P. Sycara
- JoDS VI: Special Issue on Emergent Semantics, LNCS 4090, September 2006
  Guest Editors: Karl Aberer and Philippe Cudre-Mauroux
- JoDS VII: Normal Issue, LNCS 4244, November 2006
- JoDS VIII: Special Issue on Extended Papers from 2005 Conferences, LNCS 4830, February 2007
  Coeditors: Pavel Shvaiko, Mohand-Saïd Hacid, John Mylopoulos, Barbara Pernici, Juan Trujillo, Paolo Atzeni, Michael Kifer, François Fages, and Ilya Zaihrayeu
- JoDS IX: Special Issue on Extended Papers from 2005 Conferences (continued), LNCS 4601, September 2007
  Coeditors: Pavel Shvaiko, Mohand-Saïd Hacid, John Mylopoulos, Barbara Pernici, Juan Trujillo, Paolo Atzeni, Michael Kifer, François Fages, and Ilya Zaihrayeu
- JoDS X: Normal Issue, LNCS 4900, February 2008
- JoDS XI: Special Issue on Extended Papers from 2006 Conferences, LNCS 5383, December 2008
  Coeditors: Jeff Z. Pan, Philippe Thiran, Terry Halpin, Steffen Staab, Vojtech Svatek, Pavel Shvaiko, and John Roddick

JoDS Volume XII

This volume of JoDS results from a rigorous selection among 20 full papers submissions received in response to a call for contributions issued in July 2007.

Reviews of submitted papers resulted in requests for major revisions to authors of nine papers. Reviews of revised versions eventually led to the acceptance of five papers for publication. They are listed in the table of contents hereinafter.

This volume also hosts an extended version of a paper originally published in the COOPIS 2005 Conference. This paper had been selected for an extended version proposal, but a misunderstanding between the authors and conference chairs caused a halt in the preparation and review process. Once the misunderstanding cleared, the submission was updated and positively reviewed and accepted according to normal selection rules.

We would like to thank authors of all submitted papers as well as all the reviewers who contributed to improving the papers through their detailed comments.

The forthcoming volume XIII is a JoDS special issue on semantic data warehousing.

We hope you'll enjoy reading this volume.

Stefano Spaccapietra
Editor-in-Chief
http://lbd.epfl.ch/e/Springer/

Reviewers

We are very grateful to the external reviewers listed below who helped the editorial board in the reviewing task:

Yuan An	Drexel University, USA
Daniele Barone	University of Milano-Bicocca, Italy
Jesús Bermúdez	Basque Country University, Spain
Silvana Castano	University of Milan, Italy
Jos de Bruijn	Free University of Bozen-Bolzano, Italy
Ying Ding	University of Innsbruck, Austria
Gillian Dobbie	University of Auckland, New Zealand
Federico Michele Facca	STI Innsbruck, Austria
Alfio Ferrara	Università degli Studi di Milano, Italy
Marc-Philippe Huget	University of Savoie, France
Hanjo Jeong	George Mason University, USA

JoDS Editorial Board

Table of Contents

SECCO: On Building *Semantic Links* in Peer-to-Peer Networks

Giuseppe Pirrò[1], Massimo Ruffolo[2], and Domenico Talia[1]

[1] D.E.I.S, University of Calabria
87036 Rende, Italy
{gpirro,talia}@deis.unical.it
[2] Exeura
87036 Rende, Italy
ruffolo@exeura.it

Abstract. Ontology Mapping is a mandatory requirement for enabling semantic interoperability among different agents and services relying on different ontologies. This aspect becomes more critical in Peer-to-Peer (P2P) networks for several reasons: (i) the number of different ontologies can dramatically increase; (ii) mappings among peer ontologies have to be discovered on the fly and only on the parts of ontologies "contextual" to a specific interaction in which peers are involved; (iii) complex mapping strategies (e.g., structural mapping based on graph matching) cannot be exploited since peers are not aware of one another's ontologies. In order to address these issues, we developed a new ontology mapping algorithm called Semantic Coordinator (*SECCO*). *SECCO* is composed by three individual matchers: syntactic, lexical and contextual. The *syntactic matcher*, in order to discover mappings, exploits different kinds of linguistic information (e.g., comments, labels) encoded in ontology entities. The *lexical matcher* enables discovering mappings in a semantic way since it "interprets" the semantic meaning of concepts to be compared. The *contextual matcher* relies on a "how it fits" strategy, inspired by the contextual theory of meaning, and by taking into account the contexts in which the concepts to be compared are used refines similarity values. We show through experimental results that *SECCO* fulfills two important requirements: fastness and accuracy (i.e., quality of mappings). *SECCO*, differently from other semantic P2P applications (e.g., Piazza, GridVine) that assume the preexistence of mappings for achieving semantic interoperability, focuses on the problem of finding mappings. Therefore, if coupled with a P2P platform, it paves the way towards a comprehensive semantic P2P solution for content sharing and retrieval, semantic query answering and query routing. We report on the advantages of integrating *SECCO* in the *K-link+* system.

Keywords: Ontology mapping in Peer-to-Peer networks, semantic mapping, semantic P2P applications, semantic web.

1 Introduction

Most of the information available today is in an unstructured and non-standardized form, therefore processing and exchanging it with people via computers is actually very difficult. This is because "machines" are not able to recognize the meaning of information they deal with. Solving this challenge is one of the main goals of Semantic Web technologies. The Semantic Web vision [3] aims at providing Web resources (e.g., web pages, documents) with supplementary meaningful information (i.e., metadata) in order to improve and facilitate their retrieval, enable their automatic processing by machines and make it possible the interoperability among different systems. Ontologies are key enablers towards this "new" Web of semantically rich resources. Ontologies can be exploited to give *shared conceptualizations* of knowledge domains and make *explicit* and machine understandable the meaning of the terminology adopted [24]. They aim at capturing knowledge typically shared by a group. The reference to a domain of interest indicates their usage not for modeling the whole world but rather those parts relevant to a particular task. Many ontology languages used today are based on XML (e.g., RDF(S) [28], OWL [39]) which make ontologies exploitable as semantic support in different classes of distributed applications such as Semantic Peer-to-Peer [48] and Semantic Grid [22].

In a recent interview [4], Tim Berners-Lee stated that: "*The Semantic Web is designed to smoothly interconnect personal information management, enterprise application integration, and the global sharing of commercial, scientific and cultural data...*". From this interview emerges that semantic-based data sharing is expected to begin in controlled environments smaller than the World Wide Web as for instance: enterprise networks and small-medium Peer-to-Peer (P2P) networks. Moreover, the Semantic Web is expected to follow the same path of the Internet, which started in bounded environments.

In distributed environments, it is not feasible to have a single (and universally accepted) ontology describing a knowledge domain, but there will be several (possibly overlapping) ontologies created w.r.t "the point of view" of their designers. In fact, as people see the world differently these viewpoints inevitably will be encoded in ontologies. For instance, for a computer company a computer is a product, for an economist, it is a household appliance, while for a student it is just a computer. In order to promote interoperability among these different perspectives about the world, it is necessary to ensure "reciprocal understanding". This problem has been a core issue of recent ontology research activities and in the literature is referred to as the Ontology Mapping (or Matching) Problem (OMP) [21].

OMP concerns discovering correspondences (*aka* mappings) among entities belonging to different ontologies (i.e., a *source* and a *target* ontology). The problem becomes more challenging in P2P networks for several reasons: (i) the number of overlapping ontologies can dramatically increases, in theory each peer will have its own ontology that reflects peer's needs and interests; (ii) mappings among peer ontologies must be quickly discovered and only on the parts of ontologies "contextual" to a specific interaction in which peers are involved; (iii) complex mapping strategies (e.g., structural mapping based on graph matching) cannot be exploited since peers are not aware of one another's ontologies. Thus, ontology mapping algorithms for P2P networks should ensure a trade-off between fastness (not achievable by adopting complex mapping strategies) and accuracy (i.e., quality of results).

To date, several approaches to solve the OMP have been proposed [11]. These are based on techniques borrowed from various research areas such as Bayesian Decision theory (see OMEN [36] and [38]) Graph Similarity (see GMO [51]) Information Retrieval (see LOM [41] and V-doc [42]) just to name a few of them. However, these approaches underestimate the following aspects:

- They do not adequately consider the OMP in open environments such as P2P networks. A recent survey on ontology mapping [11] contains only a bibliographic reference to mapping systems designed for P2P networks.
- They do not take into account the need for "on the fly" mappings crucial in P2P networks. In such networks, a complete mapping between peer ontologies is not a requirement for interactions among peers; they only need to quickly map the parts of their ontologies related to the specific interaction in which they are involved. Moreover, since peers are often unaware of one another's ontologies the amount of ontological information exploitable to discover mappings is quite limited.
- They do not adequately interpret the semantic meaning of ontology concepts to be compared. In addition, the context in which concepts appear is not carefully scrutinized from a semantic point of view. Even if there are some approaches addressing these issues, they often are not designed on the basis of well-founded experimental results.

In this paper, we address the OMP in P2P networks by defining a new ontology mapping algorithm called SEmantiC COordinator (*SECCO*). We especially focused on the OMP in P2P environments since P2P applications seem to be a class of applications that will take advantage of Semantic Web technologies in a near future. *SECCO* is composed by three individual matchers: *syntactic*, *lexical* and *contextual*, each of which tackles the OMP from a different perspective. In particular, the *syntactic matcher* aims at discovering mappings by an Information Retrieval approach called LOM [41] that exploits linguistic information (e.g., comments, labels) encoded in ontology entities. The *lexical matcher* assesses semantic relatedness, even among syntactically unrelated concepts (e.g., *car* and *automobile*), by combining two approaches exploiting WordNet [35] as background knowledge. The *contextual matcher* implements a new similarity strategy called "how it fits". This strategy complies with the contextual theory of meaning [34] and is founded on the idea that two concepts are related if they fit well in each other's context. This approach allows comparing the structures of two concepts both in terms of their position in the ontological taxonomy and constituent properties. This is achieved at an affordable computational cost since it is not required to take into account the whole structure of ontologies.

Specifically, the main contributions of the paper are:

- We exploit the idea of *concept mapping* with the aim to gather similarity information among concepts belonging to different peer ontologies. *Concept mappings* allow building *semantic links* among peers that can be exploited in several classes of semantic P2P applications (e.g., semantic search, semantic query routing, and community formation).
- We designed and extensively evaluated *SECCO*, which is endowed with three new mapping strategies facing the OMP from a different perspective but that share accuracy and fastness. In particular, *concept mappings* are derived by combining the results of these three mapping strategies.

- Differently from other semantic P2P applications (e.g., Piazza, GridVine) that assume the preexistence of mappings for achieving semantic interoperability, we focus on the problem of finding mappings.

Extensive experimental results, aimed at comparing *SECCO* w.r.t the state of the art, show that the combination of the proposed mapping strategies provides an adequate trade-off between accuracy (in terms of quality of mappings) and fastness (in terms of elapsed time for discovering mappings).

Moreover, we want to emphasize that *SECCO*, if coupled with a P2P platform, paves the way towards a comprehensive semantic P2P solution for content sharing and retrieval, semantic query answering and semantic routing. We report on the advantages of integrating *SECCO* within the *K-link*+ system [31].

The remainder of this paper is organized as follows. Section 2, after introducing the terminology adopted in the rest of the paper, presents the *SECCO* ontology mapping algorithm. Section 3 describes and evaluates the individual matchers of *SECCO*. In Section 3.4 we motivate the design of *SECCO*. Section 4 presents a detailed evaluation of the system in two different settings. In particular, Section 4.1 compares *SECCO* with H-Match [8, 9] and performs a sensitivity analysis of the different parameters of the algorithm (Section 4.1.1). Then, Section 4.2 evaluates the algorithm on four real-life ontologies of the OAEI 2006 benchmark test suite. Here *SECCO* is also compared with other mapping algorithms not explicitly designed for facing the OMP in P2P networks. Section 5 reviews related work. Section 6 draws some conclusions. Finally, Section 7 sketches future work.

2 The *SECCO* Ontology Mapping Algorithm

We designed the *SECCO* ontology mapping algorithm for addressing the OMP in P2P networks, since semantic P2P applications, built by interconnecting knowledge managed at personal level, seem to be the applications taking advantage of Semantic Web technologies in a near future [4]. We argue that most of the existing mapping algorithms are not suitable for P2P networks since they, to work properly, need to deal with the whole two ontologies to be mapped. For instance, top ontology mapping algorithms, i.e., Falcon [25], and RiMOM [54] have structural mapping strategies built upon graph matching techniques. These techniques are well suited to work "offline" while they are not applicable in P2P environments where the OMP has to be faced "online" and peers are not aware of one another's ontologies.

In this section, after introducing the terminology adopted in the rest of the paper (Section 2.1) we describe the ontology model exploited by *SECCO* to discover mappings (Section 2.2). Section 2.3 presents a scenario of usage of *SECCO* and provides the pseudo-code of the algorithm.

2.1 Preliminary Definitions

We consider a P2P network in which each peer owns an *ontology* (i.e., peer ontology) that represents the point of view of the peer on a particular knowledge domain. Each (*seeker*) peer can request to other (*providers*) peers a *concept mapping* whose aim is to provide information of similarity among a concept belonging to the *seeker* peer ontology and concepts belonging to ontologies of *provider* peers. The aim of the request depends on the application class, as will be discussed in Section 3.4. In this scenario, we define both *seeker* and *provider* peers as *semantic* peers since they manage, share and exchange knowledge by exploiting ontologies.

An ontology is basically composed of two parts: (i) the *intensional model*, represented by means of an ontology schema and (ii) the *extensional part*, implemented by a knowledge base. In this paper, we adopt the following simplified ontology model that is inspired by the formal ontology definition proposed in [12].

Definition 1 (*SECCO* Ontology model). The *SECCO* ontology model is a six-tuple of the form:

$$O=\langle C,R,L,\leq,\varphi_R,\varphi_L\rangle$$

consisting of a set of concepts names C, a set of relation names R and a set of strings L that contains ontology metadata like comment(s) and label(s). Concepts names are arranged in a hierarchy by means of the partial order $\leq$ (which intrinsically defines the ISA relation). The signature φ_R: $R \rightarrow C\times C$ associates each relation name $r\in R$ with a couple of concepts. Given a relation name $r\in R$, the first attribute of the tuple defines the relation domain *dom*(r)=$\pi_1(\varphi_R(r))$ and the second attribute the range *range*(r)= $\pi_2(\varphi_R(r))$. The signature φ_R: $C\cup R \rightarrow 2^L$ associates each concept and relation with a subset of strings representing its metadata.

This simplified ontology model is built up starting from OWL ontologies as described in Section 2.2.

Definition 2 (Seeker peer). A *seeker* peer is a semantic peer that sends a semantic *request* over the P2P network to *provider* peers and receives a set of *concept mappings*.

Definition 3 (Provider peer). A *provider* peer is a semantic peer that receives a *request* from a *seeker* peer and returns a *concept mapping* obtained by exploiting *SECCO*.

Definition 4 (Request). Let O be an ontology, a *request* is a two-tuple of the form $RQ=\langle c,ctx(c)\rangle$ where $c\in C$ is a concept belonging to the *seeker* peer ontology and ctx(c) is the *context* of c.

Definition 5 (Concept Context). Let O be an ontology, the set of strings ctx(c) is the *context* of the *concept* $c\in C$. This set contains names of concepts related to c by relations in R that correspond to OWL *objectype* properties [39] and the names of relations in R that correspond to OWL *datatype* properties [39]. More formally, $ctx(c)=C_{range}\cup C_{dom}\cup R_{dt}$ where: (i) C_{range} and C_{dom} are the sets of concepts names for which respectively hold the following conditions: $c\in$ *dom*(r)$\wedge c_{range}\in$ *range*(r) and $c_{dom}\in$ *dom*(r)$\wedge c\in$ *range*(r) with $r\in R$ and *range*(r) and *dom*(r) both corresponding to user defined classes; (ii) R_{dt} is the set of relation names for which either *range*(r) or *dom*(r) is defined on a data type [39].

Datatype property names, present in the original OWL ontology, are included in ctx(c) as described in Section 2.2.

The *concept mapping* between a *seeker* peer concept and the set of concepts belonging to a *provider* peer ontology is defined as follows:

Definition 6 (Concept Mapping). Given the *seeker* peer request RQ=$\langle$s,ctx(s)$\rangle$ and the ontology O belonging to a *provider* peer, a *concept mapping* M between each *provider* concept $p \in C$ and the *seeker* peer concept s is a set of 3-tuples of the form M=$\langle$s,p,$\sigma\rangle$ where $\sigma \in [0,1]$ is the similarity value between the couple of concepts s and $p \in C_p$.

Similarity values between couples of concepts are obtained by adopting the similarity measure defined as follows:

Definition 7 (Similarity Measure). Given two ontologies O_s and O_p belonging to a *seeker* and a *provider* peer respectively, a request $RQ_s=\langle c_s, ctx(c_s)\rangle$ and the set CTX_p composed by two-tuples of the form $\langle c_j, ctx(c_j)\rangle$ where $\forall c_j \in C_p$ $ctx(c_j)$ is the context of c_j; the similarity between the couples of concepts $c_s \in C_s$ and $c_p \in C_p$ is computed by the following function:

$$sim(c_s, c_p) : \{sim_{syn}(c_s, c_p), sim_{lex}(c_s, c_p), sim_{con}(c_s, c_p)\} \rightarrow [0.1]$$

where $sim_{syn}(c_s,c_p)$: $C_s \times C_p \rightarrow [0,1]$ is the *syntactic similarity*, $sim_{lex}(c_s,c_p)$: $C_s \times C_p \rightarrow [0,1]$ is the *lexical similarity*, $sim_{con}(c_s,c_p)$: $RQ_s \times CTX_p \rightarrow [0,1]$ is the *contextual similarity*. These three similarity measures are symmetric and reflexive i.e., $\forall c_s \in C_s$ and $\forall c_p \in C_p$,

$$sim(c_s,c_s)=1 \qquad \text{(reflexivity)}$$
$$sim(c_s,c_p)=sim(c_p,c_s) \qquad \text{(symmetry)}$$

How to represent mappings in SECCO

Even if the OMP has received a lot of attention from the scientific community, a standardized format for storing ontology mappings does not exist. In order to overcome this problem, there are two possible ways:

1. Exploiting features of ontology languages. For instance, OWL provides built-in constructs for representing equivalence between concepts (i.e., *owl:equivalentClass*), relations (i.e., *owl:equivalentProperty*) and instances (i.e., *owl:sameAs*). This approach allows OWL inference engines to automatically interpret the semantics of mappings and perform reasoning across different ontologies. However, by adopting this approach, a confidence value cannot be interpreted.
2. Adopting the approach described in [20]. This mapping representation exploits RDF/XML to formalize ontology mappings. Each individual mapping is represented in cells and each cell has the following attributes: *entity 1* (i.e., the concept in the *source* ontology), *entity 2* (i.e., the concept in the *target* ontology), *measure* (i.e., the confidence value), *type of mapping* (usually equivalence). Due to its different parameters, this representation can easily be exploited by several kinds of applications.

In *SECCO*, we adapt the second type of representation to the context of a P2P ontology mapping system. The adopted mapping representation is depicted in Fig.1. This representation allows a *seeker* peer (i.e., the *seeker_peer* tag), for a given *seeker* concept (i.e., the *seeker_concept* tag), to maintain both the URIs of *provider* concepts

```
<mapping>
    <seeker_peer name= ID>
    <seeker_concept=URI>
    <provider_peer name = ID>
              <provider_concept ID=URI>
                   <similarity>σ</similarity>
              </provider_concept>
              <provider_concept ID=URI>
                   <similarity> σ</similarity>
              </provider_concept>
               ...
    </provider_peer name>
      ...
 </mapping>
```

Fig. 1. Representation of mappings in *SECCO*

(i.e., *provider_concept* tag) and values of similarity (i.e., the *similarity* tag) grouped on the basis of *provider* peers (i.e., the *provider_peer* tag) that answered to the *seeker* request.

2.2 The *SECCO* Ontology Model Construction

The *SECCO* ontology model (see Definition 1) is built by exploring ontology class definitions contained in peer ontologies. To explain how the *SECCO* ontology model is constructed, let us consider the fragment of the *Ka* ontology (available at http://www.cs.man.ac.uk/~horrocks/OWL/Ontologies/ka.owl) depicted in Fig. 2.

```
<owl:Class rdf:about="file:F:/Projects/OIL/DAMLOilEd/ontologies/ka.daml#Publication">
 <rdfs:subClassOf>
     <owl:Restriction>
         <owl:onProperty rdf:resource="file:F:/Projects/OIL/DAMLOilEd/ontologies/ka.daml#title"/>
         <owl:allValuesFrom rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
     </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
   <owl:Restriction>
   <owl:onProperty rdf:resource="file:F:/Projects/OIL/DAMLOilEd/ontologies/ka.daml#describesProject"/>
         <owl:allValuesFrom>
             <owl:Class rdf:about="file:F:/Projects/OIL/DAMLOilEd/ontologies/ka.daml#Project"/>
         </owl:allValuesFrom>
   </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
     <owl:Restriction>
         <owl:onProperty rdf:resource="file:F:/Projects/OIL/DAMLOilEd/ontologies/ka.daml#abstract"/>
         <owl:allValuesFrom rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
     </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
     <owl:Restriction>
         <owl:onProperty rdf:resource="file:F:/Projects/OIL/DAMLOilEd/ontologies/ka.daml#year"/>
         <owl:allValuesFrom rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
     </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="file:F:/Projects/OIL/DAMLOilEd/ontologies/ka.daml#keyword"/>
        <owl:allValuesFrom rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="file:F:/Projects/OIL/DAMLOilEd/ontologies/ka.daml#Book">
        <rdfs:subClassOf>
            <owl:Class rdf:about="file:F:/Projects/OIL/DAMLOilEd/ontologies/ka.daml#Publication"/>
        </rdfs:subClassOf>   OMISSIS
</owl:Class>
```

Fig. 2. A fragment of the *Ka* ontology

Given an input ontology *SECCO* executes the following four main steps to construct its ontology model:

1. *Class name extraction*: for each class a concept name is created in the *SECCO* ontology model.
2. *Subclass properties* (i.e., ISA) *analysis*: for each class definition, *SECCO* scrutinizes its sub classes defined by the construct *rdfs:subClassOf* and generates the taxonomic structure.
3. *Datatype properties analysis*: values of these properties are data literals. For each datatype property *SECCO* considers the linguistic information encoded in the property name (e.g., *year* in Ka) and includes in its ontology model, a new concept name representing the property (i.e., *year*) and a relation (i.e., *has_year* as shown in Fig. 3) to relate this new concept with the original class.
4. *Object properties analysis*: these are properties for which the value is an individual. In this case, for each property, *SECCO* exploits the original OWL encoding and generates a relation, in its ontology model, that has the same name, domain and range of the original one.

The ontology fragment depicted in Fig. 2, contains the definition of a class *Publication* along with some object and datatype properties, and related classes. By running *SECCO,* we obtain the ontology model representation depicted in Fig. 3. Notice that the construction of this representation also exploits the definitions of classes (e.g., *Project*, *Event*) that are not represented in the excerpt shown in Fig.2.

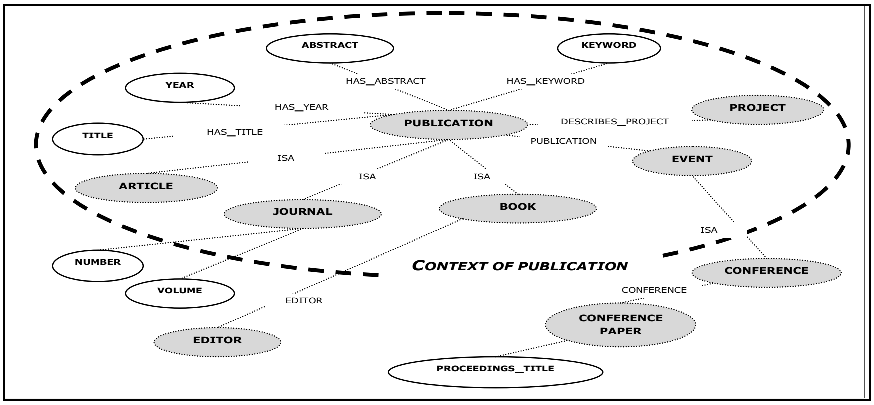


Fig. 3. The *SECCO* representation of the *Ka* ontology related to the concept *Publication*

In Fig. 3 filled oval represent ontology concepts (i.e., classes) as defined in the original OWL ontology whereas empty ovals are the concepts introduced in the *SECCO* ontology model to exploit information encoded in *Datatype properties*.

The context of the *Publication* concept, as defined in Definition 5, is the dashed area in Fig. 3. In more detail, *ctx(Publication)={title, year, abstract, keyword, Project, Event, Book, Journal, Article}*.

2.3 The *SECCO* Ontology Mapping Algorithm

SECCO aims at discovering a *concept mapping* between a *seeker* peer concept and ontology concepts belonging to ontologies of *provider* peers. Each peer in the network plays a twofold role: (i) *seeker* peer, when it sends a request to the network; (ii) *provider* peer, when it executes locally the *SECCO* algorithm. Whenever a *provider* peer receives a request, it runs *SECCO* with an input of the following form:

$$I=<c_s, ctx(c_s), O_p, T_h, w_s, w_L, w_c>$$

where: c_s is a concept belonging to a *seeker* peer ontology; $ctx(c_s)$is the context of c_s; O_p is the *provider* ontology, $T_h \in [0,1]$ is a threshold value that can be used for filtering results. Moreover, w_s, w_L, and w_c are the weights assigned to the values of *syntactic*, *lexical* and *contextual* similarity respectively. The overall similarity value is computed by the *Combiner* module that weights the similarity values provided by the individual matchers (see Fig. 4) and discards values that do not exceed a given threshold (i.e., the T_h parameter in Fig.4). Once *SECCO* has terminated, it returns a *concept mapping* as defined in Definition 6.

The overall approach is described in Fig. 4. A *seeker* peer issues an information request by picking a concept along with the related context from its ontology. This request reaches *provider* peers that run the *SECCO* algorithm on their ontologies and return to the *seeker* peer *concept mappings* that will be stored in the *mapping store* for future reuse. Fig. 5 describes the *SECCO* algorithm in a pseudo-code.

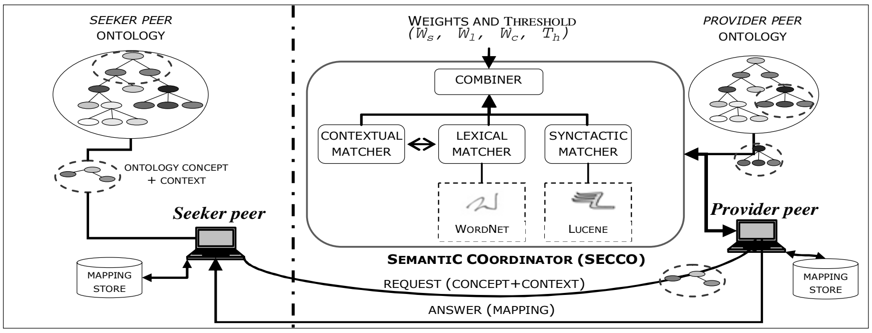


Fig. 4. The *SECCO* architecture and usage scenario

The function *evaluate_syntactic_similarity* is implemented by the *syntactic matcher* (see Section 3.1) while the function *evaluate_lexical_similarity*, is implemented by the *lexical matcher* (see Section 3.2). The function *evaluate_contextual_similarity* (see Fig. 7), implemented by the *contextual matcher* (see Section 3.3), relies on the function *evaluate_how_it_fits* (see Fig. 8) that adopts a “see how it fits” strategy that is founded on the idea that two concepts are related if they *fit well* in each other’s context (see Section 3.3). The *contextual matcher* takes as input the context obtained by the function *extract_context* (see Fig. 6).

```
The SECCO algorithm
Input: An input I=<$c_s$, ctx($c_s$), O, $T_h$, $w_s$, $w_L$, $w_C$> where O=⟨C,R⟩ is the
SECCO ontology model
Output: The concept mapping M
Method:
  1.  M=∅;
  2.  for each  c∈C do
  3.    $sim_{syn}$=evaluate_syntactic_similarity($c_s$,c);
  4.    $sim_{lex}$=evaluate_lexical_similarity($c_s$,c);
  5.     ctx(c)=extract_context(c,O); /*Fig.6*/
  6.     $sim_{con}$=evaluate_contextual_similarity($c_s$,Ctx($c_s$),c,Ctx(c));
  7.     sim=($w_S$*$sim_{syn}$+$w_L$*$sim_{lex}$+$w_C$*$sim_{con}$); /* overall similarity value
*/
  8.      if sim>$T_h$ then
  9.         m.s=$c_s$
  10.        m.p=c;
  11.        m.σ=sim
  12.        M=M∪m;
  13.     end-if
  14. end-for
  15. return M;
```

Fig. 5. The *SECCO* algorithm in pseudo-code

```
Function extract_context
Input: An ontology O=⟨C,R⟩ and a concept c∈C
Output: The context ctx(c)
Method:
    1.     $ctx_c$=∅; $ctx_r$=∅;
    2.     for each $c_c$∈C do
    3.         for each $r_c$∈R do
    4.             if ∃$r_c$(c,$c_c$)|∃$r_c$($c_c$,c) then
    5.                $ctx_c$=$ctx_c$∪{$c_c$}
    6.                $ctx_r$=$ctx_r$∪{$r_c$}
    7.             end-if
    8.         end-for
    9.     end-for
return ctx(c)=⟨$ctx_c$,$ctx_r$⟩;
```

Fig. 6. The *extract_context* function

```
Function evaluate_contextual_similarity
Input: Two concepts $c_1$ and $c_2$ and their contexts ctx($c_1$) and ctx($c_2$)
Output: A numerical value $sim_{con}$∈[0,1] representing the contextual
similarity between the concepts $c_1$ and $c_2$
Method:
    1.     s2s=evaluate_how_it_fits($c_1$, ctx($c_1$)); /* see Figure 8 */
    2.     s2t=evaluate_how_it_fits($c_1$, ctx($c_2$));
    3.     t2s=evaluate_how_it_fits($c_2$, ctx($c_1$));
    4.     t2t=evaluate_how_it_fits($c_2$, ctx($c_2$));
    5.     $sim_{con}$=((1-||s2s-t2t|-|s2t+t2s||));
    6.     return $sim_{con}$
```

Fig. 7. The *evaluate_contextual_similarity* function

```
Function evaluate_how_it_fits
Input: A concept c and a context ctx(x)= ⟨C_x,R_x⟩
Output: A numerical value m∈[0,1] representing the fitness between the
concept c and the context ctx(x)
Method:
    1.    T=0;
    2.    for each c_e∈C_x do
    3.         T+=evaluate_lexical_similarity(c,c_e);
    4.    end-for
    5.    return m=T/|ctx(x)|;
```

Fig. 8. The *evaluate_how_it_fits* function

In the next section, the individual matchers of *SECCO* are described and evaluated.

3 Individual Matchers: The Building Blocks of *SECCO*

The idea of combining different heuristics, each of which implemented by an individual matcher, for the assessment of an overall similarity value between two ontology entities is not new (see [16,17]). The main motivation of adopting such a strategy is that from some ontology mapping initiatives (e.g., [19]) emerged that a combination of mapping strategies, in general, allows to obtain better results. Moreover, it is not arguable that a single heuristic is able to exploit all the types of information (e.g., lexical, structural) encoded in ontology entities.

With these motivations in mind, we decided to endow *SECCO* with three different matchers. Each matcher respectively exploits syntactic/linguistic (*syntactic matcher*), lexical (*lexical matcher*) and contextual (*contextual matcher*) information contained in ontology entities. We adopt the *syntactic matcher*, since in our previous work on ontology mapping [41] we noticed that a merely syntactic approach can be effective and fast in discovering mappings. The *lexical matcher* is successful in discovering mappings in a semantic way, that is, by considering the semantic meaning of the compared terms and not treating them just as strings. Through this approach, it is possible, for instance, to discover that the *automobile* concept used in a *seeker* peer ontology is similar to the concept of *car* used in a *provider* peer ontology. Finally, the *contextual matcher* that relies on the *lexical matcher* allows refining similarity between concepts by considering the contexts in which they appear. The *contextual matcher* rationale complies with the contextual theory of meaning [34] according to which the relatedness between concepts can be defined in terms of their interchangeability within the contexts in which they appear. The *contextual matcher* allows the assessment of similarity between two concepts in terms of their structure/properties, but on a local basis, that is, by only considering the properties and neighbors of the two concepts and not the whole ontology structures in which they appear. Notice that for the scenario in which *SECCO* has to work (i.e., a P2P network) a structural matching strategy could affect the requirement of time accuracy given that it requires to compare entire ontologies (e.g., [51]). Indeed, in *SECCO*, a *provider* peer only receives a request (i.e., a concept along with its context) from a *seeker* peer and not its entire ontology. Through experimental evaluations (see Section 4), we prove that the lack of a structural matcher will not significantly affect mapping results. Furthermore, it is worthwhile noting that the modular architecture of *SECCO* allows easily designing and adding new matchers to be included into the algorithm. In the Sections 3.1-3.3, we provide both a description and an evaluation of the three individual matchers. Section 3.4 motivates the designing of *SECCO* by reporting on the advantages of integrating it in *K-link*+ [31], a P2P system for collaborative work and content sharing and retrieval.

3.1 The Syntactic Matcher

This matcher that implements the function *evaluate_syntactic_similarity* (see Fig. 5, line 3), mainly relies on the Lucene Ontology Matcher (LOM) described in our previous work on ontology mapping [41]. Here we provide an overall description of LOM, further details along with complete experimental results can be found in [41].

Given a *source* ontology O, in order to discover mappings, LOM aims at exploiting metadata of ontology entities (e.g., comments, and labels) contained in L (i.e., linguistic information). In particular, for each entity e in $C \cup R$ a *virtual document* that contains its metadata in L is encoded by exploiting the concept of Lucene Document [33]. Virtual documents are stored into an index maintained in main memory. Ontology mappings are derived by using values of entities of a *target* ontology as search arguments against the index created from the source ontology. Similarity values are computed by exploiting the scoring schema implemented in Lucene, which relies on Vector Space techniques [45].

In order to show the suitability of LOM in terms of both speed and accuracy, here we report on its evaluation on the OAEI 2006 benchmark test suite [37] as compared to three string matching techniques (i.e., I-Sub [49], Jaro Winkler [52] and Edit Distance [32]) that are typically exploited to perform syntactic matching of ontology entities. Ontologies in the OAEI benchmark test suite are based on one particular ontology defined in the bibliography domain and a number of variations of such ontology for which alignments are provided. There are different categories of alteration related to both linguistic aspects (variation in names and comments of entities) and structural aspects (variation in relations among entities). The benchmark is composed of five groups of tests that are constructed on the basis of the above mentioned types of alterations. Fig. 9 shows the average results obtained by LOM in terms of Precision (i.e., the number of correct mapping among all the mapping found), Recall (i.e., the number of correct mapping among all the existing mapping) and F-measure (i.e., the harmonic mean of Precision and Recall) [15].

As can be noticed, LOM outperforms the competitors. This is because it can profitably exploit all the linguistic information included in ontologies. In fact, even in test cases where entity names are altered (e.g., randomized, expressed in another language) LOM, by exploiting other linguistic information (e.g., labels, comments), can correctly assess similarity values. Moreover as discussed in [41] the average time for computing a mapping between two ontologies of the OAEI tests is 1.47s.

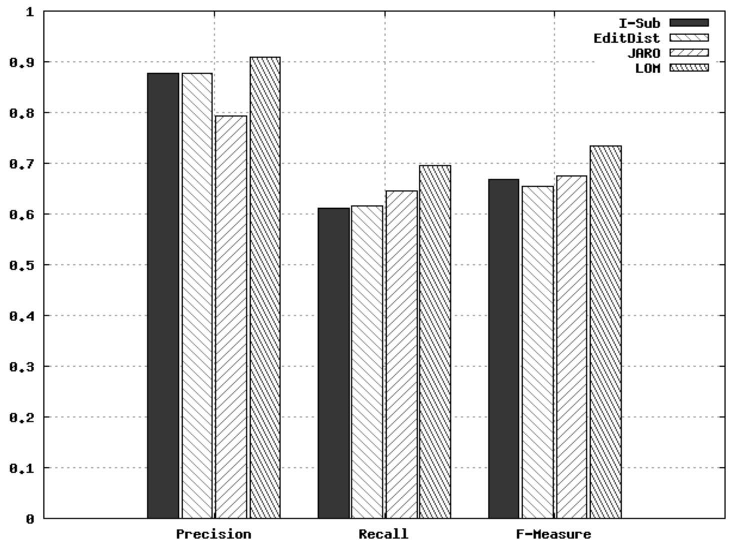


Fig. 9. Evaluation of LOM on the OEAI 2006 benchmark test suite

In the light of these considerations, LOM can be exploited as an individual matcher of *SECCO*, instead of a classical string-based approach, since it is more effective in terms of accuracy and is adequately fast.

In particular, in order to adopt LOM in *SECCO* we made the following adaptations:

- Ontologies of both *seeker* and *provider* peers are indexed. Therefore, each peer exploits its index to search for similar entities for requests coming from *seeker* peers (i.e., acts as a *provider*).
- Since in *SECCO* we do not want to compare whole ontologies, but a *seeker concept* along with its context with *provider* ontology concepts, we construct a new type of *virtual document* that contains linguistic information of the concept along with linguistic information of its context. This way, the linguistic information of a concept is augmented with linguistic information of entities in its context. Therefore, also the *syntactic matcher* takes into account a certain degree of structural information.

3.2 The Lexical Matcher

The *lexical matcher*, that implements the function *evaluate_lexical_similarity* (see Fig. 5, line 4), is the central component of the whole system. It allows implementing the semantic mapping by “interpreting” the semantic meaning of concepts to be compared. The *lexical matcher* exploits WordNet [35] as a source of knowledge about the world. WordNet is a lexical ontology organized in synsets (or senses) that encompass terms with synonymous meaning. Each synset has a gloss, which is a description in natural language of the concepts it represents. Synsets are connected to one another by a predefined set of semantic relations, some of which are reported in Table 1.

Table 1. Semantic relations between synsets in the WordNet 3.0 noun taxonomy

Relation	Description	Example
Hypernymy	*is a generalization of*	*Plant* is an hypernym of *Flower*
Hyponymy	*is a kind of*	*Tulip* is hyponym to *Flower*
Meronymy	*is a part of*	*Finger* is a meronym of *Hand*
Holonymy	*contains part*	*Tree* is a holonym of *Bark*
Antonymy	*opposite of*	*Man* is an antonym of *Woman*
Instance of	*is an instance of*	*California* is an instance of *American state*
Has instance	*has instance*	*American state* has instance *California*

Some of these relations define inheritance relations (Hypernymy and Hyponymy), other part-of relations (Holonymy and Meronymy). The Antonymy relation is used to state that a noun is the opposite of another. The relations *instance of* and *has instance* have been introduced in WordNet 3.0 and represent instantiation relations. Fig.10 shows an excerpt of the WordNet noun taxonomy.

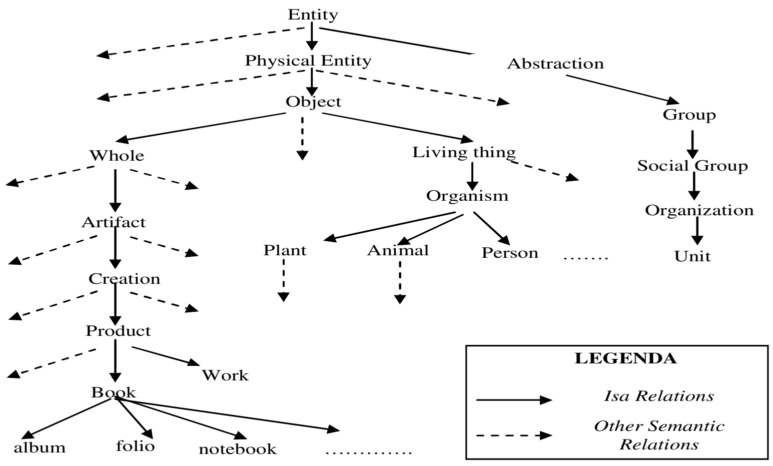


Fig. 10. An excerpt of the WordNet 3.0 noun taxonomy

Through the *lexical matcher* we aim at assessing the relatedness between ontology entities by exploiting their definitions within the WordNet database and position in the taxonomy. Semantic relatedness is the question of how related two concepts are by considering different kinds of relations connecting them. On the other hand, semantic similarity only considers the hypernymy/hyponymy relations among concepts. For instance, *Car* and *Gasoline* may be closely related to each other, e.g. because gasoline is the fuel most often used by cars. *Car* and *Bicycle* are semantically similar, not because they both have wheels and means of steering and propulsion, but because they are both instances of *Vehicle*. The relation between semantically similar and semantically related is asymmetric: if two concepts are similar, they are also related, but they are not necessarily similar just because they are related.

In literature (see [54]), there are several metrics for assessing similarity and relatedness among concepts in WordNet. In the context of ontology mapping, several approaches (e.g., [29]) compute semantic similarity between concepts by exploiting semantic similarity metrics. However, these approaches only consider the hypernymy / hyponymy relations linking synsets.

In order to take into account a wide range of semantic relations connecting synsets we included two components in the *lexical matcher*. A *similarity assessor* aimed at assessing semantic similarity and a *relatedness assessor* aimed at assessing semantic relatedness. The final lexical similarity value is obtained by combining the contribution of the two assessors.

3.2.1 The Semantic Similarity Assessor

The semantic similarity assessor aims at exploiting the structure of WordNet, which contains *per se* a certain degree of semantic information encoded in synsets. In the literature several approaches to compute semantic similarity are presented. In order to choose the most appropriate one, we evaluated results of several approaches and correlated them w.r.t human judgments of similarity. A detailed description of the dataset and evaluation methodology along with complete experimental results can be found at http://grid.deis.unical.it/similarity.

Among the evaluated metrics, the most performant are these based on the notion of Information Content (IC). IC can be considered a measure that quantifies the amount of information a concept expresses and is computed as log the negative likelihood of the occurrences of a concept in a large corpus. Resnik in [43] exploited the notion of IC for assessing semantic similarity between terms in a taxonomy. The basic intuition behind the use of the negative likelihood is that the more probable a concept is of appearing then the less information it conveys, in other words, infrequent words are more informative than frequent ones. Knowing the IC values for each concept, we may then calculate the similarity between two given concepts.

In the *lexical matcher* we adapt the Jiang and Conrath distance metric (J&C) [27]. This metric computes the semantic similarity between two concepts c_s and c_p as follows:

$$sim(c_s,c_p) = 1 - \frac{IC(c_s) + IC(c_p) - 2 * IC(sub(c_s,c_p))}{2}. \quad (1)$$

We consider the opposite of the semantic distance metric defined by J&C, as a similarity measure. Moreover, in order to quantify IC of concepts we exploit the function *IC* defined as follows [46]:

$$IC(c) = 1 - \frac{\log(hypo(c) + 1)}{\log(\max_{wn})}. \quad (2)$$

where the function *hypo* returns the number of hyponyms of a given concept *c*. Notice that concepts that represent leaves in the taxonomy will have an IC equals to one. Moreover, $\max_{wn}$ is a constant that indicates the total number of concepts in the WordNet noun taxonomy (i.e., 82115 in WordNet 3.0).

The function $sub(c_s,c_p)$ in equation 1 returns the concept (the lowest in the taxonomy) that subsumes both c_s and c_p.

3.2.2 The Semantic Relatedness Assessor

In order to select the most appropriate relatedness assessor we evaluated several approaches. A complete description of the dataset and evaluation methodology along with complete experimental results is available at the similarity experiment website: http://grid.deis.unical.it/similarity.

In our evaluation, we found that the gloss vector relatedness metric described in [40] is the most correlated w.r.t human judgment. This metric is based on the following intuition: the relatedness between two concepts can be assessed by comparing their glosses. In particular, this approach exploits "second order" vectors for glosses, that is, rather than just matching words that occur in glosses, the words in the gloss are replaced with co-occurrence (extracted from a corpus) vectors. Therefore, each gloss is represented by the average of its word vectors. Hence, pairwise comparisons can be made between vectors to measure relatedness between the concepts they represent. In the following, we summarize the step followed to compute the relatedness between two concepts c_s and c_p:

1. Get the gloss of c_s from WordNet. Create a gloss vector by adding the word vectors of all the words in the gloss.
2. Get the gloss of c_p from WordNet. Create a gloss vector by adding the word vectors of all the words in this gloss.
3. Compute the cosine of the gloss-vectors. In addition, this metric use the relations represented in Table 1 to augment the glosses of c_s and c_p, with gloss information of concepts that are directly linked to c_s and c_p. This makes the augmented glosses of c_s and c_p much bigger than the just the glossed of c_s and c_p.

If v_s and v_p are the gloss vectors for c_s and c_p, their relatedness in computed as follows:

$$relat(c_s, c_p) = \frac{v_s * v_p}{|\overrightarrow{v_s}| * |\overrightarrow{v_p}|} \cdot \qquad (3)$$

Computing the overall lexical similarity score

Overall, the lexical similarity is computed as a weighted sum of the scores provided by the two assessors:

$$\mathrm{sim}_{\mathrm{lex}}(c_s, c_p) = w_s * sim(c_s, c_p) + w_r * relat(c_s, c_p) \cdot \qquad (4)$$

From experimental evaluation, we found that equally weighting the two contributions (i.e., assigning 0.5 to both w_s and w_r) gives the best accuracy in terms of correlation w.r.t human judgment.

Reducing the elapsed time

Since WordNet is a huge lexical database, some performance issues related to its access can arise. In order to provide a fast access to the database and implement our similarity and relatedness measures we built an ad-hoc Lucene index that maintains the information about synsets. In particular, both values of IC and gloss vectors are stored in the index. The index is built by parsing the Prolog release of WordNet [53].

A running example

In order to see how the *lexical matcher* works, let us compute the lexical similarity between the concepts *Animal* and *Person*. According to eq. (4) we have to compute the semantic similarity and relatedness between the two concepts.

Computing semantic similarity

For the semantic similarity, we have to calculate the following coefficients:

- IC(*Animal*): *Animal* in the WordNet taxonomy has 3998 hyponyms, therefore according to equation (2) we have that IC(*Animal*)=0.2670.
- IC(*Person*): *Person* has 6978 hyponyms, in this case we have: IC (*Person*)=0.2178.
- IC(*Organism*): *Organism*, which subsumes both concepts (see Figure 10) has 16110 hyponyms. Therefore we have IC(*Organism*)=0.1439.

The semantic similarity value according to equation (1) is

sim(Animal, Person)= 0.9014

Computing semantic relatedness

In order to compute semantic relatedness between *Animal* and *Person* we have to compare their glosses augmented with glosses of neighbors concepts. The neighbors concepts of a given concept are concepts related to it by any of the relations reported in Table 1. In our example, the gloss of the concept Person is" *a human being…*". The gloss of the concept Animal is "*a living organism characterized by voluntary movement ...* ". Each of these concepts has a representative vector which contains for each dimension a number indicating the frequency of the word encoded in that dimension. Here we do not report the vectors of the two concepts since they have a very large dimension (about 12000). The semantic relatedness between *Animal* and *Person* is:

relat (Animal, Person)= 0.4667

Overall, the lexical similarity between *Animal* and *Person* is:

sim_{lex} *(Animal, Person)*= 0.5*0.9014+0.5*0.4667=0.6840

Compound terms

The *lexical matcher* treats compound terms by following the heuristic that in English the last token appearing on the right side of a compound term denotes the central concept, while other concepts encountered from the left side to the right side of denote a qualification of its meaning [30].

Remark

To summarize, our *lexical matcher* is a good candidate for being included in *SECCO* since it respects the requirements of fastness (by exploiting the Lucene index) and accuracy (proven by several experimental evaluations whose results are available at http://grid.deis.unical.it/similarity). The *lexical matcher* is included in the Java WordNet Similarity Library (JWSL) [26], which is a Java-based library that provides access to information about WordNet Synsets and implements a variety of similarity and relatedness metrics.

3.3 The Contextual Matcher

The aim of the *contextual matcher* is to implement the *evaluate_contextual_similarity* function (see Fig. 5, line 6) exploited to refine similarity values assessed by the *syntactic* and/or *lexical matcher*. It advances a contextual approach to semantic relatedness that builds upon Miller et al. definition in terms of the interchangeability of words in contexts [34]. Contexts help to refine the search of correct mappings since they intrinsically contain both information about the domains in which concepts to be compared are used and their structure in terms of properties and neighbors concepts. Contexts represent possible patterns of usage of concepts and the *contextual matcher* is founded on the idea that similar concepts have similar patterns of usage. If two concepts can be used in a similar context then they are related. A concept C_s (i.e., *seeker* concept) in a context $ctx(c_s)$ (i.e., *seeker* context) not similar to a concept C_p (i.e., *provider* concept) in a context $ctx(c_p)$ (i.e., *provider* context) will likely *fit bad* into $ctx(c_p)$ as well as c_p will do in $ctx(c_s)$. Conversely, if the two concepts can be interchangeably used, that is *fit well* in each other's contexts, then they can be considered related. We call this strategy *how it fits* and, in order to quantify how well a concept fits in a context, we calculate the *lexical similarity* between the concept and all the concepts in the considered context and take the average value (see Fig. 8). The overall *contextual similarity* is computed by exploiting the following similarity indicators:

1. *s2s*: indicates how the *seeker* concept fits in the *seeker* context
2. *s2t*: indicated how the *seeker* concept context fits in the *provider* context
3. *t2t*: indicated how the *provider* concept fits in the *provider* context
4. *t2s*: indicates how the *provider* concept fits in the *seeker* context

The overall contextual similarity is calculated according to the following equation.

$$sim_{con}(c_s, c_p) = 1 - (|s2s - t2t| + |s2t - t2s|) \,. \tag{5}$$

It is worthwhile noting that this strategy aims at taking into account structural information about concepts on a local basis, that is, by only considering properties and nearest neighbor concepts in the taxonomy. This is justified by the fact that a complete mapping among peer ontologies is not required; they only need to map their part of ontologies contextual to the interaction in which they are involved. Moreover, in computing a *concept mapping* by *SECCO*, a *provider* peer is not aware of the whole ontology of the *seeker* peer.

Here we provide a detailed evaluation of the *contextual matcher* on the two excerpts of ontologies depicted in Fig.11.

We consider *Book* in the ontology of the *seeker* peer, as *seeker* concept, and *Volume* in the ontology of the *provider* peer, as *provider* concept.

In order to assess the contextual similarity between *Book* and *Volume* we start with calculating the coefficients defined in equation (5). In particular, Table 2 and 3 show how the *s2t* and *t2s* coefficients are calculated.

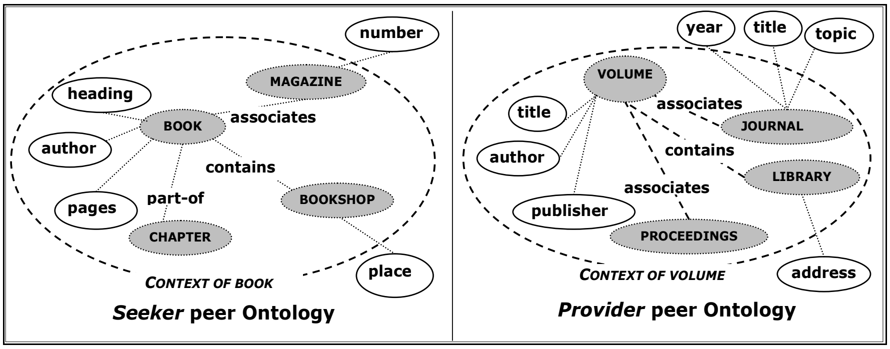


Fig. 11. Excerpts of a *seeker* and *provider* peer ontologies

Table 2. Calculation of the *s2t* coefficient

Seeker concept (*Book*)	Context of *Volume*	Lexical Similarity value
Book	*Journal*	0.8554
	Library	0.5102
	Proceedings	0.3961
	Title	0.5553
	Author	0.4735
	Publisher	0.3105
s2t value	0.5168	
Elapsed time	0.21 s	

Table 3. Calculation of the *t2s* coefficient

Provider concept (*Volume*)	Context of *Book*	Lexical Similarity value
Volume	*Magazine*	0.7088
	Bookshop	0.2574
	Chapter	0.3179
	Heading	0.3279
	Author	0.3944
	Pages	0.3967
t2s value	0.40	
Elapsed time	0.20 s	

In a similar way, *SECCO* computes values for *s2s* and *t2t*. In the considered example such values are: *s2s*=0.6578 and *t2t*=0.5467. The final contextual similarity between *Book* and *Volume* is 0.7057. Contextual similarity values for other couples of concepts are shown in Table 4.

Discussion of results

Similarity values obtained by the *contextual matcher* underline the fact that the *contextual similarity* between two concepts is affected by concepts and properties included in their contexts. For instance, even if *Book* and *Volume* can be linguistically considered very similar (their *lexical similarity* is 1), the *contextual matcher* correctly decreases their similarity value to 0.7057 (see equation 5) since they respectively appear in a *Bookshop* and *Library* context. Moreover, properties included in the definition of *Book* and *Volume* only share the concept of *author*. The highest contextual similarity value is obtained by the couple *Bookshop* and *Library*. Even being the two concepts linguistically not so much similar, their lexical similarity is 0.3467, if we only consider the contexts to which they belong, we can observe that the *seeker* context defines a *Bookshop* with *Place* as a property while the *provider* context defines a *Library* with *Address* as property. Both contexts refer to places containing books (one in which they are sold and another in which they are stored) that are characterized by

Table 4. Results obtained by the *contextual matcher* for some couples of concepts in Fig. 11

Seeker concept	Provider concept	Contextual Similarity	Elapsed time (s)*
Book	*Journal*	0.60567	0.26
Book	*Library*	0.3278	0.25
Book	*Proceedings*	0.1789	0.28
Bookshop	*Journal*	0.3878	0.24
Bookshop	*Library*	0.8067	0.25
Chapter	*Proceedings*	0.60879	0.16
Chapter	*Volume*	0.22345	0.28

* Times elapsed are computed on a P4 running at 3 GHz with 1Gb of memory

an attribute indicating their location. In this case, the high similarity value obtained by the *contextual matcher* will be refined by the lexical similarity value (which is lower) when weighing their individual contributions.

By continuing to evaluate further results, the couple *Book* and *Proceedings* receives a low contextual similarity value. They are not lexically very similar (their lexical similarity is 0.3987) and their respective contexts represent different things. The *seeker* context defines a set of properties of a *Book* (e.g., *author*, *pages*) and provides relations with its constituent parts (i.e., *chapter*), with the place where it can be sold (i.e., *Bookshop*) and so forth. Conversely, the *provider* context just provides information about the fact that a *Proceedings* is related to a *Volume*. Similar consideration can be done for the other couples of concepts.

In the light of these considerations, we can conclude that the *contextual matcher* is a suitable approach to interpret the use of concepts in different contexts. In fact, it can correctly interpret similarity between contexts, as in *Library* and *Bookshop*, while it is also able to interpret their dissimilarity, as in the case of the couple *Book* and *Proceedings*. However, it is worthwhile noting that it becomes more effective when combined with the *lexical matcher*, as in the case of the couple *Book* and *Volume*.

Finally, a consideration about elapsed time (see the last column of Table 4). We can see, as one can expect, that the elapsed time depends on the number of concepts/properties contained in the *seeker* and *provider* contexts. However, the elapsed time values, even in the case in which the dimension of the contexts in terms of number of concepts/properties is quite high (both the couples *Book* and *Volume* have 6 entities in total), never reach 0.3 s.

Since the *contextual matcher* fulfills the requirement of speed and seems to be a reasonable approach for exploiting contextual information of ontology concepts, it can also be included in *SECCO*.

3.4 Why Do We Need *SECCO?*

This section explains why *SECCO* has been designed and how it can be practically exploited. The main motivation for designing *SECCO* is to provide an ontology mapping algorithm in open environments (e.g., P2P, Grid). As pointed out in Section 1, there are several mapping algorithms, but there is a lack of algorithms especially designed for open environments. In such scenario, time accuracy is a mandatory requirement to perform "online" mapping and the amount of ontological information exploitable to discover mappings is quite limited. *SECCO* has been designed to provide the semantic foundation for the *K-link+* system [31]. *K-link+* is a P2P system for collaborative work based on the concept of workspace. The system allows workers to work concurrently in the same and shared environment (i.e., workspace) by a set of tools for sharing and exchanging knowledge in a semantic way. In such an open architecture, it would be very useful to discover and interact with semantically neighbor peers. The concept of semantic proximity can be represented by exploiting *SECCO*. In fact, mappings discovered by *SECCO*, establish *semantic links* among peers of a *K-link+* network. These links can be exploited in the following ways:

- *Semantic based search*: contents (e.g., web pages, documents) can be annotated to ontology concepts in order to provide them with an *explicit* and *machine understandable* semantic meaning. Therefore, content search can be performed by specifying ontology concepts instead of keywords. Retrieving similar concepts by *SECCO* will result in discovering contents annotated to such concepts.
- *Semantic building of workspaces*: *semantic links* between peers are supposed to reflect common interests shared by the peers involved in these links. Therefore, by following these links peers with common interests can be discovered and grouped together.
- *Semantic query routing*: *semantic links* can be exploited to forward queries. When a query reaches a peer, it can forward this query to other peers with which it has *semantic links*. This way a new semantic path between "unknown" peers can be constructed. Moreover, the amount of network traffic generated by queries (as compared with flooding techniques) can be significantly reduced by adopting a semantic-aware routing strategy.

We want to point out how, in designing a comprehensive semantic P2P solution, the central problem is to find out *semantic links* among peers. Once found, these can be exploited for several purposes. Therefore, differently from other approaches (e.g., [1, 24]) where the preexistence of mapping ensures semantic interoperability among peers, we provide a comprehensive solution that tackles the problem of designing semantic P2P systems from all the perspectives, that is, construction of the semantic overlay (provided by *SECCO*) and underling physical P2P architecture (provided by *K-link+*).

4 *SECCO: A* Double Evaluation

In this section, we show how the requirements that driven the design of *SECCO* are fulfilled in real case scenarios. In Sections 3.1-3.3 the matchers of *SECCO* have been individually described and how they cope with the requirements of fastness and accuracy has been shown. The *syntactic matcher* has been evaluated on the OAEI2006 real life ontologies (see Fig. 9). The *lexical matcher* has been extensively evaluated through the similarity experiment whose results are available at http:// grid. deis. unical.it/ similarity. The rationale of the *contextual matcher* has been described through the example depicted in Fig. 11.

In this section, we want to evaluate *SECCO* as a whole. The evaluation has been split in two parts (referred to as *Experiment 1* and *Experiment 2* in the following). In *Experiment 1* (see Section 4.1), we evaluated *SECCO* by comparing it with H-Match [8,9] that actually is the only system designed for mapping ontologies in open environments offering very similar features. In Section 4.1.1 we perform a sensitivity analysis of the assignment of weights to the individual matchers and observe how results provided by *SECCO* in *Experiment 1* and the correlation w.r.t those produced by H-Match vary. In *Experiment 2* (see Section 4.2), we evaluate how *SECCO* performs as a general mapping algorithm. In this experiment, we evaluate it on four real-life ontologies included in the OAEI 2006 benchmark test suite [37], and compare its results with those of other algorithms not explicitly designed for ontology mapping in P2P networks. We evaluate *SECCO* only on ontologies 301-304 of the OAEI 2006 in order to have an indicator of how it performs in real case scenarios.

4.1 Experiment 1*:* Comparing *SECCO* with H-Match

This section presents the comparison of *SECCO* w.r.t H-Match on two excerpts of (online available) ontologies. The first ontology (*Ka*) describes research projects while the second one (*Portal*) describes content of a Web portal. We suppose that *Ka* belongs to a *seeker* peer while *Portal* to a *provider* peer. These ontologies have also been adopted to evaluate the H-Match system as described in [8]. We have chosen to adopt the same two ontologies in order to have an objective comparison between the two approaches. Fig. 12 shows two excerpts of *Ka* and *Portal* describing the concept of *Publication* are shown.

In this evaluation, we aim at constructing, by exploiting *SECCO*, a mapping (see Definition 6) between the concept *Publication* in *Ka* and some concepts belonging to *Portal*. In particular, we want to emphasize how *SECCO* can profitably discover similarities even among terms apparently not related and how it behaves w.r.t H-Match.

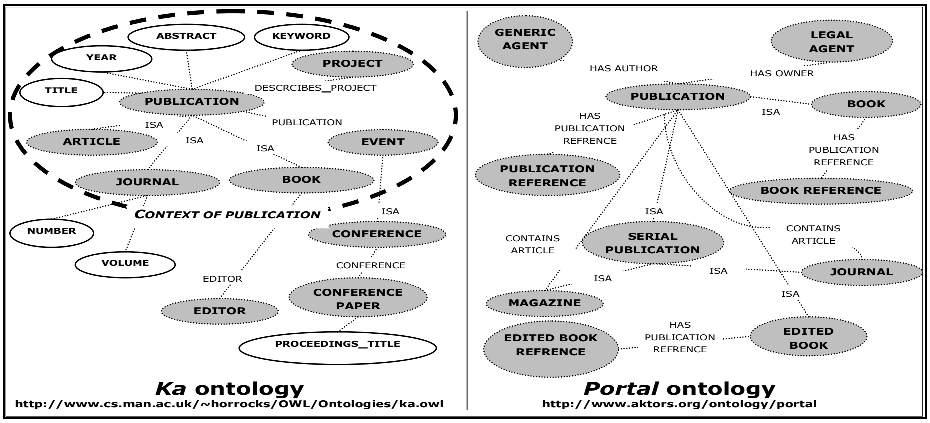


Fig. 12. Excerpts of the *Portal* and *Ka* ontologies defining the concept *Publication*. The context of *Publication* (dashed area) is also shown.

Configuration of SECCO for Experiment 1

In this experiment, the input *I* of *SECCO* (see Section 2.3) takes the values shown in Table 5. We do not set a threshold value (the T_h parameter) since we want to create one-to-many mappings.

Table 5. The input *I* of *SECCO* for *Experiment 1*

Parameter		**Value**
C_s	Seeker Concept	Publication
ctx(C_s)	Seeker Context	ctx(Publication)
O	Provider Ontology	Portal (the excerpt shown in Fig.12)
T_h	Threshold	0
w_s	Syntactic similarity weight	0.1
w_L	Lexical similarity weight	0.6
w_c	Contextual similarity weight	0.3

Since we want to give more emphasis to the semantic component of the algorithm, we consider *lexical similarity* more reliable than *syntactic similarity* or *contextual similarity* (i.e., we assign a higher value to w_L). A detailed analysis on how the assignment of weights can affect results will be provided in Section 4.1.1.

Results obtained by SECCO for Experiment 1

Table 6 shows the results obtained by *SECCO* with the input *I* (see Table 5) along with overall elapsed times.

Table 6. Results obtained by *SECCO* by considering *Publication* as *seeker* concept

		Similarity Values				
***Ka* concept**	***Portal* concept**	**Syntactic**	**Lexical**	**Contextual**	**Overall**	**Elapsed time (s)**[†]
Publication	*Publication*	1	1	0.697	**0.909**	0.49
Publication	*Book*	0	0.823	0.199	**0.553**	0.29
Publication	*Journal*	0	0.767	0.221	**0.526**	0.31
Publication	*Magazine*	0	0.737	0.088	**0.468**	0.29
Publication	*Edited Book*	0	0.823	0.674	**0.696**	0.29
Publication	*Publication Reference*	0.3	0.549	0.118	**0.395**	0.27
Publication	*Book Reference*	0	0.549	0.118	**0.365**	0.28
Publication	*Edited Book Reference*	0	0.549	0.118	**0.365**	0.31

[†] Elapsed times are computed on a P4 running at 3 GHz with 1Gb of memory

These examples show the suitability of the *lexical matcher* which allows to discover mappings in a *semantic* way. In fact, by considering the analyzed couples of concepts only from a syntactic point of view we would obtain similarity values equal to 0 apart from the couples *Publication (Ka)* and *Publication* (*Portal*) and *Publication (Ka)* and *Publication Reference* (*Portal*). In the following, we compare these results with those obtained by H-Match.

Discussion of results and comparison with H-Match

Comparing ontology mapping algorithms is a hard task, especially when an objective and reliable reference alignment is not provided. Moreover in a P2P scenario, since a mapping algorithm usually aims at finding one-to-many mappings (it provides a similarity ranking between concepts) it is very difficult to interpret ranking values. In literature, there exist very few algorithms that address the ontology mapping problem in P2P environments. The approach closer to *SECCO* is H-Match. In order to make an objective comparison between them, we considered the results obtained by H-Match for the same couples of concepts on which *SECCO* has been evaluated.

In the example depicted in Fig. 11 authors in [9] provide only a similarity value (related to the couple *Book* and *Volume*). For this couple, H-Match obtained the results shown in Table 7.

Table 7. Comparison between *SECCO* and H-Match on the example of Fig. 11

Couple	*SECCO*	H-Match			
		Shallow	*Intermediate*	*Deep*	*Average*
Book-Volume	0.8117	1	0.78	0.70	0.8266

The overall similarity value between the couple *Book* and *Volume* obtained by *SECCO* is computed as follows:

$$\text{sim}(\textit{Book}, \textit{Volume}) = w_s * \text{sim}_{syn} + w_L * \text{sim}_{lex} + w_C * \text{sim}_{con}$$

Therefore, we obtain:

$$\text{sim}(\textit{Book}, \textit{Volume}) = 0.1*0+0.6*1+0.3*0.7057=0.8117$$

The *lexical matcher* correctly interprets the linguistic similarity between the *Book* and *Volume* concepts; in fact, it gives 1 as output. The high value of lexical similarity is because the *Book* and *Volume* concepts belong to the same WordNet synset and therefore are synonyms. Same things are valid for H-Match, whose *shallow* matching model has similar features to the *lexical matcher* of *SECCO*. The *contextual matcher* of *SECCO*, since *Book* and *Volume* respectively appear in a *Bookstore* context and a *Library* context, correctly decreases the overall similarity value (this aspect has been discussed is Section 3.3).

H-Match obtains a similarity score of 0.78 with the *intermediate* matching model, which takes into account concept names and properties. Through the *deep* matching model, which considers the whole context of concepts (i.e., all the properties) H-match obtains 0.70. This matching model is the most similar to that implemented by *SECCO*. The average value given by H-Match, obtained by averaging results of the three matching models, is 0.8266, which is very close to the result obtained by *SECCO*. Therefore, in this case, we can conclude that the similarity value between *Book* and *Volume* obtained by *SECCO* is comparable with that obtained by H-Match.

A more detailed comparison between the two approaches can be done by considering the two excerpts of the *Ka* and *Portal* ontologies depicted in Fig. 12. Similarity values obtained by both *SECCO* and H-Match [8] are shown in Table 8. For the sake of comparing only semantic features of the two approaches we do not considered the contribution of the *syntactic* similarity of *SECCO* for the couples *Publication* (*Ka*) and *Publication* (*Portal*) and *Publication* (*Ka*) *Publication Reference* (*Portal*).

Table 8. Comparison between *SECCO* and H-Match on the example of Fig. 12

***Ka* concept**	***Portal* concept**	***SECCO***	**H-Match**				
			Surface	*Shallow*	*Deep*	*Intensive*	*Average*
Publication	*Publication*	**0.909**	1	0.7384	0.8047	0.7814	0.8318
Publication	*Book*	**0.553**	0.8	0.6184	0.66	0.6394	0.6795
Publication	*Journal*	**0.526**	0.64	0.5224	0.5538	0.5381	0.5636
Publication	*Magazine*	**0.468**	0.8	0.6184	0.6498	0.6341	0.6756
Publication	*Edited Book*	**0.696**	0.64	0.5224	0.5641	0.5434	0.5675
Publication	*Publication Reference*	**0.395**	0.64	0.5531	0.5741	0.5503	0.5794
Publication	*Book Reference*	**0.365**	0.64	0.5531	0.5733	0.5497	0.5790
Publication	*Edited Book Reference*	**0.365**	0.64	0.5531	0.5637	0.5420	0.5747

In this experiment, it is interesting noting that the higher similarity values obtained by *SECCO* and H-Match are related to the couple *Publication* (*Ka*) and *Publication* (*Portal*). These two values are very close. Same considerations are valid for the couple *Publication (Ka)* and *Book (Portal).* An interesting consideration can be done for the last three rows of Table 8. While *SECCO* obtains low similarity values, H-Match obtains values that always exceed 0.5. For instance, for the couple *Publication* and *Publication Reference*, *SECCO* obtains 0.395 while H-Match obtains 0.5794 as average result. However, by objectively analyzing the concepts, one can assess that these concepts are not much similar. In fact, the first describes the concept of *Publication* while the second defines a reference to a *Publication.*

It is very difficult to comparing results between the two strategies with very few matching of couples, as in the case of *Book* and *Volume* (see Table 7). Moreover, comparing mapping results, without a reference alignment, implicitly includes a certain degree of subjective interpretation.

In order to obtain an overall indicator of how the two approaches are (un)related, we computed the Pearson correlation coefficient [13] between their results. This coefficient represents an agreement between the values of two data sets (in our case between similarity results) by expressing the degree of association between them (see Table 9).

Table 9. Correlation between results of *SECCO* and H-Match shown in Table 8

	H-Match					
	Surface	*Shallow*	*Deep*	*Intensive*		*Average*
SECCO	0.898	0.8559	0.9051	0.908		0.8919

As can be noticed the higher value of correlation is 0.908 meaning that results obtained by *SECCO* are closer to these obtained by H-Match through the *intensive* matching model. Through this model of matching, H-Match considers both linguistic feature of ontology concepts and whole context of concepts (in terms of properties and semantic relations) in which they appear. In addition, also the correlation w.r.t the *deep* model is high. The average correlation value is 0.8919, which underlines how he two approaches are very close. In fact, a value of correlation higher that 0.7 can be interpreted as an indicator of high similarity [44]. It is very important notice that *SECCO* performs very close to those of H-Match even if *SECCO* does not adopt complex matching strategies.

Since both approaches heavily rely on linguistic features of ontologies, we also computed the correlation (see Table 10) between results of our *lexical matcher* that relies on WordNet and the *surface matching* model of H-Match that relies on an ad-hoc thesaurus built by exploiting WordNet.

Table 10. Correlation between the *lexical matcher* of *SECCO* and the *surface* matching model of H-Match

	H-Match
	Surface
SECCO	
Lexical matcher	0.7123

As can be noticed, the value of correlation is high even if it is very difficult to estimate which approach is more accurate. However, the *lexical matcher* of *SECCO* is not an ad-hoc thesaurus but it is able to exploit the whole structure of WordNet by including in the similarity computation a wide set of semantic relations between concepts. Moreover, the metrics included in the *lexical matcher* have been extensively evaluated by the similarity experiment [26].

4.1.1 Discussion on Similarity Aggregation and Assignment of Weights

SECCO, in order to perform similarity aggregation, adopts a weighted sum of similarity values given by the individual matchers. Do and Rahm [14] address some aspects of weights assignment and similarity aggregation for database structures. A similarity aggregation function is a function that takes results from several matchers, weights these results, and gives as output an overall similarity indicator. The weights are assigned manually or learned, e.g., using machine learning on a training set. Berkovsky et al. [5] have thoroughly investigated the effects of different weights on the alignment results.

We chose to adopt a strategy based on multiple matchers since experimental results have shown that a combination of similarity measures (provided by different matchers) leads to better alignment results than using only one matcher at a time. We realize that this technique needs a certain degree of expertise from the *SECCO* user. In facts, if the different weights are not correctly assigned, mapping results can be affected. However, notice that in a P2P scenario it is not possible to a priori analyze the structure of ontologies to be compared, in order to find the best mapping strategy (as done in [25]), since peers are not aware of the ontologies of other peers.

In the experiments, we manually settled values of the different weights (i.e., the w_s, w_L, w_c parameters). However, it would be interesting to see how the correlation coefficient w.r.t H-Match (*Experiment 1*) changes when assigning different weights. Table 11 shows correlation values between the two approaches by assigning different values of w_s, w_L and w_c. For sake of space, we do not report, for each variation of the weights, the similarity values obtained by *SECCO*.

Table 11. Correlation between results of *SECCO* and H-Match by varying the weights of the individual matchers on the couples of concepts listed in Table 8

SECCO			***H-Match***					
Syntactic similarity w_S	*Lexical similarity* w_L	*Contextual similarity* w_c	*Correlation w.r.t the different matching models*					*Average correlation*
			surface	*shallow*	*deep*	*intensive*		
0.1	***0.6***	***0.3***	0.898	0.8559	0.9051	0.908		0.8917
0.3333	***0.3333***	***0.3333***	0.9023	0.8657	0.9102	0.9117		0.8974
0.3	***0.4***	***0.3***	0.8415	0.8367	0.8567	0.8645		0.8498
0.3	***0.3***	***0.4***	0.8218	0.8123	0.8657	0.8756		0.8438
0.1	***0.3***	***0.6***	0.7656	0.7567	0.8123	0.8198		0.7886
0.1	***0.1***	***0.8***	0.4978	0.4478	0.5218	0.5123		0.4949

An interesting consideration arises from results shown in Table 11. As it can be noticed, if we assign equal weights to the matchers (row 2) the correlation raise up to 0.9117 with the *intensive* model of H-Match and to 0.8974 in the average. Moreover, it is interesting to point out that if we assign a little higher value to the *contextual matcher* (row 4), the correlation remains high (0.8756 for the intensive model and 0.8438 in the average). Even in the case in which contextual similarity has a higher value (row 5), the average correlation value remains quite high. Conversely, if we give much more emphasis to the *contextual matcher* (row 6), the average correlation drastically decrease to 0.4949 in the average. As final remark, we can conclude that assigning equal weight to the matchers can increase the correlation value w.r.t H-Match, that does not necessarily mean better results since in the considered example alignments are not provided. However, in the light of these considerations in *Experiment 2* we assign equal weights to the different matchers.

4.2 Experiment 2*:* Comparing *SECCO* with Other Ontology Mapping Algorithms Not Designed for Ontology Mapping in P2P Networks

This section provides an extensive evaluation of *SECCO* on four real-life ontologies contained in the OAEI 2006 [37] test suite. We compared *SECCO* with other mapping algorithms not explicitly designed to tackle the OMP in P2P networks. This way we want to show how much the designing strategy of *SECCO*, which has to ensure fastness and cannot exploit the whole structures of ontologies to be mapped, affects accuracy (i.e., quality of results).

In particular, we focused on the group of tests that contain four real-life ontologies (i.e. tests from 301 to 304) in order to investigate how *SECCO* performs in mapping real ontologies. For each of these ontologies the OAEI organizers provided a reference alignment. We computed measures of Precision (i.e., the number of correct mapping among all the mapping found), Recall (i.e., the number of correct mapping among all the existing mapping) and F-measure [15] (i.e., the harmonic mean of Precision and Recall). In particular, we compared results obtained by *SECCO* with those provided by the OAEI organizers.

Notice that *SECCO*, even being designed for P2P networks and therefore to work "online", can be exploited to compare entire ontologies by reiterating the process described in Section 2 for each concept in the source ontology (i.e., the reference ontology 101 contained in the OAEI tests).

Configuration of SECCO for Experiment 2

Table 12 shows the values of the input of *SECCO* for this experiment. Here we are interested in obtaining one-to-one mappings.

Table 12. *SECCO* configuration for *Experiment 2*

Parameter	Value
C_s Seeker Concept	Each concept C_i contained in the reference ontology (i.e., 101)
ctx(C_s) Seeker Context	ctx(C_i)
O Provider Ontologies	301-302-303-304
T_h Threshold	0.51
w_s Syntactic similarity weight	0.333
w_L Lexical similarity weight	0.333
w_c Contextual similarity weight	0.333

Results obtained by SECCO for Experiment 2

Fig. 13 shows values of Precision, Recall and F-Measure obtained by *SECCO*. As can be noticed, *SECCO* performs well. It always obtains a Precision around 0.9. The Recall, reaches the highest value (i.e., 0.9387) for ontology 304 while the lowest value (i.e., 0.6211) for ontology 302. However, it always remains higher than 0.5. The F-Measure values are 0.8269 for ontology 301, 0.7375 for ontology 302, 0.8012 for ontology 303 and 0.949 for ontology 304. Values of F-Measure that represent an overall indicator of the performance of a mapping algorithm are in all the cases high.

Discussion of results and comparison with other ontology mapping algorithms

In order to have an objective evaluation of *SECCO*, we decided to compare its average results with those of other ontology mapping approaches. The results are shown in Table 13. *SECCO* obtained an average Precision of 0.81, an average Recall of 0.81 and an average F-measure of 0.81. As can be noticed, *SECCO* is one of the most precise algorithms. It is only slightly outmatched by Automs and Falcon.

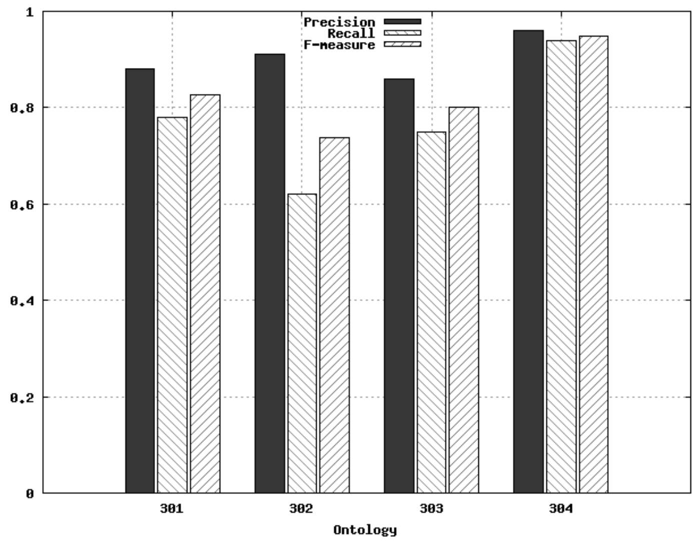


Fig. 13. Results of *SECCO* on the OAEI 2006 real life ontologies

In terms of Recall *SECCO* is outperformed only by RiMOM. In terms of F-Measure, *SECCO* is only dominated by Falcon and RiMOM.

An important consideration emerges from these results. *SECCO* is an ontology mapping algorithm that in its current implementation cannot exploit the whole structural information encoded in ontologies. Conversely, most of the presented approaches have a solid structural matching strategy. For instance, Falcon relies on the GMO approach [51] that exploits a graph-matching algorithm for discovering mappings while RiMOM exploits an adaptation of the Similarity Flooding algorithm. Such strategies require a complex analysis of the ontologies that is not conceivable in a P2P environment for two reasons: (i) peers are not aware of the whole ontologies of other peers; (ii) the fundamental requirement of fastness in P2P networks can be affected. It is worthwhile noting that *SECCO* obtains very good results without using that strategy.

Table 13. Average results obtained by some ontology mapping algorithms on the OAEI 2006 real-life ontologies, as reported in [19]

	SECCO‡	*Jhu/apl [6]*	*Automs [29]*	*Falcon [25]*	*RiMOM [55]*	*H-Match [10]*
Precision	**0.81**	**0.18**	**0.91**	**0.89**	**0.83**	**0.78**
Recall	**0.81**	**0.50**	**0.70**	**0.78**	**0.82**	**0.57**
F-Measure	**0.81**	**0.26**	**0.79**	**0.83**	**0.82**	**0.65**
Average Elapsed Time (s)	**3.05 s**	**Na**	**70.25 s**	**7.22 s**	**3.14 s**	**Na**

‡ Elapsed times are computed on a P4 running at 3 GHz with 1Gb of memory.

Notice that the elapsed time by *SECCO* is the lowest. In particular, it is 25 times lower that that obtained by *Automs* that also exploits WordNet and about 3 times lower than that of Falcon, which adopts a structural matching strategy. Moreover, the comparison with H-Match, the system actually very similar to *SECCO*, shows how *SECCO* is better in terms of Precision, Recall and F-Measure. It would be also interesting to compare the approaches in terms of elapsed time but unfortunately, authors in [10] do not provide information about execution times.

On the one side, these results show how a structural mapping strategy can improve mapping results as in the case of Falcon. On the other side, they show that *SECCO* obtains results comparable with those of the most performant ontology mapping algorithms without adopting complex structural analysis of ontologies. Finally, we can conclude *SECCO* is faster than other mapping algorithms and the cost paid, in terms of accuracy, is not so high.

5 Related Work

Recently several ontology mapping algorithms have been proposed. A detailed survey is given in [11]. In that survey, only a bibliographic reference to ontology mapping in P2P systems is listed. This underlines the fact that the OMP has not adequately been tackled in open environments. In literature, there are few approaches similar to *SECCO* explicitly designed for mapping ontologies in open environments.

The CtxMatch algorithm [7] aims at discovering mappings between Hierarchical Categories (HCs). It relies on WordNet for interpreting the correct sense of concepts in the context in which they appear. Therefore, it performs a transformation of the concepts to be compared in Description Logics axioms that are exploited to reduce the problem of discovery mappings to a SAT problem. CtxMatch similarly to *SECCO* implements a semantic based approach since it relies on WordNet. However, the main difference between these systems is that CtxMatch focuses on matching HCs and provides as output a semantic relation between terms while *SECCO* can also deal with ontologies and provides a confidence value.

H-Match [8, 9] is an algorithm for dynamically matching concepts in distributed ontologies. H-Match allows for different kinds of matching depending on the level of accuracy needed. The system aims at supporting knowledge sharing and ontology-addressable content retrieval in peer-based systems. It is actually the system closer to *SECCO*. Indeed, there are at least two main differences between these approaches:

- The lexical matcher of H-Match is based on a ad-hoc thesaurus, while that of *SECCO* is based on WordNet. H-Match defines an ad hoc similarity metric between concepts in the thesaurus. Conversely, *SECCO* can benefit from a similarity metric evaluated by the similarity experiment [26]. The metric adopted in *SECCO* is highly correlated w.r.t similarity judgments given by human[1].
- H-Match in performing contextual affinity exploits predefined weights assigned to the different types of relations among concepts. It introduces five types of relations (i.e., same-as, part-of, kind-of, contains, associates) among terms of a

[1] For preliminary experimental results refer to: http://grid.deis.unical.it/similarity

peer ontology. These relations are assessed by exploiting relations that concepts have in WordNet. Conversely, *SECCO* adopts the "how it fits" strategy from which the contextual similarity indicator can emerge by combining measures of semantic similarity, to take into account hypernymy/hyponymy relations among synsets, and relatedness to take into account a broader range of semantic relations (e.g., part of).

We deeply compared *SECCO* with H-Match concluding that results obtained by the two approaches are, for several aspects, comparable. However, *SECCO* performs a little better on real-life ontologies included in the OAEI 2006 tests. Since the two approaches are both designed to work in open environments, it would be interesting also to compare them in terms of performance (i.e., execution time for computing mappings).

Falcon-AO [25] is an automatic tool for aligning ontologies based on three alignment strategies: the I-Sub [48] metric is exploited to compare strings, the V-Doc [42] is a linguistic matcher based on Information Retrieval, while the GMO [51] is a matcher based on graph matching.

The RiMOM system [55] combines different strategies to assess ontology mappings. In particular, it includes an edit distance metric and an adaption of the similarity flooding algorithm to the context of ontology mapping.

iMapper [50] is an ontology mapping tool based on the idea of semantic enrichment. It makes use of ontology instances to calculate the similarity between concepts. The mapping process is split in two phases. In the first one (i.e., enrichment phase) documents (i.e., instances) associated to ontology concepts are analyzed thus building the enriched ontology. The association of documents to concepts can be done automatically, but user refinement it is also allowed. The output of this phase are representative vectors (one for each concept) built from the textual content of their associated documents. In the second phase (i.e., mapping phase) similarities between ontology elements are computed as the cosine between their representative vectors. Further refinements are employed to re-rank the results via the use of WordNet.

The abovementioned systems discover ontology mappings by exploiting both structural and linguistic information encoded in ontology entities. However, in order to work properly they need to scrutinize the two ontologies to be mapped. For instance, the structural matcher of Falcon is based on a graph matching algorithm (i.e., the GMO matcher) which requires to construct the adjacency matrix of the two ontologies. In addition, the lexical matcher of Falcon (i.e., the V-Doc matcher [42]) requires analyzing both the ontologies to be mapped. Similar things hold for RiMOM, which adopts as a structural matcher a variant of the Similarity Flooding algorithm. iMapper needs to access the whole two ontologies and requires ontology concepts to be associated with instances. As can be notice, a common denominator among these approaches is that they "need to know" the whole two ontologies. Conversely, *SECCO* does not impose this requirement since as usually happens in P2P networks peers are not aware of one another's ontologies. Therefore, it would be interesting to see how the abovementioned approaches perform without completely knowing the two ontologies to be mapped.

In the literature, there are some semantic P2P applications sharing common characteristics with *SECCO*.

SWAP (Semantic Web and Peer to Peer) [18] aims at combining ontologies and P2P for knowledge management purposes. SWAP allows local knowledge management through a component called LR (Local node repository), which gathers knowledge from several sources and represents it in RDF-Schema. In SWAP, each node is responsible for a single ontology: ontologies might represent different views of a same domain, multiple domains with overlapping concepts, or might be obtained by partitioning an upper level ontology. Knowledge sharing is obtained through ontology mapping and alignment.□

GridVine [1] is a semantic P2P system whose aim is to build a semantic overlay network based on two layers: *logical layer* and *physical layer*. The logical layer provides a set of functionalities such as: attribute-based search, schema management and schema mapping. The physical layer is used as support to the logical layer in constructing the overlay and forwarding queries. In GridVine, semantic interoperability is achieved by *semantic gossiping* [2]. Semantic gossiping assumes the *preexistence* of local agreements provided as mappings between different schemas. Peers introduce their own schemas and exchanging translations between them can incrementally come up with an implicit "consensus schema".

Piazza [24] is a P2P Data Management system whose main aim is to enable efficient query processing. Piazza takes into account the structure of the knowledge domain and documents in order to achieve interoperability between different information sources. Similarly, to GridVine it assumes the preexistence of mappings between data sources. Therefore, these mappings are *chained* together and exploited for query rewriting/answering.

In the SWAP system, mappings between peer ontologies are dynamically obtained by exploiting techniques based on lexical features, structure and instances of ontologies. Conversely, neither the GridVine nor the Piazza approach tackle the problem of discovering mappings among the different representations (i.e., schema, ontologies) belonging to different peers since they assume the preexistence of mappings.

6 Concluding Remarks

This paper described *SECCO*, an ontology mapping algorithm aimed at discovering *concept mappings* in P2P networks. A *concept mapping* has been defined as a similarity ranking between a *request* (composed by a concept along with its context) performed by a *seeker* peer and concepts belonging to *provider* peer ontologies. Since we assume that peers are not aware of one another's ontologies, in order to discover mappings, we designed an ad-hoc mapping strategy. This strategy aims at fulfilling two important requirements (i.e., fastness and accuracy) through three individual matchers. The main problem we faced is related to the fact that we cannot adopt sophisticated and time-consuming structural matching strategies that require to know the whole two (peer) ontologies to be compared. Hence, we adopted the notion of *context*, defined as a concept along with its properties (obtained as described in Section 2.2) and nearest neighbor concepts. Through contexts, we aim at encoding the amount of structural information needed in a particular request. We compare the contextual information of different concepts by the "how it fits" strategy that is founded on the idea that two concepts are related if they fit well in each other's context. This strategy is supported by the *lexical matcher* whose aim is to exploit an accurate (proven by the similarity experiment [26]) similarity metric in WordNet. This metric allows assessing similarity even among syntactically unrelated concepts. Moreover, in order to exploit all the linguistic information of ontology entities (i.e. ontology meta-data) we adopt the *syntactic matcher*. This matcher encodes linguistic information in *virtual documents* that are created and compared by an information retrieval approach. All these matching strategies have been extensively evaluated.

Along the paper, we discussed the exploiting of *SECCO* in the context of P2P networks and proven through experimental evaluation the suitability of the algorithm. In particular, *SECCO* has been compared (*Experiment 1*) with the H-Match algorithm, designed for ontology mapping in open environments, with very promising results. Furthermore, *SECCO* has been compared (*Experiment 2*) with other mapping algorithms not explicitly designed for mapping in P2P networks and even in this case results are satisfactory. We also performed a sensitivity analysis from which emerged an interesting aspect related to weight assignments to the different matchers.

7 Future Work

Here we briefly describe possible improvements of the algorithm. First in a future version of *SECCO* we aim at distinguishing relations between concepts from relations that describe properties of concepts. This way, the definition of *context* exploited by *SECCO* will give much emphasis to the relations that have *per se* a semantic meaning as for instance the ISA relations. We are also performing further improvements of *SECCO* along two directions.

On the one side, we are investigating a strategy for automatically tuning the weights of the different matchers and aggregating results. In particular, we are evaluating the following possibilities:

- The use of sophisticated techniques such as the Dempster Shafer theory for combining results of the different matchers. The Dempster-Shafer theory [47] is a mathematical based on *belief functions* and *plausible reasoning*, which is used to combine separate pieces of information (evidence) to calculate the probability of an event. In our case, we aim at exploiting this strategy for combining uncertain results given by the different matchers for obtaining a more reliable overall similarity value.
- The use of a linear aggregation formula. According to this strategy, a weight of 1 is given to results provided by each matcher. The overall similarity is obtained as the average similarity value given by the different matchers.

On the other side, we aim at exploiting the World Wide Web for refining similarity values among concepts. In fact, we argue that the Web could be a valuable source of knowledge. Our aim is to design a similar strategy based on the analysis of relations between terms extracted from the snippets (related to concepts to be compared) given by a search engine.

Finally, since we included *SECCO* as semantic support in our *K-link*+ [31] system, we also would like to evaluate its performance within *K-link*+. In this case, we are interested in evaluating *SECCO* in a complete semantic P2P solution for cooperation and contents sharing and retrieval.

References

1. Aberer, K., Cudré-Mauroux, P., Hauswirth, M., Van Pelt, T.: GridVine: Building internet-scale semantic overlay networks. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 107–121. Springer, Heidelberg (2004)
2. Aberer, K., Cudré-Mauroux, P., Hauswirth, M.: The Chatty Web: Emergent Semantics Through Gossiping. In: Proc. of WWW 2003, Budapest, Hungary, pp. 197–206 (2003)
3. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American (2001)
4. Berners-Lee, T.: The Semantic Web: An interview with Tim Berners-Lee. Consortium Standards Bulletin 4(6) (June 2005)
5. Berkovsky, S., Eytani, Y., Gal, A.: Measuring the relative performance of schema matchers. In: Proc. of WI 2005, Compeigne, France, pp. 366–371 (2005)
6. Bethea, W.L., Fink, C.R., Beecher-Deighan, J.S.: JHU/APL Onto-Mapology Results for OM 2006. In: Proc. of OAEI 2006, Athens, Georgia, USA (2006)
7. Bouquet, P., Serafini, L., Zanobini, S.: Semantic coordination: a new approach and an application. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 130–145. Springer, Heidelberg (2003)
8. Castano, S., Ferrara, A., Montanelli, S., Racca, G.: From Surface to Intensive Matching of Semantic web Ontologies. In: Proc. of WEBS 2004, Zaragoza, Spain, pp. 140–144 (2004)
9. Castano, S., Ferrara, A., Montanelli, S.: H-MATCH: an Algorithm for Dynamically Matching Ontologies in Peer-based Systems. In: Proc. of SWDB, Berlin, Germany, pp. 231–250 (2003)
10. Castano, S., Ferrara, A., Messa, G.: Results of the HMatch Ontology Matchmaker in OAEI. In: Proc. of OM 2006, Athens, Georgia, USA, pp. 134–143 (2006)
11. Choi, N., Song, I., Han, H.: A survey on Ontology Mapping. SIGMOD Record 35(3), 34–41 (2006)
12. Davies, J., Studer, R., Warren, P. (eds.): Semantic Web Technologies - trends and research in ontology-based systems. Wiley, Chichester (2006)
13. Devore, J.L.: Probability and Statistics for Engineering and the Sciences. International Thomson Publishing Company
14. Do, H., Rahm, E.: COMA – a system for flexible combination of schema matching approaches. In: Proc. of VLDB 2002, Hong Kong, China, pp. 610–621 (2002)
15. Do, H., Melnik, S., Rahm, E.: Comparison of schema matching evaluations. In: Proc. GI-Workshop Web and Databases, Erfurt, Germany, pp. 221–237 (2002)
16. Ehrig, M., Staab, S.: Qom - fast ontology mapping. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 289–303. Springer, Heidelberg (2004)
17. Ehrig, M., Sure, Y.: Ontology mapping - an integrated approach. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 76–91. Springer, Heidelberg (2004)
18. Ehrig, M., Tempich, C., Broekstra, J., Van Harmelen, F., Sabou, M., Siebes, R., Staab, S., Stuckenschmidt, H.: SWAP: Ontology-based Knowledge Management with Peer-to-Peer Technology. In: Proc. of WOW, Luzerne, Switzerland (2003)
19. Euzenat, J., Mochol, M., Shvaiko, P., Stuckenschmidt, H., Šváb, O., Svátek, V., van Hage, W.R., Yatskevich, M.: Results of the Ontology Alignment Evaluation Initiative 2006. In: Proc. of OM 2006, Athens, Georgia, USA (2006)
20. Euzenat, J.: An API for ontology alignment. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 698–712. Springer, Heidelberg (2004)

21. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Heidelberg (2007)
22. Goble, C.A., De Roure, D.: The Semantic Grid: Myth busting and bridge building. In: Proc. of ECAI 2004, Valencia, Spain, pp. 1129–1135 (2004)
23. Gruber, T.R.: A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition 5(2), 199–220 (1993)
24. Halevy, A.Y., Ives, Z.G., Jayant Madhavan Mork, P., Suciu, D., Tatarinov, I.: Piazza: Data Management Infrastructure for Semantic Web Applications. In: Proc. of WWW 2003, Budapest, Hungary, pp. 556–567 (2003)
25. Hu, W., Cheng, G., Zheng, D., Zhong, X., Qu, Y.: T Results of Falcon- OM in the OAEI, Campaign. In: Proc. of.OM 2006, Athens, Georgia, USA, pp. 124–133 (2006)
26. Java WordNet Similarity Library (JWSL) and the Similarity Experiment, `http://grid.deis.unical.it/similarity`
27. Jiang, J., Conrath, D.: Semantic similarity based on corpus statistics and lexical taxonomy. In: Proc. of ROCLING X, Taiwan (1997)
28. Klyne, G., Caroll, J.J.: Resource Description Framework (RDF): Concepts and abstract Syntax. W3C Recommendation (February 10, 2004) (October 2007), `http://www.w3.org/TR/rdf-concepts/`
29. Kotis, K., Valarakos, A., Vouros, G.: AUTOMS: Automated Ontology Mapping through Synthesis of methods. In: Proc. of OM 2006, Athens, Georgia, USA (2006)
30. Lauer, M.: Designing Statistical Language Learners: Experiments on Noun Compounds. In: Proc. of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL 1995), Cambridge, Massachusetts, USA, pp. 47–54 (1995)
31. Le Coche, E., Mastroianni, C., Pirrò, G., Ruffolo, M., Talia, D.: A peer-to-peer virtual office for organizational knowledge management. In: Reimer, U., Karagiannis, D. (eds.) PAKM 2006. LNCS, vol. 4333, pp. 166–177. Springer, Heidelberg (2006)
32. Levenshtein, I.V.: Binary Codes Capable of Correcting Deletions, Insertion and Reversals. Soviet Physics-Doklady 10(8), 707–710 (1966)
33. Lucene- The Apache Lucene project (October 2007), `http://lucene.apache.org`
34. Miller, G.A., Charles, W.G.: Contextual Correlates of Semantic Similarity. Language and Cognitive Processes 6(1), 1–28 (1991)
35. Miller, G.: WordNet An On-line Lexical Database. International Journal of Lexicography 3(4), 235–312 (1990)
36. Mitra, P., Noy, N.F., Jaiswal, A.R.: OMEN: A Probabilistic Ontology Mapping Tool. In: Proc. of MCN 2004, Hiroshima, Japan, pp. 71–83 (2004)
37. Ontology Alignment Evaluation Initiative (October 2007), `http://oaei.ontologymatching.org`
38. Pan, R., Ding, Z., Yu, Y., Peng., Y.: A Bayesian Network Approach to Ontology Mapping. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 563–577. Springer, Heidelberg (2005)
39. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL Web Ontology Language Semantic and Abstract Syntax. W3C Recommendation (February 10, 2004) (October 2007), `http://www.w3.org/TR/owl-semantics/`
40. Patwardhan, S., Pedersen, T.: Using WordNet-based context vectors to estimate the semantic relatedness of concepts. In: Proc. of EACL 2006, workshop, pp. 1–8 (2006)
41. Pirrò, G., Talia, D.: An approach to Ontology Mapping based on the Lucene search engine library. In: Proc. of SWAE 2007, Regensburg, Germany, pp. 407–412 (2007)
42. Qu, Y., Hu, W., Cheng, G.: Constructing Virtual Documents for Ontology Matching. In: Proc. of WWW 2006, Edinburgh, Scotland, pp. 23–31 (2006)

43. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: Proc. of IJCAI 1995, Montréal, Québec, Canada, pp. 448–453 (1995)
44. Rodgers, J.L., Nicewander, W.A.: Thirteen ways to look at the correlation coefficient. The Amer. Statistician 42, 59–65 (1988)
45. Salton, G., Wong, A., Yang, C.S.: A Vector Space Model for Automatic Indexing. Communications of the ACM 18(1), 613–620 (1975)
46. Seco, N., Veale, T., Hayes, J.: An intrinsic information content metric for semantic similarity in WordNet. In: Proc. of ECAI, Valencia, Spain, pp. 1089–1090 (2004)
47. Shafer, G.: A mathematical theory of evidence. Princeton University Press, Princeton (1976)
48. Staab, S., Stuckenschmidt, H.: Semantic Web and Peer-to-Peer. In: Decentralized Management and Exchange of Knowledge and Information. Springer, Heidelberg (2006)
49. Stoilos, G., Stamou, G., Kollias, S.: A string metric for ontology alignment. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 624–637. Springer, Heidelberg (2005)
50. Su, X., Gulla, J.A.: An information retrieval approach to ontology mapping. Data & Knowledge Engineering 1(58), 47–69 (2006)
51. Wei, H., Ningsheng, J., Yuzhong, Q., Yanbing, W.: GMO: A Graph Matching for Ontologies. In: Proc. of K-Cap 2005, Banff, Canada, pp. 43–50 (2005)
52. Winkler, W.E.: The state of record linkage and current research problems. In: Statistics of Income Division, vol. (4). Internal Revenue Service Publication (1999)
53. WordNet: a lexical database for the English language (October 2007), `http://wordnet.princeton.edu/obtain`
54. WordNet-Similarity bibliography (October 2007), `http://www.d.umn.edu/~tpederse/wnsim-bib/`
55. Yi, L., Juanzi, L., Duo, Z., Jie, T.: Result of Ontology Alignment with RiMOM at OAEI 2006. In: Proc. of OM 2006, Athens, Georgia, USA, pp. 181–191 (2006)

Towards a Scalable Query Rewriting Algorithm in Presence of Value Constraints

H. Jaudoin[1], F. Flouvat[2], J.-M. Petit[2], and F. Toumani[3]

[1] University of Rennes, ENSSAT Lannion, IRISA, UMR6074 CNRS, France
[2] University of Lyon, INSA-Lyon, LIRIS, UMR5203 CNRS, F-69621, France
[3] University of Clermont-Ferrand, LIMOS, UMR6158 CNRS, France

Abstract. In this paper, we investigate the problem of query rewriting using views in a hybrid language allowing nominals (i.e., individual names) to occur in intentional descriptions. Of particular interest, restricted form of nominals where individual names refer to simple values enable the specification of value constraints, i.e, sets of allowed values for attributes. Such constraints are very useful in practice enabling, for example, fine-grained description of queries and views in integration systems and thus can be exploited to reduce the query processing cost. We use description logics to formalize the problem of query rewriting using views in presence of value constraints and show that the technique of query rewriting can be used to process queries under the certain answer semantics. We propose a sound and complete query rewriting Bucket-like algorithm. Data mining techniques have been used to favor scalability w.r.t. the number of views. Experiments on synthetic datasets have been conducted.

1 Introduction

This work is motivated by an application in the sustainable land and water management domain[1]. We aim at providing a scalable data integration infrastructure for: (i) sharing and analyzing agricultural practices across heterogeneous agricultural data sources, and (ii) verifying their compliance with respect to national and European government regulations. We adopt a Local As View (LAV) approach [14,19] to build our data integration system and use query rewriting using views as a technique for answering queries in such a system. The process of query rewriting supplies set of query plans formed of views only that must be further evaluated on data in order to produce correct answers.

In this context, `value constraints` over attributes, i.e., sets of allowed values for those attributes, turn out to be a key feature and have a strong practical interest. Indeed, values constraints enable *fine-grained description* of queries and views in integration systems and can be exploited to reduce the query processing cost. In our application context, various data sources provide views which have

[1] This is a collaborative project with a French public research institute whose work focuses on sustainable development in non-urban areas.

identical intentional descriptions (i.e., same structures) but the possible values for certain attributes are different (i.e., different value constraints). For example, two different data sources may store information about cultural parcels that have received pesticide in different years and/or are located at different districts. Views describing these data sources can be informally defined as follows:

$S_1.V_1$: *cultural parcels of district number* 23 *or* 63 *that have received pesticide of category* c_1 *or* c_{18} *or* c_{24}.
$S_2.V_2$: *cultural parcels of district number* 03 *or* 26 *or* 43 *that have received pesticide of category* c_2 *or* c_{15} *or* c_{38}.

Therefore view V_1 of data source S_1 supplies cultural parcels only located in district 23 or 63 and that have received pesticide whose category is only c_1 or c_{18} or c_{24}. Consequently, V_1 cannot return cultural parcels from district 69. In this example, without value constraints over the attributes district number and year, views $S_1.V_1$ and $S_2.V_2$ would be identical. Therefore, the use of value constraints enables a more accurate description of the content of data sources. Moreover, value constraints can also be very useful to express more precise queries. For example, a *typical* query Q in our application can ask for cultural parcels of only district number 23 or 63 that have received pesticide of category c_{20} or c_{38} only. The user is then interested in a particular region, here the district 23 and 63 only and in a particular set of pesticide categories that are known to be compatible with a culture activity. Turning our attention to query processing techniques, the presence of such constraints provides valuable information to identify when a view is not useful for answering a query. For example, here, the view $S_2.V_2$ cannot supply correct answer to the query Q since $S_2.V_2$ gives only *cultural parcels* in *districts* 03 or 26 or 43. Consequently, capturing and exploiting value constraints may improve the query processing costs in two ways. Firstly, it reduces the number of candidate views to be considered in the query rewriting process. Secondly, it prunes the set of sources accessed to answer a query, thereby reducing the network communication cost.

More generally, value constraints play an important role in various application domains. For example, in Databases Management Systems, they allow to represent enumerated data types -e.g., a marital status is either *married*, *single*, *divorced*, or *widowed*- and then to specify the *value integrity constraints* involved by the views. Moreover, such a kind of constraints allows for incomplete information description [8], that can be very practical in open environments like the web. Indeed sometimes users and administrators of data sources are only able to enumerate possible values of attributes instead of giving its exact value. For example, the user knows that the ages of individuals in his/her databases are either 22 or 23 or 24, and cannot be another value. Therefore, it is impossible to give the right age of individuals but it is possible to give a good idea about their ages. Finally, as mentioned in our motivating example, value constraints are very useful to specify queries of the form: "I seek for individuals whose values on a given attribute cannot be outside of the set of values $\{a_1, ..., a_n\}$". For example, with such a kind of constraints, it is possible to ask for documents dealing with only a list of specific topics, or food prepared with only certain ingredients.

In this paper, we study the problem of rewriting queries using views in presence of value constraints both in the queries and in the views. The problem of rewriting queries using views, intensively investigated during the last decade [14,24], can be formally stated as follow: given a set of views $\mathcal{V}$ and a query Q, both expressed over a global schema $\mathcal{S}$, the purpose is to reformulate Q into a query expression that uses only the views in $\mathcal{V}$ and is maximally contained in Q. While much has been done on the development of query rewriting algorithms for various classes of languages (conjunctive queries, recursive datalog, description logics) [12,14,19,24], to our knowledge, none has dealt with value constraints. First, it is not possible to reuse algorithms as those proposed in the general framework of conjunctive queries in presence of constants [20] or arithmetic comparisons [2] since conjunctive queries, even with disjunction, allow only for specifying the possible values of an attribute and not for *restricting* the range of an attribute to a given set of values. Second, it is neither possible to exploit existing query rewriting algorithms proposed in the description logics setting (e.g., $\mathcal{ALCNR}$ [7]) since values constraints cannot be correctly simulated in such languages. Indeed, the implicit information on number restrictions intrinsic to the value constraints [2] is likely to be lost and then existing algorithms would lead to incomplete algorithms for the problem of answering queries using views in presence of value constraints.

Hereafter, we investigate the query rewriting problem using a Description Logic (DL) [4] based framework. DLs constitute nowadays one of the most important logic-based knowledge representation formalisms in which problems related to representing and reasoning with value constraints have been deeply studied [15]. In particular, DLs provide two constructors respectively the `OneOf` constructor, noted $\mathcal{O}$, and the universal quantifier constructor $\forall$, that allow for a correct representation of value constraints. The first one enables the enumeration of individuals and then representation of sets of values while the second one restricts the range of a given attribute. For example, the description $\forall departmentNumber.\{23, 24, 63\}$ denotes the individuals whose the department number is necessarily restricted to 23 or 24 or 63.

Contributions. First, we consider the problem of answering queries in the formal setting of the DLs $\mathcal{ALN}$ augmented with a restricted form of the `OneOf` constructor, noted $\mathcal{O}_v$. We show that query rewriting provides a *sound* and *complete* technique to process queries under the certain answers semantics [1]. Then we propose a query rewriting Bucket-like algorithm based on data mining techniques and hypergraph framework. To our knowledge, this query rewriting algorithm is the first to use data mining implementations to favor scalability of the implementation. Experiments on synthetic datasets have been conducted. They show the feasibility of our proposition.

[2] If an individual checks the constraint $\forall departmentNumber.\{03, 63\}$, then this individual has at most *two* possible values over the attribute $departmentNumber$.

The remainder of this paper is organized as follows: Section 2 presents the $\mathcal{ALN}(\mathcal{O}_v)$ logic and gives a characterization of subsumption for this logic that is appropriate to deal with our query rewriting problem. Section 3 gives a description of a LAV-mediation system in the $\mathcal{ALN}(\mathcal{O}_v)$ setting and focus on the reduction of the problem of query answering using views to the problem of query rewriting using views in $\mathcal{ALN}(\mathcal{O}_v)$. Section 4 presents how the `Bucket` algorithm has been adapted to get all certain answers to a given query. Section 5 shows how the `iZi` platform [11], that supplies efficient and generic implementations of data mining algorithms, can be used to implement the most costly steps of our Bucket-like algorithm. Finally Section 6 is devoted to implementation and experimentations while Section 7 concludes our paper. Proofs are given in Annex.

2 Preliminaries

Description Logics (DLs) [4] allow to represent domain of interest in terms of *concepts or descriptions* (unary predicates) that characterize subsets of the objects (*individuals*) in the domain, and *roles* (binary predicates) over such a domain. Concepts are denoted by expressions formed by means of special constructors. The various description logics differ from one to another based on the set of constructors they allow. For example, the constructors of the so-called $\mathcal{ALN}$ description logic are: the symbol $\top$ is a concept description which denotes the top concept while the symbol $\bot$ stands for the bottom concept; concept conjunction ($\sqcap$), e.g., the concept description $Parent \sqcap Male$ denotes the set of fathers (i.e., male parents); the value restriction ($\forall R.C$), e.g., the description $\forall child.Male$ denotes the set of individuals whose children are all male; the number restriction constructors ($\geq n\, R$) and ($\leq n\, R$), e.g., the description ($\geq 1\, child$) denotes the set of parents (i.e., individuals having at least one child), while the description ($\leq 1\, leader$) denotes the set of individuals that cannot have more than one leader; the negation restricted to atomic concepts ($\neg A$), e.g., the description $\neg Male$ denotes the class of individuals which are not males.

In this paper, we use the DL $\mathcal{ALN}$ extended with the $\mathcal{O}$ constructor that enables building concepts from a set of enumerated nominals [8,23]. More precisely, we consider a restricted form of the $\mathcal{O}$ constructor, called $\mathcal{O}_v$ and written $\{o_1, \ldots, o_n\}$, where the o_i's refer to simple values. The obtained language, called $\mathcal{ALN}(\mathcal{O}_v)$ [3], allows descriptions of the form $\forall R.\{o_1, \ldots, o_n\}$, called value constraints, where $\{o_1, \ldots, o_n\}$ denotes a set of values. For example, the concept $\forall maritalStatus.\{MARRIED, SINGLE, DIVORCED, WIDOWED\}$ denotes the set of individuals whose marital status is necessarily *married* or *single* or *divorced* or *widowed*.

$\mathcal{ALN}(\mathcal{O}_v)$ **Syntax and Semantics.** Let $\mathcal{C}$ be a set of concept names, $\mathcal{N}$ be a set of value names and let $\mathcal{R}$ be a set of role names. In the spirit of [15], we assume $\mathcal{R}$ divided in two disjointed sets: $\mathcal{R}_c$, which denotes roles whose range is the set of usual individuals, and $\mathcal{R}_v$, which denotes roles whose range is a set of

[3] In this paper, we do not consider other constructors on concrete-valued roles as the fills constructor.

values of $\mathcal{N}$. We also consider that $\top_C$ is the top classical concept while $\top_V$ is the top values concept. Let $A \in \mathcal{C}$, $R \in \mathcal{R}$, $R_C \in \mathcal{R}_C$, $R_V \in \mathcal{R}_V$, n a positive integer and $o_i \in \mathcal{N}$ with $i \in [1, n]$. $\mathcal{ALN}(\mathcal{O}_v)$ concept descriptions are built up by means of the following syntaxic rules:

$$\begin{aligned} C, D \;\rightarrow\; & A \mid \neg A \mid \top_C \mid \bot \mid \\ & C \sqcap D \mid \\ & \forall R_C.C \mid \forall R_V.\{o_1, \ldots, o_n\} \mid \forall R_V.\top_V \mid \\ & (\leq\; n\, R) \mid (\geq\; n\, R) \end{aligned}$$

Semantics of concepts is defined by an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ where $\Delta^{\mathcal{I}}$ is a non-empty set, called *interpretation domain* and $.^{\mathcal{I}}$ is a *interpretation function.* We assume that $\Delta^{\mathcal{I}}$ is divided into two disjointed sets: δ_C the set of individuals in the domain and δ_V, the set of values. Hence we have $\top_C^{\mathcal{I}} = \delta_C$ and $\top_V^{\mathcal{I}} = \delta_V$. A concept is interpreted as a subset of $\Delta^{\mathcal{I}}$. A role is interpreted as a subset of $\delta_C \times \Delta^{\mathcal{I}}$. In other words, values of δ_V cannot have any successor by roles. Values are only authorized in range of $\mathcal{R}_V$ roles. The interpretation $\mathcal{I}$ associates each value $o_i \in \mathcal{N}$ with an element $o_i^{\mathcal{I}} \in \delta_V$ such that $o_i \neq o_j$ implies that ${o_i}^{\mathcal{I}} \neq {o_j}^{\mathcal{I}}$ that is, the mapping respects the Unique Name Assumption (UNA). Furthermore the semantic of an arbitrary concept must verify the following equations:

$$\begin{aligned} & (C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}},\ (\neg A)^{\mathcal{I}} = \Delta^{\mathcal{I}} \backslash A^{\mathcal{I}} \\ & (\forall R_C.C)^{\mathcal{I}} = \{x \in \delta_C \mid \forall y : (x, y) \in {R_c}^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\} \\ & (\forall R_V.\{o_1, \ldots, o_n\})^{\mathcal{I}} = \{x \in \delta_C \mid \forall y : (x, y) \in {R_v}^{\mathcal{I}} \rightarrow y \in \{o_1^{\mathcal{I}}, \ldots, o_n^{\mathcal{I}}\} \subseteq \delta_V\} \\ & (\leq nR)^{\mathcal{I}} = \{x \in \delta_C \mid \left|\{y \mid (x, y) \in R^{\mathcal{I}}\}\right| \leq n\} \\ & (\geq nR)^{\mathcal{I}} = \{x \in \delta_C \mid \left|\{y \mid (x, y) \in R^{\mathcal{I}}\}\right| \geq n\} \end{aligned}$$

An interpretation is a *model* for a concept C iff $C^{\mathcal{I}} \neq \emptyset$. A concept is *inconsistent*, i.e., $C \equiv \bot$ iff $C^{\mathcal{I}} = \emptyset$ for all interpretation $\mathcal{I}$.

With respect to this semantics, the notions of subsumption and equivalence are defined as follows. A concept C is *subsumed* by a concept D, noted $C \sqsubseteq D$, iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ $\forall \mathcal{I}$. A concept C is *equivalent* to a concept D, noted $C \equiv D$, iff $C^{\mathcal{I}} = D^{\mathcal{I}}$ $\forall \mathcal{I}$.

Characterizing Subsumption in $\mathcal{ALN}(\mathcal{O}_v)$. We present now a normal form description for $\mathcal{ALN}(\mathcal{O}_v)$ concepts and a characterization of subsumption w.r.t. this normal form that are appropriate to deal with the problem of rewriting query using views in presence of value constraints. More precisely, we use a structural approach of subsumption in order to further reduce the research space of query rewriting.

In the sequel, we use the letter P to specify either an atomic concept (A) or its negation ($\neg A$) or a set of values (E) or a number restriction (($\leq nR$) or ($\geq nR$)), or $\bot$ concept. The normal form $\widehat{C}$ of a concept C is either the $\top$ concept, or a conjunction (nonempty) of descriptions of the form $\forall R_1.(\ldots \forall R_m.P)$ with $m \geq 0$, where $R_1, \ldots, R_m$ are (not necessarily distinct) roles. The description $\forall R_1. \ldots \forall R_m.P$ is abbreviated by $\forall R_1 \ldots R_m.P$. $R_1 \ldots R_m$ is considered as a word, noted w, over the alphabet $\mathcal{R_C} \cup \mathcal{R_V}$ of roles. More precisely, $R_1 \ldots R_{m-1}$ is a word over $\mathcal{R}_C^*$ and R_m belongs to $\mathcal{R}_C \cup \mathcal{R}_V$. If $m = 0$, then we have an empty word ϵ, i.e., $\forall\epsilon.P$ is an equivalent notation of P. If w and u are two words of $\mathcal{R}^*$, wu denotes the word obtained by the concatenation of w and u, w being a prefix of wu. Consequently, every concept in $\mathcal{ALN}(\mathcal{O}_v)$ can be expressed in its normal form as a conjunction of concepts of the form $\forall w.P$, called *conjuncts*. In the sequel we use the expression $\forall w.P \in C$ to denote that the normal form of a concept C contains a conjunct of the form $\forall w.P$ in its description.

For the sake of brevity, normalization rules that allow to transform $\mathcal{ALN}(\mathcal{O}_v)$ concepts into their normal forms are omitted. They are described in appendix A page 59.

From the normal form introduced previously, Theorem 1 gives a characterization of subsumption between two concepts in $\mathcal{ALN}(\mathcal{O}_v)$.

Theorem 1 (Subsumption). *Let C, D two concepts, expressed in their normal form. $C \sqsubseteq D$ iff one of the following conditions is verified:*

(1) $C \equiv \bot$ or $D \equiv \top_C$, or
(2) for every $\forall w.P \in D$, we have
  (2.a) $\forall w.P \in C$ or,
  (2.b) $\forall w.E' \in C$ with $E' \subseteq E$ if $P = E$ or,
  (2.c) $\forall w.(\leq kR) \in C$ with $k \leq n$ if $P =\leq nR$ or,
  (2.d) $\forall w.R.E \in C$ with $|E| \leq n$ if $P =\leq nR$ and $R \in \mathcal{R}_v$ or,
  (2.e) $\forall w.(\geq kR) \in C$ with $k \geq n$ if $P =\geq nR$ or,
  (2.f) $\forall v.(\leq 0R) \in C$ with vR prefix of w.

Informally, a concept C is subsumed by a concept D if and only if all constraints over D appears in the description of C [4]. Note that we abbreviate the concepts $\forall R.\bot$ and $\forall R.\emptyset$ by $\leq 0R$. The proof of this theorem which is given in annex A, page 59, is derived from characterization of structural subsumption of CLASSIC [8]. Indeed, logic $\mathcal{ALN}(\mathcal{O}_v)$ can be seen as a sub-language of CLASSIC [8] where constructor $\mathcal{O}_v$ can be considered as a particular case of *Host Individuals.*

Example 1. Let $C \equiv \forall received.Pesticide \sqcap CulturalParcel$, $C' \equiv \forall received.category.\{C_2, C_3\} \sqcap \forall received. \leq 2 category$, $D \equiv \forall received.Pesticide$ and $D' \equiv \forall received.category.\{C_1, C_2, C_3\} \sqcap \forall received. \leq 3\, category$.

We have $C \sqsubseteq D$ since $\forall received.Pesticide \in C$. We have $C' \sqsubseteq D'$ because $\forall received.category.\{C_2, C_3\} \in C'$ with $\{C_2, C_3\} \subseteq \{C_1, C_2, C_3\}$, and $\forall received.(\leq 2\, category) \in C'$.

3 Query Rewriting Using Views in the $\mathcal{ALN}(\mathcal{O}_v)$ Setting

In this section, we briefly introduce the $\mathcal{ALN}(\mathcal{O}_v)$-based framework used in our work. Then we focus on the problem of *query answering using views*, i.e., computing the answers of a query in presence of value constraints. In this setting, we show that the problem of query answering in the $\mathcal{ALN}(\mathcal{O}_v)$ setting can be reduced to the problem of query rewriting using views.

[4] Following the object-oriented paradigm, C must override the concept D.

3.1 A Formal Framework

A LAV-based mediation system is defined by a pair $(\mathcal{S}, \mathcal{V})$, where $\mathcal{S}$ is a global schema and $\mathcal{V}$, a set of views, i.e., named queries, expressed in terms of $\mathcal{S}$ [14]. Hereafter, we consider a mediation system $(\mathcal{S}, \mathcal{V})$ in the $\mathcal{ALN}(\mathcal{O}_v)$ setting. Therefore, the schema $\mathcal{S}$ is specified as an $\mathcal{ALN}(\mathcal{O}_v)$-terminology, i.e., a set of axioms of the forms: (i) $A \equiv D$ (concept definitions), or (ii) $A \sqsubseteq D$ (primitive specifications), where A is a concept name and D is a concept description in $\mathcal{ALN}(\mathcal{O}_v)$. Moreover, the set of views $\mathcal{V}$ is specified as a set of primitive specifications in $\mathcal{ALN}(\mathcal{O}_v)$.

The semantic of a mediation system $(\mathcal{S}, \mathcal{V})$ is derived from the notion of interpretation of a terminology in DL [4]. We say that an interpretation $\mathcal{I}$ is a model for $(\mathcal{S}, \mathcal{V})$ iff $\mathcal{I}$ is a model for every axiom in $\mathcal{S}$ and $\mathcal{V}$ (i.e., $A \sqsubseteq D$ iff $A^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ and $A \equiv D$ iff $A^{\mathcal{I}} = D^{\mathcal{I}}$). Note that, describing views as primitive specifications enables to capture Open World Assumption (OWA) in building our mediation system [1] (i.e., assuming that the data sources are incomplete). Indeed, primitive specifications are incomplete specifications in the sense that they provide only necessary, but not sufficient, conditions that must be satisfied by their instances.

The Table 1 gives an example of $\mathcal{ALN}(\mathcal{O}_v)$ mediation system $(\mathcal{S}, \mathcal{V})$. The global schema is made of two concepts: $CulturalParcel$, which denotes parcels that have at least one kind of culture and $TreatedObject$ which denotes the class of individuals that have received at least one treatment which has at least one category and whose categories take their values necessarily from the set $\{C_1, \ldots, C_{18}\}$. The terminology $\mathcal{V}$ is made of nine views (V_1, V_2, ..., V_9). The *extension* of the view V_1 is a subset of cultural parcels while extension of view V_2 is a subset of individuals that have received at least one category of treatment.

Queries are defined as $\mathcal{ALN}(\mathcal{O}_v)$ concepts expressed in terms of the elements (i.e., roles and concepts) of $\mathcal{S}$. For example, a query Q that asks for cultural parcels that have received at least one treatment and whose treatment category is either C_8 or C_9, may be expressed as follows: $Q \equiv CulturaParcel \sqcap \geq 1 treatment \sqcap \forall treatment.category.\{C_8, C_9\}$.

Let $(\mathcal{S}, \mathcal{V})$ be a $\mathcal{ALN}(\mathcal{O}_v)$ mediation system. In the sequel, we assume that the terminologies $\mathcal{S}$ and $\mathcal{V}$ are acyclic (i.e., they do not contain a concept that

Table 1. Example of mediation system

Global schema $\mathcal{S}$	
$CulturalParcel \equiv Parcel \sqcap \geq 1\, cultureType$	
$TreatedObject \equiv \geq 1 treatment \sqcap \forall treatment. \geq 1 category \sqcap \forall treatment.category.\{C_1, C_2, \ldots, C_{18}\}$	
$OrganicallyTreatedObject \equiv \forall treatment.OrganicProduct$	
Set of views $\mathcal{V}$	
$V_1 \sqsubseteq CulturalParcel$	$V_4 \sqsubseteq \forall treatment.category.\{C_9, C_{10}\}$
$V_2 \sqsubseteq \forall treatment. \geq 1 category$	$V_5 \sqsubseteq \forall treatment.category.\{C_8\}$
$V_3 \sqsubseteq \geq 1 treatment$	$V_6 \sqsubseteq \forall treatment.category.\{C_8, C_{11}\}$
$V_7 \sqsubseteq \forall treatment.category.\{C_7, C_8, C_9, C_{10}\} \sqcap \forall treatment.OrganicProduct$	
$V_8 \sqsubseteq \forall treatment.category.\{C_7, C_8, C_9\} \sqcap \forall treatment.\neg OrganicProduct$	
$V_9 \sqsubseteq \forall treatment.category.\{C_8, C_9, C_{10}\} \sqcap \forall treatment.\neg OrganicProduct$	

refers to itself in its specification or definition). We also assume that $\mathcal{V}$ is: (i) *normalized*, i.e., every primitive specification $A \sqsubseteq D$ in $\mathcal{V}$ is replaced by a definition $A \equiv \overline{A} \sqcap D$, where $\overline{A}$ is a new atomic concept [4], and (ii) *expanded*, i.e., defined concepts occurring in right-hand side of axioms are recursively replaced by their definitions. Finally, queries on $(\mathcal{S}, \mathcal{V})$ and views in $\mathcal{V}$ are assumed to be provided in their normal forms.

3.2 From Query Answering to Query Rewriting

This section focuses on the query answering problem in the $\mathcal{ALN}(\mathcal{O}_v)$ setting. Let Q be a query over a $\mathcal{ALN}(\mathcal{O}_v)$ mediation system $(\mathcal{S}, \mathcal{V})$. We consider the problem of computing the answers of Q under *certain answer* semantics [1]. Informally, an answer t is a certain answer of Q if t is an answer to Q for any database consistent with the extensions of the views in $\mathcal{V}$, i.e., the set of tuples associated with the views. We use the following notations to define the notion of certain answers under OWA in the DL setting. For a view $V \in \mathcal{V}$, we denote by V^{ext} its extension and by $\mathcal{V}^{ext}$ the union of all the extensions of the views in $\mathcal{V}$.

Definition 1 (Certain answers under OWA). *Let $(\mathcal{S}, \mathcal{V})$ be a $\mathcal{ALN}(\mathcal{O}_v)$ mediation system and Q be a query. t is a certain answer of Q iff: **(i)** $t \in \mathcal{V}^{ext}$ and **(ii)** $t \in Q^{\mathcal{I}}$, for all model $\mathcal{I}$ of $(\mathcal{S}, \mathcal{V})$ s.t. $\forall\, V \in \mathcal{V}$, $V^{ext} \subseteq V^{\mathcal{I}}$.*

The set of all the certain answers of Q is denoted by $Ans(Q, \mathcal{V}^{ext})$.

Let Q be a query over a mediation system $(\mathcal{S}, \mathcal{V})$. The problem of computing $Ans(Q, \mathcal{V}^{ext})$ can be reduced to a problem of *query rewriting using views* [14,12]. In the latter case, the goal is to reformulate Q into an expression Q' in some language, denoted $\mathcal{L}_{\mathcal{R}}$, such that Q' refers only to the views of $\mathcal{V}$ and $Q' \sqsubseteq Q$. Q', the rewriting of Q, can be viewed as a query plan in the sense that it can be evaluated over the view extensions in order to compute the certain answers of Q. A crucial point to guarantee that a rewriting Q' provides *all* the certain answers of Q lies in the definition of the rewriting language $\mathcal{L}_{\mathcal{R}}$. Below, we show that in the setting of our $\mathcal{ALN}(\mathcal{O}_v)$ mediation system it is sufficient to consider rewritings that consist in union of view conjunctions (i.e., $\mathcal{L}_{\mathcal{R}} = \{\sqcap, \sqcup\}$).

Lemma 1. *Let $(\mathcal{S}, \mathcal{V})$ be a $\mathcal{ALN}(\mathcal{O}_v)$ mediation system and Q be a query. If t is a certain answer of Q, then there exists a subset $\{V_1, \ldots, V_n\}$ of $\mathcal{V}$ s.t.: **(i)** $V_1 \sqcap \ldots \sqcap V_n \sqsubseteq Q$, and **(ii)** $t \in V_1^{ext} \cap \ldots \cap V_n^{ext}$.*

This lemma states that a certain answer of a query Q in a $\mathcal{ALN}(\mathcal{O}_v)$ mediation system is provided by a conjunction of views which is subsumed by Q. Its proof is given in annex B, page 60. Note that the rewriting language obtained in our context is the same than those usually obtained in other modeling languages as for example $\mathcal{ALCNR}$ and CARIN-$\mathcal{ALN}$ [7] or conjunctive queries [20]. Hereafter, we use the notion of *conjunctive rewriting* of a query Q to refer a conjunction of views subsumed by Q. As a consequence of lemma 1, all the certain answers of a query Q can be obtained from the union of all the conjunctive rewritings of Q. We define below the notion of *maximally-contained rewriting*, i.e., those conjunctive rewritings that return maximal sets of certain answers.

Definition 2 (Max-contained rewriting). *Let $(\mathcal{S}, \mathcal{V})$ be a mediation system and Q be a query. Q' is a maximally-contained rewriting of Q using $\mathcal{V}$ if and only if: **(i)** Q' is a conjunctive rewriting of Q, and **(ii)** there is no conjunctive rewriting Q_1 of Q s.t. $Q' \sqsubseteq Q_1 \sqsubseteq Q$ and $Q' \not\equiv Q_1$.*

The following theorem shows that the set of *all* certain answers of a query Q can be computed exclusively from the union of the maximally-contained rewritings of Q.

Theorem 2. *Let $(\mathcal{S}, \mathcal{V})$ be a mediation system and Q be a query. Let $\{Q_1, \ldots, Q_n\}$ be the set of all maximally-contained rewritings of Q using $\mathcal{V}$ and $Q_i(\mathcal{V}^{ext})$ the result of evaluating Q_i over $\mathcal{V}^{ext}$.*

$$Ans(Q, \mathcal{V}^{ext}) = \cup_{i=1}^{n} Q_i(\mathcal{V}^{ext}).$$

Therefore, to compute $Ans(Q, \mathcal{V}^{ext})$, one can restrict our attention to the problem of computing all the maximally-contained rewritings of Q using $\mathcal{V}$. Proof of this theorem is given in annex B, page 60.

To characterize the maximally-contained rewritings of Q and then compute them, we use the interesting following property that is a direct consequence of the open word assumption.

Lemma 2. *Let $\{V_1, \ldots, V_n\}$ be a subset of $\mathcal{V}$ and $Q' \equiv \sqcap_{i=1}^{n} V_i$ s.t. $Q' \sqsubseteq Q$. Q' is a maximally-contained rewriting of Q using views $\mathcal{V}$ iff for any concept Q'' obtained by removing from Q' one of its conjuncts V_i, we have $Q'' \not\sqsubseteq Q$.*

It turns out that any maximally-contained rewritings of Q is necessarily made of a minimal subset of $\mathcal{V}$ such that the conjunction of its elements is subsumed by Q. Proof of the lemma is given in annex B, page 60. Hereafter, the problem of computing $Ans(Q, \mathcal{V}^{ext})$ is then equivalent to the problem, denoted by *conj_rewrite(Q,$\mathcal{V}$,$\mathcal{ALN}(\mathcal{O}_v)$)*, of enumerating all the minimal subsets of $\mathcal{V}$ s.t. the conjunction of their elements is subsumed by Q. Next section gives an algorithm to solve *conj_rewrite(Q,$\mathcal{V}$,$\mathcal{ALN}(\mathcal{O}_v)$)*, thereby providing a sound and complete procedure for our query answering using views problem.

4 A `Bucket`-Based Algorithm for $\mathcal{ALN}(\mathcal{O}_v)$ Mediation System

In the setting of $\mathcal{ALN}(\mathcal{O}_v)$, the optimizations of the `Minicon` algorithm [22] can't be used to compute maximally-contained rewritings, since views and queries are specified as conjunction of *unary* concepts. A possible solution is to follow a "bucket-like approach" [14]. The interest of this approach is to break down the problem of rewriting maximally a query into rewriting maximally each of its subgoals, here the query conjuncts. The algorithm 1, called `ComputeRew`, is a slight adaptation of the `Bucket` algorithm [14].

Given a rewriting problem *conj_rewrite(Q,$\mathcal{V}$,$\mathcal{ALN}(\mathcal{O}_v)$)* with $Q \equiv \forall w_1.P_1 \sqcap \ldots \sqcap \forall w_n.P_n$, the algorithm `ComputeRew`, as the well-known `Bucket` algorithm is made up of two main steps:

Algorithm 1. ComputeRew

```
Require: $\mathcal{V} = \{V_1, ..., V_m\}$ a set of views and $Q$ a query
Ensure: $\mathcal{MCR}$ the set of maximally-contained rewriting of $Q$ using $\mathcal{V}$
 1: Let $Q \equiv \sqcap_{i=1}^{n} \forall w_i.P_i$
 2: /* Step 1: Buckets computation */
 3: for all conjunct $\forall w_i.P_i$ do
 4:     $B(w_i, P_i)$ = BucketBuilding($\mathcal{V}$,$\forall w_i.P_i$)
 5:     /* Pruning of inconsistent and non maximal rewritings */
 6:     $B(w_i, P_i)$:=BucketPruning($B(w_i, P_i)$)
 7: end for
 8: /* Step 2: Rewritings generation */
 9: $\mathcal{MCR}$:=Cart_Prod($B(w_i, P_i)$, $i \in \{1, ..., n\}$)
10: /* Pruning of inconsistent and non maximal rewritings */
11: $\mathcal{MCR}$:=Pruning($\mathcal{MCR}$)
12: RETURN $\mathcal{MCR}$
```

– **Buckets Computation.** For each conjunct $\forall w_i.P_i$ of Q, a *bucket*, noted $B(w_i, P_i)$, containing all the maximally-contained rewritings of this conjunct is created.
– **Rewritings Generation.** This step computes maximally-contained rewritings of Q, denoted by $\mathcal{MCR}$, by combining elements from the previously identified buckets. $\mathcal{MCR}$ is the solution to the problem *conj_rewrite(Q,$\mathcal{V}$,$\mathcal{ALN}(\mathcal{O}_v)$)*.

In the $\mathcal{ALN}(\mathcal{O}_v)$ setting, the step of Buckets computation, i.e., the step 1, must be redefined as detailed in subsection 4.1.

4.1 Bucket Algorithm for $\mathcal{ALN}(\mathcal{O}_v)$

Using a case based analysis for the language $\mathcal{ALN}(\mathcal{O}_v)$, the next lemma gives necessary conditions that should be verified by a bucket element (i.e., a rewriting of a conjunct).

Lemma 3. *For conj_rewrite(Q,$\mathcal{V}$,$\mathcal{ALN}(\mathcal{O}_v)$), let $Q \equiv \forall w.P$, l be the cardinality of the largest set of values that appears in $\mathcal{V}$ or Q, and p be the maximal depth*[5] *of the conjuncts in $\mathcal{V}$ or Q. $Q' \equiv V_{i_1} \sqcap \ldots \sqcap V_{i_k}$ is a maximally-contained rewriting of Q if Q' is made of a* ***minimal*** *subset of $\mathcal{V}$ verifying one of the following conditions:*

a) *$P \in \{A, \neg A\}$ then $\forall w.P \in Q'$ and $k = 1$ or,*
b) *$P = (\geq nR)$ then $\forall w.(\geq pR) \in Q'$ with $p \geq n$ and $k = 1$ or,*
c) *$P = (\leq nR)$ then $\forall w.(\leq pR) \in Q'$ with $p \leq n$ and $k = 1$ or,*
d) *$P = E$ then $\{V_{i_1}, ..., V_{i_k}\}$ is s.t. **(i)** for each $j \in [1, k]$, $\forall w.E_{i_j} \in V_{i_j}$, and $\cap_{j=i_1}^{i_k} E_j \subseteq E$ and **(ii)** $1 \leq k \leq l + 1$ or,*
e) *$P = (\leq n\, R_v)$, with $R_v \in \mathcal{R}_v$ then $\{V_{i_1}, ..., V_{i_k}\}$ is s.t. **(i)** for $j \in [1, k]$, $\forall w.E_{i_j} \in V_{i_j}$ and $|\cap_{j=i_1}^{i_k} E_j| \leq n$ **(ii)** $1 \leq k \leq l + 1$ or,*
f) *$\forall w'.(\leq 0v) \in Q'$ with $w'v$ a prefix of w s.t. and $1 \leq k \leq l + p$.*

Proof of this lemma is given in annex C.1, page 63.

[5] The depth of a conjunct $\forall w.P$ is equal to the length of the word w.

The algorithm 2 computes the bucket elements based on this lemma. To the best of our knowledge, algorithm 2 is the first adaptation of the bucket algorithm in the setting of $\mathcal{ALN}(\mathcal{O}_v)$. In this algorithm, we denote by $2^{\mathcal{V}}$ the powerset of $\mathcal{V}$ and by $min_{\subseteq}(S)$ s.t. $S \subseteq 2^{\mathcal{V}}$, the subsets of S that are minimal w.r.t. the set inclusion.

In this context, we can distinguish two types of rewritings: classical $\mathcal{ALN}$ rewritings [12], lines 2-14 of algorithm 2, and specific rewritings due to the presence of value constraints, lines 15-29 of algorithm 2. Note that classical $\mathcal{ALN}$ rewritings are made of only one view and correpond to cases a, b, c of Lemma 3. Rewritings $RewS_1(E, w)$ are due to value constraints while rewritings

Algorithm 2. Bucket building

```
Require: $\mathcal{V} = \{V_1, ..., V_m\}$ a set of views and $\forall w.P$ a conjunct of $Q$
Ensure: $B(w, P)$
 1: $B(w, P) := \emptyset$
 2: /* Computation of classical $\mathcal{ALN}$ rewritings */
 3: /* Condition a) of lemma 3 */
 4: if $P = A$ or $P = \neg A$ then
 5:     $B(w, P) := B(w, P) \cup \{V_i \in \mathcal{V} \mid \forall w.P \in V_i\}$
 6: end if
 7: /* Condition b) of lemma 3 */
 8: if $P = (\geq nR)$ then
 9:     $B(w, P) := B(w, P) \cup \{V_i \in \mathcal{V} \mid \forall w.(\geq pR) \in V_i, p \geq n\}$
10: end if
11: /* Condition c) of lemma 3 */
12: if $P = (\leq nR)$ then
13:     $B(w, P) := B(w, P) \cup \{V_i \in \mathcal{V} \mid \forall w.(\leq pR) \in V_i, p \leq n\}$
14: end if
15: /* Computation of specific $\mathcal{ALN}(\mathcal{O}_v)$ rewritings */
16: /* Condition d) of lemma 3 */
17: if $P = E$ then
18:     /* Computation of the rewritings $RewS_1(E, w)$ */
19:     $S_1(E, w) = min_{\subseteq}(U \in 2^{\mathcal{V}} \mid \forall V_i \in U, \forall w.E_i \in V_i$ and $\bigcap_{V_i \in U} E_i \subseteq E)$
20:     $RewS_1(E, w) = \{\sqcap_{V_i \in U} V_i \mid U \in S_1(E, w)\}$
21:     $B(w, P) := B(w, P) \cup RewS_1(E, w)$
22: end if
23: /* Condition e) of lemma 3 */
24: if $P = (\leq n R_v), R_v \in \mathcal{R}_v$ then
25:     /* Computation of the rewritings $RewS_2(n, w.R_v)$*/
26:     $S_2(n, w.R_v) = min_{\subseteq}(U \in 2^{\mathcal{V}} \mid \forall V_i \in U, \forall w.R_v.E_i \in V_i$ and $|\bigcap_{V_i \in U} E_i| \leq n)$
27:     $RewS_2(n, w.R_v) = \{\sqcap_{V_i \in U} V_i \mid U \in S_2(n, w.R_v)\}$
28:     $B(w, P) := B(w, P) \cup RewS_2(n, w.R_v)$
29: end if
30: /* Computation of both classical $\mathcal{ALN}$ and specific $\mathcal{ALN}(\mathcal{O}_v)$ implicit
    inconsistencies*/
31: /* Condition f) of lemma 3 */
32: $II = min_{\subseteq}(U \in 2^{\mathcal{V}} \mid \forall w'.(\leq 0v) \in Q' \equiv \sqcap (V_i \in U)$ and w'v is a prefix of w)
33: $B(w, P) := B(w, P) \cup \{\sqcap_{V_i \in U} V_i \mid U \in II\}$
```

$RewS_2(n, w.R_v)$ are due to the interaction between the value constraints and number restrictions constructors. Indeed a number restriction can subsume a value constraint as built up by the case (2.d) of Theorem 1. Note that each rewriting in $RewS_1(E, w)$ and in $RewS_2(n, w.R_v)$ consists of conjunction of views such that each view has a value constraint over the word w and respectively over $w.R_v$. Moreover, consequently to lemma 2, such rewritings are made of minimal subsets of views w.r.t. set inclusion, s.t. their conjunction is subsumed by $\forall w.E$, respectively by $\forall w. \leq nR_v$. Therefore to compute $RewS_1(E, w)$ and $RewS_2(n, w.R_v)$, first we must find views having value constraints over w, respectively $w.R_v$. Second, from this set of views, we have to compute the minimal subsets of views $S_1(E, w)$ and $S_2(n, w.R_v)$. A set of views is in $S_1(E, w)$ if the intersection of their value constraints is a subset of E. A set of views is in $S_2(n, w.R_v)$ if the cardinality of the intersection of their value constraints is less than n. At last, the rewritings $RewS_1(E, w)$ and $RewS_2(n, w.R_v)$ are inferred by conjunction of the views belonging to each element of $S_1(E, w)$ and $S_2(n, w.R_v)$. The maximal number of views occurring in such rewritings is $l + 1$, the cardinality of the largest set of values occurring in the views $\mathcal{V}$. Finally, the algorithm (computation of II, lines 25-27) processes $\mathcal{ALN}(\mathcal{O}_v)$ *implicit inconsistencies* [18] as built up by the case f of Lemma 3. These implicit inconsistencies are computed from $RewS_2(n, w.R_v)$ and classical implicit inconsistencies. More precisely, for each view $V_i \in \mathcal{V}$ such that $\forall w. \geq mR_v \in V_i$, we must compute $RewS_2(n, w.R_v)$ with $n < m$.

The following example illustrates the bucket building algorithm in our setting.

Example 2. Continuing the example given in Table 1, let us considering the following query made up of three conjuncts:

$Q \equiv CulturalParcel \sqcap \forall treatment.category.\{C_8, C_9\} \sqcap \geq 1 treatment.$

By applying the previous algorithm on the 9 views of the mediator given in Table 1, we get:

- $B(\epsilon, CulturalParcel)=\{V_1\}$ (case **(a)**)
- $B(\epsilon, \geq 1\, treatment) = \{V_3\}$ (case **(b)**)
- $B(treatment.category, \{C_8, C_9\}) = \{V_5\, , V_4 \sqcap V_6\, , V_7 \sqcap V_8 \sqcap V_9\, , V_7 \sqcap V_8\}$. The three first rewritings are due to the case **(d)** while the last one is due to case **(f)**.

To end up, note that the obtained buckets need to be pruned in order to remove inconsistent or not maximal rewritings (see line 6 of algorithm 1), which is not the case of the classical `Bucket` algorithm. Indeed implicit inconsistencies may appear in the rewritings, as shown in the following example.

Example 3. Continuing the example 2, the rewriting $V_7 \sqcap V_8 \sqcap V_9$ of the bucket $B(treatment.category, \{C_8, C_9\})$ is not maximal because $V_7 \sqcap V_8$, that infers an implicit inconsitency over "*treatment*" role , belongs to the same bucket. Therefore the rewriting $V_7 \sqcap V_8 \sqcap V_9$ must be deleted from the bucket $B(treatment.category, \{C_8, C_9\})$.

4.2 Max-Rewritings Generation

The second step of the algorithm 1 constructs the global rewritings of a query (i.e., the set $\mathcal{MCR}$) using the buckets generated previously. In the classical approach, it begins by generating candidate rewritings from the *cartesian product* of the buckets. However, the cost of the cartesian product is prohibitive even on medium size configuration. To cope with this limitation, we propose a new hypergraph-based characterization to avoid the use of costly cartesian product. Indeed, computation of the rewritings can be reduced to a well known problem in combinatorics, the computation of minimal transversals of a hypergraph [10].

Definition 3 (Hypergraph). *Let V be a set of vertices and E an element of the powerset $2^{|V|}$ of V.*

A hypergraph $\mathcal{H} = (V, E)$ consists of a finite collection E of sets over a finite set V. The elements of V are called the vertices of $\mathcal{H}$ while the elements of E are called the edges of $\mathcal{H}$.

Definition 4 (Transversal and minimal transversal). *Let V be a set of vertices and E an element of the powerset $2^{|V|}$ of V.*

$T \subseteq V$ is a transversal of $\mathcal{H}$ if $\forall X \in H, T \cap X \neq \emptyset$.

T is minimal if $\forall Y \subset T, Y$ is not a transversal.

Even if the best complexity of the problem of computing minimal transversal of a hypergraph is known to be in almost-polynomial time [17], efficient and scalable implementations exist since this problem is at the heart of many data mining problems [21,13].

The rewritings computation in the hypergraph framework can be formulated as follows: Let $\mathcal{H}_B = (V_B, E_B)$ be a hypergraph. Let Q be a query such that $Q \equiv \sqcap_{i=1}^{n} \forall w_i.P_i$. Each view or conjunction of views occurring in the buckets $B(w_i, P_i)$ is associated to a vertex in V_B. The set E_B consists of the set of the buckets $B(w_i, P_i)$ themselves.

Example 4. Let $Q \equiv \forall w_1.A_1 \sqcap \forall w_2.A_2 \sqcap \forall w_3.A_3 \sqcap \forall w_4.A_4$ and the associated buckets:

$B(w_1, A_1)$	$B(w_2, A_2)$	$B(w_3, A_3)$	$B(w_4, A_4)$
V_1	V_1	V_3	$V_3 \sqcap V_4$
V_2	V_4	V_4	$V_1 \sqcap V_2 \sqcap V_3$
	$V_3 \sqcap V_4$		

Let V_{34} be a representation of $V_3 \sqcap V_4$ and V_{123} of $V_1 \sqcap V_2 \sqcap V_3$. A hypergraph $\mathcal{H}_B$ can be built over $V_B = \{V_1, V_2, V_3, V_4, V_{34}, V_{123}\}$ as follows: $E_B = \{\{V_1, V_2\}, \{V_1, V_4, V_{34}\}, \{V_3, V_4\}, \{V_{34}, V_{123}\}\}$.

The following theorem shows that the minimal transversals of this hypergraph are a superset of the maximally-contained rewritings of the query Q.

Theorem 3. *Let Q be a query and its buckets $B(w_i, P_i)$, with $i \in [1, n]$. Let $\mathcal{H}_B = (V_B, E_B)$ be a hypergraph defined in terms of the $B(w_i, P_i)$. Let $T_{\mathcal{H}_B}$ be the set of minimal transversals of $\mathcal{H}_B$.*

Then $\mathcal{MCR} \subseteq T_{\mathcal{H}_B}$.

Proof of this theorem is given in annex C.2, page 64.

Then, as in conventional bucket-like approaches, the generation of the query rewritings, here the minimal transversals computation of $\mathcal{H}_B$, requires the deletion of inconsistent and non maximal rewritings. The following example illustrates the maximally-contained rewritings computation of a given query Q from the hypergraph $\mathcal{H}_B$ obtained in example 4.

Example 5. The minimal transversals of $\mathcal{H}_B$ given in example 4 are: $\{V_1, V_3, V_{34}\}$, $\{V_1, V_3, V_{123}\}$, $\{V_1, V_4, V_{34}\}$, $\{V_1, V_4, V_{123}\}$, $\{V_2, V_4, V_{34}\}$, $\{V_2, V_4, V_{123}\}$, $\{V_2, V_3, V_{34}\}$, $\{V_2, V_3, V_{123}\}$.

After expansion of the minimal transversals, we obtain the following set of candidate maximally-contained rewritings: $\{V_1 \sqcap V_3 \sqcap V_4, V_1 \sqcap V_2 \sqcap V_3, V_1 \sqcap V_2 \sqcap V_3 \sqcap V_4, V_2 \sqcap V_3 \sqcap V_4\}$.

Some of them are not minimal. The final set of maximally-contained rewritings is then: $\{V_1 \sqcap V_3 \sqcap V_4, V_1 \sqcap V_2 \sqcap V_3, V_2 \sqcap V_3 \sqcap V_4\}$.

This approach reduces significantly the number of candidate rewritings in comparison with the cartesian product. In example 4, 8 candidates are generated instead of 24 using the cartesian product.

The efficiency and scalability of our query rewriting algorithm *ComputeRew* depends on the computation of $RewS_1(E, w)$ and $RewS_2(n, w.R_v)$ since their number of candidates is exponential in the number of views. All other cases involve only one view and therefore are not concerned by scalability. The max-rewritings generation can also be costly even with the hypergraph-based characterization, due to the potentially large number of elements to generate. To cope with these difficulties, our work features the use of data mining techniques to devise an efficient algorithm that favor scalability w.r.t. the number of views in both steps. To do that, we use a data mining library called *iZi*.

5 Query Rewriting Algorithm in $\mathcal{ALN}(\mathcal{O}_v)$Using *iZi*

iZi [11] is a generic C++ library for pattern mining problems known to be "representable as sets", i.e., those problems whose solution space is isomorphic to a boolean lattice. The basic idea of *iZi* is to offer a toolbox for a rapid and easy development of efficient and robust data mining programs. This library takes advantage of a well established theoretical framework from an implementation point of view by providing efficient data structures for boolean lattice representation and several implementations of well known algorithms. By the way, these problems can be implemented with only minimal effort, i.e., programmers do not have to be aware of low-level code, customized data structures and algorithms being available for free. This library has been devised and applied to several problems such as itemset mining and constraint mining in relational databases.

Following the guidelines given with *iZi*, the rest of this section shows how three subparts of the query rewriting algorithm can take advantage of *iZi*. First we recall the underlying theoretical framework and then we point out in details how *iZi* can be used in this context.

5.1 A Theoretical Framework for Knowledge Discovery

We recall in this section the theoretical KDD framework defined in [21] for interesting pattern discovery problems, and used in *iZi*. Such a framework has been successfully applied in different contexts such as association rules [3], functional dependencies [16] and inclusion dependencies [9] to mention a few.

Given a database r, a finite language $\mathcal{L}$ for expressing patterns or defining subgroups of the data, and a predicate P for evaluating whether a pattern $\varphi \in \mathcal{L}$ is true or "interesting" in r, the discovery task is to find the theory of r with respect to $\mathcal{L}$ and P, i.e., the set $Th(r, \mathcal{L}, P) = \{\varphi \in \mathcal{L} \mid P(r, \varphi) \text{ is true}\}$.

Let us suppose a specialization/generalization relation between patterns of $\mathcal{L}$. Such a relation is a partial order $\preceq$ on the patterns of $\mathcal{L}$. We say that φ is more general (resp. more specific) than θ, if $\varphi \preceq \theta$ (resp. $\theta \preceq \varphi$).

Let $(I, \preceq)$ be a partially ordered set of elements. A set $S \subseteq I$ is *closed downwards* (resp. *closed upwards*) if, for all $X \in S$, all subsets (resp. supersets) of X are also in S.

The predicate P is said to be monotone (resp. anti-monotone) with respect to $\preceq$ if for all $\theta, \varphi \in \mathcal{L}$ such that $\varphi \preceq \theta$, if $P(r, \varphi)$ is true (resp. false) then $P(r, \theta)$ is true (resp. false). As a consequence, if the predicate is monotone (resp. anti-monotone), the set $Th(r, \mathcal{L}, P)$ is upward (resp. downward) closed, and can be represented by either of the following sets:

- its *positive border*, denoted by $\mathcal{B}d^+(\ Th(r, \mathcal{L}, P)\ )$, made up of the MOST SPECIALIZED true patterns when $Th(r, \mathcal{L}, P)$ is downward closed, and the MOST SPECIALIZED false patterns when $Th(r, \mathcal{L}, P)$ is upward closed;
- its *negative border*, denoted by $\mathcal{B}d^-(\ Th(r, \mathcal{L}, P)\ )$, made up of the MOST GENERALIZED false patterns when $Th(r, \mathcal{L}, P)$ is downward closed, and the MOST GENERALIZED true patterns when $Th(r, \mathcal{L}, P)$ is upward closed.

The union of these two borders is called the *border of* $Th(r, \mathcal{L}, P)$, and is denoted by $\mathcal{B}d(\ Th(r, \mathcal{L}, P)\ )$.

The last hypothesis of this framework is that the problem must be representable as sets via an isomorphism, i.e., the search space can be represented by a boolean lattice (or subset lattice). Let $(\mathcal{L}, \preceq)$ be the ordered set of all the patterns defined by the language $\mathcal{L}$. Let C be a finite set of elements. The problem is said to be *representable as sets* if a bijective function $f : (\mathcal{L}, \preceq) \rightarrow (2^C, \subseteq)$ exists and its inverse function f^{-1} is computable, such that:

$$X \preceq Y \Longleftrightarrow f(X) \subseteq f(Y)$$

In the sequel, a problem representable as sets will be referred to as "isomorphic to a boolean lattice".

A salient feature of this latter restriction relies on the notion of dualization [13,21], well known in combinatorics as minimal transversals of a hypergraph.

5.2 Three Scalable Components of the Query Rewriting Algorithm

In our query rewriting algorithm, three enumeration problems have been identified as possible bottleneck: $S_1(E, w)$ computation, $S_2(n, w.R_v)$ computation and minimal transversal of a hypergraph. This section shows how these three problems can be reformulated in this framework.

Reformulating the Problems As Pattern Mining Problems. Problems of the framework have to be *enumeration problems under constraints*, i.e., of the form "enumerate all the patterns that satisfy a condition". Consequently, the first step is to reformulate our problems in such pattern mining problems.

$S_1(E, w)$ **subproblem:** $S_1(E, w)$ consists in extracting the maximally-contained rewritings of the conjunct $\forall w.E$ of Q. In other words, the problem is to enumerate all minimal subsets of $\mathcal{V}$ whose intersection of their restricted set of values for the word w is included in E.
$S_1(E, w) = min_{\subseteq}(U \in 2^{\mathcal{V}} | \forall V_i \in U, \forall w.E_i \in V_i$ and $\bigcap_{V_i \in U} E_i \subseteq E)$

$S_2(n, w.R_v)$ **subproblem:** $S_2(n, w.R_v)$ consists in extracting the maximally-contained rewritings of the conjunct $\forall w. \leq n\ R_v$ of Q. In other words, the problem is to enumerate all minimal subsets of $\mathcal{V}$ whose cardinality of the intersection of their restricted set of values for the word $w.R_v$ is smaller or equal to n.
$S_2(n, w.R_v) = min_{\subseteq}(U \in 2^{\mathcal{V}} | \forall V_i \in U, \forall w.R_v.E_i \in V_i$ and $|\bigcap_{V_i \in U} E_i| \leq n)$

Max-rewritings generation (from Section 4.2) : Let $Q \equiv \sqcap_{i=1}^{n} \forall w_i.P_i$ be a query and B its buckets, i.e., $B = \bigcup_{i=1}^{n} B(w_i, P_i)$. Thanks to the hypergraph-based characterization, the maximal-contained rewritings, or $\mathcal{MCR}$, generation problem can be reformulated as enumerating all minimal transversals of the hypergraph $\mathcal{H}_B = (V_B, E_B)$, where V_B is composed of all the views or conjunction of views of the buckets ($V_B = \{v \mid v \in B(w_i, P_i), \forall i \in [1, n]\}$) and E_B consists of the set associated with each bucket ($E_B = \{e \mid e \in B\}$) .
$\mathcal{T}_{\mathcal{H}_B} = \{X \subseteq V_B \mid X$ minimal traversal of $\mathcal{H}_B\}$

Defining the Language, the Predicate and Proving Monotonicity. Once a problem is suspected to fit into the framework, the pattern language, the predicate and the partial order among patterns must be properly defined to go further. Moreover, predicate monotonicity has to be proven. In this subsection, we check all these aspects to fit our three subproblems in the theoretical framework. Proofs of properties and theorems of this subsection are given in annex D, page 65.

$S_1(E, w)$ **subproblem:**

1. The pattern language is $\mathcal{L}_{S_1(E,w)} = \{X \mid X \subseteq \mathcal{V}\}$
2. The predicate $P_{S_1(E,w)}(E, X)$ is true iff for all $V_i \in X$ and $w.E_i \in V_i$, $\bigcap_{V_i \in X} E_i \nsubseteq E$.
3. The partial order over $\mathcal{L}_{S_1(E,w)}$ is the set inclusion $\subseteq$.

Let X be a set of views satisfying $P_{S_1(E,w)}$, i.e., $P_{S_1(E,w)}(E, X) =$ true. It is clear that any subset Y of X also satisfies $P_{S_1(E,w)}$, since $\bigcap_{V_i \in X} E_i \subseteq \bigcap_{V_j \in Y} E_j$.

Property 1. The predicate $P_{S_1(E,w)}(E, X)$ is anti-monotone w.r.t. set inclusion.

The $S_1(E, w)$ problem can be reformulated as follows:

Theorem 4. $S_1(E, w) = \mathcal{B}d^-(\mathit{Th}(E, \mathcal{L}_{S_1(E,w)}, P_{S_1(E,w)}))$

$S_2(n, w.R_v)$ **subproblem:**

1. The pattern language is $\mathcal{L}_{S_2(n,w.R_v)} = \{X \mid X \subseteq \mathcal{V}\}$
2. The predicate $P_{S_2(n,w.R_v)}(n, X)$ is true iff for all $V_i \in X$ and $w.R_v.E_i \in V_i$, $|\bigcap_{V_i \in X} E_i| > n$.
3. The partial order is $\subseteq$.

Let X be a set of views satisfying $P_{S_2(n,w.R_v)}$, i.e., $P_{S_2(n,w.R_v)}(n, X) =$ true. It is clear that any subset Y of X also satisfies $P_{S_2(n,w.R_v)}$, since $|\bigcap_{V_i \in X} E_i| \leq |\bigcap_{V_j \in Y} E_j|$.

Property 2. The predicate $P_{S_2(n,w.R_v)}(n, X)$ is anti-monotone w.r.t. set inclusion.

The $S_2(n, w.R_v)$ problem can be reformulated as follows:

Theorem 5. $S_2(n, w.R_v) = \mathcal{B}d^-(\mathit{Th}(n, \mathcal{L}_{S_2(n,w.R_v)}, P_{S_2(n,w.R_v)}))$

$\mathcal{T}_{\mathcal{H}_B}$ **subproblem:**

1. The pattern language is $\mathcal{L}_{\mathcal{T}_{\mathcal{H}_B}} = \{X \mid X \subseteq V_B\}$
2. The predicate $P_{\mathcal{T}_{\mathcal{H}_B}}(\mathcal{H}_B, X)$ is true iff X is not a transversal of $\mathcal{H}_B$, i.e., if $\exists H \in E_B$ such as $X \cap H = \emptyset$.
3. The partial order is $\subseteq$.

It is also clear that any subset of non-transversal element is also non-transversal.

Property 1. The predicate $P_{\mathcal{T}_{\mathcal{H}_B}}(\mathcal{H}_B, X)$ is anti-monotone w.r.t. set inclusion.

The minimal transversals generation problem can be reformulated as follows:

Theorem 6. $\mathcal{T}_{\mathcal{H}_B} = \mathcal{B}d^-(\mathit{Th}(\mathcal{H}_B, \mathcal{L}_{MaxRew(B)}, P_{MaxRew(B)}))$

Clearly, our problems are representable as sets, i.e., isomorphic to a boolean lattice. The function f given in Section 5.1 is the identity function. Consequently, the *iZi* library can be directly used to solve these three subproblems.

6 Experimental Evaluation

A query rewriting prototype (figure [1]) has been implemented based on the theoretical investigations introduced so far. It takes as input a query Q expressed in terms of schema $\mathcal{S}$ and returns the set of all the maximally-contained rewritings of Q. The prototype is composed of two parts. The first one parses and normalizes the query Q from the schema $\mathcal{S}$ stored in a database. The second one, i.e., `ComputeRew`, is devoted to the computation of the query rewritings. This part consists of two components: `BucketsComputing` and `RewritingGeneration`. As shown by the algorithm [2], the `BucketsComputing` component requires as input the views $\mathcal{V}$ stored in a database. Moreover, as seen in Section [5], both components use the `iZi` library.

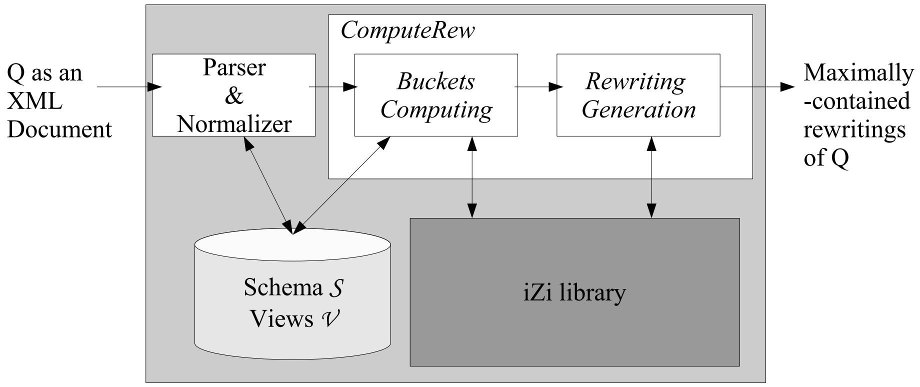


Fig. 1. Prototype description

`BucketsComputing` and `RewritingGeneration` (the most costly operations of our prototype) have been implemented using the generic APriori-like implementation provided in `iZi` [11]. The use of the APriori-like algorithm is motivated by two main reasons. First, it gives without any overhead the negative border, i.e., the solution of our subproblems. Second, several benchmarks [6,5] have shown that this algorithm is particularly efficient for discovering "not too large" solutions, which turns out to be the case in our experiments (see below).

Our implementation has been evaluated on synthetic dataset. Our objective has been to show the scalability of our proposition with respect to the number of views. More particularly, we focus on the three most costly steps of our implementation. Our first experiments focus on the computation of the sets S_1 and S_2, i.e., the computation of the rewritings due to value constraints. Second, we experiment the minimal traversals computation. The experimentations were performed on a PC with 2.6GHz P4 pro CPU and 3Go RAM.

S_1 and S_2 Computation. In this first part of the experimentations, synthetic datasets have the following characteristics. The values of constraints are chosen

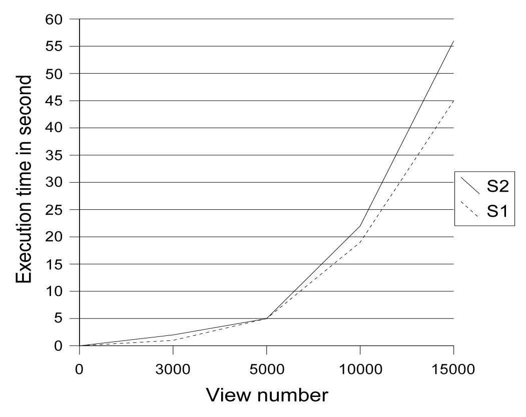


Fig. 2. Performances of S_1 and S_2 computation

among 33000. In figure 2, cardinality of the constraints is less than 10 while the number of views takes its values in the set $\{3000, 5000, 10000, 15000\}$.

In figure 3, the number of views is fixed to 5000 while the maximal cardinality of the constraints is either 10 or 20 or 30 or 40.

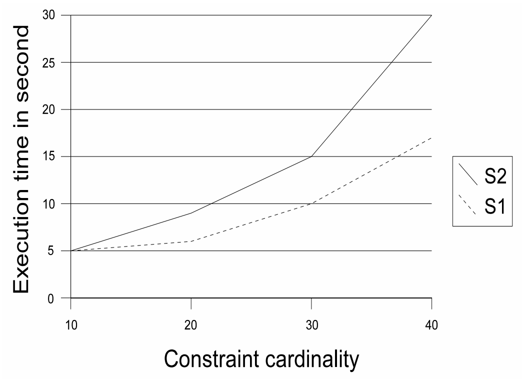


Fig. 3. Effect of value constraints cardinalities on performance

Figure 2 shows that our implementation handles up to 15000 views in an acceptable time, less than 60 seconds. Until about 10000 views, implementation remains efficient. In figure 3, we fix the number of views to 5000 and concentrate on the impact of constraints cardinality on the execution time. The implementation is very efficient for value constraints having a cardinality less than 30.

Minimal Transversal Computation. In our context, one of the problem for the minimal transversal computation is the huge number of vertices (i.e., views) that may occur in the same edge (i.e., bucket). To reduce this number, an optimization has been brought to the minimal transversal computation. Actually, the idea is to drastically reduce the number of vertices by regrouping together vertices which belong to the same edges. For example, if vertices a and b belong to the same edges, these two vertices can be replaced by a unique vertice a'. Then, the minimal transversal computation is applied on this "reduced" hypergraph. At the end, to have the solutions of the initial hypergraph, each transversal containing a' is replaced by two transversals: one with a instead of a' and one with b instead of a'. Thanks to the characteristics of our hypergraphs, i.e., a very small number of edges and huge number of vertices, this optimization is very effective in practice.

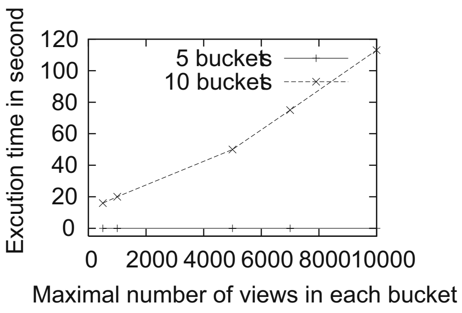


Fig. 4. Performance of the minimal transversal computation

The datasets used in the experiments are characterized by their number of buckets (i.e., number of edges of the hypergraph), their maximal number of views or conjunction of views in the buckets (i.e., the maximal size of the edges) and their total number of views or conjunction of views (i.e., the total number of vertices). The figure 4 presents the execution time for datasets with 5 and 10 buckets. The maximal number of views (or conjunction of views), in x-axis, is equal to the total number of views. As shown by this figure, our implementation can handle 10000 views instantaneously when processing 5 buckets. For the dataset with 10 buckets, it can process until 10000 views in an acceptable time. Even if this number of buckets seems small, recall that each bucket corresponds to a condition in the initial query, and consequently having more than 10 conditions for a single query stills rare.

For more buckets, despite the use of scalable data structures in the implementation, the cost of rewriting generation remains high. However, such implementation improves significantly an approach that would compute a cartesian product. In particular, our optimization for the minimal transversal computation reduced the number of vertices by a factor of 1.5 to 20 according to the datasets being studied. Moreover, even if the worst case (i.e., the cartesian product) can hardly be optimized, this case remains rare in our application since we have lot of views and a small number of buckets. On the other hand, our application supports the creation of neighborhood vertices. Consequently, our query rewriting prototype can take advantage of the two optimizations: transversal minimal computation and neighborhood vertices aggregation.

Experimental results show clearly the feasibility and scalability of our approach.

7 Conclusion

Our work supplies innovative and complementary contribution to existing works on answering queries using views by considering a new kind of constraints that can be very useful in practical situations. More precisely, we investigated this problem in the setting of the logic $\mathcal{ALN}(\mathcal{O}_v)$ that allows the expression of value constraints. We show that it is possible to compute all the certain answers of a given query Q by computing its maximally-contained rewritings. Furthermore, our work is the first to use efficient data mining techniques [11] to improve the scalability of a query rewriting Bucket-like algorithm. A query rewriting prototype has been implemented. This prototype is based on an existing data mining tool [11] for the bucket construction and for the global rewritings computation. Experimental results confirm the interest of our approach.

References

1. Abiteboul, S., Duschka, O.M.: Complexity of answering queries using materialized views. In: PODS 1998, Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 254–263. ACM Press, New York (1998)
2. Afrati, F.N., Li, C., Mitra, P.: Answering queries using views with arithmetic comparisons. In: Popa, L. (ed.) PODS 2002, Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 209–220. ACM, New York (2002)
3. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Bocca, J.B., Jarke, M., Zaniolo, C. (eds.) VLDB 1994, Proceedings of the International Conference on Very Large Data Bases, pp. 487–499. Morgan Kaufmann, San Francisco (1994)
4. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, Cambridge (2003)
5. Bayardo Jr., R.J., Goethals, B., Zaki, M.J. (eds.): FIMI 2004, Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations, UK, November 2004. CEUR Workshop Proceedings, vol. 126 (2004) CEUR-WS.org
6. Bayardo Jr., R.J., Zaki, M.J. (eds.): FIMI 2003, Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations, USA, November. CEUR Workshop Proceedings, vol. 90 (2003) CEUR-WS.org
7. Beeri, C., Halevy, A., Rousset, M.C.: Rewriting Queries Using Views in Description Logics. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) PODS 1997, Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Tucson, Arizona, May 12–14, pp. 99–108. ACM Press, New York (1997)

8. Borgida, A., Patel-Schneider, P.F.: A semantics and complete algorithm for subsumption in the classic description logic. Journal of Artificial Intelligence Research (JAIR) 1, 277–308 (1994)
9. De Marchi, F., Petit, J.-M.: Zigzag: a new algorithm for mining large inclusion dependencies in database. In: ICDM 2003, Proceedings of the IEEE International Conference on Data Mining, pp. 27–34. IEEE Computer Society, Los Alamitos (2003)
10. Eiter, T., Gottlob, G.: Identifying the minimal transversals of a hypergraph and related problems. SIAM Journal on Computing 24(6), 1278–1304 (1995)
11. Flouvat, F., De Marchi, F., Petit, J.-M.: iZi: A new toolkit for pattern mining problems. In: An, A., Matwin, S., Raś, Z.W., Ślęzak, D. (eds.) ISMIS 2008. LNCS, vol. 4994, pp. 131–136. Springer, Heidelberg (2008)
12. Goasdoué, F., Rousset, M.-C.: Answering queries using views: A krdb perspective for the semantic web. ACM Transactions on Internet Technology 4(3), 255–288 (2004)
13. Gunopulos, D., Khardon, R., Mannila, H., Saluja, S., Toivonen, H., Sharm, R.S.: Discovering all most specific sentences. ACM transactions on database systems 28(2), 140–174 (2003)
14. Halevy, A.Y.: Answering queries using views: A survey. VLDB Journal 10(4), 270–294 (2001)
15. Horrocks, I., Sattler, U.: Ontology reasoning in the shoq(d) description logic. In: Nebel, B. (ed.) IJCAI 2001, International Joint Conferences on Artificial Intelligence, pp. 199–204. Morgan Kaufmann, San Francisco (2001)
16. Huhtala, Y., Kärkkäinen, J., Porkka, P., Toivonen, H.: Tane: An efficient algorithm for discovering functional and approximate dependencies. Computer Journal 42(2), 100–111 (1999)
17. Khachiyan, L., Boros, E., Elbassioni, K.M., Gurvich, V.: An efficient implementation of a quasi-polynomial algorithm for generating hypergraph transversals and its application in joint generation. Discrete Applied Mathematics 154(16), 2350–2372 (2006)
18. Küsters, R.: Non-Standard Inferences in Description Logics. LNCS, vol. 2100. Springer, Heidelberg (2001)
19. Lenzerini, M.: Data integration: A theoretical perspective. In: PODS 2002, Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Madison, Wisconsin (2002)
20. Levy, A.Y., Rajaraman, A., Ordille, J.J.: Querying heterogeneous information sources using source descriptions. In: Vijayaraman, T.M., Buchmann, A.P., Mohan, C., Sarda, N.L. (eds.) VLDB 1996, Proceedings of the International Conference on Very Large Data Bases, pp. 251–262. Morgan Kaufmann, San Francisco (1996)
21. Mannila, H., Toivonen, H.: Levelwise search and borders of theories in knowledge discovery. Data mining and knowledge discovery 1(3), 241–258 (1997)
22. Pottinger, R., Halevy, A.Y.: Minicon: A scalable algorithm for answering queries using views. VLDB Journal 10(2-3), 182–198 (2001)
23. Schaerf, A.: Reasoning with individuals in concept languages. Data & Knowledge Engineering 13(2), 141–176 (1994)
24. Ullman, J.D.: Information integration using logical views. Theoretical Computer Science 239(2), 189–210 (2000)

A Subsumption Characterization

Normalization rules that allow to transform $\mathcal{ALN}(\mathcal{O}_v)$ concepts into their normal forms are described in the table 2. Letter A denotes an atomic concept. Letters E, E_1 and E_2 denote set of values while D and D' specify any kind of concept. To simplify the set of normalization rules, we assume that any description like $(\geq 0R)$ or $(\forall R_C.\top_C)$ or $(\forall R_V.\top_V)$ is transformed into $\top_C$ while $D \sqcap \top_C$ (if D is not a set of values otherwise $D \sqcap \top_C$ is inconsistent) and $E \sqcap \top_V$ become respectively D and E.

First rules (1) and (2) are recursively applied until a saturation point. Next the rule (3) is applied only once. Rules (4) to (10) are then applied recursively until a saturation point. Finally, the rule (11) is applied recursively until a saturation point.

Table 2. Normalization rules

(1) $\forall w.D \sqcap \forall w.D' \rightarrow \forall w.(D \sqcap D')$
(2) $E_1 \sqcap E_2 \rightarrow E$ such that $E = E_1 \cap E_2$
(3) $\forall R_v.E \rightarrow \forall R_v.E \sqcap (\leq kR_v)$, where $\lvert E \rvert = k$
(4) $\leq 0R \sqcap \forall R.D \rightarrow \leq 0R$
(5) $A \sqcap \neg A \rightarrow \bot$
(6) $(\geq nR) \sqcap (\leq mR) \rightarrow \bot$ if $n > m$
(7) $(\geq nR) \sqcap (\geq mR) \rightarrow (\geq max(n,m)R)$
(8) $(\leq nR) \sqcap (\leq mR) \rightarrow (\leq min(n,m)R)$
(9) $D \sqcap \bot \rightarrow \bot$
(10) $\forall R.\bot \rightarrow \leq 0R$
(11) $\forall w.(D \sqcap D') \rightarrow \forall w.D \sqcap \forall w.D'$

We present now the proof of the theorem 1.

Proof (Proof of theorem 1)

– completeness ($\Rightarrow$) The proof of the completeness of this theorem is derived from the structural characterization of subsumption in CLASSIC logics [8]. Let C and D be two concept descriptions. Assume that D is in its normal form, i.e., $D \equiv \forall w_1.P_1 \sqcap ... \sqcap \forall w_n.P_n$. Let G_C be the canonical description graph associated with C and assume that the subsumption algorithm given in [8] returns true with the input D and G_C, thas is, $C \sqsubseteq D$. Therefore, G_C verifies the conditions stated in [8].

We can construct a concept from the graph G_C, and then apply rule 11 of table 2 to expand it so there are no nested conjunctions. We get then a description of C in our expected normal form that verifies the following conditions:

either $C \equiv \bot$ or $D \equiv \top_C$, or
for every $\forall w.P \in D$, we have
(a) $\forall w.P \in C$ or,
(b) $\forall w.E' \in C$ with $E' \subseteq E$ if $P = E$ or,

(c) $\forall w.(\leq kR) \in C$ with $k \leq n$ if $P =\leq nR$ or,
(d) $\forall w.R.E \in C$ with $|E| \leq n$ if $P =\leq nR$ and $R \in \mathcal{R}_v$ or,
(e) $\forall w.(\geq kR) \in C$ with $k \geq n$ if $P =\geq nR$ or,
(f) $\forall v.(\leq 0R) \in C$ with vR prefix of w.

These conditions are those stated by theorem 1.

– soundness ($\Leftarrow$) Let C and D be two concept descriptions in their normal form such that $D \equiv \sqcap_{i=1}^{n} \forall w_i.P_i$. We have that either condition 1) or conditions 2) of theorem 1 are verified for each $\forall w_i.P_i$ in D. We want to show that implies $C \sqsubseteq D$.
  1) if $C \equiv \bot$ then the interpretation of C is empty and any description D subsumes it. if $D \equiv \top_C$ then its interpretation is the set of all *classic* individuals, i.e. δ_C. Since any concept in $\mathcal{ALN}(\mathcal{O}_v)$ is a subset of δ_C any concept C is subsumed by $\top_C$.
  2) otherwise one of the following condition occurs
     * if $\forall v_i.\bot$ where $w_i = v_i u$ belongs to C. Hence $C \sqsubseteq \forall v_i.\bot \sqsubseteq \forall v_i.u.P_i$ and $C \sqsubseteq \forall w_i.P_i$.
     * if $P_i = A$ or $P_i = \neg A$ then $\forall w_i.P_i$ is in the description of C and $C \sqsubseteq \forall w_i.P_i$.
     * if $P_i = (\leq nR)$ then $\forall w_i.(\leq kR)$ where $k \leq n$ is in the description of C and $C \sqsubseteq \forall w_i.(\leq kR) \sqsubseteq \forall w_i.P_i$, because $k \leq n$.
     * if $P_i = (\geq nR)$ then $\forall w_i.(\geq kR)$ where $k \geq n$ is in the description of C and $C \sqsubseteq \forall w_i.(\geq kR) \sqsubseteq \forall w_i.P_i$, because $k \geq n$.
     * if $P_i = E$ then $\forall w_i.E'$ where $E' \subseteq E$ is in the description of C and $C \sqsubseteq \forall w_i.E' \sqsubseteq \forall w_i.P_i$, because $E' \subseteq E$.

     Hence for all $\forall w_i.P_i$ in D we have $C \sqsubseteq \forall w_i.P_i$ and $C \sqsubseteq \sqcap_{i=1}^{n} \forall w_i.P_i$ and $C \sqsubseteq D$

B From Query Answering Using Views to Query Rewriting Using Views

Let us given the proof of lemma 1

Proof (Proof of lemma 1)

Let t be a certain answer.

Let $Q \equiv \forall w_1.P_1 \sqcap \ldots \sqcap \forall w_n.P_n$ a query.

Let $\mathcal{I}$ be a model of $\mathcal{S}$, s.t. $V^{ext} \subseteq V^{\mathcal{I}}, \forall\, V \in \mathcal{V}$.

We want to show that if t is a certain answer of Q, i.e., $t \in Q^{\mathcal{I}}$, then there exists a conjunction C of views from $\mathcal{V}$ s.t. $C \sqsubseteq Q$ and $t \in C^{ext}$.

According to the definition 1 on certain answers, if t is a certain answer of Q, then $t \in \mathcal{V}^{ext}$. There exists then a subset of $\mathcal{V}$ whose the views contains t in their extension.

Let M be the set of all the views that contains t in their extension. Let C_M, the conjunction of views in M. Therefore $t \in C_M^{ext}$.

It remains to show that C_M is necessarily subsumed by the query Q. To achieve that, we assume that C_M is not subsumed by Q and we show that it is possible to find an interpretation $\mathcal{J}$, model of $(\mathcal{S},\mathcal{V})$ in which t does not belong to $Q^{\mathcal{J}}$. Therefore t is not a certain answer.

We want to show that $C_M \sqsubseteq Q$, i.e., that $\forall \mathcal{I}$, model of $(\mathcal{S},\mathcal{V})$ s.t. $V^{ext} \subseteq V^{\mathcal{I}}$ for each view $V \in \mathcal{V}$, we have $C_M^{\mathcal{I}} \subseteq Q^{\mathcal{I}}$.

Assume that $C_M \not\sqsubseteq Q$, then according to theorem 1, there exists $\forall w_k.P_k \in Q$ s.t. $C_M \not\sqsubseteq \forall w_k.P_k$. Consequently, there exists an interpretation $\mathcal{K}$, model of $(\mathcal{S},\mathcal{V})$ with $V^{ext} \subseteq V^{\mathcal{K}}$ for each view $V \in \mathcal{V}$, s.t. $C_M^{\mathcal{K}} \not\subseteq \forall w_k.P_k^{\mathcal{K}}$. Therefore there exists x in $\delta_C^{\mathcal{K}}$ s.t. $x \in C_M^{\mathcal{K}}$ but s.t. $x \notin (\forall w_k.P_k)^{\mathcal{K}}$.

If x was t, then t could not be a certain answer. We are then going to define an interpretation $\mathcal{J}$, model of $(\mathcal{S},\mathcal{V})$ with $V^{ext} \subseteq V^{\mathcal{J}}$ for each view $V \in \mathcal{V}$, s.t. there exists $x = t$ in $\delta_C^{\mathcal{J}}$ with $t \in C_M^{\mathcal{J}}$ but s.t. $t \notin \forall w_k.P_k^{\mathcal{J}}$.

Let $\mathcal{J}$ an interpretation defined as follow: **(i)** $\forall i \in [1,n]$, we have $(\forall w_i.P_i)^{\mathcal{J}} = (\forall w_i.P_i)^{\mathcal{K}}$, **(ii)** $(\forall w_k.P_k)^{\mathcal{J}} = (\forall w_i.P_i)^{\mathcal{K}} - \{t\} \cup \{t'\}$ with t' a new individual ($\Delta^{\mathcal{J}} = \Delta^{\mathcal{K}} \cup \{t'\}$) and **(iii)** every $(x,t) \in R^{\mathcal{K}}$ is replaced by $(x,t') \in R^{\mathcal{J}}$ and respectively, every $(t,y) \in R^{\mathcal{K}}$ is replaced by $(t',y) \in R^{\mathcal{J}}$.

Show now that $\mathcal{J}$ remains consistent with the view extensions, i.e., $V^{ext} \subseteq V^{\mathcal{J}}$ for each view $V \in \mathcal{V}$.

For the views in M:

The views V in M are not subsumed by $\forall w_k.P_k$ and are s.t. $t \in V^{\mathcal{K}}$. Therefore, replacing t by t' in $\forall w_k.P_k^{\mathcal{J}}$ has no impact over the interpretation of $V^{\mathcal{J}}$ Moreover, the transformation of $\mathcal{K}$ in $\mathcal{J}$ preserves the semantics of every axioms in $\mathcal{V}$ thanks to the property **(iii)** that defines $\mathcal{J}$.

Therefore, for each view V in M, we have $V^{\mathcal{J}} = V^{\mathcal{K}}$ and the relationship $V^{ext} \subseteq V^{\mathcal{J}}$ is verified.

For the views in $\mathcal{V}\backslash M$:

The views V in $\mathcal{V}\backslash M$ are s.t. $t \notin V^{\mathcal{K}}$. Consequently, the deletion of t in $\forall w_k.P_k^{\mathcal{J}}$ does not modify the interpretation $V^{\mathcal{J}}$. Moreover, according to the property **(iii)** that defines $\mathcal{J}$, for each view V in $\mathcal{V}\backslash M$, we have $V^{\mathcal{J}} = V^{\mathcal{K}}$ and the relationship $V^{ext} \subseteq V^{\mathcal{J}}$ is verified.

Then there exists an interpretation $\mathcal{J}$ of $(\mathcal{S},\mathcal{V})$, s.t. $V^{ext} \subseteq V^{\mathcal{J}}$, $\forall V \in \mathcal{V}$, in which t does not belong to $Q^{\mathcal{J}}$. Consequently, t is not a certain answer of Q which contradicts the initial assumption. Therefore we have $C_M \sqsubseteq Q$.

Now follows proof of theorem 2.

Proof (Proof of theorem 2). The proof of this theorem lies on the following principle. Each Q_i is a maximally-contained rewriting of Q. Thus by definition, Q_i is subsumed by Q. Consequently, we have $Q_i(\mathcal{V}^{ext}) \subseteq Ans(Q,\mathcal{V}^{ext})$ for every $i \in \{1,\ldots,n\}$ and $\cup_{i=1}^{n} Q_i(\mathcal{V}^{ext}) \subseteq Ans(Q,\mathcal{V}^{ext})$. It remains to show that the set of certain answers is contained in $\cup_{i=1}^{n} Q_i^{ext}$. Let t be a certain answer, then there exists a conjunction Q' of views s.t. $t \in Q'(\mathcal{V}^{ext})$ and $Q' \sqsubseteq Q$. Therefore either Q' is maximally-contained in Q and there exists $i \in \{1,\ldots,n\}$ s.t. $Q' \equiv Q_i$, or there exists $Q_i \in \{Q_1,\ldots,Q_n\}$ s.t. $Q' \sqsubseteq Q_i$. Consequently, we have $t \in Q_i(\mathcal{V}^{ext})$.

The proof of lemma 2 lies on lemma 4 below that characterizes the subsumption between two consistent conjunctions of views.

Lemma 4. *Let $\mathcal{V}$ be a terminology in $\mathcal{ALN}(\mathcal{O}_v)$. Let Q_1 and Q_2 two consistent conjunctions of views in $\mathcal{V}$.*
$Q_1 \sqsubseteq Q_2$ iff Q_2 is made of a subset of views occurring in Q_1.

Proof (Proof of lemma 4)

(⇐) Let Q_{i_1} and Q_{i_2} be two conjunctions of views in $\mathcal{V}$. We are going to show that if Q_{i_2} refers a subset of views in Q_{i_1} then $Q_{i_1} \sqsubseteq Q_{i_2}$.
Assume that $Q_{i_1} \equiv V_1 \sqcap \ldots \sqcap V_n$.
Q_{i_2} refers only a subset of $\{V_1, \ldots, V_n\}$.
Assume that Q_{i_2} is equivalent to the following expression: $Q_{i_2} \equiv V_1 \sqcap \ldots \sqcap V_{j-1} \sqcap V_{j+1} \sqcap \ldots \sqcap V_n$.
Therefore, since the used formal framework is $\mathcal{ALN}(\mathcal{O}_v)$, for all interpretation $\mathcal{I}$, we have $Q_{i_1}^{\mathcal{I}} \subseteq Q_{i_2}^{\mathcal{I}}$.

(⇒) Let Q_{i_1} and Q_{i_2} be two consistent conjunctions of views from $\mathcal{V}$. We are going to show that if $Q_{i_1} \sqsubseteq Q_{i_2}$ then Q_{i_2} refers a subset of views occurring in Q_{i_1}.
Since $\mathcal{V}$ is a primitive terminology that is normalized and expanded, each view V_{i_j} is a conjunction between a concept description D_{i_j} and a unique atomic concept $\overline{V_{i_j}}$.
Assume that $Q_{i_1} \equiv V_1 \sqcap \ldots \sqcap V_n$, where $V_i \in \mathcal{V}$ for all $i \in \{1, ..., n\}$
then $Q_{i_1} \equiv (D_1 \sqcap \ldots \sqcap D_n) \sqcap (\overline{V_1} \sqcap \ldots \sqcap \overline{V_n})$.
Every concept that subsumes Q_{i_1} must contain in its description a conjunction of views from the set $\{\overline{V_1}, \ldots, \overline{V_n}\}$.
In other words, every concept that subsumes Q_{i_1} refers a subset of $\{\overline{V_1}, \ldots, \overline{V_n}\}$.
Since every concept $\overline{V_i}$ with $i \in \{1, ..., n\}$, is an unique atomic concept associated with a single view V_i, every concept that subsumes Q_{i_1} refers a subset of $\{V_1, \ldots, V_n\}$.

Proof of lemma 2 is given below.

Proof (Proof of lemma 2)

(⇒) Assume that $Q' \equiv V_1 \sqcap \ldots \sqcap V_n$ is a maximally-contained and conjunctive rewriting Q in terms of $\mathcal{V}$.
We want to show that for every concept Q'' obtained by removing from Q' one of its conjuncts V_i, we have $Q'' \not\sqsubseteq Q$.
Assume that $Q'' \equiv V_1 \sqcap ... \sqcap V_{j-1} \sqcap V_{j+1} \sqcap ... \sqcap V_n$, obtained by removing from Q' one view V_j, with $j \in \{1, ..., n\}$, is subsumed by Q (i.e. $Q'' \sqsubseteq Q$). We have also $Q' \sqsubseteq Q''$. We are going to show that in this case, Q' is not maximally contained in Q.
To achieve that, we show that Q'' is not equivalent to Q'.
Let $Q' \equiv V_1 \sqcap ... \sqcap V_n$ and $Q'' \equiv V_1 \sqcap ... \sqcap V_{j-1} \sqcap V_{j+1} \sqcap ... \sqcap V_n$. According to the open word assumption,
$Q'' \equiv (D_1 \sqcap \ldots \sqcap D_{(j-1)} \sqcap D_{(j+1)} \sqcap \ldots \sqcap D_n) \sqcap (A_1 \sqcap \ldots \sqcap A_{(j-1)} \sqcap A_{(j+1)} \sqcap \ldots \sqcap A_n)$

$Q' \equiv (D_1 \sqcap \ldots \sqcap D_{(j-1)} \sqcap D_{(j+1)} \sqcap \ldots \sqcap D_n) \sqcap (A_1 \sqcap \ldots \sqcap A_{(j-1)} \sqcap A_{(j+1)} \sqcap \ldots \sqcap A_n) \sqcap D_i \sqcap A_i$.
Q' and Q'' are not equivalent because A_i does not subsume $A_1 \sqcap \ldots \sqcap A_{(j-1)} \sqcap A_{(j+1)} \sqcap \ldots \sqcap A_n$. Indeed A_i is an atomic concept associated with a single view V_i that occurs only once in Q'.

Then there exists Q'' s.t. $Q'' \not\equiv Q'$ and $Q' \sqsubseteq Q'' \sqsubseteq Q$. Therefore Q' cannot be maximally-contained in Q.

($\Leftarrow$) We have to show that
if for all $Q'' \equiv V_{i_1} \sqcap \ldots \sqcap V_{i_{n-1}}$, s.t. $\{V_{i_1}, ..., V_{i_{n-1}}\} \subseteq \{V_1, ..., V_n\}$, we have $Q'' \not\sqsubseteq Q$,
then $Q' \equiv V_1 \sqcap \ldots V_n$ is a maximally-contained rewriting of Q.
In other words, we have to show that there is no Q'' s.t. $Q' \sqsubseteq Q''$ and $Q'' \sqsubseteq Q$ with $Q' \not\equiv Q''$.

We refer by (*) the following assumption: for all $Q'' \equiv V_{i_1} \sqcap \ldots \sqcap V_{i_{n-1}}$, s.t. $\{V_{i_1}, ..., V_{i_{n-1}}\} \subseteq \{V_1, ..., V_n\}$, we have $Q'' \not\sqsubseteq Q$.

According to hypothesis of lemma 2 $Q' \equiv V_1 \sqcap \ldots V_n$ is subsumed by Q.

Assume that Q' is not a maximally-contained rewriting of Q. Then there exists a conjunction of views Q_1 s.t. $Q' \sqsubseteq Q_1 \sqsubseteq Q$ et $Q' \not\equiv Q_1$.
According to lemma 4, Q_1 subsumes strictly Q' if Q_1 refers only a strict subset of views occurring in Q' and in this case, $Q_1 \not\equiv Q'$.
Let $Q' \equiv V_1 \sqcap ... \sqcap V_n$ et $Q_1 \equiv V_1 \sqcap ... \sqcap V_{j-1} \sqcap V_{j+1} \sqcap ... \sqcap V_n$.
$Q_1 \equiv (D_1 \sqcap \ldots \sqcap D_{(j-1)} \sqcap D_{(j+1)} \sqcap \ldots \sqcap D_n) \sqcap (A_1 \sqcap \ldots \sqcap A_{(j-1)} \sqcap A_{(j+1)} \sqcap \ldots \sqcap A_n)$
and $Q' \equiv (D_1 \sqcap \ldots \sqcap D_{(j-1)} \sqcap D_{(j+1)} \sqcap \ldots \sqcap D_n) \sqcap (A_1 \sqcap \ldots \sqcap A_{(j-1)} \sqcap A_{(j+1)} \sqcap \ldots \sqcap A_n) \sqcap D_i \sqcap A_i$.
and A_i is an atomic concept.

Therefore there exists Q_1 formed with a subset of $\{V_1, ..., V_n\}$ and that is subsumed by Q. Then each conjunction of views that uses supersets of views from Q_1 is subsumed by Q. That contradicts the assumption (*).

Therefore Q' is a maximally-contained rewriting of Q.

C A Bucket-Based Algorithm for $\mathcal{ALN}(\mathcal{O}_v)$ Mediation System

C.1 Bucket Algorithm for $\mathcal{ALN}(\mathcal{O}_v)$

Now we give proof of lemma 3:

Proof (proof of lemma 3)
We have $Q' \equiv V_{i_1} \sqcap \ldots \sqcap V_{i_k}$ s.t. Q' is maximally-contained in Q.
Therefore, according to lemma 2, Q' is formed of a minimal subset of views s.t. $Q' \sqsubseteq Q$.

- If $P \in \{A, \neg A\}$ then according to theorem 1, one of the views contains $\forall w.P$. Since the set $\{V_{i_1}, \ldots V_{i_k}\}$ is minimal, one view is sufficient to rewrite $Q \equiv \forall w.P$ and $k = 1$.

- If $P = (\geq nr)$ then according to theorem 1, one of the views contains $\forall w.(\geq mr)$, with $m \geq n$. Since the set $\{V_{i_1}, \dots V_{i_k}\}$ is minimal, one view is sufficient to rewrite $Q \equiv \forall w.P$ and $k = 1$.
- If $P = (\leq\ n\, r)$, then according to theorem 1, one of the views contains $\forall w.(\leq mr)$ with $m \leq n$. As above, one view is sufficient to rewrite $Q \equiv \forall w.P$ and $k = 1$.
- If $P = (\leq\ n\, r)$ and $r \in \mathcal{R}_v$ (*), according to normalization rules 8) and 9) and theorem 1, we can find ($k \geq 1$) views that contain respectively a conjunct $\forall w.r.E_{i_j}$ s.t. $\cap_{j=1}^{k} E_{i_j} \subseteq E'$ and $card(E') \leq n$.

  The worst case occurs when the E_{i_j}'s have in pairs a single distinct value and, we have to infer $\forall w.(\leq\ n\, r)$, with $n = 0$. If the maximal number of values in the E_{i_j}'s is l then in worst case, $(l+1)$ sets of values are necessary to obtain the empty set. Therefore if $r \in \mathcal{R}_v$ and $P = (\leq\ n\, r)$, in the worst case $(l+1)$ views are necessary to rewrite $Q \equiv \forall w.(\leq nr)$ and $1 \leq k \leq (l+1)$.
- If $P = E$, according to normalization rule 8 and theorem 1, $k \geq 1$ views contain respectively a conjunct $\forall w.E_{i_j}$ such that $\cap_{j=1}^{k} E_{i_j} \subseteq E'$ and $E' \subseteq E$. The worst case occurs when E_{i_j} have in pairs a single distinct value, and we have to infer $\forall w.E$, with $E = \emptyset$. If the maximal number of values in the E_{i_j}'s is l then $(l + 1)$ sets of values are necessary to obtain the empty set. Therefore if $P = E$, in the worst case $(l + 1)$ views are necessary to rewrite $Q \equiv \forall w.E$ and $1 \leq k \leq (l + 1)$.
- otherwise, according to theorem 1, $Q' \sqsubseteq \leq\ 0\, r_1$ with r_1 a prefix of w. In the worst case, the concept $\leq\ 0\, r_1$ can be derived by a conjunction of $(p + 1)$ views:

  $V_{i_1} \sqsubseteq \forall r_1.r_2.\dots.r_p.(\leq qr)$
  $V_{i_2} \sqsubseteq \forall r_1.r_2.\dots.r_p.(\geq mr)$, with $q < m$
  $V_{i_3} \sqsubseteq \forall r_1.r_2.\dots.r_{p-1}.(\geq 1 r_k)$,
  ...
  $V_{i(p+1)} \sqsubseteq \forall r_1.(\geq 1 r_2)$.

  This sequence of concepts has been pointed out in [18]. However we can also obtain a conjunction of views that as V_{i_1}, is subsumed by $\forall w'.r_1.r_2.\dots.r_p.(\leq q\, r)$, if $r \in \mathcal{R}_v$, as seen in (*). Such conjunction of views can take the place of view V_{i_1} if such view V_{i_1} does not exist. Therefore, at most $l + 1 + p + 1$ views are necessary to obtain a rewriting $Q' \sqsubseteq \leq\ 0\, r_1$. Hence if $w \in E(Q')$, in the worst case, $(1 \leq k \leq l + p + 2)$ views are necessary to obtain a rewriting $Q' \sqsubseteq \forall w'.(\leq\ 0\, v)$

C.2 Max-Rewritings Generation

Proof (Proof of theorem 3)

Let Q' be a maximally-contained rewriting of Q built with the elements of the Q's buckets and $T_{Q'}$ the set of views forming Q'. Assume that T'_Q is not a minimal transversal of $\mathcal{H}_B$. Therefore there exists a minimal transversal $T_{Q''}$ in $\mathcal{H}_B$ such that $T_{Q''} \subset T_{Q'}$. As $T_{Q''}$ meets every edge in E_B and thus every bucket of Q, the conjunction Q'' of views from $T_{Q''}$ is a rewriting of Q and Q' cannot be a maximally-contained rewriting of Q.

D Query Rewriting Algorithm in $\mathcal{ALN}(\mathcal{O}_v)$ Using *iZi*

We give now the proof of property 1

Proof. Let $X \in \mathcal{L}_{S_1(E,w)}$ s.t. $P_{S_1(E,w)}(E,X)$ is true.

We have to prove that for all $Y \in \mathcal{L}_{S_1(E,w)}$ s.t. $Y \subseteq X$, $P_{S_1(E,w)}(E,Y)$ is true.
Let $X = \{E_{i_1}, ..., E_{i_n}\}$, $Y = \{E_{j_1}, ..., E_{j_k}\}$, with $Y \subseteq X$
As $P_{S_1(E,w)}(E,X)$ is true, $\cap_{j=1}^{n} E_{i_j} \nsubseteq E$.
As $Y \subseteq X$ then $\cap_{j=1}^{n} E_{i_j} \subseteq \cap_{q=1}^{k} E_{j_q}$,
and thus $\cap_{j=1}^{n} E_{i_j} \nsubseteq E$ or equivalently, $P_{S_1(E,w)}(E,Y)$ is true.

We give now the proof of Theorem 4

Proof. Let $IEE = \{X \in \mathcal{L}_{S_1(E,w)}$ s.t. $P_{S_1(E,w)}(E,X)$ is true $\}$. $\mathcal{B}d^+(IEE)$ gathers the most specialized true patterns. The negative border $\mathcal{B}d^-(IEE)$ gathers the most generalized false patterns, i.e., the minimal subsets of views whose intersection of their restricted set of values for the word w is included in E, that is equivalent to $S_1(E,w)$.

Let us given the proof of property 2

Proof. Let $X \in \mathcal{L}_{S_2(n,w.R_v)}$ s.t. $P_{S_2(n,w.R_v)}(E,X)$ is true.

We have to prove that for all $Y \in \mathcal{L}_{S_2(n,w.R_v)}$ s.t. $Y \subseteq X$, $P_{S_2(n,w.R_v)}(E,Y)$ is true.
Let $X = \{E_{i_1}, ..., E_{i_n}\}$, $Y = \{E_{j_1}, ..., E_{j_k}\}$, with $Y \subseteq X$.
As $P_{S_2(n,w.R_v)}(E,X)$ is true, $|\cap_{j=1}^{n} E_{i_j}| > n$.
As $Y \subseteq X$ then $\cap_{j=1}^{n} E_{i_j} \subseteq \cap_{q=1}^{k} E_{j_q}$,
and $|\cap_{q=1}^{k} E_{i_j}| > n$ and thus $P_{S_2(n,w.R_v)}(E,Y)$ is true.

Now follows the proof of theorem 5

Proof. Let $IEN = \{X \in \mathcal{L}_{S_2(n,w.R_v)}$ s.t. $P_{S_2(n,w.R_v)}(E,X)$ is true $\}$. $\mathcal{B}d^+(IEN)$ gathers the most specialized true patterns. The negative border $\mathcal{B}d^-(IEN)$ gathers the most generalized false patterns, i.e., the minimal subsets of views whose cardinality of the intersection of their restricted set of values for the word $w.R_v$ is smaller or equal to n, that is equivalent to $S_E(n, w.R_v)$.

Let us given the proof of property 3

Proof. Let $X \in \mathcal{L}_{\mathcal{T}_{\mathcal{H}_B}}$ s.t. $P_{\mathcal{T}_{\mathcal{H}_B}}(\mathcal{H}_B, X)$ is true.
We have to prove that for all $Y \in \mathcal{L}_{\mathcal{T}_{\mathcal{H}_B}}$ s.t. $Y \subseteq X$, $P_{\mathcal{T}_{\mathcal{H}_B}}(\mathcal{H}_B, Y)$ is true.
Let $Y \in \mathcal{L}_{\mathcal{T}_{\mathcal{H}_B}}$, with $Y \subseteq X$.
As $P_{\mathcal{T}_{\mathcal{H}_B}}(\mathcal{H}_B, X)$ is true, $\exists B_i \in E_B$ s.t. $X \cap B_i = \emptyset$.
As $Y \subseteq X$ then $B_i \cap Y = \emptyset$,
and thus $P_{\mathcal{T}_{\mathcal{H}_B}}(\mathcal{H}_B, Y)$ is true.

Now follows the proof of theorem 6

Proof. Let $NT = \{X \in \mathcal{L}_{\mathcal{T}_{\mathcal{H}_B}}$ s.t.$P_{\mathcal{T}_{\mathcal{H}_B}}(\mathcal{H}_B, X)$ is true $\}$. By definition, the negative border $\mathcal{B}d^-(NT)$ gathers the most generalized false patterns w.r.t. set inclusion, i.e., the minimal subsets of views which are transversal in $\mathcal{H}_B$, that is equivalent to $\mathcal{T}_{\mathcal{H}_B}$.

Combining a Logical and a Numerical Method for Data Reconciliation

Fatiha Saïs[1], Nathalie Pernelle[1], and Marie-Christine Rousset[2]

[1] LRI, Paris-Sud 11 University, and INRIA Futurs,
2-4 rue J. Monod, F-91893 ORSAY, France
{Fatiha.Sais, Nathalie.Pernelle}@lri.fr
[2] LIG - Laboratoire d'Informatique de Grenoble
BP 72, 38402 St MARTIN D'HERES, France
Marie-Christine.Rousset@imag.fr

Abstract. The reference reconciliation problem consists in deciding whether different identifiers refer to the same data, i.e. correspond to the same real world entity. In this article we present a reference reconciliation approach which combines a logical method for reference reconciliation called L2R and a numerical one called N2R. This approach exploits the schema and data semantics, which is translated into a set of Horn FOL rules of reconciliation. These rules are used in L2R to infer exact decisions both of reconciliation and non-reconciliation. In the second method N2R, the semantics of the schema is translated in an informed similarity measure which is used by a numerical computation of the similarity of reference pairs. This similarity measure is expressed in a non linear equation system, which is solved by using an iterative method. The experiments of the methods made on two different domains, show good results for both recall and precision. They can be used separately or in combination. We have shown that their combination allows to improve runtime performance.

Keywords: Semantic Data Integration, Ontologies, Automatic reasoning, Reference reconciliation, Equation system, Iterative resolution.

1 Introduction

The data reconciliation problem is one of the main problems encountered when different sources have to be integrated. It consists in deciding whether different data descriptions refer to the same real world entity (e.g. the same person or the same publication). For example, in a standard relational database a data description is a set of tuples referring to a given identifier. In the context of data integration, data descriptions are coming from different sources. These sources are heterogeneous, built in an autonomous way and for different business requirements. In such a context, the assumption of unique identifier does not hold: two different identifiers can refer to the same real world entity. We therefore prefer to use the term of *reference* instead of *identifier*. In the following like [1] we will use "*the reference reconciliation problem*" to refer to the data reconciliation problem. This problem is also known as the record linkage or the record matching problem [2,3,4], the entity resolution problem [5,6] or the object identification problem [7].

Schema heterogeneity is a major cause of the mismatch of the data descriptions between sources. Extensive research work has been done recently (see [8,9] for surveys) to reconcile schemas and ontologies through mappings. In this work, we assume that the schema heterogeneity problem has been solved. We focus on the data heterogeneity problem when data conform to the same global schema.

The conformity to a same global schema does not indeed prevent variations between data descriptions. For example, two descriptions of persons with the same attributes Last Name, First Name, Address can vary on the values of those attributes while referring to the same person, for instance, if the First Name is given entirely in one tuple, while it is abbreviated in the second tuple.

Therefore, the reference reconciliation problem is a crucial issue for data integration and raises multiple difficulties. First, different conventions and vocabularies can be used to represent and describe data. For example, in one source a contact attribute can be represented by a set of phone numbers while in another source it is represented by an e-mail address. Second, information can be incomplete, i.e. the values of some attributes can be missing. Third, data descriptions can contain syntactic errors which are specially frequent when they are automatically extracted from the Web. Fourth, as data descriptions can be created and updated independently in different sources, their freshness over sources is a real issue which can lead to have apparently different descriptions representing the same real world entity.

Data cleaning which aims at detecting duplicates in databases is faced with the same problems. Most of the existing works (e.g., [10,11,12]) perform comparisons between strings for computing the similarity between the values of the same attribute, and then combine them for computing the similarity between tuples. In [5] the matching between data descriptions is generic but is still based on local comparisons. Some recent works [13,1,14,6] follow a global approach that exploits the dependencies possibly existing between reference reconciliations. Those dependencies often result from the semantics of the domain of interest. For example, the reconciliation between two courses described by their titles and the name of the professors in charge of them can entail the reconciliation between two descriptions of persons. This requires some knowledge of the domain to be made explicit, like the fact that a professor is a person, that a course is identified by its title and has only one professor in charge of it. In [1], such knowledge is taken into account but must be encoded in the weights of the edges of the dependency graph.

In this paper, we study the problem of reference reconciliation in the case where the data are described relatively to a rich schema expressed in RDFS [15] extended by some primitives of OWL-DL [16] and SWRL [17]. OWL-DL and SWRL are used to enrich the semantics of the classes and properties declared in RDFS. This enriched data model that we have called RDFS+ enables to express that two classes are disjoint or that some properties (or their inverse) are functional. Note that relational data and schema can be easily mapped into RDFS [18,19] and their constraints translated in RDFS+.

For the reference reconciliation problem we propose a knowledge-based and unsupervised approach, based on two methods, a logical one called L2R and a numerical one called N2R. The logical method for reference reconciliation (L2R) is based on the translation in first order logic Horn rules of some of the schema semantics. These Horn rules enable to infer both exact reconciliations and non-reconciliations among a subset of reference pairs. Rules inferring synonymies and non synonymies between basic values are also generated from the schema constraints. Since L2R is based on logical inferences, it has a 100% precision, under the assumption that the schema and data are error-free. Such an assumption is not necessarily satisfied when the data is "dirty" or the global schema is an integrated schema resulting from an automatic reconciliation process. In order to complement the partial results of L2R, we have designed a Numerical method for Reference Reconciliation (N2R). It exploits the L2R result and allows to compute similarity scores for each pair of references. The distinguishing features of N2R compared to existing numerical methods of reference reconciliation are (i) it is unsupervised and (ii) the similarity computation takes into account some of the schema semantics : the functional dependencies of the properties are captured by aggregating the similarities of the involved references and values using the maximum function. Consequently, the mutual influences between similarity scores are expressed in a non linear equation system. In order to solve it, we use an iterative method inspired from *Jacobi* method [20] for which we have proved the convergence.

In the two methods the (non) reconciliation decisions or the similarity scores are propagated to other reference pairs. Therefore, L2R and N2R are two global methods, which can be applied separately or in combination. They are based on the most recent proposals for the Semantic Web (RDF, OWL-DL and SWRL). They can be used for reconciling data in most of the applications based on the Semantic Web technologies. The experimentations done on two different data sets (scientific publication domain and tourism domain) show good results for both precision and recall. Furthermore, the recall is significantly increased if the schema is enriched by adding constraints.

The paper is organized as follows. In Section 2, we define the data model, and in Section 3, the problem of reference reconciliation that we consider. Then, in Section 4, we describe the logical method implemented in L2R, and in Section 5, the numerical method implemented in N2R. In Section 6, we present the results that we have obtained for the experimental evaluation of L2R an N2R on two real data sets. We summarize the related work in Section 7. Finally, we conclude and sketch some future work in Section 8.

2 The RDFS$^+$ Data Model

We describe the data model, that we have called RDFS$^+$ because it extends RDFS with some OWL-DL primitives and SWRL rules. RDFS$^+$ can be viewed as a fragment of the relational model (restricted to unary or binary relations) enriched with typing constraints, inclusion and exclusion between relations and functional dependencies.

2.1 Schema Representation and Its Constraints

A RDFS schema consists of a set of classes (unary relations) organized in a taxonomy and a set of typed properties (binary relations). These properties can also be organized in a taxonomy of properties. Two kinds of properties can be distinguished in RDFS: the so-called *relations*, the domain and the range of which are classes and the so-called *attributes*, the domain of which is a class and the range of which is a set of basic values (e.g. Integer, date, String). This distinction is also made in OWL which allows to declare *Object* and *datatype* properties. For example, in the RDFS schema presented in figure 1 and corresponding to a cultural application, we have as relation *Located* having as domain the class *Museum* and as range the class *City*. We also have an attribute*MuseumName* having as domain the class *Museum* and as range the data type *Literal*.

Note that, relations and attributes can be renamed in order to ensure that every attribute and every relation has only one domain and one range.

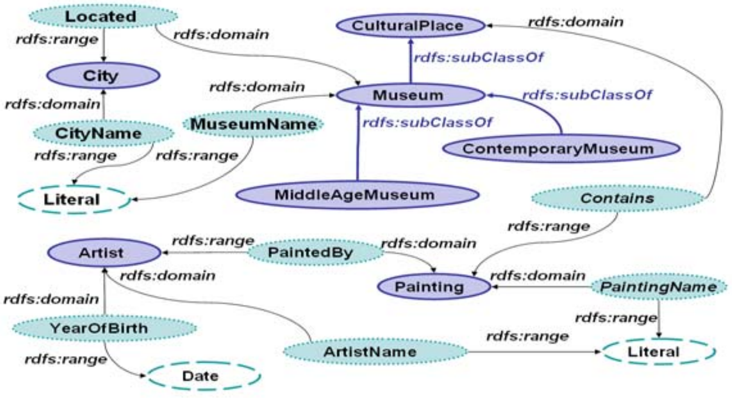


Fig. 1. Example of a RDFS schema

This schema could be extracted from the following relational schema :

CulturalPlace(<u>IDCulturalPlace</u>), Museum(<u>IDCulturalPlace#</u>, MuseumName, IDCity#, Category), Painting(<u>IDPainting</u>, PaintingName, IdArtist#, IDCulturalPlace#), Artist(<u>IDArtist</u>, ArtistName, YearOfBirth), City(<u>IDCity</u>, CityName).

We allow the declaration of constraints expressed in OWL-DL or in SWRL in order to enrich the RDFS schema. The constraints that we consider are of the following types.

- **Constraints of disjunction between classes** $DISJOINT(C, D)$ is used to declare that the two classes C and D are disjoint, for example: $DISJOINT$ $(CulturalPlace, Artist)$.
- **Constraints of functionality of properties** $PF(P)$ is used to declare that the property P (relation or attribute) is a functional property. It is similar to functional dependencies in relational databases [21]. For example, $PF(Located)$ and $PF(MuseumName)$ express respectively that a museum is located in one and only one city and that a museum has only one name. These constraints can be generalized to a set $\{P_1, \ldots, P_n\}$ of relations or attributes to state a combined constraint of functionality that we will denote $PF(P_1, \ldots, P_n)$. It means that for a set of n references which instantiate the domains of $P_1, \ldots, P_n$ there is only one reference or one value of their ranges. We note that in the RDFS schema (cf. Figure 1) all the attributes and all the relations are functional, except, the relation $Contains$.
- **Constraints of inverse functionality of properties** $PFI(P)$ is used to declare that the property P (relation or attribute) is an inverse functional property. For example, $PFI(Contains)$ expresses that a painting cannot belong to several cultural places. These constraints can be generalized to a set $\{P_1, \ldots, P_n\}$ of relations or attributes to state a combined constraint of inverse functionality that we will denote $PFI(P_1, \ldots, P_n)$. For example, $PFI(MuseumAddress, MuseumName)$ expresses that one address and one museum name cannot be associated to several museums (i.e. both are needed to identify a museum). In summary, the set of inverse functional properties of the RDFS schema (cf. Figure 1) are : $\{PFI(MuseumAddress, MuseumName)$, $PFI(PaintingName)$, $PFI(Contains)$, $PFI(ArtistName)$, $PFI(CityName)\}$.

It is important to note that the constraints of disjunction and of simple functionality (i.e., of the form $PF(P)$ or $PFI(P)$) can be expressed in OWL-DL while the constraints stating combined constraints of functionality (i.e., of the form $PF(P_1, \ldots, P_n)$ or $PFI(P_1, \ldots, P_n)$) require the expressive power of SWRL.

2.2 Data Description and Their Constraints

A datum has a reference which has the form of a URI (e.g. `http://www.louvre.fr`, `NS-S1/painting243`), and a description which is a set of RDF facts involving its reference. An RDF fact can be:

- either a class-fact $C(i)$, where C is a class and i is a reference,
- or a relation-fact $R(i_1, i_2)$, where R is a relation and i_1 and i_2 are references,
- or an attribute-fact $A(i, v)$, where A is an attribute, i a reference and v a basic value (e.g. integer, string, date).

The data description that we consider is composed of the RDF facts coming from the data sources enriched by applying the RDFS entailment rules [22]. We consider that the descriptions of data coming from different sources conform to the same RDFS$^+$ schema (possibly after schema reconciliation). In order to distinguish the data coming from different sources, we use the source identifier as the prefix of the reference of the data coming from that source. For example, Figure 2 provides examples of data coming from two RDF data sources S1 and S2 which conform to a same RDFS$^+$ schema describing the cultural application previously mentioned.

Source S1 :
MuseumName(S1_m1, *"LE LOUVRE"*); Contains(S1_m1,S1_p1); Located(S1_m1,S1_c1); CityName(S1_c1,"Paris"); PaintingName(S1_p1, *"La Joconde"*);

Source S2 :
MuseumName(S2_m1, *"musee du LOUVRE"*); Located(S2_m1,S2_c1); Contains(S2_m1,S2_p1); Contains(S2_m1, S2_p2); CityName(S2_c1, *"Ville de paris"*); PaintingName(S2_p1, *"Abricotiers en fleurs"*); PaintingName(S2_p2, *"Joconde'*);

Fig. 2. Example of RDF data of cultural places domain

We consider two kinds of constraints accounting for the Unique Name Assumption (UNA). The UNA states that two data of the same data source having distinct references refer to two different real world entities (and thus cannot be reconciled). Such an assumption is valid when a data source is clean. We have defined another kind of constraint called the Local Unique Name Assumption (denoted LUNA). The LUNA is weaker than the UNA, and states that all the references related to a same reference by relation refer to real world entities that are pairwise distinct. For example, from the facts $Authored(p, a_1)$, ..., $Authored(p, a_n)$ coming from the same data source, we can infer that the references $a_1, \ldots, a_n$ correspond to distinct authors of the paper referred to by p. In practice, it is often the case that all the values of a multi-valued property of a given instance, are provided as a group coming from a single source, like for instance the authors of a given paper, or the list of paintings in a given museum. It is therefore realistic to suppose that this group of values does not contain any duplicate. It is what LUNA means.

3 The Reference Reconciliation Problem

Let S_1 and S_2 be two data sources which conform to the same RDFS$^+$ schema. Let I_1 and I_2 be the two reference sets that correspond respectively to the data of S_1 and S_2. The problem consists in deciding whether references are reconciled or not reconciled.

A method of reference reconciliation is said complete if it provides a decision for each reference pair $(i_1, i_2) \in I_1 \times I_2$. It is numerical if the decision is based on similarity scores. It will be said symbolic if the yes/no answers for the reconciliation between reference pairs is based on symbolic inferences.

L2R is symbolic but incomplete, while N2R is complete but numerical.

Let *Reconcile*[1] be a binary predicate. *Reconcile*(X, Y) means that the two references denoted by X and Y refer to the same world entity.

The reference reconciliation problem considered in L2R consists in extracting from the set $I_1 \times I_2$ of reference pairs two subsets REC and NREC such that:

$$\begin{cases} REC = \{(i, i')|\ Reconcile(i, i')\} \\ NREC = \{(i, i')|\ \neg Reconcile(i, i')\} \end{cases}$$

The reference reconciliation problem considered in N2R consists in, given a similarity function $Sim_r : I_1 \times I_2 \longrightarrow [0..1]$, and a threshold T_{rec} (a real value in [0..1] given by an un expert, fixed experimentally or learned on a labeled data sample), computing the following set:

$$REC_{n2r} = \{(i, i') \in (I_1 \times I_2) \backslash (REC \cup NREC)|\ Sim_r(i, i') > T_{rec}\}$$

In order to evaluate the quality of the results of a reference reconciliation method, well-known measures in the Information Retrieval (IR) domain can be employed. It consists essentially in *Precision*, *Recall* and *F-Measure* defined as follows:

- *Precision* of a reconciliation method is the ratio of correct reconciliations and non-reconciliations among those found by the method.
- *Recall* of a reconciliation method is the ratio of correct reconciliations and non-reconciliations found by the method among the whole expected set of correct reconciliations and non-reconciliations.
- *F-Measure* of a reconciliation method is computed to balance the recall and precision values : $F - Measure = (2 * Recall * Precision) \div (Recall + Precision)$.

4 L2R: A Logical Method for Reference Reconciliation

In this section we present an extended version of the L2R method presented in [23]. The method is based on the inference of facts of reconciliation ($Reconcile(i, j)$) and of non reconciliation ($\neg Reconcile(i', j')$) from a set of facts and a set of rules which transpose the semantics of the data sources and of the schema into logical dependencies between reference reconciliations. Facts of synonymy ($SynVals(v_1, v_2)$) and of no synonymy ($\neg SynVals(u_1, u_2)$) between basic values (strings, dates) are also inferred. For instance, the synonymy $SynVals(\text{“}JoDS'', \text{“}Journal\ of\ Data\ Semantics'')$ may be inferred. This binary predicate is analogous to the predicate *Reconcile* but applied on basic values.

The L2R distinguishing features are that it is global and logic-based: every constraint declared on the data and on the schema in RDFS+ is automatically translated into first-order logic Horn rules (rules for short) that express dependencies between reconciliations. The advantage of such a logical approach is that

[1] Reconcile and not Reconcile can also be expressed in OWL by using *sameAs* and *differentFrom* predicates.

if the data are error-free and if the declared constraints are valid, then the reconciliations and non reconciliations that are inferred are correct, thus guaranteeing a 100 % precision of the results.

We first describe the generation of the reconciliation rules. Then we present the generation of the facts and finally the reasoning which is performed on the set of rules and facts to infer reconciliation decisions.

4.1 Generation of the Set of Reconciliation Rules

They are automatically generated from the constraints that are declared on the data sources and on their common schema. We omit to write the quantifiers applied to variables because all the variables are universally quantified in the scope of each rule. According to standard first-order logic conventions, variables will be denoted by lower case letters, and predicate names will start by a capital letter.

Translation of the Constraints on the Data Sources. We introduce the unary predicates Src1 and Src2 in order to label each reference according to its original source ($Srci(x)$ means that the reference x is coming from the source Si).

The UNA assumption, if it is stated on the sources $S1$ and $S2$, is translated automatically by the following four rules:

$$R1 : Src1(x) \wedge Src1(y) \wedge (x \neq y) \Rightarrow \neg Reconcile(x, y)$$

$$R2 : Src2(x) \wedge Src2(y) \wedge (x \neq y) \Rightarrow \neg Reconcile(x, y)$$

$$R3 : Src1(x) \wedge Src1(z) \wedge Src2(y) \wedge Reconcile(x, y) \wedge (x \neq y) \Rightarrow \neg Reconcile(z, y)$$

$$R4 : Src1(x) \wedge Src2(y) \wedge Src2(z) \wedge Reconcile(x, y) \wedge (x \neq y) \Rightarrow \neg Reconcile(x, z)$$

The first two rules express the fact that two distinct references coming from the same source cannot be reconciled. The last ones mean that one reference coming from a source $S2$ (resp. $S1$) can be reconciled with at most one reference coming from a source $S1$(resp. $S2$).
For each relation R, the LUNA assumption is translated automatically by the following rules denoted respectively $R11(R)$ and $R12(R)$:

$$R11(R) : R(z, x) \wedge R(z, y) \wedge (x \neq y) \Rightarrow \neg Reconcile(x, y)$$

$$R12(R) : R(x, z) \wedge R(y, z) \wedge (x \neq y) \Rightarrow \neg Reconcile(x, y)$$

For example, if the LUNA is declared, the two following two rules are generated for the relation *Authored* relating references to papers to references to persons: they express that there is no duplicates in the set of authors of a given paper (respectively in the set of papers of a given author) and that there is no duplicates in the set of papers of a given author.

$$R11(Authored) : Authored(z, x) \wedge Authored(z, y) \wedge (x \neq y) \Rightarrow \neg Reconcile(x, y)$$

$$R12(Authored) : Authored(x, z) \wedge Authored(y, z) \wedge (x \neq y) \Rightarrow \neg Reconcile(x, y)$$

Translation of the Schema Constraints. For each pair of classes C and D involved in a $DISJOINT(C, D)$ statement declared in the schema, or such that their disjunction is inferred by inheritance, the following rule is generated:

$$R5(C,D) : \ C(x) \ \wedge \ D(y) \ \Rightarrow \ \neg Reconcile(x,y)$$

For example, the following rule is generated if the classes *Painting* and *Artist* are declared disjoint:

$$R5(Painting, Artist) : \ Painting(x) \ \wedge \ Artist(y) \ \Rightarrow \ \neg Reconcile(x,y)$$

For each relation R declared as functional by the axiom $PF(R)$, the following rule $R6.1(R)$ is generated :

$$R6.1(R) : Reconcile(x,y) \wedge R(x,z) \wedge R(y,w) \Rightarrow Reconcile(z,w)$$

For example, the following rule is generated concerning the relation *Located* which relates references of museums to references of cities and which is declared functional:

$$R6.1(Located) : Reconcile(x,y) \wedge Located(x,z) \wedge Located(y,w) \Rightarrow Reconcile(z,w)$$

For each attribute A declared as functional by the axiom $PF(A)$, the following rule $R6.2(A)$ is generated :

$$R6.2(A) : Reconcile(x,y) \wedge A(x,z) \wedge A(y,w) \Rightarrow SynVals(z,w)$$

The binary predicate *SynVals* replaces the predicate *Reconcile* in the conclusion of the rule. For example, the following rule is generated concerning the attribute $MuseumName$ which relates references of museums to their name and which is declared functional:

$$R6.2(MuseumName) : Reconcile(x,y) \wedge MuseumName(x,z) \wedge MuseumName(y,w)$$
$$\Rightarrow SynVals(z,w)$$

For each relation R declared as inverse functional by the constraint $PFI(R)$, the following rule $R7.1(R)$ is generated:

$$R7.1(R) : Reconcile(x,y) \wedge R(z,x) \wedge R(w,y) \Rightarrow Reconcile(z,w)$$

For each attribute A declared as inverse functional by the constraint $PFI(A)$, the following rule $R7.2(A)$ is generated:

$$R7.2(A) : SynVals(x,y) \wedge A(z,x) \wedge A(w,y) \Rightarrow Reconcile(z,w)$$

Likewise, analogous rules are generated for translating constraints $PF(P_1, \ldots, P_n)$ of combined constraints of functionality and $PFI(P_1, \ldots, P_n)$ of combined constraints of inverse functionality. $PF(P_1, \ldots, P_n)$, where all the P_i's are relations, is translated by the following rule:

$$R7.1(P_1, \ldots, P_n) : \bigwedge_{i \in [1..n]} [P_i(z,x_i) \wedge P_i(w,y_i) \wedge Reconcile(x_i,y_i)] \Rightarrow Reconcile(z,w)$$

If some P_i's are attributes, the corresponding $Reconcile(x_i, y_i)$ must be replaced by $SynVals(x_i, y_i)$. For example, the declaration $PF(PaintedBy, PaintingName)$ states a composed functional dependency which expresses that the artist who painted it jointly with its name functionally determines a painting. It is translated in the following rule:

$R7.1(PaintedBy, PaintingName)$:
$PaintedBy(z, x_1) \wedge PaintedBy(w, y_1) \wedge Reconcile(x_1, y_1)$
$\wedge PaintingName(z, x_2) \wedge PaintingName(w, y_2) \wedge SynVals(x_2, y_2) \Rightarrow Reconcile(z, w)$

Similarly, $PFI(P_1, \ldots, P_n)$, where all the P_i's are relations, is translated into the rule:

$$R7.2(P_1, \ldots, P_n) : \bigwedge_{i \in [1..n]} [P_i(x_i, z) \wedge P_i(y_i, w) \wedge Reconcile(x_i, y_i)] \Rightarrow Reconcile(z, w)$$

Transitivity Rule. It allows inferring new reconciliation decisions by applying transitivity on the set of already inferred reconciliations. Its logical semantics is:

$$R8 : Reconcile(x, y) \wedge Reconcile(y, z) \Rightarrow Reconcile(x, z)$$

We note that this rule is generated only if the UNA constraint is not stated on the data sources. Indeed, when the UNA is stated, a reference can not be reconciled with more than one reference. Therefore, the transitivity rule is not meaningful in this setting. We do not generate a rule for expressing transitivity between synonymies values, since, according to Fischer [24], the synonymy between basic values is not transitive because of polysemy.

4.2 Reasoning Method for Reference Reconciliation

In order to infer sure reconciliation and non-reconciliation decisions, we apply an automatic reasoning method based on the resolution principle [25]. This method applies to the clausal form of the set of rules described in Section 4.1 and a set of facts describing the data which is generated as follows.

Generation of the Set of Facts. The set of RDF facts corresponding to the description of the data in the two sources S_1 and S_2 is augmented with the generation of:

– new class-facts, relation-facts and attribute-facts derived from the domain and range constraints that are declared in RDFS for properties, and from the subsumption statements between classes and properties that are stated in RDFS. For example if the fact *ContemporaryMuseum(i)* is present in one of the sources, the class-facts $Museum(i)$ and $CulturalPlace(i)$ are added to the description of that source;
– facts of the form $Src1(i)$ and $Src2(j)$ for each reference $i \in I_1$ and each reference $j \in I_2$,

– synonymy facts of the form $SynVals(v_1, v_2)$ for each pair (v_1, v_2) of basic values that are identical (up to some punctuation or case variations): for instance, the fact *SynVals("La Joconde", "la joconde")* is added because these two values differ only by two capital letters,
– non synonymy facts of the form $\neg SynVals(v_1, v_2)$ for each pair (v_1, v_2) of distinct basic values of a functional attribute (PF) for which it is known that each possible value of this attribute has a single form . For instance, $\neg SynVals(''2004'', ''2001'')$, $\neg SynVals(''France'', ''Algeria'')$ are added.

Resolution-Based Algorithm for Reference Reconciliation. The reasoning is applied to $\mathcal{R} \cup \mathcal{F}$: the set of rules (put in clausal form) and the set of facts generated as explained before. It aims at inferring all unit facts in the form of $Reconcile(i, j)$, $\neg Reconcile(i, j)$, $SynVals(v1, v2)$ and $\neg SynVals(v1, v2)$.

The resolution is a reasoning method for theorem proving by a successive application of the *resolution rule*[26] on the set of clauses. Several resolution strategies have been proposed so that the number of computed resolutions to obtain the theorem proof are reduced (for more details about these strategies see [26]). We have chosen to use the *unit resolution*[27], defined as follows:

Definition 1. *Unit resolution: it is a resolution strategy where at least one of the two clauses involved in the resolution is a unit clause, i.e. reduced to a single literal.*

The unit resolution is complete for refutation[2] in the case of Horn clauses without functions [27]. Furthermore, the unit resolution method is linear with respect to the size of clause set [28].

The Conjunctive Normal Form (CNF) of the knowledge base $\mathcal{R} \cup \mathcal{F}$ is made of Horn clauses and contains a lot of unit clauses. It is important to notice that the reconciliation and synonymy rules though having negative conclusions still correspond to Horn clauses. For all these reasons, unit resolution is a method which is appropriate for our problem.

The unit resolution algorithm that we have implemented consists in computing the set $SatUnit(\mathcal{R} \cup \mathcal{F})$ of unit instantiated clauses contained in $\mathcal{F}$ or inferred by unit resolution on $\mathcal{R} \cup \mathcal{F}$. Its termination is guaranteed because there are no function symbols in $\mathcal{R} \cup \mathcal{F}$. Its completeness for deriving all the facts that are logically entailed is stated in the following theorem. Because of the limited form of inequalities appearing in the rules (and thus in the clauses) we avoid to use paramodulation in combination with resolution by a preprocessing step on $\mathcal{R} \cup \mathcal{F}$. This consists in generating all the propositional rules that can be obtained from $\mathcal{R}$ by matching their conditions to facts in $\mathcal{F}$ with substitutions satisfying the inequality statements. This results into a set of clauses (a subset of which being totally instantiated) without inequalities. A simple optimization is also implemented that prunes all the unit clauses of the form $Reconcile(i, i)$ that may be generated: only unit clauses of the form $Reconcile(i, j)$ such that $i \neq j$ are then used as resolvents.

[2] Proving by refutation that a literal L is logically entailed from a theory T is viewed as the unsatisfiability of the theory $T \cup \{\neg L\}$, i.e. deduce the empty clause ($\Box$).

Theorem 1. – *Completeness of unit resolution for deriving facts from function-free Horn clauses.*

Let $\mathcal{R}$ be a set of Horn clauses without functions. Let $\mathcal{F}$ be a set of ground unit clauses. If $\mathcal{R} \cup \mathcal{F}$ is satisfiable then:

$$\forall p(\boldsymbol{a}),\ (\mathcal{R} \cup \mathcal{F} \models p(\boldsymbol{a})) \Rightarrow (p(\boldsymbol{a}) \in SatUnit(\mathcal{R} \cup \mathcal{F})),$$

where, $p(\boldsymbol{a})$ is a ground unit clause.

The theorem relies on the assumption that $\mathcal{R} \cup \mathcal{F}$ is satisfiable. $\mathcal{R} \cup \mathcal{F}$ is the logical transposition of the constraints that are declared on the schema and the data. Therefore, if those constraints are correct and if the data are error-free then $\mathcal{R} \cup \mathcal{F}$ is satisfiable. In any case, our resolution-based algorithm will detect if $\mathcal{R} \cup \mathcal{F}$ is unsatisfiable. A by-product of our logical approach is to detect whether the set of declared constraints on the schema and on the data sources is contradictory. If that it the case, this set of constraints must be revised by the database administrator in charge of the data integration.

Other reasoners, like for instance description logic reasoners, could be used for the derivation of reconciliation facts. However, description logics are not specially appropriate to express some of the reconciliation rules that we consider, which require explicit variable bindings. In addition, up to our knowledge, the existing description logic reasoners are not guaranteed to be complete for the computation of prime implicates.

Illustrative Example. We now illustrate on the example of data and RDFS$^+$ schema, given in Section 2, the resolution-based reasoning for the reference reconciliation. We will also show how reconciliation and non-reconciliation decisions are inferred and chained.

The successive application of the unit resolution on the knowledge base $\mathcal{R} \cup \mathcal{F}$, presented in the Figure 3, allows inferring the set of (non) reconciliation and synonymy facts presented in Figure 3.

The clauses R1 and R2 allow inferring a set of non-reconciliations between all the references coming from the same source, e.g. $\neg Reconcile(S1\_m1, S1\_c2)$. The clauses $R5$ translating the disjunction between classes, allow inferring non-reconciliations between references which instantiate disjoint classes, (e.g. $\neg$ $Reconcile$ $(S1\_m1, S2\_p1)$) is obtained thanks to the clause $R5(Painting, Museum)$. Furthermore, the successive application of the unit resolution between the unit clauses contained in $\mathcal{F}$ and the clause $R7.2(PaintingName)$ allows inferring $Reconcile(S2\_p1, S1\_p1)$, which means that the two paintings $S2\_p1$ and $S1\_p1$ refer to the same painting. Then, the museums $S1\_m1$ and $S2\_m2$ which contain these paintings are reconciled as well, thanks to the clause $R7.1(Contains)$. The propagation of the new reconciliation facts allows inferring the synonymie fact $SynVals(''Le\ LOUVRE'', ''musee\ du\ LOUVRE'')$ thanks to the clause $R6.2(MuseumName)$. Finally, thank to the clause $R6.1(Located)$ we entail the reconciliation $Reconcile$ $(S1\_c1, S2\_c1)$ of the two cities. Therefore, this new reconciliation leads to the entailment of the synonymy $SynVals(''ville\ de\ Paris'', ''Paris'')$ thanks to the clause $R6.2(CityName)$.

Knowledge base
$\mathcal{R}$= **{** $R1 : \neg\, Src1(x) \vee \neg\, Src1(y) \vee \neg Reconcile(x, y)$
$R2 : \neg\, Src2(x) \vee \neg\, Src2(y) \vee \neg Reconcile(x, y)$ ...
$R5(Painting, City) : \neg\, Painting(x) \vee \neg\, City(y) \vee \neg Reconcile(x, y)$
$R5(Painting, Museum) : \neg\, Painting(x) \vee \neg\, Museum(y) \vee \neg Reconcile(x, y)$
$R5(City, Museum) : \neg\, City(x) \vee \neg\, Museum(y) \vee \neg Reconcile(x, y)$ ...
$R6.1(Located) : \neg\, Reconcile(x, y) \vee \neg\, Located(x, z) \vee \neg\, Located(y, w) \vee Reconcile(z, w)$
$R6.2(MuseumName) : \neg\, Reconcile(x, y) \vee\neg\, MuseumName(x, z)$
$\vee\neg\, MuseumName(y, w) \vee SynVals(z, w)$
$R6.2(CityName) : \neg\, Reconcile(x, y) \vee \neg\, CityName(x, z) \vee \neg\, CityName(y, w) \vee SynVals(z, w)$ ...

$R7.2(PaintingName) : \neg SynVals(x, y) \vee \neg\, PaintingName(z, x) \vee$
$\neg\, PaintingName(w, y) \vee Reconcile(z, w)$
$R7.1(Contains) : \neg Reconcile(x, y) \vee \neg\, Contains(z, x) \vee \neg\, Contains(w, y) \vee Reconcile(z, w)$ ...**}**
$\mathcal{F}$= **{** $MuseumName(S1\_m1,$ "LE LOUVRE"$)$; $Contains(S1\_m1, S1\_p1)$;
$Contains(S2\_m1, S2\_p2)$; $CityName(S2\_c1,$ "Ville de paris"$)$; ...
$Src1(S1\_m1)$; $Src1(S1\_p1)$; $Src1(S1\_c1)$; $Src2(S2\_m1)$; $Src2(S2\_p1)$; $Src2(S2\_c1)$
$SynVals($"La Joconde", "Joconde"$)$**}**

Reference reconciliation result
SatUnit$(\mathcal{R} \cup \mathcal{F}) = \{...;$
$\neg Reconcile(S1\_m1, S1\_c2)$; $\neg Reconcile(S1\_m1, S1\_p1)$; $\neg Reconcile(S1\_p1, S1\_c1)$;
$\neg Reconcile(S2\_m1, S2\_p1)$; $\neg Reconcile(S2\_m1, S2\_c1)$; $\neg Reconcile(S2\_c1, S2\_p1)$;
$\neg Reconcile(S1\_m1, S2\_p1)$; $\neg Reconcile(S1\_m1, S2\_c1)$; $\neg Reconcile(S1\_p1, S2\_c1)$;
$\neg Reconcile(S1\_c1, S2\_p1)$;
$Reconcile(S2\_p1, S1\_p1)$; $Reconcile(S1\_m1, S2\_m1)$; $Reconcile(S1\_c1, S2\_c1)$;
$SynVals($"musee du LOUVRE", "LE LOUVRE"$)$; $SynVals($"ville de Paris", "Paris"$)\}$

Fig. 3. Illustrative exemple of unit resolution-based reference reconciliation

4.3 Dictionary of Synonyms and No Synonyms

The set of synonymies and no synonymies between basic values inferred by the the reference reconciliation algorithm are saved in two dictionaries. The dictionaries can also be exploited by a numerical method for reference reconciliation based on similarities between strings. We will see (see Section 5) how these (no) synonymies can be used by N2R method.

We distinguish different kinds of synonyms: (i) Codes, like *1* for *yes 75* for *Paris* and *(*)* for *star*. (ii) Abbreviations, like *apt* for *appartement*, or acronyms like *ACM* for *Association for Computing Machinery*. (iii) Real synonyms, like *good* for *comfortable*. (iv) Translations, like *Royaume-Uni* for *United Kingdom* .

The (no) synonymies can be viewed as knowledge learnt in an automatic and a unsupervised way. Indeed, this allows our method to capitalize its experience by learning more and more on the syntactic variations that characterize an application domain.

5 N2R: A Numerical Method for Reference Reconciliation

In this section we describe the numerical method for reference reconciliation (N2R) that we have designed and implemented. Like existing numerical methods (e.g., [1,6]), it computes a similarity score for each pair of references. However, N2R has two main distinguishing characteristics. First, it is fully unsupervised: in contrast with the existing methods, it does not require any training phase from manually labeled data to set up coefficients or parameters. Second, it is based on equations that model the influence between similarities. In the equations, each *variable* represents the (unknown) similarity between two references while the similarities between values of attributes are *constants* that are computed by using standard similarity measures on strings or on sets of strings. The *functions* modeling the influence between similarities are a combination of *maximum* and *average* functions in order to take into account the constraints of functionality and inverse functionality declared in the RFDS$^+$ schema in an appropriate way.

Solving this equation system is done by an iterative method inspired from the *Jacobi* method [20], which is fast converging on linear equation systems. The point is that the equation system that results for modeling the global influence of similarities is not linear, due to the use of the *max* function for the numerical translation of the functionality and inverse functionality axioms declared in the RFDS$^+$ schema. Therefore, we had to prove the convergence of the iterative method for solving the resulting non linear equation system.

N2R can be applied alone or in combination with L2R. In this case, the results of non-reconciliation inferred by L2R are exploited for reducing the reconciliation space, i.e., the size of the equation system to be solved by N2R. In addition, the results of reconciliations and of synonymies or non synonymies inferred by L2R are used to set the values of the corresponding constants or variables in the equations.

We first use a simple example to illustrate how the equation system is built from the data descriptions related to the references to reconcile. We then introduce the notations and the similarity measures that are used in order to distinguish the different types of constants and functions involved in the equations. Finally, we describe in the general case how the equation system is built from the data descriptions, and we provide the iterative method for solving it.

5.1 Illustrative Example

Let us consider the data descriptions of Figure 2. They conform to the RFDS$^+$ schema given in the Figure 1, the constraints of which are described in Section 2.1. Let us assume that the UNA is stated in both the sources S1 and S2.

Let us suppose that L2R has been applied, resulting on the non-reconcilations of all the pairs of references coming from the same source and those belonging to two disjoint classes. The only remaining pairs of references to consider for N2R are then:

$< S1\_m1, S2\_m1 >$, $< S1\_c1, S2\_c1 >$, $< S1\_p1, S2\_p1 >$ and $< S1\_p1, S2\_p2 >$.

The similarity score $Sim_r(ref, ref')$ between the references ref and ref' of each of those pairs is modeled by a variable:

- x_1 models $Sim_r(S1\_m1, S2\_m1)$,
- x_2 models $Sim_r(S1\_p1, S2\_p1)$,

– x_3 models $Sim_r(S1\_p1, S2\_p2)$,
– x_4 models $Sim_r(S1\_c1, S2\_c1)$.

We obtain the following equations that model the dependencies between those variables from the relations relating the corresponding references and the constraints declared on them in the schema:

$x_1 = max(0.68, x_2, x_3, x_4/4)$
$x_2 = max(0.1, x_1/2)$
$x_3 = max(0.9, x_1/2)$
$x_4 = max(0.42, x_1)$

The first equation expresses that the variable x_1:

– strongly and equally depends on the variables x_2 and x_3, and also on 0.68, which is the similarity score between the two strings “LE LOUVRE” and “musee du LOUVRE” computed by the Jaro-Winkler function [29],
– weakly depends on x_4.

The reason of the strong dependencies is that *Contains* is an inverse functional relation (a painting is contained in only one museum) relating $S1\_m1$ and $S2\_m1$ (the similarity of which is modeled by x_1) to $S1\_p1$ for $S1\_m1$ and $S2\_p1$ and $S2\_p2$ for $S2\_m1$, and *MuseumName* is a functional attribute (a museum has only one name) relating $S1\_m1$ and $S2\_m1$ respectively to the two strings “LE LOUVRE” and “musee du LOUVRE”.

The weak dependency of x_4 onto x_1 is expressed by the term $x_4/4$ in the equation, where the ratio 1/4 comes from that there are 4 properties (relations or attributes) involved in the data descriptions of $S1\_m1$ and $S2\_m1$. The dependency of x_4 onto x_1 is weaker than the previous ones because x_4 expresses the similarity between the two cities in which the museums modeled by x_1 are located and *Located* is not an inverse functional relation.

Conversely, because *Located* is functional, x_1 has a strong influence on the variable x_4, which translates into the second equation, where 0.42 is the Jaro-Winkler score of similarity between the strings “Paris” and “Ville de Paris” which are the respective values associated to the references $S1\_c1$ and $S2\_c1$ by the inverse functional attribute *CityName*.

The weak influence of x_1 on x_2 and x_3 is due to the fact that *Contains* is not functional. It is expressed through the two last equations, where 0.1 is the Jaro-Winkler score of similarity between the strings “*La Joconde*” and “*Abricotiers en fleurs*”, and 0.9 is the Jaro-Winkler score of similarity between the strings “La Joconde” and “*Joconde*”.

5.2 Notations and Similarity Measures on (Sets of) Basic Values

As illustrated in the previous example, the constants in the equations are similarities between basic values (e.g., String, Date, Numbers) or between sets of basic values, for which a lot of similarity measures have been extensively studied [29].

Similarity Measures on Basic Values. We denote Sim_v the similarity measure used to compute the similarity score between two basic values.

For pairs of basic values $< v_1, v_2 >$ such that $SynVal(v_1, v_2)$ (respectively $\neg SynVal(v_1, v_2)$) has been inferred by L2R , we set: $Sim_v(v_1, v_2) = 1$ (respectively $Sim_v(v_1, v_2) = 0$).

For computing the similarity scores between basic values that are not dealt with by L2R, depending on the attributes and the characteristics of their values (e.g short/long values, values containing abbreviations), we set Sim_v to the most appropriate similarity measure according to [29].

***$SSim_v$*: The Similarity Measure on Sets of Basic Values.** Some attributes are multi-valued (e.g. a person can have a set of phone numbers).

$SSim_v(S1, S2)$ denotes the similarity score between the two sets of basic values $S1$ and $S2$.

In order to compute the similarity between two sets of values we need to take into account their size and also the similarity scores of the pairs of values formed from these two sets.

We propose a similarity measure that we have named *SoftJaccard* which is inspired from *SoftTFIDF* measure defined by [29] and from the *Jaccard* measure. *SoftJaccard* allows relaxing the constraint of equality of the tokens used in *Jaccard*. $CLOSE_v(S1, S2, \theta)$ represents the values $(v1, v2) \in S1 \times S2$ that have a similarity score $Sim_v(v1, v2) > \theta$. *SoftJaccard* is defined by the expression:

$$SoftJaccard(S1, S2, \theta) = \frac{|CLOSE_v(S1, S2, \theta)|1}{|S1|},\ with\ |S1| \geq |S2|$$

Example 1. $SoftJaccard(\{"Fatiha\ \ Sais", "Marie - Christine\ \ Rousset", "Helene\ Gagliardi"\}, \{"Nathalie\ Pernelle", "Fatiha\ Sais"\}), 0.7) = 1/3.$

Common Attributes and Relations: Definitions and Notations. Given a pair of references $< i, i' >$, we will denote:

- $CAttr(< i, i' >)$ the set of its *common attributes*: A is a common attribute to $< i, i' >$ if there exists atleast a fact $A(i, v)$ in the data description of i and also atleast a fact $A(i', v')$ in the data description of i'.
- $CRel(< i, i' >)$ the set of its *common relations*: R is a common relation to $< i, i' >$ if there exists atleast a fact $R(i, r)$ in the data description of i and also atleast a fact $R(i', r')$ in the data description of i'.

Example 2. In Figure 2, we have: $CAttr(<S1\_m1, S2\_m1>) = \{MuseumName\}$
$CRel(< S1\_m1, S2\_m1 >) = \{Located, Contains\}$

Among $CAttr(< i, i' >)$ and $CRel(< i, i' >)$ we need to distinguish those which are (inverse) functional from those which are not:

- $FD_A(< i, i' >)$ denotes the set of common attributes of the reference pair $< i, i' >$ that are inverse functional.

– $FD_R(< i, i' >)$ denotes the set of common relations of the reference pair $< i, i' >$ that are functional or inverse functional.

Example 3. $FD_A(< S1\_p1, S2\_p1 >) = \{PaintingName\}$
$FD_R(< S1\_c1, S2\_c1 >) = \{Located\}$

These sets are generalized by considering the generalized constraints of (inverse) functionality which involve sets of attributes and relations. We note these sets $FD_A^M(< i, i' >)$ and $FD_R^M(< i, i' >)$. Among those attributes and relations that are not functional or inverse functional, we distinguish those for which i and i' are mono-valued from those for which i or i' are multi-valued:

– $NFD_A(< i, i' >)$ denotes the set of common attributes of $< i, i' >$ that are not inverse functional but that are mono-valued for i and for i'.
– $NFD_A^*(< i, i' >)$ denotes the set of common attributes of $< i, i' >$ that are not inverse functional and that are multi-valued for i or for i'.
– $NFD_R(< i, i' >)$ denotes the set of common relations of $< i, i' >$ that are not (inverse) functional but that are mono-valued for i and for i'.
– $NFD_R^*(< i, i' >)$ denotes the set of common relations of $< i, i' >$ that are not (inverse) functional and that are multi-valued for i or for i'.

5.3 The Equations Modeling the Dependencies between Similarities

The Variables in the Equation System. For each pair of references, its similarity score is modeled by a variable x_i and the way it depends on other similarity scores is modeled by an equation: $x_i = f_i(X)$, where n is the number of reference pairs for which we apply N2R, and $X = (x_1, x_2, \ldots, x_n)$.

When a reference pair $< ref_h, ref'_h >$, represented by a similarity score x_j, is involved in the similarity equation of another reference pair $< ref, ref' >$ corresponding to the variable x_i, a contextual suffix i is added to the variable x_j. In addition, we denote by three secondary suffixes the three types of functional dependencies that we distinguish : *df* for a functional dependency, *dfm* for a multiple functional dependency, and *ndf* for non functional dependency. Thus, for the i-th reference pair $< ref, ref' >$ we distinguish the following variables x_{ij-df}, x_{ij-dfm} and x_{ij-ndf} they depend on.

In addition, we define a variable XS_{ij-ndf} in order to express the similarity score of the reference sets S1 and S2 of the j-th common relation of the i-th reference pair $< ref, ref' >$. This variable is defined only when the relation belongs to $NFD_R^*(< ref, ref' >)$. Its value is obtained by the function $SSim_r(S1, S2)$[3].

The Constants in the Equation System. They represent the similarity score of basic values. For each pair of values (v, v') of the j-th common attribute of the i-th reference pair, a constant b_{ij-df}, b_{ij-dfm} or b_{ij-ndf} is assigned. These constants represent the similarity score obtained by the function $Sim_v(v, v')$.

In addition, we define a constant BS_{ij-ndf} in order to express the similarity score of value sets S1 and S2 of the j-th common attribute. This constant is defined only when the attribute belongs to $NFD_A^*(< ref, ref' >)$. Its value is obtained by the function $SSim_v(S1, S2)$.

[3] *SoftJaccard* applied on the sets of references.

The Equations. Each equation $x_i = f_i(X)$ is of the form:

$$f_i(X) = \max(f_{i\text{-}df}(X), f_{i-ndf}(X))$$

The function $f_{i-df}(X)$ is an aggregation function of the similarity scores of the value pairs and the reference pairs of attributes and relations with which the i-th reference pair is functionally dependent. The function $f_{i-ndf}(X)$ allows to aggregate the similarity scores of the values pairs (and sets) and the reference pairs (and sets) of attributes and relations with which the i-th reference pair is not functionally dependent.

Modeling the Influence of Functional Attributes and Functional Relations. $f_{i-df}(X)$ is defined by the maximum of similarity scores of the value pairs and reference pairs of attributes and relations with which the two references ref and ref' are functionally dependent. The maximum function allows propagating the similarity scores of the values and the references having a strong impact. $f_{i-df}(X)$ is defined as follows:

$$f_{i\text{-}df}(X) = \max(\bigcup_{j=0}^{j=|FD_A(<ref,ref'>)|} (b_{ij-df}), avg(\bigcup_{j=0}^{j=|FD_A^M(<ref,ref'>)|} (b_{ij-dfm})),$$

$$\bigcup_{j=0}^{j=|FD_R(<ref,ref'>)|} (x_{ij-df}), avg(\bigcup_{j=0}^{j=|FD_R^M(<ref,ref'>)|} (x_{ij-dfm})))$$

Note that the similarity scores of the values and the references of attributes and relations which belongs to a same multiple functional dependency are first aggregated by an *average* function.

Modeling the Influence of Non Functional Attributes and Non Functional Relations. $f_{i-ndf}(X)$ is defined by a weighted average of the similarity scores of the values and the references of attributes and relations with which the two references ref and ref' are not functionally dependent. $f_{i-ndf}(X)$ is defined as follows:

$$f_{i-ndf}(X) = \sum_{j=0}^{j=|NFD_A(<ref,ref'>)|} (\lambda_{ij} * b_{ij-ndf}) + \sum_{j=0}^{j=|NFD_A^*(<ref,ref'>)|} (\lambda_{ij} * BS_{ij-ndf}) +$$

$$\sum_{j=0}^{j=|NFD_R(<ref,ref'>)|} (\lambda_{ij} * \ x_{ij-ndf}) + \sum_{j=0}^{j=|NFD_R^*(<ref,ref'>)|} (\lambda_{ij} * \ XS_{ij-ndf})$$

Where λ_{ij} represents the weight of the j-th attribute or relation in the similarity computation of the i-th reference pair. Since we have neither expert knowledge nor training data, λ_{ij} is computed in function of the number of the common attributes and relations.

5.4 Iterative Algorithm for Reference Pairs Similarity Computation

To compute the similarity scores, we have implemented an iterative resolution method inspired from the *Jacobi* method [20] for the resolution of linear equation systems. At each iteration, the method computes the variables values by using those computed in the precedent iteration.

Iterative Similarity Scores Computation. Starting from an initial vector $X^0 = (x_1^0, x_2^0, \ldots, x_n^0)$, the value of the vector X at the k-th iteration is obtained by the expression : $X^k = F(X^{k-1})$. At each iteration k we compute the value of each x_i^k : $x_i^k = f_i(x_1^{k-1}, x_2^{k-1}, \ldots\ldots, x_n^{k-1})$ until a fix-point with precision ϵ is reached. The fix-point is reached when : $\forall\ i,\ |x_i^k\ -\ x_i^{k-1}|\ \leq\ \epsilon$. The value of ϵ is fixed at a very small positive real number. The more ϵ value is small the more the set of reconciliations may be large.

The complexity of this method is in (n^2) for each iteration, where n is the number of variables. The same kind of approach has been followed by [30] in the context of schema matching. It is important to note that the convergence of the *Jacobi* method is not always guaranteed. We have proved its convergence for the resolution of our equation system.

Illustration of the Iterative Similarity Computation. We illustrate the similarity computation on the system of equations obtained from the data descriptions of Figure 2. The constants, the variables and the weights are given in the table 1. The constants correspond to the similarity scores of pairs of basic values computed by using the Jaro-Winkler measure [29]. The weights are computed in function of the number of common attributes and common relations of the reference pairs. We assume that point-fix precision ϵ is equal to 0.005.

Table 1. The variables, the constants and the weights of the equation system

Variables	Constants	Weights
$x_1 = Sim_r(S1\_m1, S2\_m1)$	$b_{11} = Sim_v($"*LOUVRE*", "*Musee du LOUVRE*"$) = 0.68$	$\lambda_{11} = \frac{1}{4}$
$x_2 = Sim_r(S1\_p1, S2\_p1)$	$b_{21} = Sim_v($"*La Joconde*", "*Abricotiers en fleurs*"$) = 0.1$	$\lambda_{21} = \frac{1}{2}$
$x_3 = Sim_r(S1\_p1, S2\_p2)$	$b_{31} = Sim_v($"*La Joconde*", "*Joconde*"$) = 0.9$	$\lambda_{31} = \frac{1}{2}$
$x_4 = Sim_r(S1\_c1, S2\_c1)$	$b_{41} = Sim_v($"*Paris*", "*Ville de Paris*"$) = 0.42$	$\lambda_{41} = \frac{1}{2}$

The equation system is the one given in Section 5.1. The different iterations of the resulting similarity computation are provided in Table 2.

The solution of the equation system is $X = (0.9, 0.45, 0.9, 0.9)$. This corresponds to the similarity scores of the four reference pairs. The fix-point has been reached after four iterations. The error vector is then equal to 0.

This example, shows how the similarity scores are propagated between the reference pairs through the relations having a strong impact on reference pairs. For instance, at the iteration (2), the similarity 0.9 of the painting pair $< S1\_p1, S2\_p2 >$ has been propagated to the museum pair $< S1\_m1, S2\_m1 >$ through the relation *contains* which belongs to $FD_R(< S1\_m1, S2\_m1 >)$.

Table 2. Illustrative example – Iterative similarity computation

Iterations	0	1	2	3	4
$x_1 = \max(0.68, x_2, x_3, \frac{1}{4} * x_4)$	0	0.68	0.9	0.9	0.9
$x_2 = \max(0.1, \frac{1}{2} * x_1)$	0	0.1	0.34	0.45	0.45
$x_3 = \max(0.9, \frac{1}{2} * x_1)$	0	0.9	0.9	0.9	0.9
$x_4 = \max(0.42, x_1)$	0	0.42	0.68	0.9	0.9

At the following iteration (3) the same similarity score has been propagated to city pair $< S1\_c1, S2\_c1 >$ through the relation *located* which belongs to $FD_R(< S1\_c1, S2\_c1 >)$. Furthermore, we have a weaker propagation of the similarity scores. For example, at the iteration (3) the similarity score 0.90 of the museums obtained at the iteration (2) has been propagated to the pair of paintings $< S1\_p1, S2\_p1 >$. Its similarity score grows to 0.45.

If we fix the reconciliation threshold T_{rec} at 0.80, then we obtain three reconciliation decisions: two cities, two museums and two paintings.

We note that the reconciliation threshold is empirically fixed. In supervised approaches [11,3], it is learnt on labeled data.

6 Experiments

The logical method (L2R) and the numerical one (N2R) have been implemented and tested on data sets related to two different domains: the tourism domain and the scientific publications.

6.1 Presentation of the Data Sets (HOTELS and Cora)

The first real data set HOTELS, provided by an industrial partner, corresponds to a set of seven data sources which leads to a pairwise data integration problem of 21 pairs of data sources. These data sources contain 28,934 references to hotels located in Europe. The UNA is stated for each source.The hotel descriptions in the different sources are very heterogeneous. First, the instantiated properties are different from one to another. Second, the basic values are multilingual, contain abbreviations, and so on.

The second data set Cora[4] (used by [1] and [14]) is a collection of 1295 citations of 112 different research papers in computer science. In this data set, the objective of the reference reconciliation is the cleaning of a given data source (i.e. duplicates elimination). The reference reconciliation problem applies then to $I \times I$ where I is the set of references of the data source S to be cleaned. For this data set, the UNA is not stated and the RDF facts describe references which belong to three different classes (*Article*, *Conference*, *Person*).

[4] Another version of Cora is available at `http://www.cs.umass.edu/~mccallum/data/cora-refs.tar.gz`

The RDFS+ Schemas: HOTELS conforms to a RDFS schema of tourism domain, which is provided by the industrial partner. We have added a set of disjunction constraints (e.g. DISJOINT(Hotel, Service)), a set of (inverse) functional property constraints (e.g. *PF(EstablishmentName)*, *PF(Name)*, *PFI(EstablishmentName, AssociatedAddress)*).

For the Cora data set, we have designed a simple RDFS schema on the scientific publication domain, which we have enriched with disjunction constraints (e.g. DISJOINT(Article, Conference)), a set of functional property constraints (e.g. *PF(Published)*, *PF(ConfName)*) and a set of inverse functional property constraints (e.g. *PFI(Title, Year, Type)*, *PFI(ConfName, ConfYear)*).

For the Cora data set, the expected results for reference reconciliation are provided. Therefore, the recall and the precision can be easily obtained by computing the ratio of the reconciliations or non-reconciliations obtained by L2R and N2R among those that are provided.

For the HOTELS data set, we have manually detected the correct results of reconciliations or non-reconciliations between the references of two data sources containing respectively 404 and 1392 references to hotels.

6.2 L2R Results

Since the set of reconciliations and the set of non-reconciliations are obtained by a logical resolution-based algorithm the precision is of 100% by construction. Then, the measure that it is meaningful to evaluate in our experiments is the recall.

In the following, we summarize the results obtained on the HOTELS data set and then those obtained on the *Cora* data set. We emphasize the impact on the recall of increasing the expressiveness of the schema by adding constraints.

L2R Results on HOTELS Data Set. In the figure 4, we show the recall that we have obtained on the two sources on which we have manually detected the reconciliation and no reconcilation pairs. We distinguish the recall computed only on the set of reconciled references (REC) and only on not reconciled references (NREC). To examine the 532368 reference pairs we have first automatically extracted the name and address of the hotels belonging to the smaller source. Then, we have used the standard string search commands of Unix to search in the file of the second source the truncated corresponding strings (in order to be robust with typographical errors). Then, we have set the correct no reconcilations to be the remaining pairs.

As it is shown in the column named "RDFS+ (HOTELS)" of the figure 4, we have obtained a recall of 8.3%. If we only consider the reconciliations subset (REC) the recall is 54%. The REC subset corresponds to the reconciliations inferred by exploiting the inverse functional constraint *PFI(EstablishmentName, AssociatedAddress)*. It is important to emphasize that those reconciliations are inferred in spite of the irregularities in the data descriptions: not valued addresses and a lot of variability in the values, in particular in the addresses:

	RDFS+		**RDFS+ & {DA or DP}**	
	HOTELS	Cora	HOTELS	Cora
Recall (REC)	54 %	52.7 %	54 %	52.7 %
Recall (NREC)	8.2 %	50.6 %	75.9 %	94.9 %
Recall	8.3 %	50.7 %	75.9 %	94.4 %
Precision	100 %	100 %	100 %	100 %

Fig. 4. L2R results on HOTELS and Cora data sets

"*parc des fees*" vs. "*parc des fees, (nearby Royan)*". In addition, in one of the data sources, several languages are used for the basic values.

If we only consider the non-reconciliations subset (NREC) the recall is 8.2%. Actually, the only rules that are likely to infer no reconciliations are those translating the UNA assumption. Now, if we enrich the schema just by declaring pairwise disjoint specializations of the *Hotel* class (by distinguishing hotels by their countries), we obtain an impressive increasing of the recall on NREC, from 8.2% to 75.9%, as it is shown in the "RDFS+ (HOTELS) & DA" column.

L2R Results on Cora Data Set. We focus on the results obtained for the *Article* and *Conference* classes, which contain respectively 1295 references and 1292 references.

As presented in the column named "RDFS+ (Cora)" of the figure 4, the recall obtained on the Cora data set is 50.7%. This can be refined in a recall of 52.7% computed on the REC subset and a recall of 50.6% computed on NREC subset. The set of inferred reconciliations (REC subset) for references to articles is obtained by exploiting the constraint PFI(*Title, Year*) of combined inverse functionality on the properties $Title$ and $Year$. For the conferences, 35.8% of the reconciliations are obtained by exploiting the constraint PFI(*ConfName, ConfYear*) of combined inverse functionality on the attributes $ConfName$ and $ConfYear$, and 64.1% are obtained by propagating the reconciliations of references to articles, using the constraint PF(*Published*) of functionality of the relation *Published*. The set of inferred no reconciliations (NREC subset) are obtained by exploiting the constraint of disjunction between the *Article* and *Conference* classes.

For this data set, the RDFS+ schema can be easily enriched by the declaration that the property *confYear* is discriminant. When this discriminant property is exploited, the recall on the REC subset remains unchanged (52.7%) but the recall on NREC subset grows to 94.9%, as it is shown in the "RDFS+ (Cora) & DP" column. This significant improvement is due to chaining of different rules of reconciliations: the non-reconciliations on references to conferences for which the values of the *confYear* are different entail in turn non-reconciliations of the associated articles by exploiting the constraint PF(*published*).

This recall is comparable to (while a little bit lower than) the recall on the same data set obtained by *supervised* methods like e.g., [1]. The point is that L2R is *not supervised* and guarantees a 100% precision.

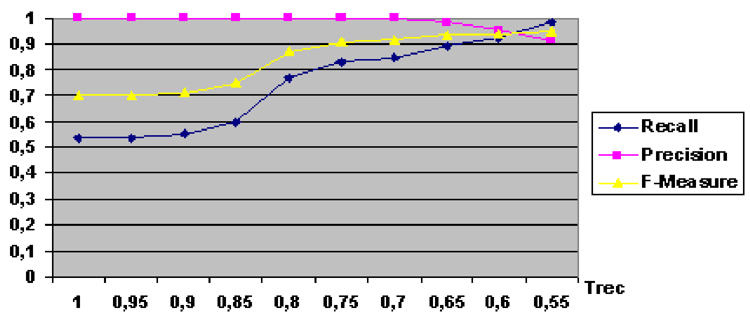


Fig. 5. N2R results obtained on HOTELS data set

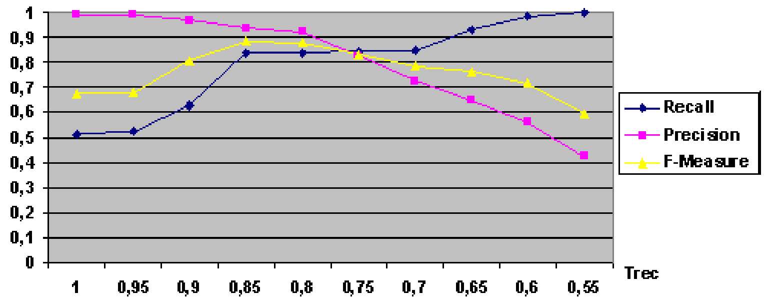


Fig. 6. N2R results obtained on *Cora* data set

6.3 N2R Results

In the following we summarize the results obtained on the HOTELS data set and on the Cora data set by N2R after the application of L2R.

N2R Results on HOTELS Data Set. The results obtained by N2R on the HOTELS data set are given in Figure 5, where the values in the x-axis correspond to values of the reconciliation threshold T_{rec}.

When $T_{rec} = 1$, N2R do not obtain more results than L2R. When T_{rec} is decreased to 0.70 the recall increases of 31 % while the precision remains at 100%. The best results have been obtained at $T_{rec} = 0.55$. For this value, the F-Measure reaches a maximum value of 94 %, with a recall of 98 % and a precision of 91 %. Some mis-reconciliations are due to the fact that some hotel references have the same name and a different addresses and vis versa.

The reconciliation space of N2R has been reduced of 75.9 % which corresponds to 427 212 of reference pairs among 562 368 reference pairs in total.

When N2R is applied independently, the results are very close than those obtained when N2R is combined with L2R.

N2R Results on Cora Data Set. The results obtained by N2R on the Cora data set are given in the Figure 6.

For $T_{rec} = 1$, N2R do not obtain more results than L2R. We also emphasize the interesting evolution of the recall and precision values in function of T_{rec}. Indeed, when the threshold is decreased to 0.85, the recall increases by 33% while the precision only falls by 6%. The best results are obtained when $T_{rec} = 0.85$. The F-measure is then at its maximum value of 88%. Besides, when the recall value is almost of 100%, for $T_{rec} = 0.5$, the precision value is still about 40%.

The exploitation of the non-reconciliation inferred by L2R allows an important reduction of the reconciliation space handled in N2R. For the *Cora* data set the size of the reconciliation space is about 37 millions of reference pairs. It has been reduced of 32.8 % thanks to the correct no reconciliations inferred by L2R. This reduction corresponds to 12 millions of reference pairs. Moreover, the reconciliation inferred by L2R are not recomputed in N2R.

These experimentations show that good results can be obtained by an automatic and unsupervised method if it exploits knowledge declared in the schema. Furthermore, the method is able to obtain F-Measure which is better than some supervised methods such that [14]. This collective record linkage method obtains an F-Measure of 87% for the same data set. Nevertheless, the results obtained by other supervised methods are slightly better than ours: [1] obtain a F-Measure of 90 % by using a method based on a dependency graph where the dependencies between reference pairs are learnt on labeled data ; and [31] obtain a F-Measure of 95 % by using an adaptive approach where the used similarity measures are learnt on labeled data and adapted to the specificities of the data sets. Since, in our numerical method, the similarity computation takes into account the schema semantics, it obtains results that are comparable to those obtained by supervised methods, even without using any labeled data.

When N2R is applied separately, we obtain only a slight regression of N2R results.

6.4 Efficiency Results

We have conducted efficiency experiments of the reconciliation methods L2R and N2R on *Cora* data set. We aim by these experiments to show how the efficiency of N2R is improved when the L2R results are exploited. We have applied the reference reconciliation methods on reference sets selected from *Cora* data set ranging from 632 to 6108 references. At each stage we have increased the data set of $\frac{1}{10}$th of the whole set of references.

We note that the reference reconciliation code was implemented in Java, and our experiments were run on 2,2GHz Intel Core 2 Duo with 2GB of RAM.

To analyze effenciency, we have measured the execution time of the methods without considering the runtime of the data loading step, because it is common for both methods. Figure 7 shows the execution time obtained for the methods of reference reconciliation L2R and N2R, when they are applied independently and when they are combined, i.e. N2R is preceeded by L2R. For N2R method we have fixed the parameter ϵ at 0.0001.

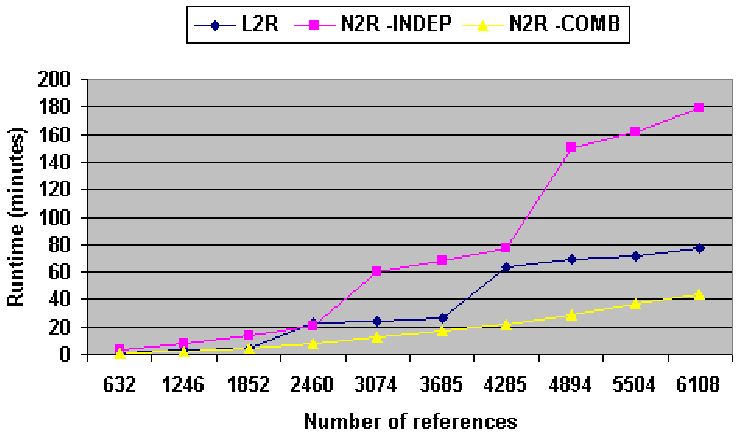


Fig. 7. Execution time of L2R and N2R methods obtained on *Cora* data set

The first result concerns the case when L2R and N2R are applied independently (curves labeled L2R and N2R-INDEP of the Figure 7. We notice that when the dataset gets larger, L2R outperforms N2R up to 59 % for 3685 references.

The second result concerns the execution time of N2R when it uses the results inferred by L2R. The curves labeled N2R-INDEP and N2R-COMB show a real improvement of N2R runtime when it is preceeded by L2R. The runtime falls of 74% for the 6108 references. This improvement in the N2R runtime is due to the large amount of non-reconciliations inferred by L2R, that are not considered by N2R i.e. they are not added in the equation system. Even the inferred reconciliations between references contribute in the speeding up of N2R. Actually, the similarity scores of these references is fixed at the maximum value then there is no need to compute their similarity scores be N2R. We have also noticed that the convergence of N2R-COMB is achieved in less iterations (3 iterations in average) than when is is applied independently (9 iterations in average). Finally, Figure 7 shows that N2R-COMB outperforms L2R up to 42% for the 6108 references.

7 Related Work

The problem of reference reconciliation was introduced by the geneticist Newcombe [32] and was first formalized by Fellegi and Sunter [2]. Since then, various approaches have been proposed in different areas and under different names – record linkage[2,3], object matching [7], or entity resolution [6,5]. We distinguish the different approaches: (i) according to the exploitation of the reference descriptions, i.e. if the relations between references are exploited in addition to the attributes ; and (ii) according to how knowledge is acquired, i.e. if knowledge is learnt on labelled data or is declared by domain expert.

The naive way to decide on the reconciliation or on the non-reconciliation of references, is the comparison of their unstructured textual description [33,34].

In these approaches, the similarity is computed by using only the textual values of the attributes in the form of a single long string without distinguishing which value corresponds to which attribute. This kind of approaches is useful in order to have a fast similarity computation [33], to obtain a set of reference pairs that are candidates for the reconciliation [34] or when the attribute-value associations may be incorrect. That is why this technique is used in *CiteSeer* portal to reconcile data which is automatically extracted from Web pages.

The traditional approaches of reference reconciliation consider the reference description as structured in several attributes. To decide on the reconciliation or on the non-reconciliation of references, some of these approches use probabilistic models [2,3,4], such as Bayesian network or SVM. However, these probabilistic models need to be trained on labeled data. This training step can be very time-consuming what is not desirable in online applications. Indeed, in such online contexts, labeled data can not be acquired and runtime constraints are very strong. Alternative approaches have been proposed like [35] where the similarity measures (see [29] for a survey) are used to compute smilarity scores between attribute values which are then gathered in a linear combination, like a weighted average. Although these approaches do not need training step, they however need to learn some parameters like weights associated to similarity scores of the different attributes. In order to improve the result quality some methods [11] use adaptive supervised algorithms that learn string similarity measures from labeled data.

The idea of exploiting relations that link references together has been recently explored in several works on reference reconciliation. The relations can be either explicitly expressed in data [12,1,14], such as foreign-keys in relational databases, or discovered [13] and then used during the reference reconciliation. To model the dependencies between reference pairs induced by the relations, [1] build a dependency graph and use it to iteratively propagate similarity scores and reconciliation decisions. However, the weights associated to the dependencies are learnt on labeled data. In [13] , the authors translate the *Context Attraction Principle* in a linear equation system and then, by its resolution they compute the connection strength between entities, through relations. In [14], a probabilistic dependency model has been proposed. It allows propagating reconciliation decisions through shared relations. In our approach, the relations between references are exploited by both logical and numerical reference reconciliation methods. The relations are used in the logical method to iteratively propagate reconciliation and non-reconciliation decisions through the logical rules. They are exploited in the numerical method to iteratively propagate similarity scores thanks to the iterative resolution of the non-linear equation system. Furthermore, our logical method infer correct non-reconciliations between references which is very usefull in applications where there are very few redundancies. These correct non-reconciliations can be used by the numerical method to reduce its reconciliation space and therefore speed-up its excecution time.

In order to improve the quality of their results, some recent methods exploit knowledge like the importance of the different attributes and relations, similarity measures or reconciliation and non-reconciliation rules. Knowledge can be either learnt on labeled data or declaratively specified by a domain expert. For instance, in [35,1], knowledge about the impacts of the different attributes or relations are encoded in weights learnt on labeled data. In [7] the rules of value normalization (i.e. date format, phone number) and of reconciliation are learnt on labeled data by using a decision tree model. In the declarative approach proposed by [36], profiles of the representative entities are expoited. These profiles that are manually specified by a domain expert contain a set of constraints on correlations between attributes which should be satisfied by the references.

In the system AJAX [10] a declarative language has been proposed to express different kinds of knowledge like knowledge on value normalization, on mapping operations and on the extraction of groups of references. Although these supervised and declarative methods ensure good results, they remain however vulnerable to changes of application domain and of data sources features. The supervised method should re-learn the knowledge on new labeled data and for the declarative methods the expert should be asked to re-specify the used knowledge. In the spirit of knowledge-based approches, we propose two declarative methods which exploit general knowledge declared on the schema and on the data sources, such as functional dependencies and Unique Name Assumption. In L2R, knowledge semantics is automatically translated into Horn rules and used to infer (non) reconciliations between reference pairs and (non) synonymies between values. Knowledge semantics is also automatically translated into non-linear equations which allow to compute similarity scores of reference pairs. Comparing to the viewed knowledge-based approches, our methods are not sensitive to domain and data changes. The exploited knowledge are general and domain-independent, indeed. Futhermore, the logical and numerical methods are unsupervised since no labeled data is needed by neither L2R nor N2R.

8 Conclusion and Future Work

We have presented the combination of a logical and numerical approach for the reference reconciliation problem. Both approaches exploit schema and data knowledge given in a declarative way by a set of axioms. This guarantees their genericity: if the domain or the sources change it is sufficient to update the set of axioms. Secondly, the relations between references are exploited either by L2R for propagating (non) reconciliation decisions through logical rules or by N2R for propagating similarity scores thanks to the resolution of the equation system. Third, the two methods are unsupervised because no labeled data set is used. Fourth, the combined approach is able to capitalize its experience by saving the correct (no) synonymies inferred by L2R in a dictionary. This allows to learn the syntactic variations of an application domain.

Furthermore, by using the logical method we obtain reconciliations and non-reconciliations that are sure. This distinguishes L2R from other existing works. This is an important point since, as it has been emphasized in [3], unsupervised approaches which deal with the reference reconciliation problem have a lot of difficulties to estimate in advance the precision of their system when it is applied to a new set of data. The numerical method complements the results of logical one. It exploits the schema and data knowledge and expresses the similarity computation in non linear equation system. This distinguishes N2R from other existing work.

The experiments show promising results for recall, and most importantly its significant increasing when axioms are added. This shows the interest and the power of the generic and flexible approach of L2R since it is quite easy to add rules to express constraints on the domain of interest.

As a future work, we first plan to exploit the results of the logical step to learn the weighting coefficients involved in the combination of the different similarity scores. We also plan to adapt the method to be used in a peer-to-peer settings. A system such SomeWhere [37] is a P2P infrastructure that exploits mappings between peer's ontologies to answer queries in a sound and complete way. This could be completed by a reference reconciliation method based on L2R and N2R to discover and exploit reconciliation decisions between references stored at different peers. Finally, we plan to study how the reference reconciliation could help the schema reconciliation and conversely how the schema reconciliation could help the reference reconciliation.

References

1. Dong, X., Halevy, A., Madhavan, J.: Reference reconciliation in complex information spaces. In: ACM SIGMOD, pp. 85–96. ACM Press, New York (2005)
2. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. Journal of the American Statistical Association 64(328), 1183–1210 (1969)
3. Winkler, W.E.: Overview of record linkage and current research directions. Technical report, Statistical Research Division U.S. Census Bureau Washington, DC 20233 (2006)
4. Verykios, V.S., Elmagarmid, A.K., Houstis, E.N.: Automating the approximate record-matching process. Inf. Sci. Inf. Comput. Sci. 126(1-4), 83–98 (2000)
5. Benjelloun, O., Garcia-Molina, H., Kawai, H., Larson, T.E., Menestrina, D., Su, Q., Thavisomboon, S., Widom, J.: Generic entity resolution in the serf project. IEEE Data Eng. Bull. 29(2), 13–20 (2006)
6. Bhattacharya, I., Getoor, L.: Entity Resolution in Graphs. Wiley, Chichester (2006)
7. Tejada, S., Knoblock, C.A., Minton, S.: Learning object identification rules for information integration. Information Systems 26(8), 607–633 (2001)
8. Rahm, E., Bernstein, P.: A survey of approaches to automatic schema matching. VLDB Journal 10, 334–350 (2001)
9. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. Journal on Data semantics (2005)
10. Galhardas, H., Florescu, D., Shasha, D., Simon, E., Saita, C.: Declarative data cleaning: Language, model and algorithms. In: VLDB 2001 (2001)
11. Bilenko, M., Mooney, R.: Adaptive duplicate detection using learnable string similarity measures. In: SIGKDD 2003 (2003)
12. Ananthakrishna, R., Chaudhuri, S., Ganti, V.: Eliminating fuzzy duplicates in data warehouses. In: ACM SIGMOD. ACM Press, New York (2002)
13. Kalashnikov, D., Mehrotra, S., Chen, Z.: Exploiting relationships for domain-independent data cleaning. In: SIAM Data Mining 2005 (2005)
14. Parag, S., Pedro, D.: Multi-relational record linkage. In: MRDM Workshop (2004)

15. McBride, B.: The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS. In: Handbook on Ontologies, pp. 51–66 (2004)
16. McGuinness, D.L., van Harmelen, F.: OWL Web Ontology Language Overview (February 2004)
17. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: A Semantic Web Rule Language Combining OWL and RuleML (submission) (May 2004)
18. Rodriguez, J.B., Gómez-Pérez, A.: Upgrading relational legacy data to the semantic web. In: WWW 2006: Proceedings of the 15th international conference on World Wide Web, pp. 1069–1070. ACM, New York (2006)
19. Murray, C., Alexander, N., Das, S., Eadon, G., Ravada, S.: Oracle spatial resource description framework (rdf). Technical report, Oracle (2005)
20. Golub, G.H., Van Loan, C.F.: Matrix Computations, 2nd edn., Baltimore, MD, USA (1989)
21. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley, Reading (1995)
22. Hayes, P.: RDF Semantics, Technical report (2004), `http://www.w3.org/tr/rdf-mt/`
23. Saïs, F., Pernelle, N., Rousset, M.-C.: L2R: A Logical Method for Reference Reconciliation. In: AAAI, pp. 329–334 (2007)
24. Fischer, S.: Two processes of reduplication in the american sign language. Foundations of Language 9, 460–480 (1973)
25. Robinson, A.: A machine-oriented logic based on the resolution principle. Journal ACM 12(1), 23–41 (1965)
26. Chin-Liang, C., Char-Tung Lee, R.: Symbolic Logic and Mechanical Theorem Proving. Academic Press, Inc., London (1997)
27. Henschen, L.J., Wos, L.: Unit refutations and horn sets. J. ACM 21(4), 590–605 (1974)
28. Forbus, K.D., de Kleer, J.: Building problem solvers. MIT Press, Cambridge (1993)
29. Cohen, W.W., Ravikumar, P., Fienberg, S.E.: A comparison of string distance metrics for name-matching tasks. In: IIWeb, pp. 73–78 (2003)
30. Euzenat, J., Valtchev, P.: Similarity-based ontology alignment in owl-lite. In: ECAI, pp. 333–337 (2004)
31. Cohen, W.W., Richman, J.: Learning to match and cluster large high-dimensional data sets for data integration. In: KDD 2002, pp. 475–480. ACM, New York (2002)
32. Newcombe, H.B., Kennedy, J.M., Axford, S.J., James, A.P.: Automatic linkage of vital records. Science 130, 954–959 (1959)
33. Cohen, W.W.: Data integration using similarity joins and a word-based information representation language. ACM Transactions on Information Systems 18(3), 288–321 (2000)
34. Bilke, A., Naumann, F.: Schema matching using duplicates. In: ICDE, pp. 69–80 (2005)
35. Dey, D., Sarkar, S., De, P.: A probabilistic decision model for entity matching in heterogeneous databases. Manage. Sci. 44(10), 1379–1395 (1998)
36. Doan, A., Lu, Y., Lee, Y., Han, J.: Object matching for information integration: A profiler-based approach. In: IIWeb, pp. 53–58 (2003)
37. Adjiman, P., Chatalic, P., Goasdoué, F., Rousset, M.-C., Simon, L.: Distributed reasoning in a peer-to-peer setting: Application to the semantic web. J. Artif. Intell. Res (JAIR) 25, 269–314 (2006)

Tightly Coupled Probabilistic Description Logic Programs for the Semantic Web*

Andrea Calì[1,2], Thomas Lukasiewicz[1,**], Livia Predoiu[3], and Heiner Stuckenschmidt[3]

[1] Computing Laboratory, University of Oxford
Wolfson Building, Parks Road, Oxford OX1 3QD, UK
{andrea.cali,thomas.lukasiewicz}@comlab.ox.ac.uk

[2] Oxford-Man Institute of Quantitative Finance, University of Oxford
Blue Boar Court, 9 Alfred Street, Oxford OX1 4EH, UK
andrea.cali@oxford-man.ox.ac.uk

[3] Institut für Informatik, Universität Mannheim
68159 Mannheim, Germany
{livia,heiner}@informatik.uni-mannheim.de

Abstract. We present a novel approach to probabilistic description logic programs for the Semantic Web in which disjunctive logic programs under the answer set semantics are tightly coupled with description logics and Bayesian probabilities. The approach has several nice features. In particular, it is a logic-based representation formalism that naturally fits into the landscape of Semantic Web languages. Tightly coupled probabilistic description logic programs can especially be used for representing mappings between ontologies, which are a common way of approaching the semantic heterogeneity problem on the Semantic Web. In this application, they allow in particular for resolving inconsistencies and for merging mappings from different matchers based on the level of confidence assigned to different rules. Furthermore, tightly coupled probabilistic description logic programs also provide a natural integration of ontologies, action languages, and Bayesian probabilities towards Web Services. We explore the computational aspects of consistency checking and query processing in tightly coupled probabilistic description logic programs. We show that these problems are decidable and computable, respectively, and that they can be reduced to consistency checking and cautious/brave reasoning, respectively, in tightly coupled disjunctive description logic programs. Using these results, we also provide an anytime algorithm for tight query processing. Furthermore, we analyze the complexity of consistency checking and query processing in the new probabilistic description logic programs, and we present a special case of these problems with polynomial data complexity.

Keywords: Probabilistic description logic programs, Semantic Web, disjunctive logic programs, answer set semantics, description logics, Bayesian probabilities, ontology mapping, inconsistency handling, merging ontology mappings, Web Services, algorithms, complexity, data tractability.

* Some preliminary results of this paper have appeared in: *Proceedings ICLP-2007* [1], *Proceedings URSW-2007* [2], and *Proceedings FoIKS-2008* [3].

** Alternative address: Institut für Informationssysteme, Technische Universität Wien, Favoritenstraße 9-11, 1040 Wien, Austria, lukasiewicz@kr.tuwien.ac.at

1 Introduction

The *Semantic Web* [4,5] aims at an extension of the current World Wide Web by standards and technologies that help machines to understand the information on the Web so that they can support richer discovery, data integration, navigation, and automation of tasks. The main ideas behind it are to add a machine-readable meaning to Web pages, to use ontologies for a precise definition of shared terms in Web resources, to use knowledge representation technology for automated reasoning from Web resources, and to apply cooperative agent technology for processing the information of the Web.

The Semantic Web consists of several hierarchical layers, where the *Ontology layer*, in form of the *OWL Web Ontology Language* [6] (recommended by the W3C), is currently the highest layer of sufficient maturity. OWL consists of three increasingly expressive sublanguages, namely *OWL Lite*, *OWL DL*, and *OWL Full*. OWL Lite and OWL DL are essentially very expressive description logics with an RDF syntax. As shown in [7], ontology entailment in OWL Lite (resp., OWL DL) reduces to knowledge base (un)satisfiability in the description logic $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$). As a next step in the development of the Semantic Web, one aims especially at sophisticated representation and reasoning capabilities for the *Rules*, *Logic*, and *Proof layers* of the Semantic Web. Several recent research efforts are going in this direction.

In particular, there is a large body of work on integrating ontologies and rules, which is a key requirement of the layered architecture of the Semantic Web. One type of integration is to build rules on top of ontologies, that is, rule-based systems that use vocabulary from ontology knowledge bases. Another form of integration is to build ontologies on top of rules, where ontological definitions are supplemented by rules or imported from rules. Both types of integration are realized in recent hybrid integrations of rules and ontologies, called *description logic programs* (or *dl-programs*), which have the form $KB = (L, P)$, where L is a description logic knowledge base and P is a finite set of rules involving either queries to L in a loose coupling [8,9] or concepts and roles from L as unary and binary predicates, respectively, in a tight coupling [10] (for detailed overviews on the different types of description logic programs, see especially [9,11,10,12]).

Other works explore formalisms for *uncertainty reasoning in the Semantic Web* (an important recent forum for approaches to uncertainty reasoning in the Semantic Web is the annual *Workshop on Uncertainty Reasoning for the Semantic Web (URSW)*; there also exists a W3C Incubator Group on *Uncertainty Reasoning for the World Wide Web*). There are especially probabilistic extensions of description logics [13,14], of Web ontology languages [15,16], and of description logic programs [17,18] (to encode ambiguous information, such as "John is a student with the probability 0.7 and a teacher with the probability 0.3", which is very different from vague / fuzzy / imprecise information, such as "John is tall with the degree of truth 0.7"). In particular, the two works [17,18] extend the loosely coupled description logic programs of [8,9] by probabilistic uncertainty as in the independent choice logic (ICL) [19]. The ICL is a powerful representation and reasoning formalism for single- and also multi-agent systems, which combines logic and probability, and which can represent a number of important uncertainty formalisms, in particular, influence diagrams, Bayesian networks, Markov decision processes, normal form games, and Pearl's structural causal models [20].

In this paper, we continue this line of research. We propose *tightly coupled probabilistic (disjunctive) description logic programs under the answer set semantics* (or *probabilistic dl-programs*), which are a tight integration of disjunctive logic programs under the answer set semantics, the description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$ (which stand behind OWL Lite and OWL DL, respectively), and Bayesian probabilities. To our knowledge, this is the first such approach. As for important applications in the Semantic Web, the new description logic programs can especially be used for representing mappings between ontologies under inconsistencies and confidence values. Furthermore, tightly coupled probabilistic description logic programs are also a natural integration of action languages, ontologies, and Bayesian probabilities, especially towards Web Services.

The problem of aligning heterogeneous ontologies via semantic mappings has been identified as one of the major challenges of Semantic Web technologies. To address this problem, a number of languages for representing semantic relations between elements in different ontologies as a basis for reasoning and query answering across multiple ontologies have been proposed [21,22]. In the presence of real world ontologies, it is unrealistic to assume that mappings between ontologies are created manually by domain experts, since existing ontologies, e.g., in the area of medicine contain thousands of concepts and hundreds of relations. Recently, a number of heuristic methods for matching elements from different ontologies have been proposed that support the creation of mappings between different languages by suggesting candidate mappings (e.g., [23]). These methods rely on linguistic and structural criteria. Evaluation studies have shown that existing methods often trade off precision and recall. The resulting mapping either contains a fair amount of errors or only covers a small part of the ontologies involved [24,25]. To leverage the weaknesses of the individual methods, it is common practice to combine the results of a number of matching components or even the results of different matching systems to achieve a better coverage of the problem [23].

This means that automatically created mappings often contain uncertain hypotheses and errors that need to be dealt with, briefly summarized as follows:

- mapping hypotheses are often oversimplifying, since most matchers only support very simple semantic relations (mostly equivalence between simple elements, i.e., only between two concepts or two relations);
- there may be conflicts between different hypotheses for semantic relations from different matching components and often even from the same matcher;
- semantic relations are only given with a degree of confidence in their correctness.

If we want to use the resulting mappings, we have to find a way to deal with these uncertainties and errors in a suitable way. We argue that the most suitable way of dealing with uncertainties in mappings is to provide means to explicitly represent uncertainties in the target language that encodes the ontology mappings. In this way, integrated reasoning with the ontologies, the mappings, and the uncertainties can be performed.

The main contributions of this paper can be summarized as follows:

- We present tightly coupled probabilistic (disjunctive) description logic programs under the answer set semantics, which combine the tightly coupled disjunctive description logic programs under the answer set semantics from [10] with Bayesian

probabilities as in the ICL [19]. The approach assumes no structural separation between the vocabularies of the ontology and the rule component. This enables us to have description logic concepts and roles in both rule bodies and rule heads.
- We show that tightly coupled probabilistic description logic programs are especially well-suited for representing mappings between ontologies. In particular, we can have concepts and roles in both rule bodies and rule heads, which is necessary if we want to use rules to combine ontologies. Furthermore, we can have disjunctions in rule heads and nonmonotonic negations in rule bodies, which gives a rich basis for refining and rewriting automatically created mappings for resolving inconsistencies. Finally, the integration with probability theory provides us with a sound formal framework for dealing with confidence values. In particular, we can interpret the confidence values as error probabilities and use standard techniques for combining them. We can also resolve inconsistencies via trust probabilities.
- Since the ICL is actually a formalism for probabilistic reasoning about actions in dynamic single- and multi-agent systems, tightly coupled probabilistic description logic programs are also a natural way of combining an action language with both description logics and Bayesian probabilities, especially towards Web Services.
- We show that consistency checking (resp., query processing) in tightly coupled probabilistic description logic programs are decidable (resp., computable), and can be reduced to their classical counterparts in tightly coupled disjunctive description logic programs. We also provide an anytime algorithm for query processing.
- We analyze the complexity of consistency checking and query processing in tightly coupled probabilistic description logic programs, which turn out to be complete for the complexity classes $\mathrm{NEXP}^{\mathrm{NP}}$ and co-$\mathrm{NEXP}^{\mathrm{NP}}$, respectively. Furthermore, we show that in the stratified normal case relative to the description logic *DL-Lite*, these two problems can be solved in polynomial time in the data complexity.

The rest of this paper is organized as follows. Sections 2 and 3 recall the expressive description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$, and the tightly coupled disjunctive description logic programs under the answer set semantics from [10], respectively. In Section 4, we introduce our new approach to tightly coupled probabilistic description logic programs. Sections 5 and 6 describe its application for representing ontology mappings and for probabilistic reasoning about actions involving ontologies, respectively. In Section 7, we explore the computational aspects of consistency checking and query processing in tightly coupled probabilistic description logic programs, and we provide an anytime algorithm for query processing. Section 8 describes a special case where consistency checking and query processing can be done in polynomial time in the data complexity. Section 9 discusses some most closely related works. In Section 10, we summarize our results and give an outlook on future research. Note that detailed proofs of all results of this paper are given in Appendix A.

2 Description Logics

We now recall the expressive description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$, which stand behind the Web ontology languages OWL Lite and OWL DL [7], respectively. Intuitively, description logics model a domain of interest in terms of concepts and roles, which represent classes of individuals and binary relations between classes of individuals, respectively. A description logic knowledge base encodes especially subset relationships between concepts, subset relationships between roles, the membership of individuals to concepts, and the membership of pairs of individuals to roles.

2.1 Syntax

We first describe the syntax of $\mathcal{SHOIN}(\mathbf{D})$. We assume a set of *elementary datatypes* and a set of *data values*. A *datatype* is either an elementary datatype or a set of data values (*datatype oneOf*). A *datatype theory* $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$ consists of a *datatype domain* $\Delta^{\mathbf{D}}$ and a mapping $\cdot^{\mathbf{D}}$ that assigns to each elementary datatype a subset of $\Delta^{\mathbf{D}}$ and to each data value an element of $\Delta^{\mathbf{D}}$. The mapping $\cdot^{\mathbf{D}}$ is extended to all datatypes by $\{v_1, \ldots\}^{\mathbf{D}} = \{v_1^{\mathbf{D}}, \ldots\}$. Let $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, and $\mathbf{I}$ be pairwise disjoint (denumerable) sets of *atomic concepts*, *abstract roles*, *datatype roles*, and *individuals*, respectively. We denote by $\mathbf{R}_A^-$ the set of *inverses* R^- of all $R \in \mathbf{R}_A$.

A *role* is any element of $\mathbf{R}_A \cup \mathbf{R}_A^- \cup \mathbf{R}_D$. *Concepts* are inductively defined as follows. Every $\phi \in \mathbf{A}$ is a concept, and if $o_1, \ldots, o_n \in \mathbf{I}$, then $\{o_1, \ldots, o_n\}$ is a concept (*oneOf*). If ϕ, ϕ_1, and ϕ_2 are concepts and if $R \in \mathbf{R}_A \cup \mathbf{R}_A^-$, then also $(\phi_1 \sqcap \phi_2)$, $(\phi_1 \sqcup \phi_2)$, and $\neg\phi$ are concepts (*conjunction*, *disjunction*, and *negation*, respectively), as well as $\exists R.\phi$, $\forall R.\phi$, $\geqslant nR$, and $\leqslant nR$ (*existential*, *value*, *atleast*, and *atmost restriction*, respectively) for an integer $n \geqslant 0$. If D is a datatype and $U \in \mathbf{R}_D$, then $\exists U.D$, $\forall U.D$, $\geqslant nU$, and $\leqslant nU$ are concepts (*datatype existential*, *value*, *atleast*, and *atmost restriction*, respectively) for an integer $n \geqslant 0$. We write $\top$ and $\bot$ to abbreviate the concepts $\phi \sqcup \neg\phi$ and $\phi \sqcap \neg\phi$, respectively, and we eliminate parentheses as usual.

An *axiom* has one of the following forms: (1) $\phi \sqsubseteq \psi$ (*concept inclusion axiom*), where ϕ and ψ are concepts; (2) $R \sqsubseteq S$ (*role inclusion axiom*), where either $R, S \in \mathbf{R}_A \cup \mathbf{R}_A^-$ or $R, S \in \mathbf{R}_D$; (3) $\mathrm{Trans}(R)$ (*transitivity axiom*), where $R \in \mathbf{R}_A$; (4) $\phi(a)$ (*concept membership axiom*), where ϕ is a concept and $a \in \mathbf{I}$; (5) $R(a, b)$ (resp., $U(a, v)$) (*role membership axiom*), where $R \in \mathbf{R}_A$ (resp., $U \in \mathbf{R}_D$) and $a, b \in \mathbf{I}$ (resp., $a \in \mathbf{I}$ and v is a data value); and (6) $a = b$ (resp., $a \neq b$) (*equality* (resp., *inequality*) *axiom*), where $a, b \in \mathbf{I}$. A *(description logic) knowledge base* L is a finite set of axioms.

We next define simple abstract roles. For abstract roles $R \in \mathbf{R}_A$, we define $\mathrm{Inv}(R) = R^-$ and $\mathrm{Inv}(R^-) = R$. Let $\sqsubseteq_L^\star$ denote the reflexive and transitive closure of $\sqsubseteq$ on $\bigcup\{\{R \sqsubseteq S,\ \mathrm{Inv}(R) \sqsubseteq \mathrm{Inv}(S)\} \mid R \sqsubseteq S \in L,\ R, S \in \mathbf{R}_A \cup \mathbf{R}_A^-\}$. An abstract role S is *simple* relative to L iff for each abstract role R with $R \sqsubseteq_L^\star S$, it holds that (i) $\mathrm{Trans}(R) \notin L$ and (ii) $\mathrm{Trans}(\mathrm{Inv}(R)) \notin L$. Informally, an abstract role S is simple iff it is neither transitive nor has transitive subroles. For decidability, number restrictions in description logic knowledge bases L are restricted to simple abstract roles [26].

The syntax of $\mathcal{SHIF}(\mathbf{D})$ is as the above syntax of $\mathcal{SHOIN}(\mathbf{D})$, but without the oneOf constructor and with the atleast and atmost constructors limited to 0 and 1.

Example 2.1 (University Database). A university database may use a description logic knowledge base L to characterize students and exams. For example, suppose that (1) every bachelor student is a student; (2) every master student is a student; (3) every Ph.D. student is a student; (4) professors are not students; (5) every student is either a bachelor student or a master student or a Ph.D. student; (6) only students take exams and only exams are taken; (7) every student takes between one and ten exams, and every exam is taken by at most 25 students; (8) John is a student, Mary is a master student, java is an exam, and John has taken it; and (9) John is the same person as John Miller, and John and Mary are different persons. These relationships are expressed by the following axioms in L (where $\mathbf{A} = \{\mathit{bachelor\_student}, \mathit{master\_student}, \mathit{student}, \mathit{professor}, \mathit{exam}\}$, $\mathbf{R}_A = \{\mathit{taken}\}$, $\mathbf{R}_D = \emptyset$, and $\mathbf{I} = \{\mathit{john}, \mathit{john\_miller}, \mathit{mary}, \mathit{java}\}$):

$$\begin{array}{l}
(1)\ \mathit{bachelor\_student} \sqsubseteq \mathit{student};\ \ (2)\ \mathit{master\_student} \sqsubseteq \mathit{student};\\
(3)\ \mathit{phd\_student} \sqsubseteq \mathit{student};\ \ (4)\ \mathit{professor} \sqsubseteq \neg \mathit{student};\\
(5)\ \mathit{student} \sqsubseteq \mathit{bachelor\_student} \sqcup \mathit{master\_student} \sqcup \mathit{phd\_student};\\
(6)\ \geqslant 1\ \mathit{taken} \sqsubseteq \mathit{student};\ \ \geqslant 1\ \mathit{taken}^- \sqsubseteq \mathit{exam};\\
(7)\ \mathit{student} \sqsubseteq \exists \mathit{taken}.\mathit{exam};\ \ \mathit{student} \sqsubseteq\ \leqslant 10\ \mathit{taken};\ \ \mathit{exam} \sqsubseteq\ \leqslant 25\ \mathit{taken}^-;\\
(8)\ \mathit{student}(\mathit{john});\ \ \mathit{master\_student}(\mathit{mary});\ \ \mathit{exam}(\mathit{java});\ \ \mathit{taken}(\mathit{john}, \mathit{java});\\
(9)\ \mathit{john} = \mathit{john\_miller};\ \ \mathit{john} \neq \mathit{mary}.
\end{array}$$

2.2 Semantics

An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ relative to a datatype theory $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$ consists of a nonempty (*abstract*) *domain* $\Delta^{\mathcal{I}}$ disjoint from $\Delta^{\mathbf{D}}$, and a mapping $\cdot^{\mathcal{I}}$ that assigns to each atomic concept $\phi \in \mathbf{A}$ a subset of $\Delta^{\mathcal{I}}$, to each individual $o \in \mathbf{I}$ an element of $\Delta^{\mathcal{I}}$, to each abstract role $R \in \mathbf{R}_A$ a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and to each datatype role $U \in \mathbf{R}_D$ a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathbf{D}}$. We extend the mapping $\cdot^{\mathcal{I}}$ to all concepts and roles as usual (where $\#S$ denotes the cardinality of a set S):

– $(R^-)^{\mathcal{I}} = \{(y,x) \mid (x,y) \in R^{\mathcal{I}}\}$,
– $\{o_1, \ldots, o_n\}^{\mathcal{I}} = \{o_1^{\mathcal{I}}, \ldots, o_n^{\mathcal{I}}\}$, $(\neg\phi)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus \phi^{\mathcal{I}}$,
– $(\phi_1 \sqcap \phi_2)^{\mathcal{I}} = \phi_1^{\mathcal{I}} \cap \phi_2^{\mathcal{I}}$, $(\phi_1 \sqcup \phi_2)^{\mathcal{I}} = \phi_1^{\mathcal{I}} \cup \phi_2^{\mathcal{I}}$,
– $(\exists R.\phi)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y\colon (x,y) \in R^{\mathcal{I}} \wedge y \in \phi^{\mathcal{I}}\}$,
– $(\forall R.\phi)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y\colon (x,y) \in R^{\mathcal{I}} \rightarrow y \in \phi^{\mathcal{I}}\}$,
– $(\geqslant nR)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#(\{y \mid (x,y) \in R^{\mathcal{I}}\}) \geqslant n\}$,
– $(\leqslant nR)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#(\{y \mid (x,y) \in R^{\mathcal{I}}\}) \leqslant n\}$,
– $(\exists U.D)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y\colon (x,y) \in U^{\mathcal{I}} \wedge y \in D^{\mathbf{D}}\}$,
– $(\forall U.D)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y\colon (x,y) \in U^{\mathcal{I}} \rightarrow y \in D^{\mathbf{D}}\}$,
– $(\geqslant nU)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#(\{y \mid (x,y) \in U^{\mathcal{I}}\}) \geqslant n\}$,
– $(\leqslant nU)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#(\{y \mid (x,y) \in U^{\mathcal{I}}\}) \leqslant n\}$.

The *satisfaction* of an axiom F in $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ relative to $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$, denoted $\mathcal{I} \models F$, is defined as follows: (1) $\mathcal{I} \models \phi \sqsubseteq \psi$ iff $\phi^{\mathcal{I}} \subseteq \psi^{\mathcal{I}}$; (2) $\mathcal{I} \models R \sqsubseteq S$ iff $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$; (3) $\mathcal{I} \models \mathrm{Trans}(R)$ iff $R^{\mathcal{I}}$ is transitive; (4) $\mathcal{I} \models \phi(a)$ iff $a^{\mathcal{I}} \in \phi^{\mathcal{I}}$; (5) $\mathcal{I} \models R(a,b)$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$; (6) $\mathcal{I} \models U(a,v)$ iff $(a^{\mathcal{I}}, v^{\mathbf{D}}) \in U^{\mathcal{I}}$; (7) $\mathcal{I} \models a = b$ iff $a^{\mathcal{I}} = b^{\mathcal{I}}$; and (8) $\mathcal{I} \models a \neq b$ iff $a^{\mathcal{I}} \neq b^{\mathcal{I}}$. We say $\mathcal{I}$ *satisfies* the axiom F, or $\mathcal{I}$ is a *model* of F, iff $\mathcal{I} \models F$. We say $\mathcal{I}$ *satisfies* a description logic knowledge base L, or $\mathcal{I}$ is a *model* of L, denoted $\mathcal{I} \models L$, iff $\mathcal{I} \models F$ for all $F \in L$. We say L is *satisfiable* iff L has a model. An axiom F is a *logical consequence* of L, denoted $L \models F$, iff every model of L satisfies F.

Example 2.2 (University Database cont'd). Consider again the description logic knowledge base L of Example 2.1. It is not difficult to verify that L is satisfiable, and that

$professor \sqsubseteq \neg master\_student$, $student(mary)$, $(bachelor\_student \sqcup master\_student \sqcup phd\_student)(john)$, and $taken(john, java)$ are logical consequences of L.

3 Tightly Coupled Disjunctive DL-Programs

In this section, we recall the *tightly coupled* approach to *disjunctive description logic programs* (or simply *disjunctive dl-programs*) $KB = (L, P)$ under the answer set semantics from [10], where KB consists of a description logic knowledge base L and a disjunctive logic program P. The semantics of KB is defined in a modular way as in [8,9], but it allows for a much tighter coupling of L and P. Note that we do not assume any structural separation between the vocabularies of L and P. The main idea behind the semantics of KB is to interpret P relative to Herbrand interpretations that are compatible with L, while L is interpreted relative to general interpretations over a first-order domain. Thus, we modularly combine the standard semantics of logic programs and of description logics, which allows for building on the standard techniques and results of both areas. As another advantage, the novel disjunctive dl-programs are decidable, even when their components of logic programs and description logic knowledge bases are both very expressive. See especially [10] for further details on the novel approach to disjunctive dl-programs and for a detailed comparison to related works.

3.1 Syntax

We assume a first-order vocabulary Φ with finite nonempty sets of constant and predicate symbols, but no function symbols. We use Φ_c to denote the set of all constant symbols in Φ. We also assume a set of data values $\mathbf{V}$ (relative to a datatype theory $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$) and pairwise disjoint (denumerable) sets $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, and $\mathbf{I}$ of atomic concepts, abstract roles, datatype roles, and individuals, respectively, as in Section 2. We assume that (i) Φ_c is a subset of $\mathbf{I} \cup \mathbf{V}$, and that (ii) Φ and $\mathbf{A}$ (resp., $\mathbf{R}_A \cup \mathbf{R}_D$) may have unary (resp., binary) predicate symbols in common.

Let $\mathcal{X}$ be a set of variables. A *term* is either a variable from $\mathcal{X}$ or a constant symbol from Φ. An *atom* is of the form $p(t_1, \ldots, t_n)$, where p is a predicate symbol of arity $n \geqslant 0$ from Φ, and $t_1, \ldots, t_n$ are terms. A *literal* l is an atom p or a default-negated atom $not\, p$. Note that the default-negated atom $not\, p$ refers to the lack of evidence about the truth of the atom p, and thus has a different meaning than the classically negated atom $\neg p$, which refers to the presence of knowledge asserting the falsehood of the atom p. A *disjunctive rule* (or simply *rule*) r is an expression of the form

$$\alpha_1 \vee \cdots \vee \alpha_k \leftarrow \beta_1, \ldots, \beta_n, not\, \beta_{n+1}, \ldots, not\, \beta_{n+m}\,, \tag{1}$$

where $\alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_{n+m}$ are atoms and $k, m, n \geqslant 0$. We call $\alpha_1 \vee \cdots \vee \alpha_k$ the *head* of r, while the conjunction $\beta_1, \ldots, \beta_n, not\, \beta_{n+1}, \ldots, not\, \beta_{n+m}$ is its *body*. We define $H(r) = \{\alpha_1, \ldots, \alpha_k\}$ and $B(r) = B^+(r) \cup B^-(r)$, where $B^+(r) = \{\beta_1, \ldots, \beta_n\}$ and $B^-(r) = \{\beta_{n+1}, \ldots, \beta_{n+m}\}$. We say that r is a *fact* iff $n = m = 0$. A *disjunctive (logic) program* P is a finite set of disjunctive rules of the form (1). We say that P is *positive* iff $m = 0$ for all disjunctive rules (1) in P. We say that P is a *normal (logic) program* iff $k \leqslant 1$ for all disjunctive rules (1) in P.

A *ground instance* of a rule r is obtained from r by replacing every variable that occurs in r by a constant symbol from Φ_c. We denote by $\mathit{ground}(P)$ the set of all ground instances of rules in P. The *Herbrand base* relative to Φ, denoted HB_Φ, is the set of all ground atoms constructed with constant and predicate symbols from Φ. A disjunctive program P is *acyclic* iff a mapping κ from HB_Φ to the non-negative integers exists such that $\kappa(p) > \kappa(q)$ for all $p, q \in HB_\Phi$ where p (resp., q) occurs in the head (resp., body) of some rule in $\mathit{ground}(P)$. A *(local) stratification* of P is a mapping $\lambda\colon HB_\Phi \to \{0, 1, \ldots, k\}$ such that $\lambda(\alpha) \geqslant \lambda(\beta)$ (resp., $\lambda(\alpha) > \lambda(\beta)$) for each $r \in \mathit{ground}(P)$, $\alpha \in H(r)$, and $\beta \in B^+(r)$ (resp., $\beta \in B^-(r)$), where $k \geqslant 0$ is the *length* of λ. A disjunctive program P is *(locally) stratified* iff it has a stratification λ of some length $k \geqslant 0$. Note that every acyclic disjunctive program is also stratified.

A *tightly coupled disjunctive description logic program* (or simply *disjunctive dl-program*) $KB = (L, P)$ consists of a description logic knowledge base L and a disjunctive program P. We say that $KB = (L, P)$ is *positive* iff P is positive. We say that $KB = (L, P)$ is a *normal dl-program* iff P is a normal program.

Example 3.1 (University Database cont'd). Consider the disjunctive dl-program $KB = (L, P)$, where L is the description logic knowledge base from Example 2.1, and P is the following set of rules, which express that (1) Bill is either a master student or a Ph.D. student (which is encoded by a rule that has the form of a disjunction of ground atoms), and Mary has taken an exam in unix; (2) every student who has taken an exam in knowledge bases is either a master student or a Ph.D. student (which is encoded by a rule with a disjunction in its head); (3) every student who is not known to be a master student or a Ph.D. student is a bachelor student (which is encoded by a rule with default negations in its body); (4) the relation of being a prerequisite enjoys the transitive property; (5) if a student has taken an exam, then he/she has taken every exam that is a prerequisite for it; and (6) unix is a prerequisite for java, and java is a prerequisite for programming languages:

$$
\begin{array}{l}
(1)\ \mathit{master\_student}(\mathit{bill}) \vee \mathit{phd\_student}(\mathit{bill});\ \ \mathit{taken}(\mathit{mary}, \mathit{unix});\\
(2)\ \mathit{master\_student}(X) \vee \mathit{phd\_student}(X) \ \leftarrow\ \mathit{taken}(X, \mathit{knowledge\_bases});\\
(3)\ \mathit{bachelor\_student}(X) \ \leftarrow\\
\qquad\qquad \mathit{student}(X),\ \mathit{not}\ \mathit{master\_student}(X),\ \mathit{not}\ \mathit{phd\_student}(X);\\
(4)\ \mathit{prerequisite}(X, Z) \ \leftarrow\ \mathit{prerequisite}(X, Y),\ \mathit{prerequisite}(Y, Z);\\
(5)\ \mathit{taken}(X, Z) \ \leftarrow\ \mathit{taken}(X, Y),\ \mathit{prerequisite}(Z, Y);\\
(6)\ \mathit{prerequisite}(\mathit{unix}, \mathit{java});\ \ \mathit{prerequisite}(\mathit{java}, \mathit{programming\_languages}).
\end{array}
$$

The above disjunctive dl-program also shows the advantages and flexibility of the tight coupling between rules and ontologies (compared to the loose coupling in [8,9]): Observe that the predicate symbol taken in P is also a role in L, and it freely occurs in both rule bodies and rule heads in P (which is both not possible in [8,9]). Moreover, we can easily use L to express additional constraints on the predicate symbols in P. For example, we may use the two axioms $\geqslant 1\ \mathit{prerequisite} \sqsubseteq \mathit{exam}$ and $\geqslant 1\ \mathit{prerequisite}^{-1} \sqsubseteq \mathit{exam}$ in L to express that $\mathit{prerequisite}$ in P relates only exams.

3.2 Semantics

We now define the answer set semantics of (tightly coupled) disjunctive dl-programs as a generalization of the answer set semantics of ordinary disjunctive logic programs. In the sequel, let $KB = (L, P)$ be a disjunctive dl-program.

We use DL_Φ to denote the set of all ground atoms in HB_Φ that are constructed from atomic concepts in $\mathbf{A}$, abstract roles in $\mathbf{R}_A$, and datatype roles in $\mathbf{R}_D$.

An *interpretation* I is any subset of HB_Φ. Informally, every such I represents the Herbrand interpretation in which all $a \in I$ (resp., $a \in HB_\Phi - I$) are true (resp., false). We say an interpretation I is a *model* of a description logic knowledge base L, denoted $I \models L$, iff $L \cup I \cup \{\neg a \mid a \in HB_\Phi - I\}$ is satisfiable. We say I is a *model* of a ground atom $a \in HB_\Phi$, or I *satisfies* a, denoted $I \models a$, iff $a \in I$. We say I is a *model* of a ground rule r, denoted $I \models r$, iff $I \models \alpha$ for some $\alpha \in H(r)$ whenever $I \models B(r)$, that is, $I \models \beta$ for all $\beta \in B^+(r)$ and $I \not\models \beta$ for all $\beta \in B^-(r)$. We say I is a *model* of a set of rules P iff $I \models r$ for every $r \in ground(P)$. We say I is a *model* of a disjunctive dl-program $KB = (L, P)$, denoted $I \models KB$, iff I is a model of both L and P.

We now define the answer set semantics of disjunctive dl-programs by generalizing the ordinary answer set semantics of disjunctive logic programs. We generalize the definition via the FLP-reduct [27] (which is equivalent to the definition via the Gelfond-Lifschitz reduct [28]). Given a disjunctive dl-program $KB = (L, P)$, the *FLP-reduct* of KB relative to an interpretation $I \subseteq HB_\Phi$, denoted KB^I, is the disjunctive dl-program (L, P^I), where P^I is the set of all $r \in ground(P)$ such that $I \models B(r)$. An interpretation $I \subseteq HB_\Phi$ is an *answer set* of KB iff I is a minimal model of KB^I. A disjunctive dl-program KB is *consistent* (resp., *inconsistent*) iff it has an (resp., no) answer set.

Example 3.2 (University Database cont'd). Consider again the disjunctive dl-program $KB = (L, P)$ of Example 3.1. It is not difficult to verify that KB is consistent and that it has two answer sets, which contain both in particular the ground atoms

$$\begin{array}{l} student(john),\ student(john\_miller),\ bachelor\_student(john), \\ bachelor\_student(john\_miller),\ master\_student(mary),\ student(mary), \\ student(bill),\ exam(java),\ taken(john, java),\ taken(john\_miller, java), \\ taken(mary, unix),\ prerequisite(java, programming\_languages), \\ prerequisite(unix, java),\ prerequisite(unix, programming\_languages), \end{array}$$

as well as either $master\_student(bill)$ or $phd\_student(bill)$.

We finally define the notion of *cautious* (resp., *brave*) *reasoning* from disjunctive dl-programs under the answer set semantics as follows. A ground atom $a \in HB_\Phi$ is a *cautious* (resp., *brave*) *consequence* of a disjunctive dl-program KB under the answer set semantics iff every (resp., some) answer set of KB satisfies a.

Example 3.3 (University Database cont'd). Consider again the disjunctive dl-program $KB = (L, P)$ of Example 3.1. By Example 3.2, the ground atom $student(bill)$ is a cautious consequence of KB, while $phd\_student(bill)$ is a brave consequence of KB.

3.3 Semantic Properties

We now summarize some important semantic properties of disjunctive dl-programs under the above answer set semantics. In the ordinary case, every answer set of a disjunctive program P is also a minimal model of P, and the converse holds when P is positive. This result holds also for disjunctive dl-programs.

As another important semantic property, the answer set semantics of disjunctive dl-programs faithfully extends its ordinary counterpart. That is, the answer set semantics of a disjunctive dl-program with empty description logic knowledge base coincides with the ordinary answer set semantics of its disjunctive program. Furthermore, the answer set semantics of disjunctive dl-programs also faithfully extends (from the perspective of answer set programming) the first-order semantics of description logic knowledge bases. That is, a ground atom $\alpha \in HB_\Phi$ is true in all answer sets of a positive disjunctive dl-program $KB = (L, P)$ iff α is true in all first-order models of $L \cup ground(P)$. In particular, a ground atom $\alpha \in HB_\Phi$ is true in all answer sets of $KB = (L, \emptyset)$ iff α is true in all first-order models of L. Note that this result holds also when α is a ground formula constructed from HB_Φ using the operators $\wedge$ and $\vee$.

Another important feature of disjunctive dl-programs $KB = (L, P)$ concerns the *unique name assumption*, which says that any two distinct constant symbols in Φ_c represent two distinct domain objects (and which is quite usual in logic programming). It turns out that we do not have to make the unique name assumption here, since the description logic knowledge base of a disjunctive dl-program may very well contain or imply equalities between individuals. Intuitively, since we have no unique name assumption in L, we also do not have to make the unique name assumption in P.

Example 3.4. The unique answer set of the disjunctive dl-program $KB = (L, P) = (\{a = b\}, \{p(a)\})$, where $a, b \in \Phi_c \cap \mathbf{I}$ and $p \in \Phi \cap \mathbf{A}$, contains both ground atoms $p(a)$ and $p(b)$, since L contains the equality axiom $a = b$, and P contains the fact $p(a)$.

The tight coupling of ontologies and rules semantically behaves very differently from the loose coupling. This makes the former more (and the latter less) suitable for representing ontology mappings (see Section 5) and for combining sophisticated reasoning formalisms from Artificial Intelligence (such as reasoning about actions) with ontologies (see Section 6). The following example illustrates this difference.

Example 3.5 (Client Database). The normal dl-program $KB = (L, P)$, where

$$\begin{aligned} L &= \{\mathit{person}(a),\ \mathit{person} \sqsubseteq \mathit{male} \sqcup \mathit{female}\} \text{ and} \\ P &= \{\mathit{client}(X) \leftarrow \mathit{male}(X),\ \mathit{client}(X) \leftarrow \mathit{female}(X)\} \end{aligned}$$

implies $\mathit{client}(a)$, while the normal dl-program $KB' = (L', P')$ as in [8,9]

$$\begin{aligned} L' &= \{\mathit{person}(a),\ \mathit{person} \sqsubseteq \mathit{male} \sqcup \mathit{female}\} \text{ and} \\ P' &= \{\mathit{client}(X) \leftarrow DL[\mathit{male}](X),\ \mathit{client}(X) \leftarrow DL[\mathit{female}](X)\} \end{aligned}$$

does *not* imply $\mathit{client}(a)$, since the two queries are evaluated independently from each other, and neither $\mathit{male}(a)$ nor $\mathit{female}(a)$ follows from L'. To obtain the conclusion $\mathit{client}(a)$ in [8,9], one has to directly use the rule $\mathit{client}(X) \leftarrow DL[\mathit{male} \sqcup \mathit{female}](X)$.

4 Tightly Coupled Probabilistic DL-Programs

In this section, we present a *tightly coupled* approach to *probabilistic disjunctive description logic programs* (or simply *probabilistic dl-programs*) under the answer set semantics. Differently from [17] (in addition to being a tightly coupled approach), the probabilistic dl-programs here also allow for disjunctions in rule heads. Similarly to the probabilistic dl-programs in [17], they are defined as a combination of dl-programs with the ICL [19], but using the tightly coupled disjunctive dl-programs of [10] (see Section 3), rather than the loosely coupled dl-programs of [8,9]. The ICL is based on ordinary acyclic logic programs P under different "choices", where every choice along with P produces a first-order model, and one then obtains a probability distribution over the set of all first-order models by placing a probability distribution over the different choices. We use the tightly integrated disjunctive dl-programs under the answer set semantics of [10], instead of ordinary acyclic logic programs under their canonical semantics (which coincides with their answer set semantics). We first introduce the syntax of probabilistic dl-programs and then their answer set semantics.

Observe that tightly coupled probabilistic dl-programs generalize a simplified version of Poole's (multi-agent) ICL, where we have only "nature" as agent, and the Herbrand base is finite. Since the latter can represent (discrete and finite) Bayesian networks [29] and (binary and finite) structural causal models [20], it thus follows immediately that tightly coupled probabilistic dl-programs can also represent (discrete and finite) Bayesian networks and (binary and finite) structural causal models. Furthermore, since Poole's ICL can represent influence diagrams, Markov decision processes, and normal form games [19], it follows that a multi-agent version of tightly coupled probabilistic dl-programs (where we additionally have a finite set of agents (including "nature"), which each control certain alternatives on the choice space, and the probability on the choice space only concerns the alternatives controlled by nature), can also represent influence diagrams, Markov decision processes, and normal form games.

4.1 Syntax

We now define the syntax of (tightly coupled) probabilistic dl-programs and probabilistic queries to them. We first introduce choice spaces and probabilities on choice spaces.

A *choice space* $\mathcal{C}$ is a set of pairwise disjoint and nonempty sets $A \subseteq HB_\Phi - DL_\Phi$. Any $A \in \mathcal{C}$ is an *alternative* of $\mathcal{C}$ and any element $a \in A$ an *atomic choice* of $\mathcal{C}$. Intuitively, every alternative $A \in \mathcal{C}$ represents a random variable and every atomic choice $a \in A$ one of its possible values. A *total choice* of $\mathcal{C}$ is a set $B \subseteq HB_\Phi$ such that $|B \cap A| = 1$ for all $A \in \mathcal{C}$ (and thus $|B| = |\mathcal{C}|$). Intuitively, every total choice B of $\mathcal{C}$ represents an assignment of values to all the random variables. A *probability* μ on a choice space $\mathcal{C}$ is a probability function on the set of all total choices of $\mathcal{C}$. Intuitively, every probability μ is a probability distribution over the set of all joint variable assignments. Since $\mathcal{C}$ and all its alternatives are finite, μ can be defined by (i) a mapping $\mu\colon \bigcup \mathcal{C} \rightarrow [0,1]$ such that $\sum_{a \in A} \mu(a) = 1$ for all $A \in \mathcal{C}$, and (ii) $\mu(B) = \Pi_{b \in B} \mu(b)$ for all total choices B of $\mathcal{C}$. Intuitively, (i) defines a probability over the values of each random variable of $\mathcal{C}$, and (ii) assumes independence between the random variables.

Example 4.1 (University Database cont'd). A choice space $\mathcal{C}$ for the University Database may be defined by $\mathcal{C} = \{\{\mathit{choice}_u,\ \mathit{not\_choice}_u\},\ \{\mathit{choice}_o,\ \mathit{not\_choice}_o\}\}$, which represents two random variables X_u and X_o with the binary domains $\{\mathit{choice}_u,\ \mathit{not\_choice}_u\}$ and $\{\mathit{choice}_o,\ \mathit{not\_choice}_o\}$, respectively. A probability μ on $\mathcal{C}$ may be given as follows. We first define μ for the atomic choices by $\mu\colon\ \mathit{choice}_u,\ \mathit{not\_choice}_u,\ \mathit{choice}_o,\ \mathit{not\_choice}_o \mapsto 0.9,\ 0.1,\ 0.7,\ 0.3$, and then we naturally extend μ to all total choices by assuming independence between the alternatives. For example, the total choice $B = \{\mathit{choice}_u,\ \mathit{not\_choice}_o\}$ (which represents the joint variable assignment $X_u = \mathit{choice}_u$, $X_o = \mathit{not\_choice}_o$) has the probability $\mu(B) = 0.9 \cdot 0.3 = 0.27$.

Observe that, as in Poole's ICL [19], all atomic choices in the alternatives of choice spaces are ground. But one may also use non-ground atomic choices in the alternatives by assuming that every non-ground alternative abbreviates all its ground instances, which allows for a more compact representation of choice spaces and their probabilities.

Example 4.2 (University Database cont'd). The non-ground choice space $\mathcal{C} = \{\{p(X),\ \mathit{not\_p}(X)\}\}$ along with the probability $\mu\colon\ p(X),\ \mathit{not\_p}(X) \mapsto 0.9,\ 0.1$ (assuming independence between the alternatives) abbreviates the ground choice space $\mathcal{C}' = \{\{p(c),\ \mathit{not\_p}(c)\} \mid c \in \Phi_c\}$ along with the probability $\mu'\colon\ p(c),\ \mathit{not\_p}(c) \mapsto 0.9,\ 0.1$ for every $c \in \Phi_c$ (assuming independence between the alternatives). Informally, for every constant $c \in \Phi_c$, we have one random variable X_c with the binary domain $\{p(c),\ \mathit{not\_p}(c)\}$ and the probability $\mu'\colon\ p(c),\ \mathit{not\_p}(c) \mapsto 0.9,\ 0.1$ on it. Similarly, one may also have one random variable X_c for every constant $c \in \Phi_s$, where Φ_s is a certain subset of Φ_c only, such as the set of all constants in Φ_c encoding exams.

A *tightly coupled probabilistic disjunctive description logic program* (or simply *probabilistic dl-program*) $\mathit{KB} = (L, P, \mathcal{C}, \mu)$ consists of a (tightly coupled) disjunctive dl-program (L, P), a choice space $\mathcal{C}$ such that no atomic choice in $\mathcal{C}$ coincides with the head of any rule in $\mathit{ground}(P)$, and a probability μ on $\mathcal{C}$. Intuitively, since the total choices of $\mathcal{C}$ select subsets of P, and μ is a probability distribution on the total choices of $\mathcal{C}$, every probabilistic dl-program is the compact representation of a probability distribution on a finite set of disjunctive dl-programs. Observe here that P is fully general (it may have disjunctions in rule heads and default negations in rule bodies, and it may not necessarily be acyclic or stratified). We say KB is *normal* iff P is normal (see Section 3.1). An *event* α is any Boolean combination of atoms (that is, constructed from atoms via the Boolean operators "$\wedge$" and "$\neg$"). A *conditional event* is of the form $\beta|\alpha$, where α and β are events. A *probabilistic query* to KB has the form $\exists(\beta|\alpha)[r, s]$, where $\beta|\alpha$ is a conditional event, and r and s are either two variables or two reals from $[0, 1]$. Note that dealing with probabilities of conditional events in both knowledge bases and queries is commonly regarded as being an indispensable feature of probabilistic reasoning formalisms in Artificial Intelligence [30,31,32]. In our approach, conditional events are explicitly allowed in queries, and probabilities of conditional events in the knowledge base can be modeled in the same way as in Poole's ICL [19].

Example 4.3 (University Database cont'd). A probabilistic dl-program $\mathit{KB} = (L, P, \mathcal{C}, \mu)$ is given by the choice space $\mathcal{C}$ and the probability μ on $\mathcal{C}$ of Example 4.1, and the disjunctive dl-program (L, P), which is nearly the same as the one given in Example 3.1, except that P now also contains the following two (probabilistic) rules:

$taken(X, operating\_systems) \leftarrow master\_student(X), taken(X, unix), choice_u;$
$taken(X, databases) \leftarrow master\_student(X), taken(X, operating\_systems), choice_o.$

Here, the new (probabilistic) rules express that if a master student has taken an exam in $unix$ (resp., $operating\_systems$), then there is a probability of 0.9 (resp., 0.7) that he/she has also taken an exam in $operating\_systems$ (resp., $databases$). Note that probabilistic facts can be encoded by rules with only atomic choices in their body.

Querying for the entailed tight interval for the probability that Bill is a student (resp., master student) can be expressed by the probabilistic query $\exists(student(bill))[R, S]$ (resp., $\exists(master\_student(bill))[R, S]$). Querying for the entailed tight interval for the probability that Mary has taken an exam in databases (resp., Mary has taken an exam in databases given that she has taken an exam in operating systems) can be expressed by the probabilistic query $\exists(taken(mary, databases))[R, S]$ (resp., $\exists(taken(mary, databases)|taken(mary, operating\_systems))[R, S]$). In the latter case, we model a conditioning of all probability distributions that are compatible with KB on the observation that Mary has taken an exam in operating systems (which is not encoded in KB). Querying for the exams that John has taken along with their tight probability intervals can be expressed by the probabilistic query $\exists(taken(john, E))[R, S]$.

4.2 Semantics

We now define an answer set semantics of (tightly coupled) probabilistic dl-programs, and we introduce the notions of consistency, consequence, tight consequence, and correct and tight answers for probabilistic queries to probabilistic dl-programs.

Given a probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$, a *probabilistic interpretation* Pr is a probability function on the set of all $I \subseteq HB_\Phi$. We say Pr is an *answer set* of KB iff (i) every interpretation $I \subseteq HB_\Phi$ with $Pr(I) > 0$ is an answer set of $(L, P \cup \{p \leftarrow \mid p \in B\})$ for some total choice B of $\mathcal{C}$, and (ii) $Pr(\bigwedge_{p \in B} p) = \sum_{I \subseteq HB_\Phi,\, B \subseteq I} Pr(I) = \mu(B)$ for every total choice B of $\mathcal{C}$. Informally, Pr is an answer set of $KB = (L, P, \mathcal{C}, \mu)$ iff (i) every interpretation $I \subseteq HB_\Phi$ of positive probability under Pr is an answer set of the dl-program (L, P) under some total choice B of $\mathcal{C}$, and (ii) Pr coincides with μ on the total choices B of $\mathcal{C}$. We say KB is *consistent* iff it has an answer set Pr.

Example 4.4 (University Database cont'd). Let the probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$ be as in Example 4.3. Let S_1, S_2, S_3, and S_4 be answer sets of $(L, P \cup \{choice_u, choice_o\})$, $(L, P \cup \{choice_u, not\_choice_o\})$, $(L, P \cup \{not\_choice_u, choice_o\})$, and $(L, P \cup \{not\_choice_u, not\_choice_o\})$, respectively. Then, $Pr\colon S_1, S_2, S_3, S_4 \mapsto 0.63, 0.27, 0.07, 0.03$ is an answer set of KB, which also shows that KB is consistent.

If additionally S_1', S_2', S_3', and S_4' are answer sets of $(L, P \cup \{choice_u, choice_o\})$, $(L, P \cup \{choice_u, not\_choice_o\})$, $(L, P \cup \{not\_choice_u, choice_o\})$, and $(L, P \cup \{not\_choice_u, not\_choice_o\})$, respectively, different from S_1, S_2, S_3, and S_4, respectively, then $Pr\colon S_1, S_1', S_2, S_2', S_3, S_3', S_4, S_4' \mapsto 0.63, 0, 0.22, 0.05, 0.06, 0.01, 0.02, 0.01$ is an answer set of KB.

Given a ground event α, the *probability* of α in a probabilistic interpretation Pr, denoted $Pr(\alpha)$, is the sum of all $Pr(I)$ such that $I \subseteq HB_\Phi$ and $I \models \alpha$. Given two ground events α and β, and two reals $l, u \in [0, 1]$, we say $(\beta|\alpha)[l, u]$ is a *consequence* of a consistent probabilistic dl-program KB under the answer set semantics, denoted $KB \mathrel{\|\!\!\sim} (\beta|\alpha)[l,u]$, iff $Pr(\alpha \wedge \beta) \,/\, Pr(\alpha) \in [l,u]$ for all answer sets Pr of KB with $Pr(\alpha) > 0$. We say $(\beta|\alpha)[l,u]$ is a *tight consequence* of a consistent probabilistic dl-program KB under the answer set semantics, denoted $KB \mathrel{\|\!\!\sim}_{tight} (\beta|\alpha)[l,u]$, iff l (resp., u) is the infimum (resp., supremum) of $Pr(\alpha \wedge \beta) \,/\, Pr(\alpha)$ subject to all answer sets Pr of KB with $Pr(\alpha) > 0$. Note that this infimum (resp., supremum) is naturally defined as 1 (resp., 0) iff no such Pr exists. The *tight answer* (resp., *correct answer*) for a probabilistic query $Q = \exists(\beta|\alpha)[r,s]$ to KB under the answer set semantics, where r and s are two variables (resp., two reals from $[0,1]$), is the set of all ground substitutions θ (for the variables in Q) such that $(\beta|\alpha)[r,s]\theta$ is a tight consequence (resp., consequence) of KB under the answer set semantics. For ease of presentation, since tight (and correct) answers for probabilistic queries $Q = \exists(\beta|\alpha)[r,s]$ with non-ground $\beta|\alpha$ are easily reducible to tight answers for probabilistic queries $Q = \exists(\beta|\alpha)[r,s]$ with ground $\beta|\alpha$,[1] we consider only the latter type of probabilistic queries in the following.

Example 4.5 (University Database cont'd). Consider again the probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$ of Example 4.3. It is not difficult to verify that the tight answers to the probabilistic queries $\exists(\mathit{student}(\mathit{bill}))[R, S]$ and $\exists(\mathit{master\_student}(\mathit{bill}))[R, S]$ are given by $\theta = \{R/1, S/1\}$ and $\theta = \{R/0, S/1\}$, respectively. Furthermore, the tight answers to the two probabilistic queries $\exists(\mathit{taken}(\mathit{mary}, \mathit{databases}))[R, S]$ and $\exists(\mathit{taken}(\mathit{mary}, \mathit{databases})|\mathit{taken}(\mathit{mary}, \mathit{operating\_systems}))[R, S]$ are given by $\theta = \{R/0.63, S/0.63\}$ and $\theta = \{R/0.7, S/0.7\}$, respectively.

5 Representing Ontology Mappings

In this section, we show that tightly coupled probabilistic dl-programs are well-suited for representing ontology mappings. We first describe the requirements of a formal language for representing and combining correspondences produced by different matching components or systems. We then show how tightly coupled disjunctive dl-programs can be used for representing (possibly inconsistent) ontology mappings (without confidence values). We finally show how tightly coupled probabilistic dl-programs can be used for representing (possibly inconsistent) ontology mappings with confidence values.

5.1 Representation Requirements

The problem of ontology matching can be defined as follows [23]. Ontologies are theories encoded in a certain language L. In this work, we assume that ontologies are encoded in OWL DL or OWL Lite. Further, as in [23], we assume that the use of mappings between ontologies is restricted to certain constructs in each ontology that can be linked to parts of other ontologies. We call these constructs within one ontology matchable elements and denote them by $Q(O)$. In principle, $Q(O)$ can be any valid expressions in the

[1] Every probabilistic query $Q = \exists(\beta|\alpha)[r,s]$ with non-ground $\beta|\alpha$ is reduced to all ground instances $Q\theta$ of Q relative to Φ_c: The tight answer to $Q = \exists(\beta|\alpha)[r,s]$, where r and s are two variables, is given by the set of all $\theta \circ \theta'$ such that (i) θ is a ground substitution for $\beta|\alpha$ and (ii) θ' is the tight answer to $Q\theta$. As Φ_c is finite, also the set of all ground instances of Q is finite.

language L (e.g., predicate names or whole formulas). In practice, $Q(O)$ depends on the semantics of the concrete mapping language used. If we encode a mapping in standard description logics, the matchable elements in the connected ontologies correspond to class expressions over the signature of the respective ontology. If we use disjunctive Datalog, the set of matchable elements in the source ontology consists of all possible rule bodies (as defined in Section 3) over the terminology of the source ontology, while the set of matchable elements in the target ontology consists of all possible rule heads (as defined in Section 3) over the terminology of the target ontology.

Given two ontologies O_1 and O_2, the task of matching is now to determine correspondences between the matchable elements in the two ontologies. Correspondences are 5-tuples (id, e, e', r, p) such that

- id is a unique identifier for referring to the correspondence;
- $e \in Q(O_1)$ and $e' \in Q(O_2)$ are matchable elements from the two ontologies;
- $r \in R$ is a semantic relation (in this work, we consider the case where the semantic relation can be interpreted as an implication);
- p is a degree of confidence in the correctness of the correspondence.

From the above general description of automatically generated correspondences between ontologies, we can derive a number of requirements for a formal language for representing the results of multiple matchers as well as the contained uncertainties:

– *Tight integration of mapping and ontology language:* The semantics of the language used to represent the correspondences between elements in different ontologies has to be tightly integrated with the semantics of the ontology language used (in this case OWL). This is important if we want to use the correspondences to reason across different ontologies in a semantically coherent way. In particular, this means that the interpretation of the mapped elements depends on the definitions in the ontologies.

– *Support for mappings refinement:* The language should be expressive enough to allow the user to refine oversimplifying correspondences suggested by the matching system. This is important to be able to provide a more precise account of the true semantic relation between elements in the mapped ontologies. In particular, this requires the ability to describe correspondences that include several elements from the two ontologies.

– *Support for repairing inconsistencies:* Inconsistent mappings are a major problem for the combined use of ontologies because they can cause inconsistencies in the mapped ontologies. These inconsistencies can make logical reasoning impossible, since everything can be derived from an inconsistent ontology. The mapping language should be able to represent and reason about inconsistent mappings in an approximate fashion.

– *Representation and combination of confidence:* The confidence values provided by matching systems are an important indicator for the uncertainty that has to be taken into account. The mapping representation language should be able to use these confidence values when reasoning with mappings. In particular, it should be able to represent the confidence in a mapping rule and to combine confidence values on a sound formal basis.

– *Decidability and efficiency of instance reasoning:* An important use of ontology mappings is the exchange of data across different ontologies. In particular, we normally want to be able to ask queries using the vocabulary of one ontology and receive answers that do not only consist of instances of this ontology but also of ontologies connected through ontology mappings. To support this, query answering in the combined formalism consisting of ontology language and mapping language has to be decidable and there should be efficient algorithms for answering queries.

Throughout this section, we use real data form the Ontology Alignment Evaluation Initiative[2] to illustrate the different aspects of mapping representation. In particular, we use examples from the benchmark and the conference data set. The benchmark dataset consists of five OWL ontologies (tests 101 and 301–304) describing scientific publications and related information. The conference dataset consists of about 10 OWL ontologies describing concepts related to conference organization and management. In both cases, we give examples of mappings that have been created by the participants of the 2006 evaluation campaign. In particular, we use mappings created by state-of-the-art ontology matching systems like falcon and hmatch.

5.2 Deterministic Ontology Mappings

We now show how tightly coupled disjunctive dl-programs $KB = (L, P)$ can be used for representing (possibly inconsistent) mappings (without confidence values) between two ontologies. Intuitively, L encodes the union of the two ontologies, while P encodes the mappings between the ontologies, where disjunctions in rule heads and nonmonotonic negations in rule bodies in P can be used to resolve inconsistencies.

Tightly coupled disjunctive dl-programs $KB = (L, P)$ naturally represent two heterogeneous ontologies O_1 and O_2, and mappings between O_1 and O_2 as follows. The description logic knowledge base L is the union of two independent description logic knowledge bases L_1 and L_2, which encode the ontologies O_1 and O_2, respectively. Here, we assume that L_1 and L_2 have signatures $\mathbf{A}_1$, $\mathbf{R}_{A,1}$, $\mathbf{R}_{D,1}$, $\mathbf{I}_1$ and $\mathbf{A}_2$, $\mathbf{R}_{A,2}$, $\mathbf{R}_{D,2}$, $\mathbf{I}_2$, respectively, such that $\mathbf{A}_1 \cap \mathbf{A}_2 = \emptyset$, $\mathbf{R}_{A,1} \cap \mathbf{R}_{A,2} = \emptyset$, $\mathbf{R}_{D,1} \cap \mathbf{R}_{D,2} = \emptyset$, and $\mathbf{I}_1 \cap \mathbf{I}_2 = \emptyset$. Note that this can easily be achieved for any pair of ontologies by a suitable renaming. A mapping between elements e_1 and e_2 from L_1 and L_2, respectively, is then represented by a simple rule $e_2(\mathbf{x}) \leftarrow e_1(\mathbf{x})$ in P, where $e_1 \in \mathbf{A}_1 \cup \mathbf{R}_{A,1} \cup \mathbf{R}_{D,1}$, $e_2 \in \mathbf{A}_2 \cup \mathbf{R}_{A,2} \cup \mathbf{R}_{D,2}$, and $\mathbf{x}$ is a suitable variable vector. Informally, such a rule encodes that every instance of (the concept or role) e_1 in O_1 is also an instance of (the concept or role) e_2 in O_2. Note that this can be easily extended to con- and disjunctions of atoms $e_1(\mathbf{x})$ and $e_2(\mathbf{x})$, respectively. Note also that demanding the signatures of L_1 and L_2 to be disjoint guarantees that the rule base that represents mappings between different ontologies is stratified as long as there are no cyclic mappings. Note furthermore that the restriction to such simple mapping rules is not imposed by us but by the limitations of the matchers used, which are shared by most matchers existing nowadays.

Example 5.1. Taking an example from the conference data set of the OAEI challenge 2006, we find e.g. the following mappings that have been created by the hmatch system for mapping the CRS Ontology (O_1) on the EKAW Ontology (O_2):

$$O_2: \mathit{EarlyRegisteredParticipant}(X) \leftarrow O_1: \mathit{Participant}(X);$$
$$O_2: \mathit{LateRegisteredParticipant}(X) \leftarrow O_1: \mathit{Participant}(X).$$

[2] http://oaei.ontologymatching.org/2006/

Informally, these two mapping relationships express that every instance of the concept *Participant* of the ontology O_1 is also an instance of the concepts *EarlyRegistered-Participant* and *LateRegisteredParticipant*, respectively, of the ontology O_2.

We now encode the two ontologies and the mappings by a tightly coupled disjunctive dl-program $KB = (L, P)$, where L is the union of two description logic knowledge bases L_1 and L_2, encoding the ontologies O_1 and O_2, respectively, and P encodes the mappings. However, we cannot directly use the two mapping relationships as two rules in P, since this would introduce an inconsistency in KB. More specifically, recall that a model of KB has to satisfy both L and P. Here, the two mapping relationships interpreted as rules in P would require that if there is a participant Alice ($\mathit{Participant}(\mathit{alice})$) in the ontology O_1, an answer set of KB contains both $\mathit{EarlyRegisteredParticipant}(\mathit{alice})$ and $\mathit{LateRegisteredParticipant}(\mathit{alice})$ at the same time. Such an answer set, however, is invalidated by the ontology O_2, which requires the concepts *EarlyRegisteredParticipant* and *LateRegisteredParticipant* to be disjoint. Therefore, these mappings are useless, since they do not actively participate in the creation of any model of KB.

In [33], we present a method for detecting such inconsistent mappings. There are different approaches for resolving this inconsistency. The most straightforward one is to drop mappings until no inconsistency is present any more. Peng and Xu [34] have proposed a more suitable method for dealing with inconsistencies in terms of a relaxation of the mappings. In particular, they propose to replace a number of conflicting mappings by a single mapping that includes a disjunction of the conflicting concepts. In the example above, we would replace the two mapping rules by the following one:

$$\begin{aligned}&O_2\colon \mathit{EarlyRegisteredParticipant}(X) \vee \\ &\qquad O_2\colon \mathit{LateRegisteredParticipant}(X) \leftarrow O_1\colon \mathit{Participant}(X).\end{aligned}$$

This new mapping rule can be represented in our framework and resolves the inconsistency. More specifically, for a particular participant Alice ($\mathit{Participant}(\mathit{alice})$) in the ontology O_1, it imposes the existence of two answer sets

$$\begin{aligned}&\{O_2\colon \mathit{EarlyRegisteredParticipant}(\mathit{alice}),\, O_1\colon \mathit{Participant}(\mathit{alice})\};\\ &\{O_2\colon \mathit{LateRegisteredParticipant}(\mathit{alice}),\, O_1\colon \mathit{Participant}(\mathit{alice})\}.\end{aligned}$$

None of these answer sets is invalidated by the disjointness constraints imposed by the ontology O_2. However, we can deduce only $\mathit{Participant}(\mathit{alice})$ cautiously, the other atoms can be deduced bravely. More generally, with such rules, instances that are only available in the ontology O_2 cannot be classified with certainty.

We can solve this issue by refining the rules again and making use of nonmonotonic negation. In particular, we can extend the body of the original mappings with the following additional requirement:

$$\begin{aligned}&O_2\colon \mathit{EarlyRegisteredParticipant}(X) \leftarrow \\ &\qquad O_1\colon \mathit{Participant}(X) \wedge O_1\colon \mathit{RegisteredBeforeDeadline}(X);\\ &O_2\colon \mathit{LateRegisteredParticipant}(X) \leftarrow \\ &\qquad O_1\colon \mathit{Participant}(X) \wedge \mathit{not}\ O_1\colon \mathit{RegisteredBeforeDeadline}(X).\end{aligned}$$

This refinement of the mapping rules resolves the inconsistency and also provides a more correct mapping because background information has been added. A drawback of this approach is the fact that it requires manual post-processing of mappings because the additional background information is not obvious. In the next section, we present a probabilistic extension of tightly integrated disjunctive dl-programs that allows us to directly use confidence estimations of matching engines to resolve inconsistencies and to combine the results of different matchers.

5.3 Ontology Mappings with Confidence Values

We next show how tightly coupled probabilistic dl-programs $KB = (L, P, \mathcal{C}, \mu)$ can be used for representing (possibly inconsistent) mappings with confidence values between two ontologies. Intuitively, L encodes the union of the two ontologies, while P, $\mathcal{C}$, and μ encode the mappings between the ontologies, where confidence values can be encoded as error probabilities, and inconsistencies can also be resolved via trust probabilities (in addition to using disjunctions and nonmonotonic negations in P). Note, however, that again we need to employ an additional method for detecting inconsistent mappings as mentioned in Section 5.2. Here, we show how the previously detected inconsistencies can be resolved by taking into account the uncertainty represented by the confidence values that the matchers produce. Furthermore, we also show how we can combine possibly inconsistent results of different matchers by adding the representation of trust to the matchers by means of Bayesian probabilities. The trust values can be adjusted manually, but it is also conceivable to adjust them automatically by providing the background knowledge of the domain and by using a statistical pre-evaluation on some benchmarking ontologies of different domains.

The probabilistic extension of tightly coupled disjunctive dl-programs $KB = (L, P)$ to tightly coupled probabilistic dl-programs $KB' = (L, P, \mathcal{C}, \mu)$ provides us with a means to explicitly represent and use the confidence values provided by matching systems. In particular, we can interpret the confidence value as an *error probability* and state that the probability that a mapping introduces an error is $1 - p$. Conversely, the probability that a mapping correctly describes the semantic relation between elements of the different ontologies is $1 - (1 - p) = p$. This means that we can use the confidence value p as a probability for the correctness of a mapping. The indirect formulation is chosen, because it allows us to combine the results of different matchers in a meaningful way. In particular, if we assume that the error probabilities of two matchers are independent, we can calculate the joint error probability of two matchers that have found the same mapping rule as $(1 - p_1) \cdot (1 - p_2)$. This means that we can get a new probability for the correctness of the rule found by two matchers which is $1 - (1 - p_1) \cdot (1 - p_2)$. This way of calculating the joint probability meets the intuition that a mapping is more likely to be correct if it has been discovered by more than one matcher because $1 - (1 - p_1) \cdot (1 - p_2) \geqslant p_1$ and $1 - (1 - p_1) \cdot (1 - p_2) \geqslant p_2$.

In addition, when merging inconsistent results of different matching systems, we weigh each matching system and its result with a (e.g., user-defined) *trust probability*, which describes our confidence in its quality. All these trust probabilities sum up to 1. For example, the trust probabilities of the matching systems m_1, m_2, and m_3 may be 0.6, 0.3, and 0.1, respectively. That is, we trust most in m_1, medium in m_2, and less in m_3.

Example 5.2. We illustrate this approach using an example from the benchmark data set of the OAEI 2006 campaign. In particular, we consider the case where the publication ontology in test 101 (O_1) is mapped on the ontology of test 302 (O_2). Below we show some mappings that have been detected by the matching system hmatch that participated in the challenge. The mappings are described as rules in P, which contain a conjunct indicating the matching system that has created it and a subscript for identifying the mapping. These additional conjuncts are atomic choices of the choice space $\mathcal{C}$ and link probabilities (which are specified in the probability μ on the choice space $\mathcal{C}$) to the rules:

$$O_2: \mathit{Book}(X) \leftarrow O_1: \mathit{Collection}(X) \wedge \mathit{hmatch}_1;$$
$$O_2: \mathit{Proceedings}(X) \leftarrow O_1: \mathit{Proceedings}(X) \wedge \mathit{hmatch}_2.$$

We define the choice space according to the interpretation of confidence described above. The resulting choice space is $\mathcal{C} = \{\{\mathit{hmatch}_i, \mathit{not\_hmatch}_i\} \mid i \in \{1,2\}\}$. It comes along with the probability μ on $\mathcal{C}$, which assigns the corresponding confidence value p to each atomic choice hmatch_i and the complement $1-p$ to the atomic choice $\mathit{not\_hmatch}_i$. In our case, we have $\mu(\mathit{hmatch}_1) = 0.62$, $\mu(\mathit{not\_hmatch}_1) = 0.38$, $\mu(\mathit{hmatch}_2) = 0.73$, and $\mu(\mathit{not\_hmatch}_2) = 0.27$.

The benefits of this explicit treatment of uncertainty becomes clear when we now try to merge this mapping with the result of another matching system. Below are two examples of rules that describe correspondences for the same ontologies that have been found by the falcon system:

$$O_2: \mathit{InCollection}(X) \leftarrow O_1: \mathit{Collection}(X) \wedge \mathit{falcon}_1;$$
$$O_2: \mathit{Proceedings}(X) \leftarrow O_1: \mathit{Proceedings}(X) \wedge \mathit{falcon}_2.$$

Here, the confidence encoding yields the choice space $\mathcal{C}' = \{\{\mathit{falcon}_i, \mathit{not\_falcon}_i\} \mid i \in \{1,2\}\}$ along with the probabilities $\mu'(\mathit{falcon}_1) = 0.94$, $\mu'(\mathit{not\_falcon}_1) = 0.06$, $\mu'(\mathit{falcon}_2) = 0.96$, and $\mu'(\mathit{not\_falcon}_2) = 0.04$.

Note that directly merging these two mappings as they are would not be a good idea for two reasons. The first one is that we might encounter an inconsistency problem like shown in Section 5.2. For example, in this case, the ontology O_2 imposes that the concepts $\mathit{InCollection}$ and Book are to be disjoint. Thus, for each publication pub belonging to the concept $\mathit{Collection}$ in the ontology O_1, the merged mappings infer $\mathit{Book}(\mathit{pub})$ and $\mathit{InCollection}(\mathit{pub})$. Therefore, the first rule of each of the mappings cannot contribute to a model of the knowledge base. The second reason is that a simple merge does not account for the fact that the mapping between the $O_1: \mathit{Proceedings}$ and $O_2: \mathit{Proceedings}$ concepts has been found by both matchers and should therefore be strengthened. Here, the mapping rule has the same status as any other rule in the mapping and each instance of $O_1: \mathit{Proceedings}$ has two probabilities at the same time.

Suppose we associate with hmatch and falcon the trust probabilities 0.55 and 0.45, respectively. Based on the interpretation of confidence values as error probabilities, and on the use of trust probabilities when resolving inconsistencies between rules, we can now define a merged mapping set that consists of the following rules:

$$O_2: \mathit{Book}(X) \leftarrow O_1: \mathit{Collection}(X) \wedge \mathit{hmatch}_1 \wedge \mathit{sel\_hmatch}_1;$$
$$O_2: \mathit{InCollection}(X) \leftarrow O_1: \mathit{Collection}(X) \wedge \mathit{falcon}_1 \wedge \mathit{sel\_falcon}_1;$$
$$O_2: \mathit{Proceedings}(X) \leftarrow O_1: \mathit{Proceedings}(X) \wedge \mathit{hmatch}_2;$$
$$O_2: \mathit{Proceedings}(X) \leftarrow O_1: \mathit{Proceedings}(X) \wedge \mathit{falcon}_2.$$

The new choice space $\mathcal{C}''$ and the new probability μ'' on $\mathcal{C}''$ are obtained from $\mathcal{C} \cup \mathcal{C}'$ and $\mu \cdot \mu'$ (which is the product of μ and μ', that is, $(\mu \cdot \mu')(B \cup B') = \mu(B) \cdot \mu'(B')$ for all total choices B of $\mathcal{C}$ and B' of $\mathcal{C}'$), respectively, by adding the alternative $\{sel\_hmatch_1, sel\_falcon_1\}$ and the probabilities $\mu''(sel\_hmatch_1) = 0.55$ and $\mu''(sel\_falcon_1) = 0.45$ for resolving the inconsistency between the first two rules.

It is not difficult to verify that, due to the independent combination of alternatives, the last two rules encode that the rule $O_2 : Proceedings(X) \leftarrow O_1 : Proceedings(X)$ holds with the probability $1 - (1 - \mu''(hmatch_2)) \cdot (1 - \mu''(falcon_2)) = 0.9892$, as desired. Informally, any randomly chosen instance of $Proceedings$ of the ontology O_1 is also an instance of $Proceedings$ of the ontology O_2 with the probability 0.9892. In contrast, if the mapping rule would have been discovered only by falcon or hmatch, respectively, such an instance of $Proceedings$ of the ontology O_1 would be an instance of $Proceedings$ of the ontology O_2 with the probability 0.96 or 0.73, respectively.

A probabilistic query Q asking for the probability that a specific publication pub in the ontology O_1 is an instance of the concept $Book$ of the ontology O_2 is given by $Q = \exists(Book(pub))[R, S]$. The tight answer θ to Q is $\theta = \{R/0, S/0\}$, if pub is not an instance of the concept $Collection$ in the ontology O_1 (since there is no mapping rule that maps another concept than $Collection$ to the concept $Book$). If pub is an instance of the concept $Collection$, however, then the tight answer to Q is $\theta = \{R/0.341, S/0.341\}$ (as $\mu''(hmatch_1) \cdot \mu''(sel\_hmatch_1) = 0.62 \cdot 0.55 = 0.341$). Informally, pub belongs to the concept $Book$ with the probabilities 0 and 0.341, respectively.

6 Probabilistic Reasoning about Actions Involving Ontologies

The ICL [19] is in fact a language for probabilistic reasoning about actions in single- and multi-agent systems. It allows to describe the preconditions and effects of actions in single- and multi-agent dynamic systems; for further details including a solution to the frame problem, see especially [19]. Note that such action descriptions are closely related to action descriptions in the situation calculus. Hence, our approach to (tightly coupled) probabilistic dl-programs also constitutes a natural way of integrating reasoning about actions, description logics, and Bayesian probabilities.

Such an integration is especially useful in the context of Web Services, where action descriptions in the situation calculus have been successfully used

(a) as background action theories for defining the preconditions and effects of primitive actions, which are then composed to more complex programs using the programming constructs of Golog, in order to formulate Web Services [36,35];
(b) to provide a formal semantics for a subset of service descriptions in DAML-S, serving as an intermediate step towards a Petri net representation, which are then used for Web Service simulation, verification, and composition [37,38].

We now describe how action descriptions in the situation calculus along with Golog programs can be used for defining and composing Web Services. In addition to an example from Web Services, we then also provide an example from mobile robotics.

The situation calculus [39,40] is a second-order language for representing dynamically changing worlds. Its main ingredients are *actions*, *situations*, and *fluents*. Informally, actions have changes to the world as effects, and situations encode sequences of actions, while fluents represent a world or an agent property that may change when executing an action. In the situation calculus, a dynamic domain is represented by a *basic action theory*, which encodes (i) foundational axioms for situations, (ii) unique name axioms for actions, (iii) the initial state of the domain, (iv) action precondition axioms, and (v) successor state axioms to describe how the fluents change by the actions.

Golog is an agent programming language that is based on the situation calculus [41,40]. It allows for constructing complex actions from the primitive actions defined in a basic action theory AT, where standard (and not so standard) Algol-like constructs can be used, in particular, (i) program sequences: $p_1; p_2$; (ii) tests of conditions: $\phi?$; (iii) nondeterministic choices of two programs: $p_1|p_2$; (iv) nondeterministic choices of program argument: $\pi x\,(p(x))$; and (v) conditionals, while-loops, and procedures.

The following example (adapted from [37]) illustrates the use of probabilistic dl-programs along with Golog programs for defining and composing Web Services.

Example 6.1. We sketch how a (tightly coupled) probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$ can be used to model a product database along with the (effects of the) elementary operations of transactions on the product database. The product database itself may be modeled by a description logic knowledge base L, such as:

$$\begin{array}{l}
\mathit{textbook} \sqsubseteq \mathit{book};\ \mathit{pc} \sqcup \mathit{laptop} \sqsubseteq \mathit{electronics};\ \mathit{pc} \sqsubseteq \neg\mathit{laptop};\\
\mathit{book} \sqcup \mathit{electronics} \sqsubseteq \mathit{product};\ \mathit{book} \sqsubseteq \neg\mathit{electronics};\ \mathit{offer} \sqsubseteq \mathit{product};\\
\mathit{product} \sqsubseteq\ \geqslant 1\,\mathit{related};\ \ \geqslant 1\,\mathit{related} \sqcup \geqslant 1\,\mathit{related}^- \sqsubseteq \mathit{product};\\
\mathit{related} \sqsubseteq \mathit{related}^-;\ \mathit{related}^- \sqsubseteq \mathit{related};\\
\mathit{textbook}(\mathit{tb\_ai});\ \mathit{textbook}(\mathit{tb\_lp});\ \mathit{related}(\mathit{tb\_ai}, \mathit{tb\_lp});\\
\mathit{pc}(\mathit{pc\_ibm});\ \mathit{pc}(\mathit{pc\_hp});\ \mathit{related}(\mathit{pc\_ibm}, \mathit{pc\_hp});\\
\mathit{provides}(\mathit{ibm}, \mathit{pc\_ibm});\ \mathit{provides}(\mathit{hp}, \mathit{pc\_hp}).
\end{array}$$

Intuitively, the knowledge base L adds to P different kinds of products and relations between them (namely, all its entailed atomic concept and role membership axioms). Every answer set of KB contains in particular the following ground atoms:

$$\begin{array}{l}
\mathit{textbook}(\mathit{tb\_ai});\ \mathit{book}(\mathit{tb\_ai});\ \mathit{product}(\mathit{tb\_ai});\\
\mathit{textbook}(\mathit{tb\_lp});\ \mathit{book}(\mathit{tb\_lp});\ \mathit{product}(\mathit{tb\_lp});\\
\mathit{pc}(\mathit{pc\_ibm});\ \mathit{electronics}(\mathit{pc\_ibm});\ \mathit{product}(\mathit{pc\_ibm});\\
\mathit{pc}(\mathit{pc\_hp});\ \mathit{electronics}(\mathit{pc\_hp});\ \mathit{product}(\mathit{pc\_hp});\\
\mathit{related}(\mathit{tb\_ai}, \mathit{tb\_lp});\ \mathit{related}(\mathit{pc\_ibm}, \mathit{pc\_hp});\\
\mathit{provides}(\mathit{ibm}, \mathit{pc\_ibm});\ \mathit{provides}(\mathit{hp}, \mathit{pc\_hp}).
\end{array}$$

The rule component P encodes the preconditions and the effects of the primitive actions. For example, the following rule in P encodes that a client C can buy a product O from an online shop M if (i) C does not already own O, (ii) M has O in stock and not reserved for someone else, and (iii) C's account status is above M's price for O (as usual, we use parameterized actions for a more compact representation, that is, $\mathit{buy}(\mathit{cl}_i, \mathit{obj}_j, \mathit{shop}_k)$ represents the non-parameterized action $\mathit{buy\_cl}_i\mathit{\_obj}_j\mathit{\_shop}_k$):

$$\begin{array}{l}
\mathit{Poss}(\mathit{buy}(C, O, M), S) \leftarrow \mathit{not}\ \mathit{own}(C, O, S),\ \mathit{in\_stock}(M, O, S),\\
\quad \mathit{not}\ \mathit{reserved}(M, O, S),\ \mathit{account}(C, B, S),\ \mathit{price}(M, O, B', S),\ B > B'.
\end{array}$$

The following two rules in P encode that (i) if a client C buys a product O from an online shop M, then C's account status reduces by M's price for O in the next situation, and (ii) else C's account status does not change in the next situation:

$$\begin{aligned}&\mathit{account}(C,B,\mathit{do}(A,S)) \leftarrow A=\mathit{buy}(C,O,M),\ \mathit{account}(C,B',S),\\&\qquad \mathit{price}(M,O,B'',S),\ B=B'-B'';\\&\mathit{account}(C,B,\mathit{do}(A,S)) \leftarrow A\neq \mathit{buy}(C,O,M),\ \mathit{account}(C,B,S).\end{aligned}$$

The following two rules in P encode that (i) if a client C buys a product O from an online shop M, then C owns O in the next situation, and (ii) if a client C does not sell an owned product O to an online shop M, then C also owns O in the next situation:

$$\begin{aligned}&\mathit{own}(C,O,\mathit{do}(A,S)) \leftarrow A=\mathit{buy}(C,O,M);\\&\mathit{own}(C,O,\mathit{do}(A,S)) \leftarrow A\neq \mathit{sell}(C,O,M),\ \mathit{own}(C,O,S).\end{aligned}$$

The choice space $\mathcal{C}$ and the probability μ may then be used to define probabilistic effects of actions. For example, a client's buying may only succeed with the probability 0.9, which can be expressed through the rule

$$\mathit{own}(C,O,\mathit{do}(A,S)) \leftarrow A=\mathit{buy}(C,O,M),\ \mathit{succ},$$

the choice space $\mathcal{C}=\{\mathit{succ},\mathit{fail}\}$, and the probability μ: $\mathit{succ},\mathit{fail}\mapsto 0.9, 0.1$.

The primitive actions, which we defined via a probabilistic action description, can now be composed to more complex programs using Golog constructs. For example, the following small Golog program encodes the Web Service for buying a birthday gift, which divides into buying either a book or a CD (from a given list of books (resp., CDs) at a given list of online shops) and eventually also a flower (from a given list of flowers at a given list of online shops) if more than € 20 are left on the client's account:

```
**proc** $\mathit{buy\_birthday\_gift}(C, \mathit{Shops}, \mathit{Books}, \mathit{CDs}, \mathit{Flowers})$
$\pi_{M\in \mathit{Shops}}(\pi_{O\in \mathit{Books}}(\mathit{buy}(C,O,M)))\ |$
    $\pi_{M\in \mathit{Shops}}(\pi_{O\in \mathit{CDs}}(\mathit{buy}(C,O,M)))$;
**if** $\mathit{account}(C,B)\wedge B\geqslant 20$ **then**
    $\pi_{M\in \mathit{Shops}}(\pi_{O\in \mathit{Flowers}}(\mathit{buy}(C,O,M)))$
**end**.
```

The next example describes another application in mobile robotics.

Example 6.2. Consider a mobile robot that should pick up some objects. We now sketch how this scenario can be modeled using a (tightly coupled) probabilistic dl-program $KB=(L,P,\mathcal{C},\mu)$. The ontology component L encodes background knowledge about the domain. For example, concepts may encode different kinds of objects and different kinds of positions, while roles may express different kinds of relations between positions (in a 3×3 grid), which is expressed by the following description logic axioms in L:

$$\begin{aligned}&\mathit{ball}\sqsubseteq \mathit{light\_object};\ \ \mathit{light\_object}\sqsubseteq \mathit{object};\ \ \mathit{heavy\_object}\sqsubseteq \mathit{object};\\&\mathit{central\_position}\sqsubseteq \mathit{position};\ \ \mathit{central\_position}\sqsubseteq\ \leqslant 9\,\mathit{neighbor}^-.\mathit{position};\\&\mathit{central\_position}\sqsubseteq(\leqslant 1\,\mathit{west\_of}^-.\mathit{position})\sqcap(\leqslant 1\,\mathit{north\_of}^-.\mathit{position});\\&\exists \mathit{west\_of}.\top\sqsubseteq \mathit{position};\ \ \exists \mathit{west\_of}^-.\top\sqsubseteq \mathit{position};\\&\mathit{object}(\mathit{obj}_1);\ \ \mathit{light\_object}(\mathit{obj}_2);\ \ \mathit{heavy\_object}(\mathit{obj}_3);\ \ \mathit{ball}(\mathit{obj}_4);\ \ \mathit{obj}_2=\mathit{obj}_4;\\&\mathit{position}(\mathit{pos}_1);\ \ldots;\ \mathit{position}(\mathit{pos}_9);\ \ \mathit{central\_position}(\mathit{pos}_5);\\&\mathit{neighbor}(\mathit{pos}_1,\mathit{pos}_2);\ \ldots;\ \mathit{west\_of}(\mathit{pos}_1,\mathit{pos}_2);\ \ldots;\ \mathit{north\_of}(\mathit{pos}_1,\mathit{pos}_4);\ \ldots\ .\end{aligned}$$

The rule component P encodes the dynamics (within a finite time frame). For example, the following rule in P says that if (i) the robot performs a pickup of object O, (ii) both the robot and the object O are at the same position, and (iii) the pickup of O succeeds (which is an atomic choice associated with a certain probability), then the robot is carrying O in the next situation:

$$\begin{aligned}&\mathit{carrying}(O, \mathit{do}(A, S)) \leftarrow A = \mathit{pickup}(O),\ \mathit{at}(\mathit{robot}, \mathit{Pos}, S),\ \mathit{at}(O, \mathit{Pos}, S),\\ &\quad \mathit{pickup\_succeeds}(O, S),\ \mathit{object}(O),\ \mathit{position}(\mathit{Pos}).\end{aligned}$$

The next rule in P says that if (i) the robot is carrying a heavy object O, (ii) performs no pickup and no putdown operation, and (iii) keeps carrying O (which is an atomic choice associated with a certain probability), then the robot also keeps carrying O in the next situation (we can then use a similar rule for light object with a different probability):

$$\begin{aligned}&\mathit{carrying}(O, \mathit{do}(A, S)) \leftarrow \mathit{carrying}(O, S),\ A \neq \mathit{pickup}(O),\ A \neq \mathit{putdown}(O),\\ &\quad \mathit{keeps\_carrying}(O, S),\ \mathit{heavy\_object}(O).\end{aligned}$$

To encode the probabilities for the above rules, the choice space $\mathcal{C}$ contains ground instances of $\{\mathit{keeps\_carrying}(O, S), \mathit{not\_keeps\_carrying}(O, S)\}$ and $\{\mathit{pickup\_succeeds}(O, S), \mathit{not\_pickup\_succeeds}(O, S)\}$. We then define a probability μ on each alternative $A \in \mathcal{C}$ (for example, $\mu(\mathit{keeps\_carrying}(\mathit{obj}_1, S_0)) = 0.9$ and $\mu(\mathit{not\_keeps\_carrying}(\mathit{obj}_1, S_0)) = 0.1$) and extend it to a probability μ on the set of all total choices of $\mathcal{C}$ by assuming independence between the alternatives of $\mathcal{C}$.

An example of a small Golog program for this domain is

$$\textbf{while}\ \neg \mathit{carrying}(\mathit{obj}) \wedge \exists \mathit{Pos}\ (\mathit{at}(\mathit{robot}, \mathit{Pos}) \wedge \mathit{at}(\mathit{obj}, \mathit{Pos}))\ \textbf{do}\ \mathit{pickup}(\mathit{obj}),$$

which stands for "while the robot *robot* is not carrying the object *obj*, and they are both at the same position, the robot *robot* tries to pick up the object *obj*".

7 Algorithms and Complexity

In this section, we characterize the consistency and the query processing problem in probabilistic dl-programs under the answer set semantics in terms of the consistency and the cautious/brave reasoning problem in disjunctive dl-programs under the answer set semantics (which are all decidable [10]). These characterizations show that the consistency and the query processing problem in probabilistic dl-programs under the answer set semantics are decidable and computable, respectively, and they also directly reveal algorithms for solving these problems. In particular, the second characterization can be used for an anytime algorithm for tight query processing in probabilistic dl-programs under the answer set semantics. We describe this anytime algorithm along with soundness and error estimation results. We also give a precise picture of the complexity of deciding consistency and correct answers for probabilistic dl-programs under the answer set semantics.

7.1 Algorithms

The following theorem shows that a probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$ is consistent iff the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$ is consistent, for every total choice B of $\mathcal{C}$ with $\mu(B) > 0$. Thus, deciding whether a probabilistic dl-program is consistent can be reduced to deciding whether a disjunctive dl-program is consistent.

Theorem 7.1. *Let $KB = (L, P, \mathcal{C}, \mu)$ be a probabilistic dl-program. Then, KB is consistent iff $(L, P \cup \{p \leftarrow \mid p \in B\})$ is consistent for each total choice B of $\mathcal{C}$ with $\mu(B) > 0$.*

The next theorem shows that computing tight answers for probabilistic queries $Q = \exists(\beta|\alpha)[r, s]$ with ground $\beta|\alpha$ to consistent probabilistic dl-programs KB under the answer set semantics can be reduced to brave and cautious reasoning from disjunctive dl-programs. Informally, the tight lower (resp., upper) bound is computed from values a (resp., b) and c (resp., d), where (1) a (resp., b) is the sum of all $\mu(B)$ such that (1.i) B is a total choice of $\mathcal{C}$ and (1.ii) $\alpha \wedge \beta$ a cautious (resp., brave) consequence of the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$, and (2) c (resp., d) is the sum of all $\mu(B)$ such that (2.i) B is a total choice of $\mathcal{C}$ and (2.ii) $\alpha \wedge \neg\beta$ a brave (resp., cautious) consequence of the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$.

Theorem 7.2. *Let $KB = (L, P, \mathcal{C}, \mu)$ be a consistent probabilistic dl-program, and let $Q = \exists(\beta|\alpha)[r, s]$ be a probabilistic query with ground conditional event $\beta|\alpha$. Let a (resp., b) be the sum of all $\mu(B)$ such that (i) B is a total choice of $\mathcal{C}$ and (ii) $\alpha \wedge \beta$ is true in every (resp., some) answer set of the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$. Let c (resp., d) be the sum of all $\mu(B)$ such that (i) B is a total choice of $\mathcal{C}$ and (ii) $\alpha \wedge \neg\beta$ is true in some (resp., every) answer set of the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$. Then, the tight answer θ for Q to KB under the answer set semantics is given as follows:*

$$\theta = \begin{cases} \{r/1,\ s/0\} & \text{if } b = 0 \text{ and } c = 0; \\ \{r/0,\ s/0\} & \text{if } b = 0 \text{ and } c \neq 0; \\ \{r/1,\ s/1\} & \text{if } b \neq 0 \text{ and } c = 0; \\ \{r/\frac{a}{a+c},\ s/\frac{b}{b+d}\} & \text{otherwise.} \end{cases} \tag{2}$$

By the above theorem, computing the tight answer for probabilistic queries $Q = \exists(\beta|\alpha)[r, s]$ with ground $\beta|\alpha$ to consistent probabilistic dl-programs $KB = (L, P, \mathcal{C}, \mu)$ under the answer set semantics can be reduced to (1) computing the set of all answer sets of each disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$ such that B is a total choice of $\mathcal{C}$ and (2) performing brave and cautious reasoning from these answer sets. The number of all total choices B is generally a non-neglectable source of complexity. We thus propose (i) to compute the tight answer for Q to KB only up to an error within a given threshold $\epsilon \in [0, 1]$, (ii) to process the B's along decreasing probabilities $\mu(B)$, and (iii) to eventually stop the calculation after a given time interval.

Given a consistent probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$, a probabilistic query $Q = \exists(\beta|\alpha)[r, s]$ with ground $\beta|\alpha$, and an error threshold $\epsilon \in [0, 1]$, Algorithm tight_answer (see Fig. 1) computes some $\theta = \{r/l',\ s/u'\}$ such that $|l - l'| + |u - u'| \leqslant \epsilon$, where $\{r/l,\ s/u\}$ is the tight answer for Q to KB under the answer set semantics. More concretely, it computes the bounds l' and u' by first initializing the variables a, b, c, and d (which play the same role as in Theorem 7.2). It then computes the answer set semantics S of the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B_i\})$, for every total choice B_i of $\mathcal{C}$, checks whether $\alpha \wedge \beta$ and $\alpha \wedge \neg\beta$ are true or false in all $s \in S$, and

```
Algorithm tight_answer
Input: consistent probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$, probabilistic query
        $Q = \exists(\beta|\alpha)[r, s]$ with ground $\beta|\alpha$, and error threshold $\epsilon \in [0, 1]$.
Output: $\theta = \{r/l', s/u'\}$ such that $|l - l'| + |u - u'| \leqslant \epsilon$, where $\{r/l, s/u\}$ is the tight
        answer for $Q$ to $KB$ under the answer set semantics.
Notation: $B_1, \ldots, B_k$ is a sequence of all total choices $B$ of $\mathcal{C}$ with $\mu(B_1) \geqslant \cdots \geqslant \mu(B_k)$.

 1. $a := 0$; $b := 1$; $c := 1$; $d := 0$; $v := 1$; $i := 1$;
 2. while $i \leqslant k$ and $v > 0$ and $\frac{v}{a+c} + \frac{v}{b+d} > \epsilon$ do begin
 3.   compute the set $S$ of all answer sets of $(L, P \cup \{p \leftarrow \mid p \in B_i\})$;
 4.   if $\alpha \wedge \beta$ is true in every $s \in S$ then $a := a + \mu(B_i)$
 5.     else if $\alpha \wedge \beta$ is false in every $s \in S$ then $b := b - \mu(B_i)$;
 6.   if $\alpha \wedge \neg\beta$ is false in every $s \in S$ then $c := c - \mu(B_i)$
 7.     else if $\alpha \wedge \neg\beta$ is true in every $s \in S$ then $d := d + \mu(B_i)$;
 8.   $v := v - \mu(B_i)$;
 9.   $i := i + 1$
10. end;
11. if $b = 0$ and $c = 0$ then return $\theta = \{r/1, s/0\}$
12.   else if $b = 0$ and $c \neq 0$ then return $\theta = \{r/0, s/0\}$
13.     else if $b \neq 0$ and $c = 0$ then return $\theta = \{r/1, s/1\}$
14.       else return $\theta = \{r/\frac{a}{a+c}, s/\frac{b}{b+d}\}$.
```

Fig. 1. Algorithm tight_answer

updates a, b, c, and d accordingly. If the possible error in the bounds falls below ϵ, then it stops and returns the bounds computed thus far. Hence, in the special case where $\epsilon = 0$, the algorithm computes in particular the tight answer for Q to KB under the answer set semantics. The following theorem shows that tight_answer is sound.

Theorem 7.3. *Let KB be a consistent probabilistic dl-program, let $Q = \exists(\beta|\alpha)[r, s]$ be a probabilistic query with ground $\beta|\alpha$, and let $\theta = \{r/l, s/u\}$ be the tight answer for Q to KB under the answer set semantics. Let $\epsilon \in [0, 1]$ be an error threshold. Then, Algorithm tight_answer always terminates on KB, Q, and ϵ. Let $\theta' = \{r/l', s/u'\}$ be the output computed by tight_answer for KB, Q, and ϵ, and let v' be the value of the variable v. Then, if $v' = 0$, then $l = l'$ and $u = u'$; otherwise, $|l - l'| + |u - u'| \leqslant \epsilon$.*

The following example illustrates how Algorithm tight_answer works.

Example 7.1 (University Database cont'd). Consider again the probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$ and the probabilistic query $Q = \exists(\beta|\alpha)[r, s] = \exists(\mathit{taken}(\mathit{mary}, \mathit{databases})|\mathit{taken}(\mathit{mary}, \mathit{operating\_systems}))[R, S]$ of Example 4.3, and suppose $\epsilon = 0$. After the initialization in line 1, we observe that (i) $\alpha \wedge \beta$ is true in every answer set of exactly $(L, P \cup \{\mathit{choice}_u, \mathit{choice}_o\})$, (ii) $\alpha \wedge \beta$ is false in every answer set of exactly $(L, P \cup \{\mathit{choice}_u, \mathit{not\_choice}_o\})$, $(L, P \cup \{\mathit{not\_choice}_u, \mathit{choice}_o\})$, and $(L, P \cup \{\mathit{not\_choice}_u, \mathit{not\_choice}_o\})$, (iii) $\alpha \wedge \neg\beta$ is false in every answer set of exactly $(L, P \cup \{\mathit{choice}_u, \mathit{choice}_o\})$, $(L, P \cup \{\mathit{not\_choice}_u, \mathit{choice}_o\})$, and $(L, P \cup \{\mathit{not\_choice}_u, \mathit{not\_choice}_o\})$, and (iv) $\alpha \wedge \neg\beta$ is true in every answer set of exactly $(L, P \cup \{\mathit{choice}_u, \mathit{not\_choice}_o\})$. We thus obtain (i) $a = 0 + 0.63 = 0.63$, (ii) $b = 1 - 0.27 - 0.07$

$-0.03=0.63$, (iii) $c=1-0.63-0.07-0.03=0.27$, and (iv) $d=0+0.27=0.27$, respectively, and return the tight answer $\theta=\{r/\frac{0.63}{0.63+0.27},\ s/\frac{0.63}{0.63+0.27}\}=\{r/0.7,\ s/0.7\}$.

Algorithm tight_answer is actually an *anytime algorithm*, since we can always interrupt it, and return the bounds computed thus far. The following theorem shows that these bounds deviate from the tight bounds with an exactly measurable error (note that it can also be shown that the computed lower and upper bounds are increasing and that the possible error is decreasing along the iterations of the while-loop of the algorithm). For this reason, Algorithm tight_answer also iterates through the total choices B_i of $\mathcal{C}$ in a way such that the probabilities $\mu(B_i)$ are decreasing, so that the error in the computed bounds is very likely to be low already after few iteration steps.

Theorem 7.4. *Let KB be a consistent probabilistic dl-program, let* $Q=\exists(\beta|\alpha)[r,s]$ *be a probabilistic query with ground conditional event* $\beta|\alpha$*, let* $\epsilon\in[0,1]$ *be an error threshold, and let* $\theta=\{r/l, s/u\}$ *be the tight answer for Q to KB under the answer set semantics. Assume we run Algorithm tight_answer on KB, Q, and* ϵ*, and we interrupt it after line (9). Let the returned* $\theta'=\{r/l',\ s/u'\}$ *be as specified in lines (11) to (14), and let* a'*,* b'*,* c'*,* d'*, and* v' *be the values of the variables a, b, c, d, and v, respectively. Then, if* $v'=0$*, then* $\theta=\theta'$*; otherwise,* $|l-l'|+|u-u'|\leqslant\frac{v'}{a'+c'}+\frac{v'}{b'+d'}$.

7.2 Complexity

The following theorem shows that deciding whether a probabilistic dl-program under the answer set semantics is consistent is complete for $\mathrm{NEXP}^{\mathrm{NP}}$ (and thus has the same complexity as deciding consistency of ordinary disjunctive logic programs under the answer set semantics). Note that the lower bound follows from the $\mathrm{NEXP}^{\mathrm{NP}}$-hardness of deciding whether an ordinary disjunctive logic program has an answer set.

Theorem 7.5. *Given a first-order vocabulary* Φ *and a probabilistic dl-program* $KB=(L,P,\mathcal{C},\mu)$*, where L is defined in* $\mathcal{SHIF}(\mathbf{D})$ *or* $\mathcal{SHOIN}(\mathbf{D})$*, deciding whether KB is consistent under the answer set semantics is complete for* $\mathrm{NEXP}^{\mathrm{NP}}$.

The next theorem shows that deciding correct answers for probabilistic queries $Q=\exists(\beta|\alpha)[l,u]$, where $\beta|\alpha$ is a ground conditional event, to a consistent probabilistic dl-program *KB* under the answer set semantics is complete for co-$\mathrm{NEXP}^{\mathrm{NP}}$.

Theorem 7.6. *Given a first-order vocabulary* Φ*, a consistent probabilistic dl-program* $KB=(L,P,\mathcal{C},\mu)$*, where L is defined in* $\mathcal{SHIF}(\mathbf{D})$ *or* $\mathcal{SHOIN}(\mathbf{D})$*, a ground conditional event* $\beta|\alpha$*, and reals* $l,u\in[0,1]$*, deciding whether* $(\beta|\alpha)[l,u]$ *is a consequence of KB under the answer set semantics is complete for* co-$\mathrm{NEXP}^{\mathrm{NP}}$.

8 Tractability Results

We now describe a special class of (tightly coupled) probabilistic dl-programs for which deciding consistency and query processing can both be done in polynomial time in the data complexity. These programs are normal, stratified, and defined relative to *DL-Lite* [42] (which allows for deciding knowledge base satisfiability in polynomial time).

We first recall *DL-Lite*. Let $\mathbf{A}$, $\mathbf{R}_A$, and $\mathbf{I}$ be pairwise disjoint sets of atomic concepts, abstract roles, and individuals, respectively. A *basic concept in DL-Lite* is either an atomic concept from $\mathbf{A}$ or an existential restriction on roles $\exists R.\top$ (abbreviated as $\exists R$), where $R \in \mathbf{R}_A \cup \mathbf{R}_A^-$. A *literal in DL-Lite* is either a basic concept b or the negation of a basic concept $\neg b$. *Concepts in DL-Lite* are defined by induction as follows. Every basic concept in *DL-Lite* is a concept in *DL-Lite*. If b is a basic concept in *DL-Lite*, and ϕ_1 and ϕ_2 are concepts in *DL-Lite*, then also $\neg b$ and $\phi_1 \sqcap \phi_2$. An *axiom in DL-Lite* is either (1) a concept inclusion axiom $b \sqsubseteq \phi$, where b is a basic concept in *DL-Lite*, and ϕ is a concept in *DL-Lite*, or (2) a *functionality axiom* $(\mathsf{funct}\ R)$, where $R \in \mathbf{R}_A \cup \mathbf{R}_A^-$, or (3) a concept membership axiom $b(a)$, where b is a basic concept in *DL-Lite* and $a \in \mathbf{I}$, or (4) a role membership axiom $R(a, c)$, where $R \in \mathbf{R}_A$ and $a, c \in \mathbf{I}$. A *knowledge base in DL-Lite* L is a finite set of axioms in *DL-Lite*. Note that the semantics of *DL-Lite* assumes standard names, which includes the unique name assumption.

The following two results from [42] show that deciding whether a knowledge base in *DL-Lite* is satisfiable is possible in polynomial time, and that every knowledge base in *DL-Lite* can be rewritten to have only literals in the heads of concept inclusion axioms.

Theorem 8.1 (see [42]). *Given a knowledge base L in DL-Lite, deciding whether L is satisfiable can be done in polynomial time.*

Proposition 8.1 (see [42]). *Every knowledge base L in DL-Lite can be transformed into an equivalent knowledge base $trans(L)$ in DL-Lite in which every concept inclusion axiom is of form $b \sqsubseteq \ell$, where b (resp., ℓ) is a basic concept (resp., literal) in DL-Lite.*

Given a knowledge base L in *DL-Lite*, we define $trans_L(P) = P \cup \{b'(X) \leftarrow b(X) \mid b \sqsubseteq b' \in trans(L),\ b' \text{ is a basic concept}\} \cup \{\exists R(X) \leftarrow R(X, Y) \mid R \in \mathbf{R}_A \cap \Phi\} \cup \{\exists R^-(Y) \leftarrow R(X, Y) \mid R \in \mathbf{R}_A \cap \Phi\}$. Intuitively, we make explicit all the relationships between the predicates in P that are implicitly encoded in L.

We define stratified normal dl- and stratified normal probabilistic dl-programs in *DL-Lite* as follows. A normal dl-program $KB = (L, P)$ with L in *DL-Lite* is *stratified* iff $trans_L(P)$ is stratified (see Section 3.1). A normal probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$ with L in *DL-Lite* is *stratified* iff every of KB's represented dl-programs is stratified.

Example 8.1 (University Database cont'd). Consider the probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$, where L is the description logic knowledge base of Example 2.1 without the axioms in (5), (7), and (9), P is the set of rules of Example 4.3 without the disjunctive rules in (1) and (2) of Example 3.1, and $\mathcal{C}$ and μ are as in Example 4.3. It is then not difficult to verify that L is definable in *DL-Lite*, and KB is normal and stratified.

The following result shows that stratified normal probabilistic dl-programs in *DL-Lite* allow for consistency checking and query processing with a polynomial data complexity. It follows from Theorems 7.1 and 7.2 and that consistency checking and cautious/brave reasoning in stratified normal dl-programs can be done in polynomial time in the data complexity [10]. Here, the notion of *data complexity* is defined as usual, that is, we keep all of $KB = (L, P, \mathcal{C}, \mu)$ fixed except for Φ_c and the facts in P.

Theorem 8.2. *Given a first-order vocabulary Φ and a stratified normal probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$ with L in DL-Lite, (a) deciding whether KB has an answer set, and (b) computing $l, u \in [0, 1]$ for a given ground conditional event $\beta|\alpha$ such that $KB \,\|\!\!\sim_{tight} (\beta|\alpha)[l, u]$ can both be done in polynomial time in the data complexity.*

9 Related Work

In this section, we give a comparison to most closely related works on (i) tightly coupled description logic programs, (ii) probabilistic description logic programs, (iii) representing ontology mappings, and (iv) reasoning about actions involving ontologies.

9.1 Tightly Coupled Description Logic Programs

Some other tight integrations of ontologies and rules are in particular due to Donini et al. [43], Levy and Rousset [44], Grosof et al. [45], Motik et al. [46], Heymans et al. [47], and Rosati [48,49]. SWRL [50] and WRL [51] also belong to this category. Among the above works, closest in spirit to the tightly coupled disjunctive dl-programs used in this paper are perhaps Rosati's [48,49] and Heymans et al.'s [47]. Like here, Rosati's hybrid knowledge bases also consist of a description logic knowledge base L and a disjunctive program (with default negations) P, where concepts and roles in L may act as predicate symbols in P. However, differently from this paper, Rosati partitions the predicates of L and P into description logic predicates and logic program predicates, where the former are interpreted under the classical model-theoretic semantics, while the latter are interpreted under the answer set semantics (and thus in particular default negations of concepts and roles are not allowed in P). Furthermore, differently from this paper, he also assumes a syntactic restriction on rules (called *weak safeness*) to gain decidability, and he makes the standard names assumption, which includes the unique name assumption. The approach of Heymans et al. [47] is very similar to Rosati's. The main differences are that on the one hand, the rules are interpreted under the open answer set semantics [52], and on the other hand, different syntactic restrictions on rules are imposed, namely, instead of the ones in Rosati's approach, solely the presence of a guard in every non-free rule. A guard is an atom that contains all variables of a rule. The main differences to Levy and Rousset's integration of ontologies and rules in CARIN [44] are that (i) we allow for disjunctions in rule heads and default negations in rule bodies, while Levy and Rousset allow only for Horn clause rules, and that (ii) our integration of ontologies and rules is developed from the perspective of logic programming, using ontological knowledge as constraints on models of logics programs, while their integration of ontologies and rules is developed from the perspective of description logics, adding rules to a description knowledge base. As a consequence, we easily gain decidability in the general case, while their formalism is undecidable in general. In the case without disjunctions in rule heads and default negations in rule bodies, our approach corresponds to adding the grounding of all rules to a description logic knowledge base (cf. Section 3.3), while Levy and Rousset's one essentially corresponds to adding a first-order rewriting of all rules to the description logic knowledge base. Another difference is that Levy and Rousset make the unique name assumption, while we do not.

9.2 Probabilistic Description Logic Programs

It is important to point out that the probabilistic description logic programs here are very different from the ones in [17] (and their recent tractable variant in [18]). First, they are based on the tight integration between the ontology component L and the rule component P of [10], while the ones in [17,18] realize the loose query-based integration between the ontology component L and the rule component P of [8]. This implies in particular that the vocabularies of L and P here may have common elements (see also Example 3.1), while the vocabularies of L and P in [17,18] are necessarily disjoint. Furthermore, the probabilistic description logic programs here behave semantically very differently from the ones in [17,18] (see Example 3.5). As a consequence, the probabilistic description logic programs here are especially useful for sophisticated probabilistic reasoning tasks involving ontologies (including representing and reasoning with ontology mappings under probabilistic uncertainty and inconsistency, as well as probabilistic reasoning about actions involving ontologies), while the ones in [17,18] can especially be used as query interfaces to Web databases (including RDF theories). Second, differently from the programs here, the ones in [17,18] do not allow for disjunctions in rule heads. Third, differently from here, the works [17,18] do not explore the use of probabilistic description logic programs for representing and reasoning with ontology mappings under probabilistic uncertainty and inconsistency, and their use for probabilistic reasoning about actions involving ontologies.

9.3 Representing Ontology Mappings

There are several languages for representing mappings between ontologies [21,22]. However, all of them, except for Bayesian description logic programs (BDLPs) [22] represent mappings deterministically. Compared to tightly coupled probabilistic dl-programs, BDLPs suffer from a rather low expressivity, since they are based on a probabilistic extension of a subset of the description logic variants underlying OWL, namely, the description logic programs (DLPs) by Grosof et al. [45]. It is important to point out that the description logic programs of [45] are different from the description logic programs of [8], which have been extended with probabilities in [17,18]. The former correspond to definite clause logic with several further restrictions, which is essentially the intersection of the description logic behind OWL and logic programming, while the latter consists of a knowledge base L in the description logic behind OWL and of a logic program P with negation under the answer set semantics (cf. also Section 9.2).

Using BDLPs, only ontologies with a very restricted expressiveness can be mapped. Note that for reasoning with ontologies in DLPs and mappings expressed in BDLPs, the ontologies need to be translated into logic programming syntax. Concerning the requirements that we impose on a mapping language as mentioned in Section 5.1, BDLPs provide a *tight integration of mapping and ontology language* only for ontologies lying in DLPs. The requirement *support for mappings refinement* can be partly fulfilled, since it is possible to add further positive conjuncts in the body of a rule. *Support for repairing inconsistencies* is also rather limited by BDLPs, because disjunctions are disallowed in rule heads. However, by means of the probabilities attached to each rule, inconsistencies may be resolved probabilistically. *Representation and combination of confidence* is possible with BDLPs as shown in [22]. The requirement *decidability and efficiency of instance reasoning* is also fulfilled by BDLPs due to the restricted expressiveness of BDLPs and due to the fact that BDLPs are a probabilistic rule language. Rule languages can deal with reasoning tasks like instance retrieval much more efficiently. This holds especially if the TBoxes are small and the ABoxes big [53].

There are probabilistic extensions of different Web languages that are conceivable to be used as mapping languages in the context of ontology mapping [54]. Examples of such probabilistic extensions are probabilistic extensions of description logics like P-CLASSIC, which is a probabilistic extension of CLASSIC [55], PR-OWL, which is an ontology language that describes multi-entity Bayesian networks [16], BayesOWL, which provides a probabilistic extension of a subset of OWL [56], and P-$\mathcal{SHOQ}(\mathbf{D})$, which is a probabilistic extension of $\mathcal{SHOQ}(\mathbf{D})$ [13] (see also P-$\mathcal{SHIF}(\mathbf{D})$ and P-$\mathcal{SHOIN}(\mathbf{D})$ in [14]). P-CLASSIC and BayesOWL have the disadvantage of a too low expressivity. PR-OWL does not provide a tight formal integration between ontologies and the probabilistic model that they describe. Although P-$\mathcal{SHOQ}$ is quite expressive and provides a tight integration between the description logic and the probabilistic model, it does not have a rule component and cannot solve the instance retrieval reasoning task efficiently compared to rule languages. Probabilistic extensions of rule languages for the Web besides the already mentioned BDLPS are also pOWLLite^{-} and pOWLLite^{EQ} [57]. These two languages differ only by equality, which is disallowed in pOWLLite^{-}. Both support also only the description logic programming fragment (possibly enriched with equality) that is supported by BDLPs and thus have the same expressivity drawback. Note that except for BDLPs, none of these languages have been considered for an application in the area of ontology mappings.

9.4 Reasoning about Actions Involving Ontologies

One of the earliest works on combining reasoning about actions with ontologies is due to De Giacomo et al. [58], who exploit the correspondence between propositional dynamic logics and description logics to develop a formalism for reasoning about actions based on description logics, and who implement it on a robotic soccer team. Recent work by Iocchi et al. [59] extends this action formalism by sensing under qualitative and probabilistic uncertainty. Other recent works by Baader et al. [60], Milicic [61], and Drescher and Thielscher [62] also use description logics for developing formalisms for reasoning about actions. The main conceptual difference between our formalism and all the above approaches is that we use ontologies as a background theory to further constrain action descriptions, guided by the idea of exploiting existing ontologies in reasoning about actions, while they perform a reduction of action formalisms to ontologies, guided by the idea of developing decidable formalisms for reasoning about actions, related to existing ones like the situation calculus and the fluent calculus.

10 Conclusion

We have presented tightly coupled probabilistic (disjunctive) dl-programs under the answer set semantics, which are a tight combination of disjunctive logic programs under the answer set semantics, description logics, and Bayesian probabilities. We have described applications in representing and reasoning with ontology mappings and in probabilistic reasoning about actions involving ontologies. We have shown that consistency checking and query processing in tightly coupled probabilistic dl-programs are decidable and computable, respectively, and that they can be reduced to their classical counterparts in tightly coupled disjunctive dl-programs. We have also given an anytime algorithm for query processing, and we have analyzed the complexity of consistency checking and query processing. Furthermore, we have delineated a special case of these problems that can be solved in polynomial time in the data complexity.

As for representing ontology mappings, the new formalism supports the resolution of inconsistencies on a symbolic and a numeric level. While the use of disjunction and nonmonotonic negation allows the rewriting of inconsistent rules, the probabilistic extension of the language allows us to explicitly represent numeric confidence values as error probabilities, to resolve inconsistencies by using trust probabilities, and to reason about these on a numeric level. While being expressive and well-integrated with description logic ontologies, the new formalism is still decidable and has data-tractable subsets, which make it particularly interesting for practical applications.

We leave for future work the implementation of tightly coupled probabilistic dl-programs. Another interesting topic for future work is to explore whether the tractability results can be extended to an even larger class of tightly coupled probabilistic dl-programs. One way to achieve this could be to approximate the answer set semantics through the well-founded semantics (which may be defined similarly as in [18]). Furthermore, it would be interesting to investigate whether one can develop an efficient top-k query technique for the presented tightly coupled probabilistic dl-programs: Rather than computing the tight probability interval for a given ground conditional event, such a technique returns the k most probable ground instances of a given non-ground atom.

Acknowledgments. Andrea Calì has been supported by the Engineering and Physical Sciences Research Council (EPSRC) under the project "Schema Mappings and Automated Services for Data Integration and Exchange" (EP/E010865/1) and by the European Union under the STREP FET project TONES (FP6-7603). Thomas Lukasiewicz has been supported by the German Research Foundation (DFG) under the Heisenberg Programme and by the Austrian Science Fund (FWF) under the project P18146-N04. Heiner Stuckenschmidt and Livia Predoiu have been supported by an Emmy-Noether Grant of the German Research Foundation (DFG). We thank the reviewers of this paper for their constructive and useful comments, which helped to improve this work.

References

1. Calì, A., Lukasiewicz, T.: Tightly integrated probabilistic description logic programs for the semantic web. In: Dahl, V., Niemelä, I. (eds.) ICLP 2007. LNCS, vol. 4670, pp. 428–429. Springer, Heidelberg (2007)
2. Calì, A., Lukasiewicz, T., Predoiu, L., Stuckenschmidt, H.: A framework for representing ontology mappings under probabilities and inconsistency. In: Proceedings URSW 2007. CEUR Workshop Proceedings, vol. 327 (2008) CEUR-WS.org
3. Calì, A., Lukasiewicz, T., Predoiu, L., Stuckenschmidt, H.: Tightly integrated probabilistic description logic programs for representing ontology mappings. In: Hartmann, S., Kern-Isberner, G. (eds.) FoIKS 2008. LNCS, vol. 4932, pp. 178–198. Springer, Heidelberg (2008)

4. Berners-Lee, T.: Weaving the Web. Harper, San Francisco (1999)
5. Fensel, D., Wahlster, W., Lieberman, H., Hendler, J. (eds.): Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential. MIT Press, Cambridge (2002)
6. W3C: OWL Web Ontology Language Overview (2004) W3C Recommendation (February 10, 2004), http://www.w3.org/TR/2004/REC-owl-features-20040210/
7. Horrocks, I., Patel-Schneider, P.F.: Reducing OWL entailment to description logic satisfiability. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 17–29. Springer, Heidelberg (2003)
8. Eiter, T., Lukasiewicz, T., Schindlauer, R., Tompits, H.: Combining answer set programming with description logics for the Semantic Web. In: Proceedings KR 2004, pp. 141–151. AAAI Press, Menlo Park (2004)
9. Eiter, T., Ianni, G., Lukasiewicz, T., Schindlauer, R., Tompits, H.: Combining answer set programming with description logics for the Semantic Web. Artif. Intell. 172(12/13), 1495–1539 (2008)
10. Lukasiewicz, T.: A novel combination of answer set programming with description logics for the semantic web. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 384–398. Springer, Heidelberg (2007)
11. Motik, B., Horrocks, I., Rosati, R., Sattler, U.: Can OWL and logic programming live together happily ever after? In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 501–514. Springer, Heidelberg (2006)
12. Heymans, S., de Bruijn, J., Predoiu, L., Feier, C., Van Nieuwenborgh, D.: Guarded hybrid knowledge bases (2007)
13. Giugno, R., Lukasiewicz, T.: P-$\mathcal{SHOQ}(\mathbf{D})$: A probabilistic extension of $\mathcal{SHOQ}(\mathbf{D})$ for probabilistic ontologies in the semantic web. In: Flesca, S., Greco, S., Leone, N., Ianni, G. (eds.) JELIA 2002. LNCS, vol. 2424, pp. 86–97. Springer, Heidelberg (2002)
14. Lukasiewicz, T.: Expressive probabilistic description logics. Artif. Intell. 172(6/7), 852–883 (2008)
15. da Costa, P.C.G.: Bayesian Semantics for the Semantic Web. PhD thesis, George Mason University, Fairfax, VA, USA (2005)
16. da Costa, P.C.G., Laskey, K.B.: PR-OWL: A framework for probabilistic ontologies. In: Proceedings FOIS 2006, pp. 237–249. IOS Press, Amsterdam (2006)
17. Lukasiewicz, T.: Probabilistic description logic programs. Int. J. Approx. Reasoning 45(2), 288–307 (2007)
18. Lukasiewicz, T.: Tractable probabilistic description logic programs. In: Prade, H., Subrahmanian, V.S. (eds.) SUM 2007. LNCS, vol. 4772, pp. 143–156. Springer, Heidelberg (2007)
19. Poole, D.: The independent choice logic for modelling multiple agents under uncertainty. Artif. Intell. 94(1/2), 7–56 (1997)
20. Finzi, A., Lukasiewicz, T.: Structure-based causes and explanations in the independent choice logic. In: Proceedings UAI 2003, pp. 225–232. Morgan Kaufmann, San Francisco (2003)
21. Serafini, L., Stuckenschmidt, H., Wache, H.: A formal investigation of mapping language for terminological knowledge. In: Proceedings IJCAI 2005, Professional Book Center, pp. 576–581 (2005)
22. Predoiu, L., Stuckenschmidt, H.: A probabilistic framework for information integration and retrieval on the Semantic Web. In: Proceedings InterDB 2007 Workshop on Database Interoperability (2007)
23. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Heidelberg (2007)

24. Euzenat, J., Mochol, M., Shvaiko, P., Stuckenschmidt, H., Svab, O., Svatek, V., van Hage, W.R., Yatskevich, M.: First results of the ontology alignment evaluation initiative 2006. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273. Springer, Heidelberg (2006)
25. Euzenat, J., Stuckenschmidt, H., Yatskevich, M.: Introduction to the ontology alignment evaluation, In: Proceedings K-CAP 2005 Workshop on Integrating Ontologies (2005)
26. Horrocks, I., Sattler, U., Tobies, S.: Practical reasoning for expressive description logics. In: Ganzinger, H., McAllester, D., Voronkov, A. (eds.) LPAR 1999. LNCS, vol. 1705, pp. 161–180. Springer, Heidelberg (1999)
27. Faber, W., Leone, N., Pfeifer, G.: Recursive aggregates in disjunctive logic programs: Semantics and complexity. In: Alferes, J.J., Leite, J. (eds.) JELIA 2004. LNCS, vol. 3229, pp. 200–212. Springer, Heidelberg (2004)
28. Gelfond, M., Lifschitz, V.: Classical negation in logic programs and disjunctive databases. New Generation Comput. 9(3/4), 365–386 (1991)
29. Poole, D.: Probabilistic Horn abduction and Bayesian networks. Artif. Intell. 64(1), 81–129 (1993)
30. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Francisco (1988)
31. Frisch, A.M., Haddawy, P.: Anytime deduction for probabilistic logic. Artif. Intell. 69(1/2), 93–122 (1994)
32. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 2nd edn. Prentice Hall, San Francisco (2002)
33. Meilicke, C., Stuckenschmidt, H., Tamilin, A.: Repairing ontology mappings. In: Proceedings AAAI 2007, pp. 1408–1413. AAAI Press, Menlo Park (2007)
34. Wang, P., Xu, B.: Debugging ontology mapping: A static method. Computing and Informatics 27(1), 21–36 (2008)
35. McIlraith, S.A., Son, T.C., Zeng, H.: Semantic Web Services. IEEE Intelligent Systems 16(2), 46–53 (2001)
36. McIlraith, S.A., Son, T.C.: Adapting Golog for composition of Semantic Web Services. In: Proceedings KR 2002, pp. 482–496. Morgan Kaufmann, San Francisco (2002)
37. Narayanan, S., McIlraith, S.A.: Simulation, verification and automated composition of Web Services. In: Proceedings WWW 2002, pp. 77–88. ACM Press, New York (2002)
38. McIlraith, S.A., Martin, D.L.: Bringing semantics to Web Services. IEEE Intelligent Systems 18(1), 90–93 (2003)
39. McCarthy, J., Hayes, P.J.: Some philosophical problems from the standpoint of Artificial Intelligence. In: Machine Intelligence, vol. 4, pp. 463–502. Edinburgh University Press (1969)
40. Reiter, R.: Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems. MIT Press, Cambridge (2001)
41. Levesque, H.J., Reiter, R., Lespérance, Y., Lin, F., Scherl, R.: GOLOG: A logic programming language for dynamic domains. J. Logic Program. 31(1–3), 59–84 (1997)
42. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: DL-Lite: Tractable description logics for ontologies. In: Proceedings AAAI 2005, pp. 602–607. AAAI Press MIT Press (2005)
43. Donini, F.M., Lenzerini, M., Nardi, D., Schaerf, A.: $\mathcal{AL}$-log: Integrating Datalog and description logics. J. Intell. Inf. Syst. 10(3), 227–252 (1998)
44. Levy, A.Y., Rousset, M.C.: Combining Horn rules and description logics in CARIN. Artif. Intell. 104(1/2), 165–209 (1998)
45. Grosof, B.N., Horrocks, I., Volz, R., Decker, S.: Description logic programs: Combining logic programs with description logics. In: Proceedings WWW 2003, pp. 48–57. ACM Press, New York (2003)

46. Motik, B., Sattler, U., Studer, R.: Query answering for OWL-DL with rules. J. Web Sem. 3(1), 41–60 (2005)
47. Heymans, S., Van Nieuwenborgh, D., Vermeir, D.: Nonmonotonic ontological and rule-based reasoning with extended conceptual logic programs. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, pp. 392–407. Springer, Heidelberg (2005)
48. Rosati, R.: On the decidability and complexity of integrating ontologies and rules. J. Web Sem. 3(1), 61–73 (2005)
49. Rosati, R.: $\mathcal{DL}+log$: Tight integration of description logics and disjunctive Datalog. In: Proceedings KR 2006, pp. 68–78. AAAI Press, Menlo Park (2006)
50. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: A Semantic Web rule language combining OWL and RuleML, W3C Member Submission (May 2004), `http://www.w3.org/Submission/SWRL/`
51. Angele, J., Boley, H., de Bruijn, J., Fensel, D., Hitzler, P., Kifer, M., Krummenacher, R., Lausen, H., Polleres, A., Studer, R.: Web Rule Language (WRL), W3C Member Submission (September 2005), `http://www.w3.org/Submission/WRL/`
52. Heymans, S., Van Nieuwenborgh, D., Vermeir, D.: Guarded open answer set programming. In: Baral, C., Greco, G., Leone, N., Terracina, G. (eds.) LPNMR 2005. LNCS, vol. 3662, pp. 92–104. Springer, Heidelberg (2005)
53. Motik, B., Sattler, U.: A comparison of reasoning techniques for querying large description logic aBoxes. In: Hermann, M., Voronkov, A. (eds.) LPAR 2006. LNCS, vol. 4246, pp. 227–241. Springer, Heidelberg (2006)
54. Predoiu, L., Stuckenschmidt, H.: Probabilistic extensions of Semantic Web languages — a survey. In: The Semantic Web for Knowledge and Data Management: Technologies and Practices. Idea Group (to appear)
55. Koller, D., Levy, A.Y., Pfeffer, A.: P-CLASSIC: A tractable probabilistic description logic. In: Proceedings AAAI 2007, pp. 390–397. AAAI Press, Menlo Park (1997)
56. Ding, Z., Peng, Y., Pan, R.: BayesOWL: Uncertainty modeling in Semantic Web ontologies. In: Soft Computing in Ontologies and Semantic Web, pp. 3–28. Springer, Heidelberg (2006)
57. Nottelmann, H., Fuhr, N.: Adding probabilities and rules to OWL Lite subsets based on probabilistic Datalog. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 14(1), 17–42 (2006)
58. De Giacomo, G., Iocchi, L., Nardi, D., Rosati, R.: Moving a robot: The KR&R approach at work. In: Proceedings KR 1996, pp. 198–209. Morgan Kaufmann, San Francisco (1996)
59. Iocchi, L., Lukasiewicz, T., Nardi, D., Rosati, R.: Reasoning about actions with sensing under qualitative and probabilistic uncertainty. ACM Trans. Computat. Logic (in press)
60. Baader, F., Lutz, C., Milicic, M., Sattler, U., Wolter, F.: Integrating description logics and action formalisms: First results. In: Proceedings AAAI 2005, pp. 572–577. AAAI Press/ MIT Press (2005)
61. Milicic, M.: Planning in action formalisms based on DLs: First results. In: Proceedings DL 2007. CEUR Workshop Proceedings, vol. 250 (2007) CEUR-WS.org
62. Drescher, C., Thielscher, M.: Integrating action calculi and description logics. In: Hertzberg, J., Beetz, M., Englert, R. (eds.) KI 2007. LNCS, vol. 4667, pp. 68–83. Springer, Heidelberg (2007)

Appendix A: Proofs

Proof of Theorem 7.1. Recall first that KB is consistent iff KB has an answer set Pr, which is a probabilistic interpretation Pr such that (i) every interpretation $I \subseteq HB_\Phi$

with $Pr(I) > 0$ is an answer set of the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$ for some total choice B of $\mathcal{C}$, and (ii) $Pr(\bigwedge_{p\in B} p) = \mu(B)$ for each total choice B of $\mathcal{C}$.

($\Rightarrow$) Suppose that KB is consistent. We now show that the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$ is consistent, for every total choice B of $\mathcal{C}$ with $\mu(B) > 0$. Towards a contradiction, suppose the contrary. That is, $(L, P \cup \{p \leftarrow \mid p \in B\})$ is not consistent for some total choice B of $\mathcal{C}$ with $\mu(B) > 0$. It thus follows that $Pr(\bigwedge_{p\in B} p) = 0$. But this contradicts $Pr(\bigwedge_{p\in B} p) = \mu(B) > 0$. This shows that $(L, P \cup \{p \leftarrow \mid p \in B\})$ is consistent, for every total choice B of $\mathcal{C}$ with $\mu(B) > 0$.

($\Leftarrow$) Suppose that the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$ is consistent, for every total choice B of $\mathcal{C}$ with $\mu(B) > 0$. That is, there exists some answer set I_B of $(L, P \cup \{p \leftarrow \mid p \in B\})$, for every total choice B of $\mathcal{C}$ with $\mu(B) > 0$. Let the probabilistic interpretation Pr be defined by $Pr(I_B) = \mu(B)$ for every total choice B of $\mathcal{C}$ with $\mu(B) > 0$ and by $Pr(I) = 0$ for all other $I \subseteq HB_\Phi$. Then, Pr is an interpretation that satisfies (i) and (ii). That is, Pr is an answer set of KB. Thus, KB is consistent. □

Proof of Theorem 7.2. The statement of the theorem is immediate for the three cases where $b = 0$ or $c = 0$, since $b = 0$ (resp., $c = 0$) iff $Pr(\alpha \wedge \beta) = 0$ (resp., $Pr(\alpha \wedge \neg\beta) = 0$) for all models Pr of KB. Thus, in the following, suppose $b \neq 0$ and $c \neq 0$. Observe first that the probability $\mu(B)$ of all total choices B of $\mathcal{C}$ such that δ is true in all (resp., some) answer sets of the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$ definitely contributes (resp., "can be made to" contribute) to the probability $Pr(\delta)$. As for the lower bound, every $\mu(B)$ of all total choices B of $\mathcal{C}$ such that $\alpha \wedge \beta$ is true in all answer sets of $(L, P \cup \{p \leftarrow \mid p \in B\})$ definitely contributes to both $Pr(\alpha \wedge \beta)$ and $Pr(\alpha)$. Hence, we obtain the smallest value of $Pr(\alpha\wedge\beta) \,/\, Pr(\alpha) = Pr(\alpha\wedge\beta) \,/\, (Pr(\alpha\wedge\beta) + Pr(\alpha\wedge\neg\beta))$, if we additionally take the probabilities $\mu(B)$ of all total choices B of $\mathcal{C}$ such that $\alpha\wedge\neg\beta$ is true in some answer sets of $(L, P \cup \{p \leftarrow \mid p \in B\})$ and make them contribute to $Pr(\alpha)$. Similarly, as for the upper bound, every $\mu(B)$ of all total choices B of $\mathcal{C}$ such that $\alpha \wedge \neg\beta$ is true in all answer sets of $(L, P \cup \{p \leftarrow \mid p \in B\})$ definitely does not contribute to $Pr(\alpha \wedge \beta)$ but contributes to $Pr(\alpha)$. Thus, we obtain the largest value of $Pr(\alpha \wedge \beta) \,/\, Pr(\alpha) = Pr(\alpha \wedge \beta) \,/\, (Pr(\alpha \wedge \beta) + Pr(\alpha \wedge \neg\beta))$ if we additionally take the $\mu(B)$'s of all total choices B of $\mathcal{C}$ such that $\alpha \wedge \beta$ is true in some answer sets of $(L, P \cup \{p \leftarrow \mid p \in B\})$ and make them contribute to both $Pr(\alpha \wedge \beta)$ and $Pr(\alpha)$. □

Proof of Theorem 7.3. Since the number of all total choices B of $\mathcal{C}$ is finite, Algorithm tight_answer always terminates. Let a', b', c', d', i', and v' be the final values of the variables a, b, c, d, i, and v, respectively. If $v' = 0$, then the algorithm has processed all the total choices B of $\mathcal{C}$ with $\mu(B) > 0$. Hence, in this case, by Theorem 7.2, the algorithm returns the exact tight answer for Q to KB. Suppose now $v' > 0$ (and thus $i' \leqslant k$, $b' \neq 0$, and $c' \neq 0$). Then, the returned lower and upper bounds are given by $l' = \frac{a'}{a'+c'} \leqslant \frac{a'+v'}{a'+c'}$ and $u' = \frac{b'}{b'+d'} \leqslant \frac{b'+v'}{b'+d'}$, respectively. Furthermore, the exact tight lower and upper bounds l and u are of the form $l' \leqslant l = \frac{a'+v_a}{a'+v_a+c'-v_c} \leqslant \frac{a'+v'}{a'+c'}$ and $u' \leqslant u = \frac{b'+v_b}{b'+v_b+d'-v_d} \leqslant \frac{b'+v'}{b'+d'}$, respectively. So, $|l - l'| + |u - u'| \leqslant \frac{v'}{a'+c'} + \frac{v'}{b'+d'} \leqslant \epsilon$. □

Proof of Theorem 7.4. Immediate by the proof of Theorem 7.3. □

Proof of Theorem 7.5. We first show membership in NEXP$^{\text{NP}}$. By Theorem 7.1, we check whether the disjunctive dl-program $(L, P \cup \{p \leftarrow \mid p \in B\})$ is consistent, for every total choice B of $\mathcal{C}$ with $\mu(B) > 0$. Observe then that the number of all total choices B of $\mathcal{C}$ with $\mu(B) > 0$ is exponential in the size of $\mathcal{C}$. As shown in [10], deciding whether a disjunctive dl-program has an answer set is in NEXP^{NP}. In summary, this shows that deciding whether KB is consistent is in NEXP^{NP}.

Hardness for NEXP^{NP} follows from the NEXP^{NP}-hardness of deciding whether a disjunctive dl-program has an answer set [10], since by Theorem 7.1 a disjunctive dl-program $KB = (L, P)$ has an answer set iff the probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$ has an answer set, for the choice space $\mathcal{C} = \{\{a\}\}$, the probability function $\mu(a) = 1$, and any ground atom $a \in HB_{\Phi}$ that does not occur in $\mathit{ground}(P)$. □

Proof of Theorem 7.6. We first show membership in $\text{co-NEXP}^{\text{NP}}$. We show that deciding whether $(\beta|\alpha)[l, u]$ is not a consequence of KB under the answer set semantics is in NEXP^{NP}. Observe that $(\beta|\alpha)[l, u]$ is not a consequence of KB under the answer set semantics iff there are sets $\mathcal{B}_{\beta\wedge\alpha}$ and $\mathcal{B}_{\beta\wedge\neg\alpha}$ of total choices B of $\mathcal{C}$ with $\mu(B) > 0$ such that either (a.1) $\beta \wedge \alpha$ is true in some answer set of $(L, P \cup \{p \leftarrow | \, p \in B\})$, for every $B \in \mathcal{B}_{\beta\wedge\alpha}$, (a.2) $\beta \wedge \neg\alpha$ is false in some answer set of $(L, P \cup \{p \leftarrow | \, p \in B\})$, for every $B \in \mathcal{B}_{\beta\wedge\neg\alpha}$, and (a.3) $b > u \cdot (b + d)$, where $b = \sum_{B \in \mathcal{B}_{\beta\wedge\alpha}} \mu(B)$ and $d = 1 - \sum_{B \in \mathcal{B}_{\beta\wedge\neg\alpha}} \mu(B)$, or (b.1) $\beta \wedge \alpha$ is false in some answer set of $(L, P \cup \{p \leftarrow | \, p \in B\})$, for every $B \in \mathcal{B}_{\beta\wedge\alpha}$, (a.2) $\beta \wedge \neg\alpha$ is true in some answer set of $(L, P \cup \{p \leftarrow | \, p \in B\})$, for every $B \in \mathcal{B}_{\beta\wedge\neg\alpha}$, and (a.3) $a > l \cdot (a + c)$, where $a = 1 - \sum_{B \in \mathcal{B}_{\beta\wedge\alpha}} \mu(B)$ and $c = \sum_{B \in \mathcal{B}_{\beta\wedge\neg\alpha}} \mu(B)$. Since the number of all total choices B of $\mathcal{C}$ with $\mu(B) > 0$ is exponential in the size of $\mathcal{C}$, guessing $\mathcal{B}_{\beta\wedge\alpha}$ and $\mathcal{B}_{\beta\wedge\neg\alpha}$ can be done in nondeterministic exponential time. As shown in [10], deciding whether $\beta \wedge \alpha$ or $\beta \wedge \neg\alpha$ is true or false in some answer set of a disjunctive dl-program is in NEXP^{NP}. In summary, guessing the sets $\mathcal{B}_{\beta\wedge\alpha}$ and $\mathcal{B}_{\beta\wedge\neg\alpha}$, and verifying that either (a.1)–(a.3) or (b.1)–(b.3) hold is in NEXP^{NP}. Hence, deciding whether $(\beta|\alpha)[l, u]$ is not a consequence of KB under the answer set semantics is in NEXP^{NP}. It thus follows that deciding whether $(\beta|\alpha)[l, u]$ is a consequence of KB under the answer set semantics is in $\text{co-NEXP}^{\text{NP}}$.

Hardness for $\text{co-NEXP}^{\text{NP}}$ follows from the $\text{co-NEXP}^{\text{NP}}$-hardness of deciding whether a ground atom q is true in all answer sets of a disjunctive dl-program [10], since by Theorem 7.2 a ground atom q is true in all answer sets of a disjunctive dl-program $KB = (L, P)$ iff $(q)[1, 1]$ is a consequence of the probabilistic dl-program $KB = (L, P, \mathcal{C}, \mu)$ under the answer set semantics, for the choice space $\mathcal{C} = \{\{a\}\}$, the probability function $\mu(a) = 1$, and any $a \in HB_{\Phi}$ that does not occur in $\mathit{ground}(P)$. □

Proof of Theorem 8.2. As shown in [10], deciding the existence of (and computing) the answer set of a stratified normal dl-program (L, P) with L in *DL-Lite* can be done in polynomial time in the data complexity. Notice then that in the case of data complexity, the choice space $\mathcal{C}$ (and so the set of all its total choices) is fixed. By Theorems 7.1 and 7.2, it thus follows that the problems of (a) deciding whether KB has an answer set, and (b) computing the reals $l, u \in [0, 1]$ for a given ground conditional event $\beta|\alpha$ such that $KB \,\|\!\!\sim_{\mathit{tight}} (\beta|\alpha)[l, u]$ can both be done in polynomial time in the data complexity. □

Intensional First-Order Logic for P2P Database Systems

Zoran Majkić

ETF, Applied Mathematics Department
University of Belgrade, Serbia
majkic@etf.bg.ac.yu
http://www.geocities.com/zoran_it/

Abstract. The meaning of concepts and views defined over a database ontology can be considered as intensional objects which have a particular extension in a given possible world: for instance in the *actual* world. Thus, non invasive mapping between completely independent peer databases in a P2P systems can be naturally specified by the set of couples of views, which have the same meaning (intension), over two different peers. Such a kind of mapping has very different semantics from standard view-based mappings based on material implication, commonly used for Data Integration Systems. The introduction of an intensional equivalence generates the quotient intensional FOL fundamental for a query answering in P2P systems. In this paper we introduce this formal intensional FOL by fusing Bealer's intensional algebraic FOL with a possible-world semantics of the Montague's FOL modal approach to natural language. We modify the Bealer's intensional algebra in order to deal with relational databases and views, by introducing the join operation of relational algebra. Then we adopt the S5 Kripke frame in order to define an *intensional equivalence* relation between views for peer databases. Finally, we define an embedding of P2P database system into this quotient intensional FOL, and the computing of its extensionalization mapping in the actual Montague's world.

1 Introduction

Ontologies play a prominent role on the Semantic Web. An ontology specifies a conceptualization of a domain in terms of concepts, attributes and relations. A key challenge in building the Semantic Web is finding semantic mappings among the ontologies (relational schemas of peer databases). Given the de-centralized nature of the development of the Semantic Web, there will be an explosion in the number of ontologies. Many of these ontologies will describe similar domains, but using different terminologies, and others will have overlapping domains. To integrate data from disparate ontologies, we must know the *semantic correspondence* between their elements [1]. Recently a number of different architecture solutions have been introduced [2,3,4,5,6,7].

The first seminal work which introduces *autoepistemic semantics* for P2P databases, based on known (i.e. certain) answers from peers is presented by Lenzerini and the author in [8], successively compared to Franconi's approach in [9]. This modal logic framework for P2P database systems guarantees also the *decidability* for query answering, non supported by first-order semantics. This autoepistemic semantics for peer databases can be used as an example for the foundation of a sound and complete implementation of a query answering mechanism for a *single* peer database. The intensional logic for *inter-peer* mappings, instead, extends this peer database semantics into the whole P2P system query-answering semantics. This is the principal issue of this work.

Contemporary use of the term 'intension' derives from the traditional logical doctrine that an idea has both an extension and an intension. Although there is divergence in its formulation, it is accepted that the extension of an idea (or concept) consists of the subjects to which the idea applies, and the intension consists of the attributes implied by the idea. The intension is the concept expressed by the expression, and the extension is the set of items to which the expression applies. This usage resembles Frege's use of 'Bedeutung' and 'Sinn'. Intensional entities are such things as concepts, propositions and properties. What make them 'intensional' is that they violate the principle of extensionality; the principle that extensional equivalence implies identity. All (or most) of these intensional entities have been classified at one time or another as kinds of Universals [10]. We adopt the non-reductionist approaches [11] and we will show how they correspond to possible world semantics.

The fundamental entities are *intensional abstracts* or so called, 'that'-clauses. We assume that they are singular terms; Intensional expressions like 'believe', 'mean', 'assert', 'know', are standard two-place predicates that take 'that'-clauses as arguments. Expressions like 'is necessary', 'is true', and 'is possible' are one-place predicates that take 'that'-clauses as arguments. For example, in the intensional sentence "it is necessary that A", where A is a proposition, the 'that A' is denoted by the $\lessdot A \gtrdot$, where $\lessdot \_ \gtrdot$ is the intensional abstraction operator which transforms a logic formula into a *term*. So that the sentence "it is necessary that A" is expressed by the logic atom $N(\lessdot A \gtrdot)$, where N is the unary predicate 'is necessary'. In this way we are able to avoid having higher-order syntax for our *intensional* logic language (predicates appear in variable places of other predicates),as, for example HiLog [12] where the *same* symbol may denote a predicate, a function, or an atomic formula.

In what follows we abbreviate $A \Rightarrow B \wedge B \Rightarrow A$ by $A \equiv B$.

Comparative analysis. We are able to individuate at least two extreme scenarios for inter-peer mappings presented in the literature developed from the initial article [8] followed in Lenzerini's approach for Data Integration Systems: *strongly-coupled* and *weakly-coupled* P2P database systems. Let us consider two different peer databases, $P_i,\ P_k$, named John and Peter, with K_i, K_k their "know" operators respectively.

We consider *a view* definition $q_k(\mathbf{x})$ as a conjunctive query, with a tuple of variables in $\mathbf{x}$, $head(q_k) \leftarrow body(q_k)$ where $body(q_k)$ is a sequence $b_1, b_2, ..., b_m$, where each b_j is an atom over a global relation name of a peer P_i. In what follows we will consider a view as a virtual predicate with a tuple $\mathbf{x}$ of free variables in the head of a query. Let $q_{P_i}(\mathbf{x})$ and $q_{P_k}(\mathbf{x})$ be a two views (conjunctive queries) over P_i and P_j respectively, which represent the same concept "Italian art in the 15'th century" for their local knowledge. Then these two approaches may be paraphrased by:

1. The 'strong' (*extensional*) Global or Local As View (GLAV) mapping which directly extends the data-integration paradigm used for a single peer, for P2P systems [13] also. It may be paraphrased, for example, by an *imperative* sentence '*John must know all the facts about "Italian art in the 15'th century" known by Peter*'.

This *extensional* multi-modal mapping has been introduced in [6] by the formula $K_i q_{P_i}(\mathbf{x}) \Rightarrow K_k q_{P_k}(\mathbf{x})$, where $' \Rightarrow'$ is the standard (material) logic implication. It

uses a single S5 modality [9,7], and recently adopted, a K45 multi-modality [14]. With such a mapping the local knowledge of P_k *is not independent* from the local knowledge of other peers. The implication between concepts of these two peers will cause that the facts known by P_i about this concept will be imposed as the local knowledge of P_k (must be true for P_k also when such facts do not exist in its local knowledge), with the consequence that such forcing can make the knowledge of P_k inconsistent.
2. The 'weak' (*intensional*) mapping, may be paraphrased by a *belief*-sentence '*John believes that Peter also knows something about "Italian art in the 15'th century"*'. Such a mapping is weaker than internal extensional GLAV mappings of a peer, so that it grantees the independence of peer individuality also in the presence of mappings between peers. With such mappings we may distinguish the answers of a peer 'John' from answers of 'Peter', i.e., we may distinguish the local independent answer of each peer. This *intensional* mapping is defined by the "formula" $K_i q_{P_i}(\mathbf{x}) \approx_{in} K_k q_{P_k}(\mathbf{x})$, where $' \approx_{in}'$ is the informal symbol for the intensional equivalence [15,6,16,17,18], and formally in Definition 6 by the logic modal formula $\Diamond q_{P_i}(\mathbf{x}) \equiv \Diamond q_{P_k}(\mathbf{x})$ of the intensional FOL introduced in this paper. This mapping tells only that these two peers have the knowledge about the *same* concept, without any constraint for extensions of this concept in these two peers respectively.

The motivation for the second approach, adopted in this paper, is the following: The logic implications used in order to impose the strong semantic relationships between peer ontologies, based on GLAV mappings, are much too *constrictive*. Internal peer mapping and external mapping between peers have the *same* (extensional) expressive (GLAV) power, so that peer individuality is completely destroyed.

Example 1. Let us consider strong mapping, used in [7], with $K_i q_{P_i}(x_1, x_2) \Rightarrow K_k q_{P_k}(x_1, x_2)$, $K_j q_{P_j}(x_1, x_2) \Rightarrow K_k q_{P_k}(x_1, x_2)$, with local knowledge $K_i q_{P_i}(c_1, c_2)$ and $K_j q_{P_j}(c_1, c_3)$ of peers P_i, P_j respectively, with $c_2 \neq c_3$, and with the key-constraint for the attribute x_1 in the data schema of the third peer P_k. Then both "imported" facts for P_k, $K_k q_{P_k}(c_1, c_3)$ and $K_k q_{P_k}(c_1, c_2)$ have to be true making its logic *inconsistent*, independently of its local knowledge! Moreover, the query agent must know instantly the knowledge of all peers in order to be able to know the extension of local knowledge of *a single* peer: it reassembles a kind of global and centrally controlled knowledge (the centralized controlled knowledge for all peers in the network must exist). If we consider thousands of dynamically modified peers, each of them with millions of local facts, we can understand that this query-agent has to have a practically impossible capacity of memory and elaboration. Consequently it is not possible to map peers into a grid for parallel query computation of each independent peer answer.

All these problems are avoided in the second, weak, approach, where a query agent will consider only one peer at time and will use inter-peer mappings only in order to move to another peers which have some knowledge about the user query [19,20]. □

The detailed differences between these two approaches above can be found also in the recent article in the Special issue on Emergent Semantics [15]. So, we will expose only the comparative analysis with this last, and presumably most recent, article.

What we argue is the *full* epistemic independency of peer databases (where there is not any imposition of local knowledge of one peer into the local knowledge of some other peer). The peers can change their ontology and/or extension of their knowledge independently and without any communication to other peers. In this way we intend to obtain very robust P2P systems, able to answer to user queries also when intended mappings between peers do not correspond the modified ontologies (relational database schemas) of peers. We are able also to naturally map P2P database systems into the grid computation. In fact, having fully independent peers it is enough now to associate each pair (peer, query formulae) to a particular resource of grid computing, in order to obtain a known answer from such a peer.

In the paper [15] such a framework has been defined for the P2P database systems and semantics for intensional equivalence based on Montague's modal semantics for natural languages. But, this definition is not part of the *logic framework* for the query answering in a P2P system. It is instead considered as a part of a sound algebraic query rewriting *algorithm* only, and is used as a functional mean in order to define the global semantics for the standard extensional FOL of a P2P semantics.

The full, complete logic integration of the *intensional equivalence* for a query answering in P2P systems needed the definition of the new *intensional* FOL. The formalization of non omniscient intensional contextual reasoning for query-agents in P2P systems within this logic framework can be found in [19].

The main motivation for this paper is to provide clear *semantics* for P2P database systems with *weak mapping* between peers. We intend to provide a clear mathematical framework for its query answering computation which, successively, can be implemented by a massive grid computing framework.

The main contributions in this paper are the following: We define a two-level modal logic framework for P2P database systems. At the higher level we define the Intensional S5 modal FOL with the intensional identity, extended by the new intensional equivalence which can be used for inter-peer mappings. The extensionalization function of the intensional FOL for a P2P system, in the actual world, is modeled by the lower, "computational" level. This level is represented as an *extensional* (standard) multi-modal logic where the set of worlds is the union of preferred subsets of minimal Herbrand models of each peer database. We extend the epistemic modal logic semantics of peers into the whole (global) P2P system, and we provide a formal definition for the global P2P epistemic modal operator $\mathcal{K}$ ("know").

The Plan. After a brief preliminary for P2P systems, in Section 2 we introduce a syntax for an intensional FOL with intensional abstraction operator. We combine the work developed by Bealer on intensional algebra and the framework of Montague modal logic based on possible worlds, in order to define formally the *intensional equivalence* between logic formulae.

In Section 3 we present an embedding of a P2P system into this intensional logic in order to obtain the higher level modal framework for P2P systems. Finally, in Section 4 we define the lower level modal framework, which models the *extensionalization function* of this Intensional logic in the actual world, which is represented by the formal semantics for extensional (standard) epistemic multi-modal P2P logic based on the P2P mappings. We define a *global* P2P modal operator for query answering, based on sound query rewriting algorithms, used to obtain the answers from intensionally equivalent rewritten queries over peers of a P2P network.

1.1 Technical Preliminaries: Peer-to-Peer Systems

In what follows we will consider reach ontology of peer databases, formally expressed as a global schema of a Data Integration System (DIS). A DIS [21] is a triple $\mathcal{I}_i = (\mathcal{G}_i, \mathcal{S}_i, \mathcal{M}_i)$, where $\mathcal{G}_i = (\mathcal{O}_i, \Sigma_{T_i})$ is a global schema, expressed in a language $\mathcal{L}_{\mathcal{O}}$ over an alphabet $\mathcal{A}_{\mathcal{G}_i}$, Σ_{T_i} are the integrity constraints, $\mathcal{S}_i$ is a source schema and $\mathcal{M}_i$ is a set of mappings between a global relational database schema (ontology) $\mathcal{O}_i$ and a source relational schema $\mathcal{S}_i$ of data extracted by wrappers. In what follows we will consider the case of Global-As-View (GAV) mappings between source and global schema, with existential quantifiers also included (for mapping an incomplete information from source to global schema). A DIS with constraints (for example, key-constraints) can become a *locally inconsistent* w.r.t. data sources extracted by wrappers. Such local inconsistences can be avoided by inconsistency repairing technics [22].

We assume that logical theory for a peer database (see for example [23]) has *one or more* minimal 2-valued Herbrand models, one for each possible completion of incomplete Web information, or possible local inconsistency repairing. In practice we will use the *preferred* Herbrand models only, corresponding to the subset of *minimal* inconsistency repairing [22]. Consequently the logic theory of a peer database is generally non monotonic (in fact it has a cumulative non monotonic inference [24]). It is known also that a global schema of DIS with key and foreign-key constraints can have also an infinite canonical database with the Skolem constants [25].

We conceive a peer P_i as a software module, which encapsulates a DIS $\mathcal{I}_i$. The internal structure of a peer database is hidden to users. It is encapsulated in the way that only its logical relational schema $\mathcal{O}_i$ can be seen by users. A peer database is able to respond to the union of conjunctive queries by *known* answers (true in all models of a peer-database). Consequently, we consider that each peer database P_i is an independent epistemic logic theory with *incomplete* information where all local inconsistencies are repaired. Such a peer, for a given conjunctive query $q(x)$ over its ontology, responds by known answers (true in a preferred subset of all minimal Herbrand models of such theory) only. More interested readers can see in [24] the kind of plausible query answering for such a peer (DIS) with incomplete and inconsistent information in the framework of classic 2-valued logic; the alternative solution for databases, with incomplete and locally inconsistent information, based on Autoepistemic bilattice-based logic programming can be found in [26]. Thus, we consider that each peer P_i is an epistemic normal modal logic theory, where the possible worlds are the preferred Herbrand models of a peer database, with epistemic operator "peer P_i knows", K_i.

Notice that such epistemic semantics of peers is presented in [6] based on the *hybrid mono-modal* language with a unique universal modal operator $\Box$ [27], where $K_i = @_i\Box$. The modal operator, @, for this hybrid logic enables the "retrieving" of worlds: a formula of the form $@_i\varphi$ is an instruction to *move* to the world labeled by the variable i and evaluate φ there.

P2P Network Definition: Syntax [6]

In order to be able to share the knowledge with other peers P_j in the network $\mathcal{N}$, each peer P_i has also an export-interface module $\mathcal{M}^{ij}$ composed by groups of ordered pairs of intensionally equivalent logical views (conjunctive queries over peer's ontologies), denoted by (q_i, q_j). In what follows we will consider only the intensional version for P2P mapping based on considerations presented in [18].

Definition 1. *The P2P network system $\mathcal{N}$ is composed by $2 \leq N$ independent peers, where each peer module P_i is defined as follows:* $P_i := \langle \mathcal{O}_i,\ \mathcal{M}_i \rangle$, *where* $\mathcal{M}_i =< \mathcal{M}^{i1}, ..., \mathcal{M}^{iN} >$ *is an interface tuple with* $\mathcal{M}^{ij}$, $1 \leq j \leq N$ *a (possibly empty) interface to other peer* P_j *in the network, defined as a group of intensionally equivalent query connections, denoted by* $(q^{ij}_{1k}, q^{ij}_{2k})$ *where* $q^{ij}_{1k}(\boldsymbol{x})$ *is a conjunctive query defined over* $\mathcal{O}_i$*, while* $q^{ij}_{2k}(\boldsymbol{x})$ *is a conjunctive query defined over the ontology* $\mathcal{O}_j$ *of the connected peer* P_j *:* $\mathcal{M}^{ij} = \{(q^{ij}_{1k}, q^{ij}_{2k}) \mid\ 1 \leq k \leq\mid ij \mid\}$,
and $\mid ij \mid$ *is the total number of query connections of the peer* P_i *toward a peer* P_j*.*

Intuitively, when an user defines a conjunctive query over the ontology $\mathcal{O}_i$ of a peer P_i, the intensionally equivalent concepts between this peer and other peers will be used in order to obtain the answers from a P2P system.

They will be a "bridge" which a query agent can use in order to rewrite the original user query over a peer P_i into an *intensionally equivalent* query over a peer P_j.

The answers of other peers will be epistemically considered as *possible* answers because the are based on the *belief* which peer P_i has about the knowledge of peer P_j. This belief is formally represented by the supposition of a peer P_i that a pair of queries $(q^{ij}_{1k}, q^{ij}_{2k}) \in \mathcal{M}^{ij}$ is intensionally equivalent.

Example 2. Let us consider the P2P system in a Fig.1:

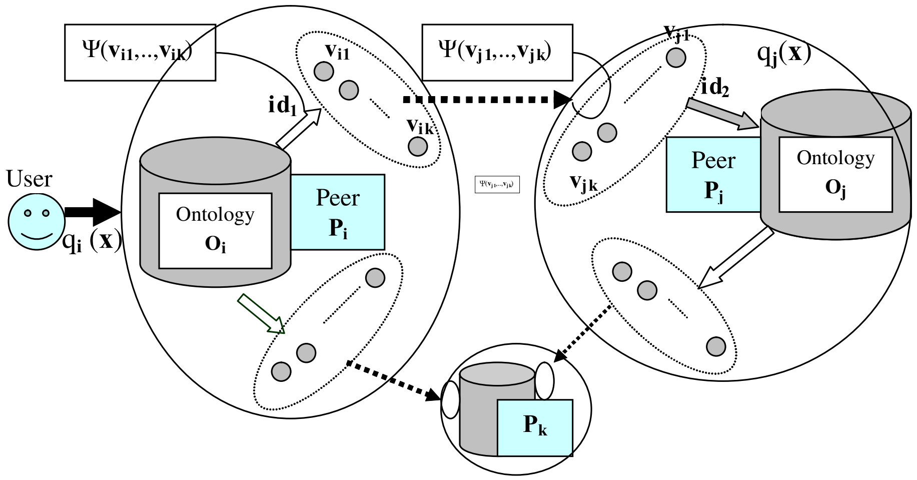


Fig. 1. Intensional mappings between peers

Let us consider a peer P_i, with an ontology $\mathcal{O}_i$ and an interface $\mathcal{M}^{ij} = \{(v_{im}, v_{jm}) \mid\ 1 \leq m \leq k\}$ toward a peer P_j, and a peer P_j with an ontology $\mathcal{O}_j$.

The query answering for a given user query can be obtained by the following step of query-rewriting between two semantically interconnected peers. The idea is the following: given a user query $q_i(\mathbf{x})$ over a peer P_i, a query agent will rewrite it (if it is possible) into the *identical* query $\Psi(v_{i1}, ..., v_{ik})$ over the set of views $\{v_{i1}, ..., v_{ik}\}$ of a peer P_i. Then it will use the set of intensional equivalences $(v_{im}, v_{jm}) \in \mathcal{M}^{ij}, 1 \leq m \leq k$, in order to obtain the *intensionally equivalent* query $\Psi(v_{j1}, ..., v_{jk})$ over the set of views $\{v_{j1}, ..., v_{jk}\}$ of a peer P_j. After that it will rewrite this query into the *identical* query $q_j(\mathbf{x})$ over the ontology $\mathcal{O}_j$ of the peer P_j.

The *known answers* of both peers P_i, P_j to the queries $q_i(\mathbf{x})$ and $q_j(\mathbf{x})$ will constitute the subset of the global P2P answer to the original user query. Another possible answer to the same user query can be obtained by the similar method from the intensionally equivalent queries over a peer P_k obtained by using the intensional mappings from P_i to P_k and from P_j to P_k respectively.

2 Intensional FOL Language and Intensional Equivalence

In the First-order logic (FOL) with intensional abstraction we have more fine distinction between an atom A and its use as a *term* "that A", denoted by $\lessdot A \gtrdot$ and considered as intensional "name", inside some other predicate. For example, we may have the first-order formula $\neg A \wedge P(t, \lessdot A \gtrdot)$ instead of the syntactically second-order HiLog formula $\neg A \wedge P(t, A)$.

Definition 2. *The syntax of the First-order Logic language with intensional abstraction* $\lessdot\gtrdot$*, called* $\mathcal{L}_\omega$ *in [28], is as follows:*

Logic operators $(\wedge, \neg, \exists)$*; Predicate letters in P (functional letters are considered as a particular case of predicate letters); Variables* $x, y, z, ..$ *in* Var*; Abstraction* $\lessdot\_\gtrdot$*, and punctuation symbols (comma, parenthesis). With the following simultaneous inductive definition of* term *and* formula*:*

1. All variables and constants (0-ary functional letters in P) are terms.
2. If $t_1, ..., t_k$ *are terms, then* $A(t_1, ..., t_k)$ *is a formula (*$A \in P$ *is a k-ary predicate letter).*
3. If A and B are formulae, then $(A \wedge B)$*,* $\neg A$*, and* $(\exists x)A$ *are formulae.*
4. If A is a formula and $\alpha =< x_1, ..., x_n >$*, is a sequence (tuple) of* distinct *variables (a subset of free variables in A), then* $\lessdot A \gtrdot_\alpha$ *is a term. The externally quantifiable variables are the* free *variables not in* α*. When* $n = 0$*,* $\lessdot A \gtrdot$ *is a term which denotes a proposition, for* $n \geq 1$ *it denotes a n-ary relation-in-intension.*

An occurrence of a variable x_i *in a formula (or a term) is* bound *(*free*) iff it lies (does not lie) within a formula of the form* $(\exists x_i)A$ *(or a term of the form* $\lessdot A \gtrdot_{x_1 ... x_i ... x_m}$*). A variable is free (bound) in a formula iff it has (does not have) a free occurrence in that formula.*

A sentence *is a formula having no free variables. The binary predicate letter* F_1^2 *is singled out as a distinguished logical predicate and formulae of the form* $F_1^2(t_1, t_2)$ *are to be rewritten in the form* $t_1 = t_2$*. The logic operators* $\forall, \vee, \Rightarrow$ *are defined in terms of* $(\wedge, \neg, \exists)$ *in the usual way.*

For example, "x believes that A" is given by formula $B(x, \lessdot A \gtrdot)$ (B is binary 'believe' predicate), "Being a bachelor is the same thing as being an unmarried man" is given by identity of terms $\lessdot B(x) \gtrdot_x = \lessdot U(x) \wedge M(x) \gtrdot_x$ (with B for 'bachelor', U for 'unmarried', and M for 'man', unary predicates).

Let A be any well-formed formula in FOL, and let $x_1, ..., x_m$ be distinct variables, where $m \geq 0$. (It is possible to have free variables in A that are not among these variables $x_1, ..., x_m$. Such variables can be externally quantified). Then $\lessdot A \gtrdot_{x_1,...,x_m}$ is a singular term whose semantics correlate is an intensional entity of degree m. If $m = 0$, the semantics correlate of this singular term is the proposition that A; if $m = 1$, the semantical correlate is the property of bing something x_1 such that A; if $m > 1$, then the semantical correlate is the relation among $x_1, ..., x_m$ such that A.

Certain complex nominative expressions - namely, gerundive and infinitive phrases - are best represented as singular terms of the sort provided by our generalized bracket notation $\lessdot A \gtrdot_{x_1,...,x_m}$, where $m \geq 1$. $\mathcal{L}_\omega$ differs from standard First Order Logic (FOL) only in having these singular terms $\lessdot A \gtrdot_{x_1,...,x_m}$.

We will extend the intensional FOL language of Bealer, given in the introduction, by Definition 2, by other operators for intensional entities. Thus, analogously to Boolean algebras which are extensional models of propositional logic, we introduce an intensional algebra as follows. We consider a non empty domain $\mathcal{D} = D_{-1} \bigcup D_I$, where a subdomain D_{-1} is made of particulars (extensional entities), and the rest $D_I = D_0 \bigcup D_1 ... \bigcup D_n ...$ is made of universals (D_0 for propositions (the 0-ary relation-in- intensions), and $D_n, n \geq 1$, for n-ary relations-in-intension.

Definition 3. *(SYNTAX): Intensional algebra is a structure*
$Alg_{int} = \; < \mathcal{D}, conj, disj, impl, neg, pred, \tau, f, t >$, *with binary operations* $conj : D_I \times D_I \rightarrow D_I$, $pred : D_i \times \mathcal{D} \rightarrow D_{i-1}$, *for* $i \geq 1$, *and unary operation* $neg : D_i \rightarrow D_i$, *for each* $i \geq 0$*; the disjunctions and implications are defined in a standard way by* $disj(u, v) = neg(conj(neg(u), neg(v)))$, $impl(u, v) = disj(neg(u), v)$, *for any* $u, v \in D_I$*;*
τ *is a set of auxiliary operations [29] intended to be semantic counterparts of the syntactical operations of repeating the same variable one or more times within a given formula and of changing around the order of the variables within a given formula;*
f, t *are empty set* $\{\}$ *and set* $\{<>\}$ *(with the empty tuple* $<> \in D_{-1}$ *i.e. the unique tuple of 0-ary relation) which may be thought of as falsity and truth, as those used in the relational algebra, respectively.*

Remark. This definition differs from the original work in [29] where t is defined as $\mathcal{D}$, and $conj : D_i \times D_i \rightarrow D_i$, $i \geq 0$, here we are using the *relational algebra* semantics for the conjunction. So that we are able to support also structural composition for abstracted terms necessary for supporting relational conjunctive queries, as, for example, $\lessdot A(x, y) \wedge B(y, z) \gtrdot_{xyz}$, which is not possible in the reduced syntactic version of Bealer's algebra. In the original work [29] this "algebraization" of the intensional FOL is extended also to logic quantifiers, but for our purpose it is not necessary, because in the embedding of a P2P system into the intensional FOL for query answering, we will use only the predicates from the global schema of each peer databases both with the queries (virtual predicates) used for intensional mapping between peers. □

The distinction between intensions and extensions is important especially because we are now able to have and *equational theory* over intensional entities (as $\lessdot A \gtrdot$), that is predicate and function "names", that is separate from the extensional equality of relations and functions. Thus, intensional FOL has the simple Tarski first-order semantics, with a decidable unification problem, but we need also the actual world mapping which maps any intensional entity to its *actual world extension*. In what follows we will identify a *possible world* by a particular mapping which assigns to intensional entities their extensions in such possible world. It is the direct bridge between intensional FOL and possible worlds representation [30,31,32,33,34], where intension of a proposition is a *function* from a set of possible worlds $\widetilde{\mathcal{W}}$ to truth-values, and properties and functions from $\widetilde{\mathcal{W}}$ to sets of possible (usually not-actual) objects.

In what follows we will use one simplified S5 modal logic framework (we will not consider time as one independent parameter as in Montague's original work) with a model $\mathcal{M} = (\widetilde{\mathcal{W}}, \mathcal{R}, \mathcal{D}, V)$, where $\widetilde{\mathcal{W}}$ is a set of possible worlds, $\mathcal{R}$ is a reflexive, symmetric and transitive accessibility relation between worlds ($\mathcal{R} = \widetilde{\mathcal{W}} \times \widetilde{\mathcal{W}}$), $\mathcal{D}$ is a non-empty domain of individuals given by Definition 3, while V is a function defined for the following two cases:

1. $V : \widetilde{\mathcal{W}} \times F \rightarrow \bigcup_{n<\omega} \mathcal{D}^{\mathcal{D}^n}$, with F a set of functional symbols of the language, such that for any world $w \in \widetilde{\mathcal{W}}$ and a functional symbol $f \in F$, we obtain a function $V(w, f) : \mathcal{D}^{arity(f)} \rightarrow \mathcal{D}$.
2. $V : \widetilde{\mathcal{W}} \times P \rightarrow \bigcup_{n<\omega} \mathbf{2}^{\mathcal{D}^n}$, with P a set of predicate symbols of the language and $\mathbf{2} = \{t, f\}$ is the set of truth values (true and false, respectively), such that for any world $w \in \widetilde{\mathcal{W}}$ and a predicate symbol $p \in P$, we obtain a function $V(w, p) : \mathcal{D}^{arity(p)} \rightarrow \mathbf{2}$, which defines the extension $[p] = \{\mathbf{a} | \mathbf{a} \in \mathcal{D}^{arity(p)}$ and $V(w, p)(\mathbf{a}) = t\}$ of this predicate p in the world w.

The extension of a formula A, w.r.t. a model $\mathcal{M}$, a world $w \in \widetilde{\mathcal{W}}$ and an assignment $g : Var \rightarrow \mathcal{D}$ is denoted by $[A]^{\mathcal{M},w,g}$ or by $[A/g]^{\mathcal{M},w}$ where A/g ia a ground formula obtained from A by assigning values to all its free variables. Thus, if $p \in F \bigcup P$ then for a given world $w \in \widetilde{\mathcal{W}}$ and the assignment function for variables g, $[p]^{\mathcal{M},w,g} = V(w, p) : \mathcal{D}^n \rightarrow \mathbf{2}$, that is, for any set of terms $t_1, .., t_n$, where n is the arity of p, we have $[p(t_1, .., t_n)]^{\mathcal{M},w,g} = V(w, p)([t_1]^{\mathcal{M},w,g}, .., [t_n]^{\mathcal{M},w,g}) \in \mathbf{2}$.

For any formula A, $\mathcal{M} \vDash_{w,g} A$ is equivalent to $[A]^{\mathcal{M},w,g} = t$, means 'A is true in the world w of a model $\mathcal{M}$ for assignment g'. The additional semantic rules relative to the modal operators $\Box$ and $\Diamond$ are as follows:

$\mathcal{M} \vDash_{w,\ g} \Box A$ iff $\mathcal{M} \vDash_{w',\ g} A$ for every w' in $\widetilde{\mathcal{W}}$ such that $w\mathcal{R}w'$.

$\mathcal{M} \vDash_{w,\ g} \Diamond A$ iff there exists a w' in $\widetilde{\mathcal{W}}$ such that $w\mathcal{R}w'$ and $\mathcal{M} \vDash_{w',\ g} A$.

A formula A is said to be *true in a model* $\mathcal{M}$ if $\mathcal{M} \vDash_{w,\ g} A$ for each g and $w \in \widetilde{\mathcal{W}}$.
A formula is said to be *valid* if it is true in each model.

Montague defined the *intension* of a formula A as follows:
$[A]_{in}^{\mathcal{M},g} =_{def} \{w \mapsto [A]^{\mathcal{M},w,g} \mid w \in \widetilde{\mathcal{W}}\}$,
i.e., as graph of the function $[A]_{in}^{\mathcal{M},g} : \widetilde{\mathcal{W}} \rightarrow \bigcup_{w \in \mathcal{W}_N} [A]^{\mathcal{M},w,g}$.

One thing that should be immediately clear is that intensions are more general than extensions: if the intension of an expression is given, one can determine its extension with respect to a particular world but not viceversa, i.e., $[A]^{\mathcal{M},w,g} = [A]_{in}^{\mathcal{M},g}(w)$.

In particular, if c is a non-logical constant (individual constant or predicate symbol), the definition of the extension of c is, $[c]^{\mathcal{M},w,g} =_{def} V(w,c)$. Hence, the intensions of the non-logical constants are the following functions: $[c]^{\mathcal{M},g}_{in} : \widetilde{\mathcal{W}} \rightarrow \bigcup_{w \in \widetilde{\mathcal{W}}} V(w,c)$. The extension of variable is supplied by the value assignment g only, and thus does not differ from one world to the other; if x is a variable we have $[x]^{\mathcal{M},g}_{in} = g(x)$.

Thus the intension of a variable will be a constant function on worlds which corresponds to its extension. Finally, the connection between Bealer's non-reductionistic and Montague's possible world approach to intensional logic can be given by the isomorphism (its meaning is that basically we can use the extensionalization functions in the place of Montague's possible worlds):

$$\mathcal{F} : \widetilde{\mathcal{W}} \simeq \mathcal{E},$$

where $\mathcal{E}$ is a set of possible extensionalization functions: Each extensionalization function $h \in \mathcal{E}$ assigns to the intensional elements of $\mathcal{D}$ an appropriate extension as follows: for each proposition $u \in D_0$, $h(u) \in \mathbf{2} = \{f,t\}$ is its extension (true or false value); for each n-ary relation-in-intension $u \in D_n$, $h(u)$ is a subset of $\mathcal{D}^n$ (n-th Cartesian product of $\mathcal{D}$); in the case of particulars $u \in D_{-1}$, $h(u) = u$. We require that operations $conj, disj$ and neg in this intensional algebra behave in the expected way with respect to each extensionalization function (for example, for all $u \in D_0$, $h(neg(u)) = t$ iff $h(u) = f$, etc..), that is

$$h = h_{-1} + h_0 + \sum_{i \geq 1} h_i : \sum_{i \geq -1} D_i \longrightarrow D_{-1} + \mathbf{2} + \sum_{i \geq 1} \mathcal{P}(D^i)$$

where $h_{-1} = id : D_{-1} \rightarrow D_{-1}$ is identity, $h_0 : D_0 \rightarrow \mathbf{2}$ assigns truth values in $\mathbf{2} = \{f,t\}$, to all propositions, and $h_i : D_i \rightarrow \mathcal{P}(D^i)$, $i \geq 1$, assigns extension to all relations-in-intension, where $\mathcal{P}$ is the powerset operator. Thus, intensions can be seen as *names* of abstract or concrete entities, while extensions correspond to various rules that these entities play in different worlds.

Among the possible functions in $\mathcal{E}$ there is a distinguished function $\Bbbk$ which is to be thought as the *actual* extensionalization function: it tells us the extension of the intensional elements in $\mathcal{D}$ in the actual (current) world.

In what follows we will use the join operator $\bowtie$, such that for any two relations r_1, r_2 their join is defined by: $r_1 \bowtie r_2 = \{(\mathbf{a},\mathbf{c},\mathbf{b}) \mid (\mathbf{a},\mathbf{c}) \in r_1 \text{ and } (\mathbf{c},\mathbf{b}) \in r_2\}$, where $\mathbf{a},\mathbf{c},\mathbf{b}$ are tuples (also empty) of constants, so that $r_1 \bowtie \{\} = \{\}$ and $r_1 \bowtie \{<>\} = r_1$.

Definition 4. *(SEMANTICS): The operations of the algebra Alg_{int} must satisfy the following conditions, for any $h \in \mathcal{E}$, with $f = \{\}, t = \{<>\}$, and $u_1,..,u_i \in \mathcal{D}$:*
1. $h(conj(u,v)) = h(u) \bowtie h(v)$, *for* $u,v \in D_I$.
2.1 $h(neg(u)) = t$ *iff* $h(u) = f$, *for* $u \in D_0$.
2.2 $< u_1,..,u_i > \in h(neg(u))$ *iff* $< u_1,..,u_i > \notin h(u)$, *for* $u \in D_i, i \geq 1$.
3.1 $h(pred(u,u_1)) = t$ *iff* $u_1 \in h(u)$, *for* $u \in D_1$.
3.2 $< u_1,..,u_{i-1} > \in h(pred(u,u_i))$ *iff*
$< u_1,..,u_{i-1},u_i > \in h(u)$, *for* $u \in D_i, i \geq 2$.

Notice that this definition for the conjunction operation is different from the original work in [28] where

1.1 $< u_1, .., u_i > \in h(conj(u, v))$ iff
$< u_1, .., u_i > \in h(u) \bigcap h(v)$, for $u, v \in D_i, i \geq 1$.
1.2 $h(conj(u, v)) = t$ iff $h(u) = h(v) = t$, for $u, v \in D_0$.

Once one has found a method for specifying the denotations of singular terms of $\mathcal{L}_\omega$ (taken into consideration the particularity of abstracted terms), the Tarski-style definitions of truth and validity for $\mathcal{L}_\omega$ may be given in the customary way. An *intensional interpretation* I [29] maps each i-ary predicate letter of $\mathcal{L}_\omega$ to i-ary relations-in-intention in D_i. It can be extended to all formulae in usual way. What is being considered specifically is a method for characterizing the denotations of singular terms of $\mathcal{L}_\omega$ in such a way that a given singular term $\lessdot A \gtrdot_{x_1 \ldots x_m}$ will denote an appropriate property, relation, or proposition, depending on the value of m. Thus, the mapping of intensional abstracts (terms obtained by abstraction operator $\lessdot \_ \gtrdot$) in $A_{BS} \subset \mathcal{L}_\omega$ into $\mathcal{D}$, given in original version of Bealer [29], will be called denotation $den : A_{BS} \rightarrow \mathcal{D}$, such that the denotation of $\lessdot A \gtrdot$ is equal to the meaning of a proposition A, that is, $den(\lessdot A \gtrdot) = I(A) \in D_0$. In the case when A is an atom $F^m(x_1, .., x_m)$ then $den \lessdot F^m(x_1, .., x_m) \gtrdot_{x_1,..,x_m} = I(F^m) \in D_m$. The denotation of a more complex abstract $\lessdot A \gtrdot_\alpha$ is defined in terms of the denotation(s) of the relevant syntactically simpler abstract(s) [29]. For example $I(A(x) \wedge B(x)) = conj(I(A(x)), I(B(x)))$, $I(\neg p) = neg(I(p))$. A sentence A is true relative to I and the intensional algebra, iff its *actual* extention is equal to t, that is, $Tr(\lessdot A \gtrdot)$ iff $\Bbbk(I(A)) = t$, where Tr is a unary predicate for true sentences.

For predicate calculus with individual constants (variables with fixed assignment, proper names, and intensional abstracts) we introduced an additional binary algebraic operation $pred$ (singular predication, or *membership* relation), such that for any two $u, v \in \mathcal{D}$, for any extensionalization function h holds $h(pred(u, v)) = t$ iff $v \in h(u)$. So we are able to assign appropriate intensional value (propositional meaning) to a ground atom $A(c) \in \mathcal{L}_\omega$ with individual constant c.

That is, $I(A(c)) = pred(I(A(x)), I(c))$ is an expression in this intensional algebra with $I(A(x)) \in D_1$ and $I(c) \in D_{-1}$. So that $h(I(A(c))) = h(pred(I(A(x)), I(c)) = t$ iff $I(c) \in h(I(A(x)))$. That is, in the 'world' h, $A(c)$ is true (that is, the extension of the propositional meaning of $A(c)$ is equal to t) iff the interpretation of c is in the extension of the interpretation of the predicate $A(x)$. Or, for example, for a given formula with intensional abstract, $B(\lessdot A(x, y) \gtrdot_{x,y}) \in \mathcal{L}_\omega$, we have that $h(I(B(\lessdot A(x, y) \gtrdot_{x,y}))) = h(pred(I(B(z)), den(\lessdot A(x, y) \gtrdot_{x,y}))) = t$ iff $den(\lessdot A(x, y) \gtrdot_{x,y}) \in h(I(B(z)))$, where $I(B(z)) \in D_1$ and $den(\lessdot A(x, y) \gtrdot_{x,y}) \in D_2$.

We can connect $\mathcal{E}$ with a possible-world semantics, where $w_0 = \mathcal{F}^{-1}(\Bbbk)$ denotes the actual world in which intensional elements have the extensions defined by $\Bbbk$. Such a correspondence, not present in original intensional theory [10], is a natural identification of intensional logics with modal Kripke based logics.

Definition 5. *(Model): A model for the intensional FOL is the S5 Kripke structure $\mathcal{M}_{int} = (\widetilde{\mathcal{W}}, \mathcal{R}, \mathcal{D}, V)$, with intensional identity defined as follows:*
$\lessdot A \gtrdot_\alpha = \lessdot B \gtrdot_\alpha$ *iff* $\Box(A \equiv B)$
where $\widetilde{\mathcal{W}} = \{\mathcal{F}^{-1}(h) \mid h \in \mathcal{E}\}$, $\mathcal{R} = \widetilde{\mathcal{W}} \times \widetilde{\mathcal{W}}$. The symbol $\Box$ is the universal "necessity" S5 modal operator.

Remark: This semantics is equivalent to the algebraic semantics for $\mathcal{L}_\omega$ in [28] for the case of the conception where intensional entities are considered to be *identical* if and only if they are *necessarily equivalent*. Intensional identity is stronger than the standard *extensional equality* in the actual world, just because it requires the extensional equality in *all* possible worlds, in fact, if $\lessdot A \gtrdot_\alpha = \lessdot B \gtrdot_\alpha$ then $h(den(\lessdot A\gtrdot_\alpha)) = h(den(\lessdot B\gtrdot_\alpha))$ for all extensionalization functions $h \in \mathcal{E}$ (that is possible worlds $\mathcal{F}^{-1}(h) \in \widetilde{\mathcal{W}}$). But we can have the extensional equality in the possible world $w = \mathcal{F}^{-1}(h)$, while $den(\lessdot A\gtrdot_\alpha) \neq den(\lessdot B\gtrdot_\alpha)$, that is, when A and B are not intensionally equal, so that each intensional identity class of elements is the subset of the extensional equivalence class.

Example 3. Let two predicate forms $A(x)$ and $B(x)$ be intensionally equal, that is $I(A(x)) = I(B(x))$, then for any $h \in \mathcal{E}$ holds that $h(I(A(x))) = h(I(B(x)))$, i.e., have the same extension, thus $A(x) \equiv B(x)$ is true, (or $(A(x) \Rightarrow B(x)) \wedge (B(x) \Rightarrow A(x))$ is true), in each world $\mathcal{F}^{-1}(h)$. Consequently $\Box(A(x) \equiv B(x))$ is true, and from the definition holds the intensional identity for their intensional abstracts, $\lessdot A(x)\gtrdot_x = \lessdot B(x)\gtrdot_x$, and finally, $den(\lessdot A(x)\gtrdot_x) = den(\lessdot B(x)\gtrdot_x)$.
Vice versa, if $\Box(A(x) \equiv B(x))$ then $\lessdot A(x)\gtrdot_x = \lessdot B(x)\gtrdot_x$, and $den(\lessdot A(x)\gtrdot_x) = den(\lessdot B(x)\gtrdot_x)$, and from the fact that a denotation of $\lessdot A(x)\gtrdot_x$ is equal to the meaning of $A(x)$, that is, equal to $I(A(x))$, we obtain that $I(A(x)) = I(B(x))$, and consequently $A(x)$ and $B(x)$ are intensionally equal: so the modal formula $\Box(A(x) \equiv B(x))$ corresponds to the intensional equality of $A(x)$ and $B(x)$. $\Box$

Moreover, for this intensional FOL soundness and completeness hold true: For all formulae A in $\mathcal{L}_\omega$, A is valid if and only if A is a theorem of this First-order S5 modal logic with intensional equality [28]. It is easy to verify that intensional equality means that in every possible world $w \in \widetilde{\mathcal{W}}$ the intensional entities A and B have the same extensions (as in Montague's approach). Moreover:

Proposition 1. *(*Bealer-Montague connection*): For any intensional entity $\lessdot A/g\gtrdot$ its extension in a possible world $w \in \widetilde{\mathcal{W}}$ is equal to* $\mathcal{F}(w)(den(\lessdot A/g\gtrdot)) = [A]_{in}^{\mathcal{M},g}(w)$.

Proof. Directly from the definition of the identification of a possible world w of Montague's approach with the extensional function $h = \mathcal{F}(w) \in \mathcal{E}$ in the Bealer's approach, where $[A]_{in}^{\mathcal{M},g}$ is the "functional" intension of Montague, and $\lessdot A/g\gtrdot$ is a intensional term of Bealer's logic for a ground formula A/g. $\Box$

Now we can introduce the new intensional equivalence relation:

Definition 6. *(Intensional Equivalence $\approx$) : the two intensional entities $\lessdot A\gtrdot_\alpha, \lessdot B \gtrdot_\alpha$ without free variables (ground terms) are intensionally equivalent*

$\lessdot A\gtrdot_\alpha \approx \lessdot B \gtrdot_\alpha$ *iff* $\Diamond A \equiv \Diamond B$, *where* $\Diamond = \neg\Box\neg$.

This equivalence defines the QUOTIENT *algebra $Alg_{int}/_\approx$ for a quotient-intensional FOL $\mathcal{L}_\omega/_\approx$, as follows:*

Given an intensional logic $\mathcal{L}_\omega$ with a basic, user defined, set of intensional equivalences S_{eq}, and its deductive inference relation $\vdash_{in}$ of the S5 modal logic with intensional equality theory, then, for any intensional entity $\lessdot A(\boldsymbol{x})\gtrdot_{\boldsymbol{x}}$, where $\boldsymbol{x} =< x_1, .., x_k >$ is a tuple of free variables in A, we obtain an intensional-equivalence class

$\mathcal{C} = \{\lessdot A_i(\boldsymbol{x}) \gtrdot_{\boldsymbol{x}} \mid A_i(\boldsymbol{x}) \in \mathcal{L}_\omega$ *, such that* $\mathcal{L}_\omega, S_{eq} \vdash_{in} \lessdot A_i(\boldsymbol{x})\gtrdot_{\boldsymbol{x}} \approx \lessdot A(\boldsymbol{x})\gtrdot_{\boldsymbol{x}}\}$. *If we denote by* $\lessdot A(\boldsymbol{x})\gtrdot_{\boldsymbol{x}} \in Alg_{int}/_{\approx}$ *the quotient intensional entity for this equivalence class, its extension in a world* w *is defined by*
$\mathcal{F}(w)(den(\lessdot A(\boldsymbol{x})\gtrdot_{\boldsymbol{x}})) = \{\boldsymbol{t} \in \mathcal{D}^k \mid A_i(\boldsymbol{t})$ *is true in* w, $A_i(\boldsymbol{x}) \in \mathcal{C}\}$
$= \bigcup_{1\leq i\leq m} \mathcal{F}(w)(den(\lessdot A_i(\boldsymbol{x})\gtrdot_{\boldsymbol{x}}))$.

This definition of equivalence relation is the flat-accumulation case presented in [6,18]: if the first predicate is true in some world then the second must be true in some world also, and vice versa. In what concerns this paper we will consider *only* the actual world $w_0 = \mathcal{F}^{-1}(\Bbbk)$. Moreover, the set of basic intensional equivalences are designed by users/developers. Consequently, the definition above has a theoretical importance only, but is useful to understand the meaning of the intensional equivalence. The (omniscient) deductive inference relation $\vdash_{in}$ of this logic, able to derive all other intensionally equivalent formulae, is that of the S5 modal logic with intensional equality theory.

The following theorem considers the class of peers and queries where the substitutivity of intensionally equivalent formulae holds.

Theorem 1. *Let us consider the class of peers with integrity constraints which does not contain negative clauses of the form* $\neg A_1 \vee ... \vee \neg A_m,\ m \geq 2$. *Then, the intensional equivalence is preserved by conjunction logic operation, that is,*
if $\varphi \equiv (b_1 \wedge ... \wedge b_k)$, $k \geq 1$, *and* $\lessdot b_i\gtrdot \approx \lessdot c_i\gtrdot,\ 1 \leq i \leq k,$ *then* $\lessdot \varphi\gtrdot \approx \lessdot\psi\gtrdot$ *where* $\psi \equiv (c_1 \wedge .... \wedge c_k)$.

Proof. By structural induction on the number of conjuncts in the expression: it is enough to prove for expressions composed by two conjuncts. Let us define $lub(\phi(\mathbf{x})) = \bigcup_{w\in\widetilde{\mathcal{W}}} \mathcal{F}(w)(d(\lessdot\phi(\mathbf{x})\gtrdot_{\mathbf{x}}))$, so that we have $\lessdot\phi(\mathbf{x})\gtrdot_{\mathbf{x}} \approx \lessdot\phi_1(\mathbf{x}) \gtrdot_{\mathbf{x}}$ iff $\Diamond\phi(\mathbf{x}) \equiv \Diamond\phi_1(\mathbf{x})$ iff $lub(\phi(\mathbf{x})) = lub(\phi_1(\mathbf{x}))$.

Let b_1, b_2 be any two (virtual) predicates over a peer P_i, $q_{i1}(x,y)$ and $q_{i2}(y,z)$ respectively, and c_1, c_2 (equal to $q_{j1}(x,y)$ and $q_{j2}(y,z)$ respectively) any two (virtual) predicates over a peer P_j, such that $\lessdot b_i\gtrdot \approx \lessdot c_i\gtrdot,\ i = 1,2$. We have to prove that $lub(\varphi(x,z)) = lub(\psi(x,z))$, where
$\varphi(x,z) \equiv (q_{i1}(x,y) \wedge q_{i2}(y,z))$ and $\psi(x,z) \equiv (q_{j1}(x,y) \wedge q_{j2}(y,z))$.

From the fact that $lub(q_{i1}(x,y)) = lub(q_{j1}(x,y))$ and $lub(q_{i2}(y,z)) = lub(q_{j2}(y,z))$, we define $S_L = \{(a,c) \mid \exists b.((a,b) \in lub(q_{i1}(x,y)) \wedge (b,c) \in lub(q_{i2}(y,z)))\} = = \{(a,c) \mid \exists b.((a,b) \in lub(q_{j1}(x,y)) \wedge (b,c) \in lub(q_{j2}(y,z)))\}$.

Let us prove that $lub(\varphi(x,z)) = \bigcup_{w\in\widetilde{W}}\{(a,c) \mid \exists b.((a,b) \in \mathcal{F}(w)(d(\lessdot q_{i1}(x,y)\gtrdot_{x,y})) \wedge (b,c) \in \mathcal{F}(w)(d(\lessdot q_{i2}(x,y)\gtrdot_{x,y})))\}$ is equal to S_L .

First, from $\mathcal{F}(w)(d(\lessdot q_{ik}(x,y)\gtrdot_{x,y})) \subseteq lub(q_{ik}(x,y)),\ k = 1,2$ holds that $lub(\varphi(x,z)) \subseteq S_L$.

Let us prove, that also $lub(\varphi(x,z)) \supseteq S_L$, i.e. that for any $(a,b) \in S_L$ also $(a,b) \in lub(\varphi(x,z))$. Let us suppose that there is one (a,c) such that $(a,c) \in S_L$ but $(a,c) \notin lub(\varphi(x,z))$, i.e., that for all possible worlds for this P2P system, $w \in \widetilde{\mathcal{W}}$, holds that $\pi_2(\mathcal{F}(w)(d(\lessdot q_{i1}(a,y)\gtrdot_y))) \bigcap \pi_1(\mathcal{F}(w)(d(\lessdot q_{i2}(y,c)\gtrdot_y))) = \{\}$ (is empty), where π_1, π_2 are the first and the second projections. That is, the following logic formula must hold $\neg q_{i1}(a,y) \vee \neg q_{i2}(y',c) \vee \neg(y = y')$.

But such constraint (negative clause) cannot exist in this class of peers, thus the supposition is false, and we conclude that $S_L = lub(\varphi(x,z))$. By the same way we obtain that $S_L = lub(\psi(x,z))$, thus $\lessdot \varphi(x,z) \gtrdot_{x,z} \approx \lessdot \psi(x,z) \gtrdot_{x,z}$. $\square$

The quotient intensional FOL $\mathcal{L}_\omega/_\approx$ is fundamental for *query answering* in intensional P2P database mapping systems: given a query $q(\mathbf{x})$ over a peer P_i, the answer to this query is defined as the extension of the quotient-intensional concept $\lessdot q(\mathbf{x}) \gtrdot_{\mathbf{x}}$, in the intensional P2P logic $\mathcal{L}_\omega/_\approx$.

3 An Embedding of P2P Database Systems into Intensional FOL

The formal semantic framework for P2P database systems, presented also in [6] as a hybrid modal logic, in this paper will be defined as a quotient (by intensional equivalence) intensional FOL.

We will consider *only* the actual world $w_0 = \mathcal{F}^{-1}(\Bbbk)$, correspondent to the extensionalization function $\Bbbk$ of the quotient intensional FOL $\mathcal{L}_\omega/_\approx$ (the actual world for $\mathcal{L}_\omega/_\approx$ corresponds to the actual extension of peer databases). The answer to a conjunctive query $q(\mathbf{x})$, over an ontology $\mathcal{O}_i$ of a peer database P_i, is computed in this actual world w_0, that is in the actual extension of all peer databases in a P2P network $\mathcal{N} = \{P_i \mid 1 \leq i \leq N\}$.

Definition 7. *Let $\mathcal{N} = \{P_i \mid 1 \leq i \leq N\}$ be a P2P database system. The intensional FOL $\mathcal{L}_\omega$ for a query answering in a P2P network $\mathcal{N}$ is composed by:*

1. The set of basic intensional entities is a disjoint union of entities of peers $S_I = \biguplus_{1 \leq i \leq N} \{r(\mathbf{y}) \mid r(\mathbf{y}) \in \mathcal{O}_i\}$. The intensional interpretation of the set of all intensional entities define the domains $D_n, n \geq 1$;
2. The extensional part of a domain, D_{-1}, corresponds to the disjoint union of domains of peer databases. The intension-in-proposition part, D_0, is defined by disjoint union of peer's Herbrand bases.
3. The basic set of the equivalence relation $\approx$ is defined as a disjoint union for each peer P_i as follows ($\boldsymbol{x}$ is a tuple of variables of queries):
if $(q^{ij}_{1k}(\boldsymbol{x}), q^{ij}_{2k}(\boldsymbol{x})) \in \mathcal{M}^{ij}$, then $\lessdot q^{ij}_{1k}(\boldsymbol{x}) \gtrdot_{\boldsymbol{x}} \approx \lessdot q^{ij}_{2k}(\boldsymbol{x}) \gtrdot_{\boldsymbol{x}}$.
The COMPLETE *P2P answer to a conjunctive query $q(\boldsymbol{x})$ over a peer P_i is equal to the extension of the quotient-intensional concept $\lessdot q(\boldsymbol{x}) \gtrdot_{\boldsymbol{x}}$, whose equivalence class is determined by the deductive omniscient closure of $\vdash_{in}$, in the quotient intensional P2P logic $\mathcal{L}_\omega/_\approx$.*

Notice that in this embedding of a P2P system into the intensional FOL $\mathcal{L}_\omega$, we do not use any existential quantifier, so that the intensional algebra in Definition 3 is sufficient for a P2P query answering. We need to finish the modeling of the intensional logic $\mathcal{L}_\omega/_\approx$ by defining its extensionalization function $\Bbbk$ for the actual world w_0.
For this aim we will consider as actual world w_0, of the intensional logic, the actual *extensional* FOL multi-modal P2P database system:

1 - What we will obtain is a two-level modal framework: the higher, or *P2P query answering*, level is the Bealer's intensional logic (without quantifiers) with S5 Montague's possible-worlds $\widetilde{\mathcal{W}}$ modal structure, where $w_0 \in \widetilde{\mathcal{W}}$ is an actual world for a P2P system. The lower, "computational", level is the extensional FOL multi-modal epistemic

logic (with existential quantifiers also), where the set of worlds $\mathcal{W}$ is a disjoint union of the set $\mathcal{W}_i$ of preferred Herbrand models of each peer database P_i: a modal "know" operator K_i of a peer P_i is based on the accessibility relation $R_i = \{P_i\} \times \mathcal{W}_i$. Consequently, each Bealer/Montague's possible world contains a particular set of low-level worlds of the extensional FOL multi-modal epistemic P2P system. We can see this "computational" level as a sophisticated wrapper (based on a Data Integration System which is encapsulated into a peer as an Abstract Data Type (ADT) [15]). Each peer is considered as an *independent* (from other peers) sophisticated wrapper, which extracts the exact extension (of *only known* facts) for all predicates used in upper intensional P2P query answering logic layer.
2 - Here we will use the standard *extensional* multi-modal logic framework, of this lower "computational" level of each independent peer. We will differentiate the accessibility relation between peers (network's (or *global*) conceptual level) from the accessibility relations which model the epistemic query answering semantics of single *local* peers. This multi-modal logic will define the extensionalization function $\Bbbk$ for the actual world w_0 in the Bealer-Montague's framework.

Let us summarize the obtained P2P architecture. What we obtained is a relatively simple intensional predicate (without quantifiers) logic, with only a subset of predicates used in the global schema of peer databases with the set of views (virtual predicates) defined for intensional mapping between peers. The extension of these predicates is wrapped by the ADT of each peer independently. The logic specification for these sophisticated wrappers can be obtained by using the epistemic multi-modal extensional logic [15] of each single peer database. Here a peer is considered as a Data Integration System with Global-As-View (GAV) mappings between its source and its global database schemas [21], and with integrity constraints over global schema also, which possibly can use the existential quantifiers.

This P2P database architecture uses strong (extensional) semantic mapping, based on views, *inside* each peer database, as in standard Data Integration Systems [35,24], with the possibility to use also logic negation [36]. The weak (intensional) semantic mapping based on views, is used for mapping *between* the peers.

This architecture takes advantage of both semantic approaches:

1. Extensional, for a building of independent peer databases (a development of any particular peer database can be done by a group of developers, dedicated to developing and maintaining its functionalities).
2. Intensional, for robust and non invasive mapping between peers, based on beliefs of developers of one peer about the intensionally equivalent knowledge contained in other peers (which are not under their control).

Example 4. Let us consider the *cyclic* P2P system in a Fig.2, with a sound but generally incomplete deduction [19], which can be easily implemented by *non-omniscient* query agents: we have P_i, with the ontology $\mathcal{O}_i$ and the interface $\mathcal{M}^{ij} = \{(v_{im}, v_{jm}) \mid 1 \leq m \leq k_1\}$ toward the peer P_j, and the peer P_j, with the ontology $\mathcal{O}_j$ and the interface $\mathcal{M}^{ji} = \{(w_{jm}, w_{im}) \mid 1 \leq m \leq n_1\}$ toward the peer P_i. First we traduce a pair (v_{im}, v_{jm}) by intensional equivalence $\lessdot v_{im} \gtrdot \approx \lessdot v_{jm} \gtrdot$. In what follows, the subscript of a query identifies the peer relative to such a query.

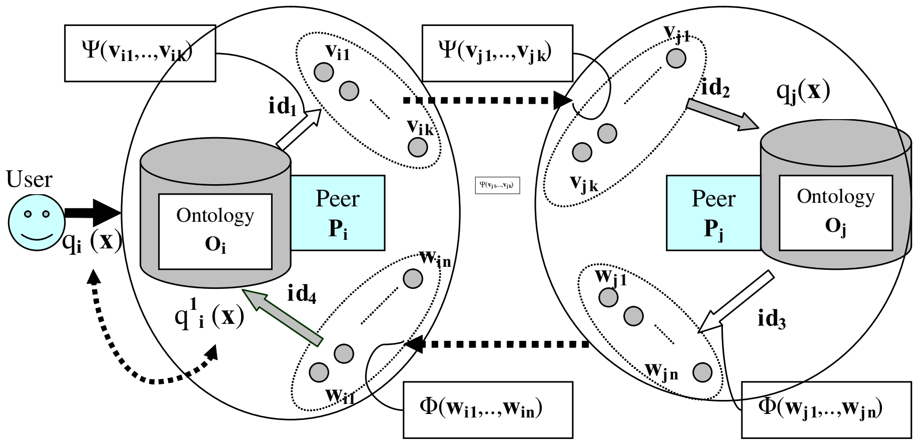


Fig. 2. Derivation of intensionally equivalent queries

Let $q_i(\mathbf{x})$ be the original user's conjunctive query over the ontology $\mathcal{O}_i$ of the peer database P_i. If this query can be rewritten [37], by the query rewriting algorithm id_1,in the equal query over the set of views $\{v_{i1},...,v_{ik}\} \subseteq \pi_1\mathcal{M}^{ij}$, where π_1 is the first projection, we will obtain an identical (to original query $q_i(\mathbf{x})$) conjunctive query $\Psi(v_{i1},...,v_{ik})$, that is, in the intensional logic language holds the identity
$id_1 : \lessdot q_i(\mathbf{x})\gtrdot_{\mathbf{x}} = \lessdot\Psi(v_{i1},...,v_{ik})\gtrdot_{\mathbf{x}}$, or, equivalently, $\Box(q_i(\mathbf{x}) \equiv \Psi(v_{i1},...,v_{ik}))$.

From the set of intensional equivalences in $\mathcal{M}^{ij}$, $\lessdot v_{im}\gtrdot \approx \lessdot v_{jm}\gtrdot, 1 \leq m \leq k$, we obtain that $\lessdot \Psi(v_{i1},..,v_{ik})\gtrdot_{\mathbf{x}} \approx \lessdot\Psi(v_{j1},..,v_{jk})\gtrdot_{\mathbf{x}}$, or $\Diamond\Psi(v_{i1},..,v_{ik}) \equiv \Diamond\Psi(v_{j1},..,v_{jk})$, in the top-horizontal arrow in Fig.2.

In the next step the conjunctive query formula $\Psi(v_{j1},..,v_{jk})$ over the set of views $\{v_{j1},...,v_{jk}\} \subseteq \pi_1\mathcal{M}^{ji}$ of the peer database P_j, will be rewritten (by simply unfolding) to the conjunctive query $q_j(\mathbf{x})$ directly over the ontology $\mathcal{O}_j$ of the peer P_j, that is, in the intensional logic language holds the identity
$id_2 : \lessdot\Psi(v_{j1},...,v_{jk})\gtrdot = \lessdot q_j(\mathbf{x})\gtrdot$, or, equivalently, $\Box(q_j(\mathbf{x}) \equiv \Psi(v_{j1},...,v_{jk}))$.

If we compose algebraically these mappings we obtain the one-step P2P query rewriting $id_2\circ \approx \circ id_1 : q_i(\mathbf{x}) \mapsto q_j(\mathbf{x})$, that is, from $id_2\circ \approx \circ id_1 = \approx$ we obtain the intensional equivalence $\lessdot q_j(\mathbf{x})\gtrdot \approx \lessdot q_i(\mathbf{x})\gtrdot$, that is, $\Diamond q_j(\mathbf{x}) \equiv \Diamond q_i(\mathbf{x})$.

In the same way (see the inverse bottom horizontal arrows of a diagram in Fig.2), based on the interface specification of the peer P_j, $\mathcal{M}^{ji} = \{(w_{jm}, w_{im}) \mid 1 \leq m \leq n\}$, toward the peer P_i, we obtain also $\lessdot q_i^1(\mathbf{x})\gtrdot_{\mathbf{x}} \approx \lessdot q_j(\mathbf{x})\gtrdot_{\mathbf{x}}$, that is, $\Diamond q_i^1(\mathbf{x}) \equiv \Diamond q_j(\mathbf{x})$. Thus we obtain the three intensionally equivalent queries $q_i(\mathbf{x})$, $q_j(\mathbf{x})$ and $q_i^1(\mathbf{x})$, where two of them, $q_i(\mathbf{x}), q_i^1(\mathbf{x})$ are over the *same* peer P_i: the first one is the original user query, while the second is the intensionally equivalent *derived* query (based on P2P interface intensional specification).

These three query formulae, $\{q_i(\mathbf{x}), q_j(\mathbf{x}), q_i^1(\mathbf{x})\}$, are the subset of the equivalent class $\mathcal{C}$ for the given user query, which in the intensional FOL $\mathcal{L}_\omega/_\approx$ is represented by the quotient intensional entity $Q(\mathbf{x})$, whose extension (from Definition 6) in the actual world w_0 is defined by $\mathcal{F}(w_o)(den(\lessdot Q(\mathbf{x})\gtrdot_{\mathbf{x}})) = \{\mathbf{t} \in \mathcal{D}^k \mid Q(\mathbf{t}), Q(\mathbf{x}) \in \mathcal{C}\} =$ $= \bigcup_{1\leq i\leq m} \mathcal{F}(w_0)(den(\lessdot Q(\mathbf{x})\gtrdot_{\mathbf{x}}))$, that is, the union of known answers of these three queries is a subset of the extension of this quotient intensional entity $Q(\mathbf{x})$. Only in the case when this sound query rewriting algorithm deduces *all* intensionally equivalent queries in a P2P network, that is when it is also *complete*, the union of known answers for queries obtained from such algorithm will be equal to the extension of the intensional entity $Q(\mathbf{x})$.

4 Computing of the Extensionalization Function

The actual world w_0, with correspondent extensionalization function $\Bbbk = \mathcal{F}(w_0)$, is represented as an *extensional* FOL multi-modal logic theory for a P2P database system, composed by a number of peers $\{P_i \mid 1 \leq i \leq N\}$, defined as follows:

Definition 8. *We consider a model $\mathcal{M}$, for the extensional multi-modal logic translation of a P2P database system composed by N peers, a four-tuple $(\mathcal{W}, \{\mathcal{R}_i\}, \mathcal{D}, \mathcal{V})$, where:*

- *The set of points is a disjoint union $\mathcal{W} = \sum_{1 \leq i \leq N}(\mathcal{W}_i \bigcup \{P_i\})$, with $\mathcal{W}_i = Mod(P_i)$, where:*
  1. Each point P_i is considered as a FOL theory with incomplete information, composed by an extensional (ground atoms/facts) and, possibly, an intensional part (logic formulae with variables).
  2. For each peer database P_i, the set of points $\mathcal{W}_i = Mod(P_i)$, $1 \leq i \leq N$ is the set of all preferred Herbrand models of such peer database. Each $w \in Mod(P_i)$ can be seen as a logical theory also, composed by only ground terms (only extensional part).
- *$\mathcal{R}_0$ is a binary accessibility relation between peers, such that $(P_i, P_j) \in \mathcal{R}_0$ iff a mapping exists from peer P_i to peer P_j. Then we close this relation for its reflexivity and transitivity properties.*
- *$\mathcal{R}_i = \{P_i\} \times \mathcal{W}_i$, $1 \leq i \leq N$ is a binary accessibility relation for a i-th peer universal modal operator K_i, so that, for a given view $q(\boldsymbol{x})$ over a peer P_i, and assignment g, $\mathcal{M} \models_{P_i,g} K_i q(\boldsymbol{x})$ iff $\forall w((P_i, w) \in \mathcal{R}_i$ implies $\mathcal{M} \models_{w,g} q(\boldsymbol{x}))$.*
- *$\mathcal{V}$ is a function which assigns to each pair consisting of an n-place predicate constant r and of an element $w \in \mathcal{W}$ a function $\mathcal{V}(r, w)$ from $\mathcal{D}^n$ to $\{1, 0\}$.*

So, the extensionalization function $\Bbbk = \mathcal{F}(w_0)$ for basic intensional entities of the intensional P2P logic $\mathcal{L}_\omega/_\approx$, is defined as follows: for any $\lessdot r(\boldsymbol{y})\gtrdot$, where r is an n-ary (virtual) predicate of a peer P_i, and $\boldsymbol{y}, \boldsymbol{c}$ are n-tuples of variables and constants in $\mathcal{D}$ respectively, we define

- *for any n-ary relation-in-intension $den(\lessdot r(\boldsymbol{y})\gtrdot_{\boldsymbol{y}}) \in D_n$, $n \geq 1$,*
  $\Bbbk(den(\lessdot r(\boldsymbol{y})\gtrdot_{\boldsymbol{y}})) = \{g(\boldsymbol{y}) \mid \mathcal{M} \models_{P_i,g} K_i r(\boldsymbol{y}),$ *and assignment* $g : Var \rightarrow \mathcal{D}\}$.
- *for intensional propositions $den(\lessdot r(\boldsymbol{c})\gtrdot)$ in D_0,*
  $\Bbbk(den(\lessdot r(\boldsymbol{c})\gtrdot)) = t$ *if* $\mathcal{M} \models_{P_i} K_i r(\boldsymbol{c})$; f *, otherwise.*

In this way the binary relation of each partition (peer database), $\mathcal{R}_i, i \geq 1$, models the *local* universal epistemic modal operator K_i for each peer database. In fact it holds that $\mathcal{M} \models_{P_i,g} K_i q(\mathbf{x})$ iff $\forall w \in Mod(P_i)(\mathcal{M} \models_{w,g} q(\mathbf{x}))$, i.e., $K_i q(g(\mathbf{x}))$ is true iff $q(g(\mathbf{x}))$ is true in *all* preferred models of a peer P_i.

The binary relation $\mathcal{R}_0$, instead, models the *global epistemic* P2P modal operator $\mathcal{K}$ in this extensional multi-modal logic, whose semantics is defined as follows: for any query formula $q(\mathbf{x})$, with a tuple of variables in $\mathbf{x}$, defined over the ontology of a peer P_i, and a tuple of constants $\mathbf{c}$ of the P2P database domain, it holds that
$\mathcal{M} \models_{P_i} \mathcal{K}q(\mathbf{c}) \quad$ iff $\quad \exists P_n(\ (P_i, P_n) \in \mathcal{R}_0$ and $\mathcal{M} \models_{P_n} K_n\, q_n(\mathbf{c})\ )$,
where $K_n\, q_n(\mathbf{c})$ is a modal formula with $\lessdot q_n(\mathbf{x}) \gtrdot_{\mathbf{x}} \approx \lessdot q(\mathbf{x}) \gtrdot_{\mathbf{x}}$.

In practice we can obtain that $q_n(\mathbf{x}) = Rew(q(\mathbf{x}), P_n)$ is the rewritten query over a peer P_n. Here the Rew algorithm, as for example those described in [6], tries to rewrite, based on the basic set of intensional equivalences in a P2P system, a conjunctive query over another peer P_n; it returns with an empty $\{\}$ query in the case of a failure.

This definition corresponds to the fact that when we define a query $q(\mathbf{x})$ over a peer P_i, any known answer of any peer accessible from a peer P_i will constitute the global P2P answer to this query. From the definition of the peer-accessibility relation $\mathcal{R}_0$ we have that it is *reflexive*, thus also the answers of the same peer will be part of the whole global P2P answer to this query. Another peer P_n, accessible from P_i (consider that $\mathcal{R}_0$ is also *transitive*), such that $\mathcal{L}_\omega \vdash_{in} \Diamond q(\mathbf{x}) \equiv \Diamond q_n(\mathbf{x})$ is a valid omniscient deduction in $\mathcal{L}_\omega$, can contribute to the *global* P2P answer by his *known local* answer to $q_n(\mathbf{x})$.

Proposition 2. *The P2P global operator $\mathcal{K}$ is an existential normal modal operator.*

Proof. From the definition above it holds that $\mathcal{K}$ is an existential modal operator, modeled by the accessibility relation $\mathcal{R}_0$. Let us prove that it is a normal modal operator. In fact, for any false ground query f over a peer P_i we have that $\mathcal{M} \not\models_{P_i} \mathcal{K}f$: suppose that $\mathcal{M} \models_{P_i} \mathcal{K}f$, then from definition must exist a peer P_n such that $\mathcal{M} \models_{P_i} K_n f$, what is a contradiction because all peers are modeled by normal modal logic. It is easy to verify that $\mathcal{K}(A \vee B) \equiv \mathcal{K}(A) \vee \mathcal{K}(B)$: (left-to-right) suppose that $\mathcal{M} \models_{P_i} \mathcal{K}(A \vee B)$ then from the definition a peer P_n must exist such that $\mathcal{M} \models_{P_i} K_n(A \vee B)$, and from the normal modal operator K_n we obtain that $\mathcal{M} \models_{P_i} K_n(A)$ or $\mathcal{M} \models_{P_i} K_n(B)$, so that $\mathcal{M} \models_{P_i} \mathcal{K}(A)$ or $\mathcal{M} \models_{P_i} \mathcal{K}(B)$ holds. Analog result holds for the right-to-left proof.□

From the implementation point of view, it will be the task of a query agent in a Web P2P system to coordinate the query rewriting over different peers and to collect their local answers. Such a global P2P query answering, for a P2P network with a finite number of peers, will have the biggest fixpoint which is mathematically the final coalgebraic semantics for this query answering, as is defined in the next Section.

Proposition 3. *Let Rew be any conjunctive query rewriting algorithm which satisfies the definition of a global P2P modal operator $\mathcal{K}$. Then it is a* SOUND *algorithm w.r.t. the quotient intensional FOL $\mathcal{L}_\omega/_\approx$. Given a conjunctive query $q(\boldsymbol{x})$ over a peer P_i, the global query answer in a network $\mathcal{N}$ of a P2P database system, will be the following set of tuples:*
$[q(\boldsymbol{c})]^{P_i} = \{\boldsymbol{c} \mid \mathcal{M} \models_{P_i} \mathcal{K}q(\boldsymbol{c})\ \} \subseteq \Bbbk(den(\lessdot q(\boldsymbol{x}) \gtrdot_{\boldsymbol{x}}))$.

Proof. Sketch: it is based on the fact that any query rewriting algorithm, used to define the semantics of a global P2P modal operator $\mathcal{K}$, is based on the intensional equivalence of peers based on views (for example the sound but incomplete algorithm in [6]). If this algorithm is perfect, that is creates ALL possible equivalent queries over ontologies of other peers, for a given P2P network $\mathcal{N}$ and the set of intensional equivalences defined in P2P mappings, than it will be also *complete*. In this case it will give exactly, in the actual world $\Bbbk$, the extension of the quotiented intensional element $\lessdot q(\mathbf{x})\gtrdot)$ in the logic theory $\mathcal{L}_\omega/_\approx$ obtained by the *omniscient* entailment $\vdash_{in}$ of this S5 modal logic. □

Context-dependent query answering: notice, that the answer to any query depends on the topology of the P2P network, that is, it depends on the peer's accessibility relation $\mathcal{R}_0$, so that for equivalent queries, but formalized over different peers we will generally obtain different answers. Now we are able to synthesize the definition of intensionally equivalent views used for mappings between peers, in this two-leveled Kripke model framework:

Definition 9. *(Intensional FOL for P2P systems): A two-level Kripke model for the intensional FOL of a P2P database system $\mathcal{N}$, given in Definition [7], is the S5 Kripke structure $\mathcal{M}_{int} = (\widetilde{\mathcal{W}}, \mathcal{R}, \mathcal{D}, V)$, where each Montague's possible world $w_n \in \widetilde{\mathcal{W}}$ is the multi-modal translation of a P2P database system in that world, given by Definition [8], that is $w_n = (\mathcal{W}, \{\mathcal{R}_i\}, \mathcal{D}, \mathcal{V})_n \in \widetilde{\mathcal{W}}$, so that an intensional equivalence of views, $q_i(\boldsymbol{x})$ and $q_j(\boldsymbol{x})$, defined as conjunctive queries over peers P_i and P_j respectively, is formally given by the following modal formulae of the intensional FOL:*

$$\Diamond q_i(\boldsymbol{x}) \equiv \Diamond q_j(\boldsymbol{x}) \quad \textit{i.e.,} \quad (\Diamond q_i(\boldsymbol{x}) \Rightarrow \Diamond q_j(\boldsymbol{x})) \wedge (\Diamond q_j(\boldsymbol{x}) \Rightarrow \Diamond q_i(\boldsymbol{x}))$$

This definition tells us, intuitively, that any possible world (for a given time-instance) of the intensional logic for P2P database system $\mathcal{N}$, represents (that is models) a particular state of this P2P database, that is, the structure and the extensions of all peer databases in such a time instance. The set of possible worlds $w_n \in \widetilde{\mathcal{W}}$ corresponds to the whole evolution in time of the given P2P system. Such an evolution is result of all possible modifications of an initially defined P2P database system: a simple modification of extensions of peer databases, an inserting of a new peer, or a deleting of an existing peer in this network $\mathcal{N}$.

Thus, the modal formulae used to specify the intensional equivalence of two views, $\lessdot q_i(\mathbf{x})\gtrdot_{\mathbf{x}} \approx \lessdot q_j(\mathbf{x})\gtrdot_{\mathbf{x}}$, means that if, for a given tuple of constants $\mathbf{c}$, $K_i q_i(\mathbf{c})$ is true in some world $w \in \widetilde{\mathcal{W}}$, than also $K_j q_j(\mathbf{c})$ may be true in some world $w' \in \widetilde{\mathcal{W}}$.

The Coalgebraic specification of query-answering in this intensional predicate logic for P2P database systems can be found in [38].

5 Conclusion

Integrating heterogeneous computational resources and databases, which are distributed over highly dynamic computer networks, is the crucial challenge at the current evolutionary stage of IT infrastructures. Peer-to-peer data integration aims at overcoming these drawbacks by modeling autonomous information systems as peers, and establishing mapping among peers without resorting to any hierarchical structure.

As this paper has shown, the problem of defining the semantics for intensional ontology mappings between peer databases, can be expressed in a quotient intensional FOL language, which for the actual P2P world can be computationally reduced to a multi-modal logic. A P2P answer to conjunctive queries is based on the known answers to intensionally equivalent queries on peers, and is strictly connected with the features of the multi-modal epistemic logic.

This intensional FOL for P2P system is obtained by the particular fusion of Bealer's intensional algebra and Montague's possible world modal logic for the semantics of the natural language. In this paper we enriched such a logic framework by a kind of intensional equivalence, which can be used in order to define the intensional view-based mapping between peer's local ontologies. We have shown how such intensional mapping can be used during a query answering process and we defined the global existential modal operator for a query answering in a P2P database system. The semantics of this global modal P2P system operator is modeled by a kind of the accessibility relation between peers, based on the intensional view-based mappings between peers, and on the particular sound query rewriting algorithm.

We concluded that the multi-modal logic is a good candidate language for the specification of such P2P database systems and its coalgebraic translation is an abstract specification for a grid query answering computation. Each particular peer database P_i can be seen as a local epistemic extensional modal logic with a proper epistemic modal operator K_i, "Peer P_i knows that ..", independently from all other peers, and can be translated as a module for a grid computation node during the query answering transaction. It is the responsibility of a query agent to rewrite the original user query over a peer P_i to all other intensionally equivalent queries over other peers in a P2P network, so that a sound non-omniscient query rewriting algorithm is a practical solution which may be implemented into such an intelligent software object.

In a future work we will explore the behavioral equivalence in the framework of P2P systems, to study also its other dynamic properties, and to define the logic inference system for a sound and complete query answering.

References

1. Ushold, M.: Where is the semantics in the semantic web. In: Workshop on Ontologies in Agent Systems (OAS) at the 5th International Conference on Autonomous Agents (2001)
2. Gribble, S., Halevy, A., Ives, Z., Rodrig, M., Suciu, D.: What can databases do for Peer-to-Peer? In: WebDB Workshop on Databases and the Web (2001)
3. Serafini, L., Giunchiglia, F., Mylopoulos, J., Bernstein, P.A.: The local relational model: Model and proof theory. Technical Report 0112-23, ITC-IRST (2001)
4. Ghidini, C., Giunchiglia, F.: Local models semantics or contextual reasoning = locality + compatibility. Artificial Intelligence 127, 221–259 (2001)
5. Reiter, R.: Towards a logical reconstruction of relational database theory. In: Brodie, M.L., Mylopoulos, J., Schmidt, J.W. (eds.) On Conceptual Modeling: Perspectives from Artificial Intelligence Databases and Programming Languages (1984)
6. Majkić, Z.: Weakly-coupled ontology integration of P2P database systems. In: 1st Int. Workshop on Peer-to-Peer Knowledge Management (P2PKM), Boston, USA, August 22 (2004)
7. Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Logical foundations of Peer-to-Peer data integration. In: PODS 2004, Paris, France, June 14-16 (2004)
8. Lenzerini, M., Majkić, Z.: General framework for query reformulation. Semantic Webs and Agents in Integrated Economies, D3.1, IST-2001-34825 (February 2003)

9. Franconi, E., Kuper, G., Lopatenko, A., Serafini, L.: A robust logical and computational characterization of Peer-to-Peer data systems. Technical Report DIT-03-051, University of Trento, Italy (September 2003)
10. Bealer, G.: Universals. The Journal of Philosophy 90, 5–32 (1993)
11. Bealer, G.: A solution to Frege's puzzle. In: Tomberlin, J. (ed.) Philosophical Perspectives, vol. 7, pp. 17–61. Ridgeview Press, Atascadero (1993)
12. Chen, W., Kifer, M., Warren, D.S.: HiLog: A foundation for higher-order logic programming. Journal of Logic Programming 15, 187–230 (1993)
13. Calvanese, D., Damaggio, E., De Giacomo, G., Lenzerini, M., Rosati, R.: Semantic data integration in P2P systems. In: Proc. of the Int. Workshop On Databases, Inf. Systems and P2P Computing, Berlin, Germany (September 2003)
14. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Inconsistency tollerance in P2P data integration: an epistemic approach. In: Proc. 10th Int. Workshop on Database Programming Language (2005)
15. Majkić, Z.: Intensional semantics for P2P data integration. In: LNCS Journal on Data Semantics VI, Special Issue on 'Emergent Semantics', April 15 (2006)
16. Majkić, Z.: Weakly-coupled P2P system with a network repository. In: 6th Workshop on Distributed Data and Structures (WDAS 2004), Lausanne, Switzerland, July 5-7 (2004)
17. Majkić, Z.: Massive parallelism for query answering in weakly integrated P2P systems. In: Workshop GLOBE 2004, Zaragoza, Spain, August 30-September 3 (2004)
18. Majkić, Z.: Intensional logic and epistemic independency of intelligent database agents. In: 2nd International Workshop on Philosophy and Informatics (WSPI 2005), Kaiserslautern, Germany, April 10-13 (2005)
19. Majkić, Z.: Non omniscient intensional contextual reasoning for query-agents in P2P systems. In: 3rd Indian International Conference on Artificial Intelligence (IICAI 2007), Pune, India, December 17-19 (2007)
20. Majkić, Z., Prasad, B.: Soft query-answering computing in P2P systems with epistemically independent peers. In: Book on Soft Computing Applications in Industry. STUDFUZZ, vol. 226, pp. 331–356. Springer, Berlin (2008)
21. Lenzerini, M.: Data integration: A theoretical perspective, pp. 233–246 (2002)
22. Greco, G., Greco, S., Zumpano, E.: A logic programming approach to the integration, repairing and querying of inconsistent databases. In: Codognet, P. (ed.) ICLP 2001. LNCS, vol. 2237, pp. 348–364. Springer, Heidelberg (2001)
23. Gelfond, M., Lifshitz, V.: The stable model semantics for logic programming. In: Proc. of the Fifth Logic Programming Symposium, pp. 1070–1080. MIT Press, Cambridge (1988)
24. Majkić, Z.: Plausible query-answering inference in data integration. In: 18th International Florida Artificial Intelligence Conference (FLAIRS 2005), Clearwater Beach, USA, May 15-17 (2005)
25. Majkić, Z.: Fixpoint semantics for query answering in data integration systems. In: AGP 2003 - 8.th Joint Conference on Declarative Programming, Reggio Calabria, pp. 135–146 (2003)
26. Majkić, Z.: Autoepistemic logic programming for reasoning with inconsistency. In: International Symposium on Logic-based Program Synthesis and Transformation (LOPSTR), Imperial College, London, UK, September 7-9 (2005)
27. Blackburn, P.: Representation, reasoning, and relational structures: a hybrid logic manifesto. Methods for Modalities 1, Logic Journal of the IGPL 8, 339–365 (2000)
28. Bealer, G.: Theories of properties, relations, and propositions. The Journal of Philosophy 76, 634–648 (1979)
29. Bealer, G.: Quality and concept. Oxford University Press, USA (1982)
30. Lewis, D.K.: On the prularity of worlds. Blackwell, Oxford (1986)

31. Stalnaker, R.: Inquiry. MIT Press, Cambridge (1984)
32. Montague, R.: Universal grammar. Theoria 36, 373–398 (1970)
33. Montague, R.: The proper treatment of quantification in ordinary English. In: Hintikka, J., et al. (eds.) Approaches to Natural Language, pp. 221–242. Reidel, Dordrecht (1973)
34. Montague, R.: Formal philosophy. In: Thomason, R. (ed.) selected papers of Richard Montague, pp. 108–221. Yale University Press, New Haven (1974)
35. Lembo, D., Lenzerini, M., Rosati, R.: Source inconsistency and incompleteness in data integration. CEUR Electronic Workshop Proceedings (2002), http://ceur-ws.org/Vol-54/
36. Majkić, Z.: Querying with negation in data integration systems. In: 9th International Database Engineering and Application Symposium (IDEAS), Montreal, Canada, July 25-27, pp. 58–70. IEEE Computer Society, Los Alamitos (2005)
37. Levy, A., Mendelzon, A., Sagiv, Y.: Answering queries using views. In: Proc. 14th ACM Symp. on Principles of Database Systems, pp. 95–104 (1995)
38. Majkić, Z.: Coalgebraic specification of query computation in intensional P2P database systems. In: Int. Conference on Theoretical and Mathematical Foundations of Computer Science (TMFCS 2008), Orlando FL, USA, July 9-11 (2008)

Multi-faceted Visualisation of Worklists

Ross Brown[1] and Hye-young Paik[2]

[1] Faculty of Information Technology,
Queensland University of Technology, Brisbane, Australia
r.brown@qut.edu.au

[2] School of Computer Science and Engineering,
University of New South Wales, Sydney, Australia
hpaik@cse.unsw.edu.au

Abstract. Although business process management has been a major area of ICT research, no coherent approach has been developed to address the problem of business process visualisation to aid workers in the process of task prioritisation. In this paper we describe the development of a new, coherent approach to worklist visualisation, via analysis and development of a resource-centric view of the worklist information. We use instances of generic resource types as workflow elements that may be considered by workers when interacting with worklists. We then propose a generic 2D framework for visualising the resources, creating an effective mapping between a task and the capabilities of the resources. This aims to aid the process of task selection and prioritisation by workers. A worklist visualisation system has been implemented as an extension to an open-source workflow system, YAWL (Yet Another Workflow Language).

1 Introduction

Visualisation techniques offer powerful tools for understanding data and processes within complex systems. However, visualisation in the area of Business Process Management (BPM), and in particular workflow management systems, lags behind the state of the art in other areas such as medicine, engineering and mining [1].

Workflow Management Systems (WfMS) play a vital role in BPM in that the business process models are implemented and executed through a WfMS, which routes and dispatches the tasks defined in a model to the individual workers[1]. The result of *routing* tasks is presented to the workers as a *worklist*. A worklist can be understood as a to-do list of tasks that the workers need to carry out in order to complete the process defined by the model.

The success of business process models depends on communicating them to the model consumers effectively. However, modern workflow systems have largely overlooked the needs of the workers in understanding their given tasks in a manner that would help manage them efficiently. For example, it is quite common

[1] The workers are the *consumers* of the model who will carry out the tasks. In this paper, we use the terms *workers* and *model consumers* interchangeably.

that workers would have questions such as "how urgent is this task?", "who else can do the task?", "where do you have to go to carry out the task?" (eg., where is this meeting room B809), "do I have enough resources?" (eg., are there enough chairs for 20 people in the meeting room B809), etc.

A typical representation of a worklist includes a list of tasks with short textual descriptions, and/or attachments (eg., email, document forms, etc). It, however, does not include any support (context) information about the tasks that may assist the worker in planning the tasks. At any point in time, a given worker may be involved in many workflows and may thus be presented with a large to-do list. The worker needs to have available tools to help them decide which would be the best task to undertake next.

We believe that visualisation techniques can be applied to many areas of BPM due to their previous use in application domains that support decision making processes. In decision support systems, information is typically provided to enable the user to be adequately informed as to the direction to be taken for a particular scenario. This applies to all levels of business systems, and to BPM as a whole. For the purpose of this paper, we limit the scope of the work to the area of workflow management, in particular, managing worklists. We apply a visualisation technique to provide workers with information about the context of a task, in order to improve their understanding of the process models and the communication of such models between the model designers and the consumers. The visual information is designed to help workers make decisions in managing worklists (eg., accepting, postponing, delegating, or rejecting tasks).

In this paper, we present a generic visualisation framework that is used to provide support (context) information about the tasks in a worklist. Our contributions are three fold:

- An analysis of the decision making process in managing workflow tasks, especially in relation to the resources available to the worker;
- A novel and generic visualisation technique for worklists;
- The implementation of the framework as a proof of concept.

The rest of the paper is organised as follows. Section 2 investigates the state of the art in worklist visualisation. Section 3 describes the general approach to using these visualisations in workflow systems. Section 4 details the development of a resource centric approach to the management of worklists. Sections 5 and 6 explain the mapping of important worklist resources to appropriate visualisation techniques to aid the process of task selection by workers. Section 7 details the implementation of a visualisation system incorporated into a workflow system. The paper then concludes with a discussion of future work in Section 8.

2 Related Work

Computerised data visualisation is a broad field that has its beginning in pictorial representations used in pre computer technology times [2]. Today it has developed to the point of being one of the main areas of application for computer graphics, and is a powerful tool for the analysis and presentation of data.

Many areas in science and mathematics have benefited from the exploitation of modern computing power in data exploration [1]. Business experts are now using advanced computerised visualisations to analyse multi-dimensional business data and processes. The main purpose of the technique is to allow the users to observe trends or patterns by exploring structured and domain-specific information. (e.g., Figure 1 shows an example of a software solution for business data visualisation).

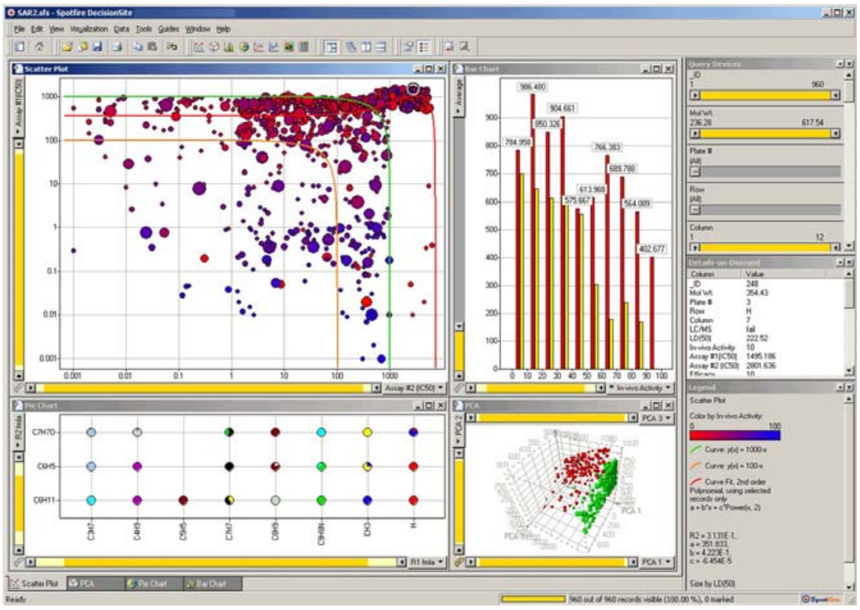

SAR2.sfs - Spotfire DecisionSite
Scatter Plot
Bar Chart
Query Devices
Details-on-Demand
Pie Chart
PCA
Legend
960 out of 960 records visible (100.00 %), 0 marked
Ready

Fig. 1. Spotfire by Tibco: Visualisation of Process Data for Analysis

The focus of our discussion in this paper, though, is in the area of visualising business processes and individual tasks generated by them. Hence, we discuss the related work in the context of business process visualisation and worklist/tasks visualisation.

2.1 Business Process Visualisation

The present state of play for the visualisation of business processes uses various graphical notations to abstract internal (complete) processes into public (basic) processes in order to hide the complexity and implementation details in the process specifications[3,4,5].

For instance, research has been performed into developing approaches to viewing complex business process via aggregating related activities[4]. [5] applies similar principles to a YAWL (Yet Another Workflow Language) business process models, in order to efficiently remove and add components, while maintaining the validity of the YAWL model representation during zoom operations. Some have explored the use of 3D extensions to 2D representations using such techniques as Cone Trees [6,7], to full virtual reality implementations for distributed interaction with process models [8].

To provide more interactive way of analysing business processes, the idea of graphically querying business processes is proposed in [9,10]. For example, Figure 2 shows a user querying 'Does this business process requires login before searching'. In [10], an exploratory (or browsing) approach, rather than directly querying, is used to visualise automated and repetitive scientific workflow specifications. Currently, such querying is only made possible for static processes (i.e., specifications) and does not apply to 'running' instances of the processes.

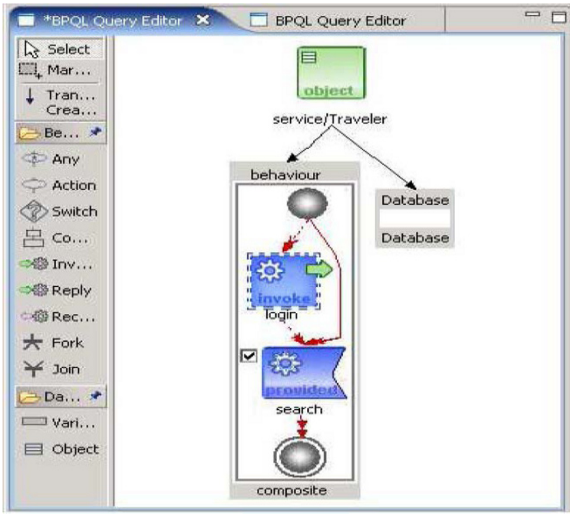


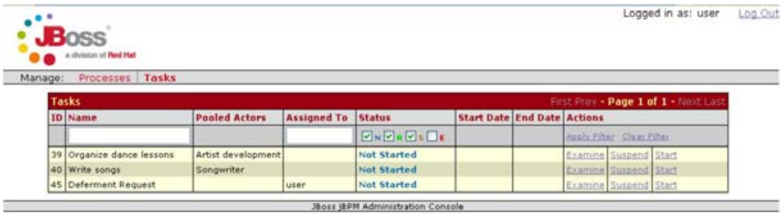
Fig. 2. Visually Querying a Business Process

2.2 Worklist/Tasks Visualisation

The visualisation techniques mentioned above can help the user better understand the business process specification. However, they cannot assist him/her with effectively managing the tasks allocated to them by the processes, as visualisation does not provide take the concept of individual 'activities' or 'tasks' and any background information about them into consideration (e.g., how important is this task?, how is this related to my previous task?).

Figure 3 shows a typical worklist view of modern business process management systems. From the user's view point, tasks generated by a running business process are only represented as a list of brief textual descriptions.

Most project management software will support creation, allocation and tracking of task completion in a business organisation environment. However, the



Logged in as: user Log Out

Manage: Processes | Tasks

Tasks — First Prev - Page 1 of 1 - Next Last

ID	Name	Pooled Actors	Assigned To	Status	Start Date	End Date	Actions
				✓ N ✓ R ✓ S ☐ E			Apply Filter Clear Filter
39	Organize dance lessons	Artist development		Not Started			Examine | Suspend | Start
40	Write songs	Songwriter		Not Started			Examine | Suspend | Start
45	Deferment Request		user	Not Started			Examine | Suspend | Start

JBoss jBPM Administration Console

Fig. 3. JBPM by JBoss: A Worklist

process of generating and allocating tasks is highly ad-hoc and manual and visualisation techniques used are rather crude.

More creative approaches to organising tasks can be found in a PC desktop environment. Many desktop calendar applications will let the users create events or tasks which are visualised in a two dimensional space (time and dates). iChronos (Figure 4) attempts to provide an integration between multiple PC desktop applications (calendar, address book, file folders, etc.). The idea of providing an integrated environment for managing tasks and task related data is presented in the newly emerging research area called “Personal Information Management (PIM)” [11].

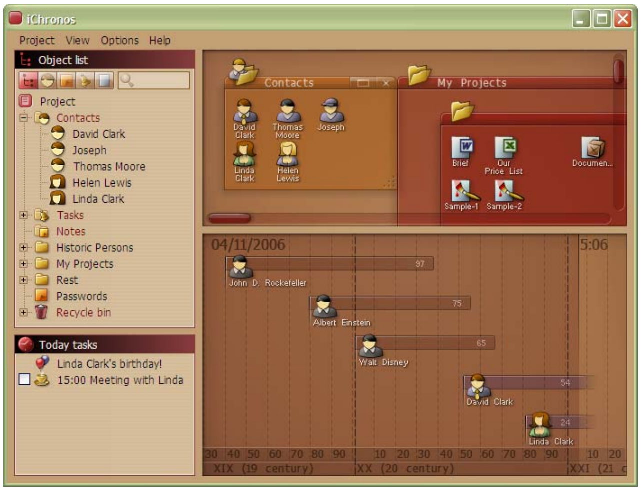


Fig. 4. iChronos

However, the approaches in PIM are heavily focused on capturing and modelling information stored in a person’s computer. The applications have not yet incorporated the business process concepts such as “process definition”, “process execution” “tasks” or “tasks allocation”, etc. into account. As demonstrated by the research prototypes such as Haystack [12] and iDM [13], a PIM system aims to provide a conceptual framework in which the relationship between personal "information" is defined according to how the user perceived and retrieved in a way that is meaningful to the user.

Although our project can benefit from the lessons learned in the PIM area, we believe that there is much work to be done in appropriate visualisation of tasks and business processes that produce the tasks. While there has been evidence of research into user requirements for business process modelling [3,14], much work still remains with regards to the following:

- Data gathering for requirements analysis, the current research is often tied to software implementations which restrict creative solutions;
- No real evidence of systematic analysis of sophisticated 2D and 3D visualisation techniques for use in complex business process models;
- Abstract and leading edge representational techniques, such as Self Organising Maps and Parallel Coordinates amongst others, are often ignored despite their power in representing multi-dimensional data that occurs in business systems;
- Application domain information is not factored into the representations;
- No assessment of visualisation effectiveness via real case studies.

What is needed is a thorough data gathering-based analysis of user requirements for the visualisation of business processes, and the analysis of 2D/3D techniques and visualisation wisdom for such representations. In particular, the area of concurrent process visualisation [15] is expected to provide many useful visualisation techniques. Furthermore, there is a need to provide an approach to visualisation of business processes that accounts for domain specific factors in their representations. Such a visualisation approach needs to allow for both the designers and the users of the business process model, as both these people have different requirements for visualisations, with regards to design, analysis, and usage tasks [16].

What these other workflow visualisation techniques lack is a focus on supporting information to assist the worker in managing the tasks in a worklist. Each of the techniques provides a presentation of the worklist that is rudimentary in nature, lacking any support information for the main task required by a workflow system; deciding to accept, delegate or suspend a presented task. We believe this should be the main reason for such workflow visualisations, and that an analysis of this choice process and derivation of appropriate visualisation techniques is required to support this process.

Analysis of such requirements is best taken from a resource oriented point of view [17], as the available resources influence the acceptance, delegation, reprioritisation or rejection of the task[2] in a worklist. We now proceed to analyse this worklist management problem from a resource perspective, in order to derive appropriate worklist visualisations.

[2] By rejection of tasks, we mean choosing not to accept the task. Such tasks can be delegated, suspended, or re-allocated by the workflow system.

3 Worklist Management

Before we introduce the concept of resources and their visualisations in worklists, we first discuss a theoretical background for generating a worklist from a workflow. That is, how tasks are generated for the worker.

3.1 Worklist Generation

Russell [17] offers a meta model of a resource. A resource can be human or non-human, although in the paper the discussion is focused on human resources. In the meta model, a human resource has a set of capabilities (eg., a manager can approve orders) and may occupy one or more positions. This means that s/he can perform multiple roles in an organisation, or even in a particular workflow. A human resource also can delegate activities to other resources (either human or non-human). Also, it is reasonable to assume that a resource can be substituted by another resource with similar capacity or characteristics. In the following, we explain how tasks are allocated to human resources, which will create a worklist for the worker.

For the sake of clarity, we use the term *worker* to mean a human resource. The way in which worklists are generated is related to how a worker becomes bound to a specific work item (task) for execution. A worklist for worker A can be understood as a list of tasks bound to worker A by the workflow system.

For a better understanding of worklist generation, we look at Fig. 5 which explains the life cycle of a task, from the time of creation to its final state. A final state could be either completion or failure. Each box is the state of the task in a running workflow case. The prefix S and R refer to actions enacted by the Workflow System and the Resource (worker) respectively. Once created, a task can be offered to a single worker, directly allocated to a single worker, or offered to multiple workers.

Being "offered" means that the worker may have a choice in his/her worklist, in that the task can be accepted or rejected. The "allocated" tasks are normally expected to receive firm commitment from the worker. After this offering and allocation process, tasks will appear in workers' worklists and each worker may start the task. Again, once the task is started, it is reasonable to assume that the task may (i) be completed, (ii) suspended (eg., the worker may have to wait for some events to occur or acquire extra resources), (iii) fail. In the following, we look into some of the patterns in this life cycle [17] which, in turn, will help us understand the *patterns in worklist generation.*

3.2 Patterns for Worklist Generation

Push Patterns. The name comes from the fact that the system will "push" a task to a worker. As explained in the life cycle, pushing can be either by allocation or offering (refer to Transitions S:offer_s, S:offer_m, S_allocate_s and S_allocate_m in Fig. 5). Also, a task can be offered to a worker or multiple workers. Note that when a task is pushed to more than one worker, the system is looking for a volunteer to accept the task.

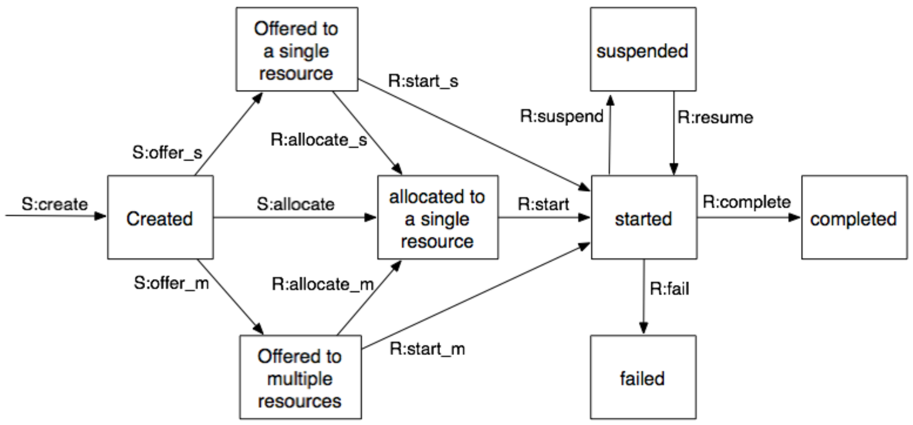


Fig. 5. Illustration of task life cycle; modified from [17]

Visualisation and worklist interactions: It should be clear to the worker (eg., using colour coding) whether a task is allocated, or offered, as it implies a different level of responsibility from the worker. Also, when a task is offered to multiple workers, the worklist should be able to present such tasks as a form of advertisement (eg., a popup window, rolling text, etc.) so that the offer can be noticed by workers immediately. The allocation process often targets the worklists with the smallest number of tasks (ie., shortest queue). Again, a predefined colour-coding of worklists might be used to warn the owner of such candidate worklists of possible future allocation of tasks.

Pull Patterns. This pattern considers the issue of allocation and offering from the worker's point of view (refer to Transitions R:allocate_s, R_allocate_m, R_start_s, R_start_m in Fig. 5. When a worker is "pulling" a task, s/he may intend to start the task immediately. However, it is possible that s/he may only intend to signal the intention to execute the task at some point, but not immediately. In the second case, the task is pulled by the worker and allocated to (ie., bound to) the specific worker. It will not be allocated to another resource.

Visualisation and worklist interactions: The worklist should provide the worker with a way of clearly marking/tagging tasks that s/he intend to do, but not immediately. These tasks will remain allocated in the worklist and cannot be offered to others. Hence, it will also be useful to indicate deadlines for each "pulled-for-later" tasks.

Detour Patterns. In real world scenarios, work allocations may have to be reconsidered due to interruptions by the workflow system, or to the changing states of the workers. Detour patterns describe nine such cases; the following is a direct quote from [17], describing each case:

– *Delegation* - where a worker allocates a work item previously allocated to it, to another worker;

- *Escalation* - where the workflow system attempts to progress a work item that has stalled by offering or allocating it to another worker;
- *De-allocation* - where the system makes a previously allocated or started work item available for offer and subsequent allocation;
- *Reallocation* - where a worker allocates a work item that it has started to another;
- *Resource suspension/resumption* - where a worker temporarily suspends execution of a work item or recommences execution of a previously suspended work item;
- *Skipping* - where a worker elects to skip the execution of a work item allocated to it
- *Redo* - where a worker repeats execution of a work item completed earlier;
- *Pre-do* - where a worker executes a work item that is ahead of the current execution point of a workflow case.

Visualisation and worklist interactions: It is quite obvious from the above that understanding and implementing detour patterns is crucial to providing a worklist with flexibility. For example, before going on leave, a worker must be able to re-allocate his/her tasks to other resources so that the workflow can continue. Appropriate visualisation techniques can assist workers to plan and organise detour activities. For example, a task has been stalled for a certain period of time, the visualisation can highlight such tasks for the worker. Once such a task is brought to the attention of the worker, s/he may choose to escalate it or complete it herself/himself. Another useful technique will be to highlight, for example, tasks that can wait or those that have to be re-allocated for a given period of time. This will help a potential holiday leaver to plan whether to re-allocate tasks or simply to suspend them for a while.

3.3 A Generic Approach to Managing a Worklist

Based on what we discussed so far, we introduce a generalised algorithm that workers may use to manage their tasks. Inherent in the work allocation process by the workflow system is the choice by the system of whom to give the task, via the offer actions.

This paper concentrates on the visual interface to evaluating resource capabilities with regards to work tasks to be performed. Automatic resource allocation is assumed to be performed behind the scenes mainly in push-based workflow systems, where the worklist items are delivered in a final state to the worker [17]. However, the worker may need to allow for local knowledge of resources in order to make a decision about worklist items to choose, as the centralised resource allocation system may be unaware of resource issues at the worksite in question. For this paper, the assumption in the scenario is that the worker is able to then select the work from the system, based upon their assessment of the resources available using the visual presentation. As the focus here is the visual presentation of resources allocated to work, the examination of the automatic allocation of resources is beyond the scope of the paper.

As mentioned before, the workflow system will have a resource view that evaluates the capabilities of the intended recipient of the task. The worker upon receiving the task, must make a decision about accepting or not accepting the task. This process is out of the control of the workflow system, as it only can push tasks to the worker to request acceptance. The workers' responses have thus been characterised by detour processes. A worker may delegate, de-allocate (reject, in other words), re-allocate and suspend/resume.

The question for the worker is the choice of adding or rejecting (ie., detouring) a task from his/her worklist and such a choice could be based on push or pull pattern approaches. Assuming a more pull oriented model of worklist task selection, providing resource information relevant to a worklist will aid the worker in this worklist management task, as they are able to decide which item to choose based upon critical resource issues.

The workflow system may offer a number or only one instance of the task to the worker, and at this point the worker may decide to perform the task by checking them out and adding them to a list of active tasks, or the user may decide to return the task to the unallocated pool via the detour process. Furthermore, the worker upon completion of the task checks the task in, thus removing it from the active checked out worklist. This task acceptance process may be represented by the following formula for the acceptance process, them being the check out processes respectively:

$$W_r = W_r \cup \{I\} \iff C_{W_r,T} \geq C_{\{I\},T} \tag{1}$$

where:

- W_r is the set of worklist items for worker(s) r;
- I is the new worklist item to be added;
- $C_{x,y}$ is the capability for the task(s) x of type y;
- T is the type of resource being processed (eg. Computer Equipment).

So, at any stage a worker will make a decision about whether to add a worklist item to their set of worklist items, by looking at the capabilities of the worker for the present worklist as compared to the requirements of the new task. This can be automated, but in order to promote healthy workforce relations, the worker must be allowed to make such decisions as well. It must also be recognised that people will simply decide not to do a task, if they do not want to or decide to prioritise using undefined criteria. Furthermore, these visualisations may give information to the worker regarding the reasoning behind the choice of been allocated the task, and so the worker is left in an informed state about the reasons for work allocation.

4 Resource Perspective in Workflow

In this section, we discuss a resource perspective on workflow. [18] argues that an effective workflow model would consider all of the following perspectives:

- Functional perspective: the activities being performed
- Behavioural perspective: when, how and in what order activities are begin performed
- Organisational perspective: the organisational context in which the activities are being performed (ie., where and by whom in the organisation).
- Information perspective: the information and data associated with the activities.

In general, a resource is referred to as the entity that actually performs an activity/task. Such entity can be a human or non-human (eg., a computer program). In our paper, we take a broader view of the term and we consider resources to be the workers as well as any work environment element or context that may be required/considered when workers make decisions in managing their tasks. Therefore, in the context of our work, introducing a resource perspective into workflow is an attempt to create an integrated view of organisational perspective which deals with human resources, and information perspective which deals with non-human, work environment element or context that may be relevant to the tasks.

It is important for a workflow system to be aware of the characteristics of each resource (eg., availability, utilisation, cost, etc.) so that it can make smart decisions when allocating tasks. Also, it can be argued that it is necessary to highlight resource related information to the workers so that they can make smart decisions when managing their worklists.

Most commercial workflow systems are quite mature in their support for other perspectives (in the form of control-flow and data-flow), but hold a very simplistic view of the resource perspective [19]. Presently, the only definitive representation of resources and their relationship to workflow has been for human resources [17]. Non-human resources, while being noted in their importance, have not been extensively modelled in workflow systems. Russell [17] simply describes non-human resources as a tuple; (`ResourceType`, `Description`, `Capability`).

An enterprise resource ontology has been developed which seeks to generically model the resources in an organisation [20]. It was developed from a manufacturing viewpoint, but easily transfers to aspects of workflow management, as the resources are generically defined to allow application to other domains.

However, the resources to be considered by the workers for managing worklists may differ depending on the nature of the tasks, the skill level of the workers, or the kind of roles the workers play in an organisation. Indeed, we believe that a thorough study into the requirements of the workers in making decisions as well as a survey of effective visualisation techniques have not been explored. This is an important part of our current on-going investigations, in which we look at identifying various types of resources that a worklist can provide to help the workers carry out the tasks.

In the following, we discuss a few generic resource types that we identified as relevant to our project goal. The list is by no means complete, but we believe it is generic enough that their visualisations could be implemented in many workflow systems.

4.1 Generic Resource Types

We tabulate in Table 1 a list of general resources, with illustrative examples. Most resources we described in Table 1 can be represented in the generic non-human resource model described in [17], or the enterprise resource model proposed in [20]. For example, *equipment*, *services*, *location* or *materials* can have their name, description and capability (eg., print speed for printer) recorded against them in the workflow database, which can be queried. Some resources can be derived from information available within the workflow system. For example, data about *people* come from staff/organisational information in the workflow administration data. A *active worklist* can be obtained from querying the workflow engine to list unfinished tasks. Also, deadlines (an instance of *time*) are available from the workflow engine as they can be assigned to each task when a workflow process is instantiated.

For these resources to be effectively visualised for a worklist, an association between a resource (eg., locations of meeting rooms) and a task (eg., Staff meeting at Room K17 401) needs to be made. In our implementation, this link is manually created by the workflow process designer through the Worklist Visualisation Editor (see Fig. 12). Currently, we do not store resource data in the workflow system for visualisation purposes. Instead, the designer would provide external resources (eg., an image of a floor plan, an XML data file of people or GPS coordinates, etc.) which will then be imported into the visual worklist handler to create mappings between the tasks and the resource.

4.2 Generic Resource Queries

The visualisations in the worklist can be seen as results of queries of the resources associated with the tasks being examined, and thus being presented to the worker in a visual manner. This is a complex requirement, that can be structured as a series of *competency queries* on the resources within an enterprise[3]. These queries form part of an enterprise resource ontology proposed by Fadel [20].

We formalised the comparison previously in Equation 1, to illustrate the fundamental process involved in making work task choices. In order to aid understanding of the resource factors assessed by the capability function $C_{(x,y)}$, we list the generic resource queries enumerated in [20]:

- Quantity – how much of the resource exists at time t;
- Consumption – how much is to be consumed by the worklist item;
- Divisibility – can the resources be divided up to service the work list items;
- Structure – does the structure of the resource fit the worklist item;
- Capacity – can the resource be shared with other work list items;
- Location – where is the resource;
- Commitment – is the resource available at time t;
- Trend – capability trend of resource.

[3] For the sake of congruence with other workflow resource research we replace the word *competency* with *capability* to mean the ability of the resource to perform work.

Table 1. Generic types of resources

Resources	Description and examples
Space	Size or dimensional information relevant to the tasks. It may be a diagram showing available storage rooms and their sizes, or meeting rooms and their capacities. This type of resource may be used to determine, for example, where 20 computers should be stored. It is separate from location, as sometimes the visualisation may not relate space with actual location of the space – space to store computers, but not interested in where.
Materials	Materials or consumable information relevant to the tasks. It may be an inventory list of materials to be used in the task, and whether you have enough of those things: number, volumes, weights, etc. Some of these measures will be discrete and others will be continuous.
Equipment	Equipment information relevant to the tasks. It may be, for example, an inventory of barcode scanners required for the worklist item.
Services	Internal or external services information. It may be a list of travel booking agencies, printing services, or messaging services and their contacts/availabilities.
Time	Any "time" information relevant to tasks. It could be deadlines (eg., the time each task should be completed by), opening hours (eg., the time a particular service, for example a printing centre, is available) or a calendar showing working days. This type of resource will be useful in the planning of task execution sequences.
Location	Geographical "location" information relevant to tasks. It could be a map of a campus showing locations of university facilities, a floor plan of an office block, or a diagram showing relative distances between locations. This type of resource also can be used in scheduling of tasks. We separate this resource from space as our model uses location in both the sense of a resource (maps), and as a generic place holder for the work item location in the visualisation (grid layout).
People	Information about people and their roles in an organisation. It could be an organisational chart showing roles and responsibilities of people. This type of resource may be used in finding the right person to seek for specific help or to delegate a task.
Active Worklist	Current (active) tasks that are being carried out by the worker. This type of resource will help the worker determine the desirable workload, and effectively manage the current/future tasks. This resource is specific to workflow research, as the number of active work items allowed is idiosyncratic to the worker involved, and may be influenced by management or worker originating factors.

In the enterprise resource ontology previously referred to, the above capabilities are defined with appropriate relationships to allow queries of the enterprise model. These query results can be mapped to different worklist visualisations, allowing the workers to assess the availability of resources to meet a task. The above queries are applied to the generic resources in Table 1 to yield a capacity value for those resources with regards to accepting the task.

5 Mapping Resources to a Worklist Visualisation

Domain specific issues have a major role in determining the type of queries for mapping the task to visualisation, but general mappings can also be inferred from generalisations of the tasks to be performed by the user of the visualisation.

In this section, we describe our generic visualisation construction framework for worklist visualisation based on the resources we presented earlier. To illustrate our concept, we use the following simple workflow as running examples throughout the paper[4].

5.1 Example Scenario

The workflow describes a stocktaking process given to an asset management officer who has to record all computer assets managed by a company. Figure 6 describes that, after stocktaking is announced, the officer has to plan and schedule field trips to various sites to physically locate an asset and record the asset number using a barcode scanner. This process will continue until all the sites have been visited.

For the asset management officer to be able to carry out each task, some context information may be required. For example, s/he may want to know how far rooms are located from each other, how many assets are to be collected at each location, etc. to schedule the field trips efficiently.

5.2 Backgrounds and Overlays

For illustration purposes, we choose the four resources; time, location, people and active worklist. The visualisation framework is based on a layered approach, in which background and overlay planes are used. A 2D representation of any of the resources forms the background layer. Thus the background plane allows the comparison of resource values for each of the active worklist items being presented. For example:

– The location resource uses a coordinate representation that shows whereabouts and distance between locations (eg., Street maps);
– The people resource uses a chart or social network form of representation (eg., organisational charts);
– The time resource uses a constrained time line form of representation (eg., Gantt chart).

The overlay plane consists of the tasks in the worklist being viewed by the worker. Each task is given `(x, y)` coordinates in relation to the background, which is the resource information allocated to the task. We name the resource that is allocated to the background layer, the *Principal Resource* (PR). The final `(x, y)` coordinates for the work item are then a mapping of the input

[4] The reader should note that the example is simplified for illustration purposes.

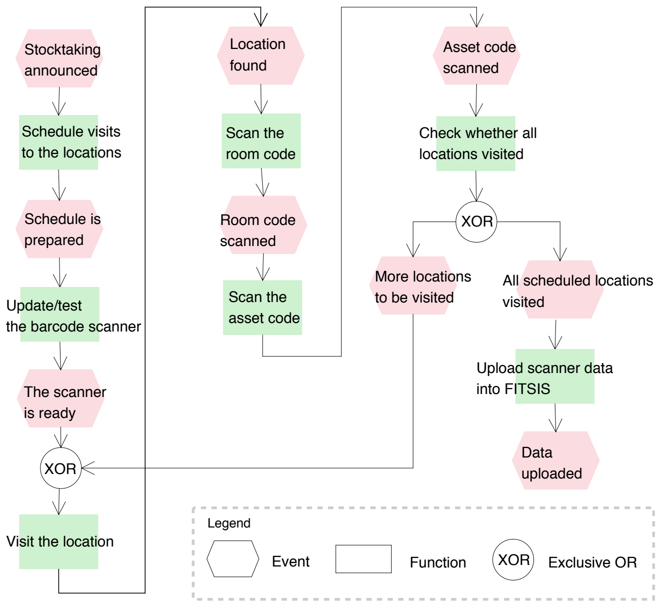


Fig. 6. Event-Process-Chain diagram of the stocktaking process

components of the PR vector to two dimensions in normalised device coordinates, formalised as a general mapping function here:

$$f : R^n \rightarrow R^2 : r_p \mapsto w_c \tag{2}$$

where:

– w_c is the 2D work item normalised window coordinate `x = [0.0,1.0]` and `y = [0.0,1.0]`;
– r_p is a vector of information defining the PR for the visualisation.

We group these mappings into three functional types:

– *Coordinate* mappings are an arbitrary mapping of specified locations with x,y coordinates mapped to the visualisation device, for example, GPS coordinates, that have no constrains with respect to each other (refer to Fig. 7). Each object may overlap other objects in the visualisation.
– *Regional* mappings use an ID to lookup the region they are constrained to in the visualisation, for example, organisational unit name. However, the regions themselves may or may not be constrained relative to each other (refer to Fig. 8), and may overlap each other.
– *Grid* mappings generate visualisation coordinates from enforced discrete rows and columns assigned to a worklist item, for example, the commencement time of a task (refer to Fig. 9). This represents a coordinate system separate to the final visualisation device coordinates. The grid is the *default*

mapping of work item tasks, due to the fact that a simple list of work items is a form of column-based grid visualisation. The objects do not overlap each other, being constrained to grid cells.

It should be noted that each of these visualisations requires fine tuning for an implementation, via computer graphics implementations. The graphics library is responsible for mapping the normalised device coordinates w_c to pixel coordinates in the viewing window of the application [21]. Therefore, the mappings presented in this document represent a general form of the visualisation work item coordinate generation process.

All of the worklist items on a case appear on every *view*. If the system selectively placed worklist items on a view, then the worker may miss items, or get confused by the changes from view to view. In addition, the item may require assessment from a number of resource viewpoints before being accepted or rejected by the worker. Therefore, we ensure that the entire worklist is shown on each view. The workers can easily switch from one view to another via tabbed windows. These tabs can be undocked to allow a free form window layout, if the user wishes.

The framework is generic in that any types of resources can be presented using the overlaying technique. The same worklist can be viewed from different resource perspectives. Furthermore, the background and foreground can contain iconic representations of an arbitrary nature, to represent the worklist items using appropriate images that match domain specific metaphors, for example, a PC icon can represent a computer to be collected [22].

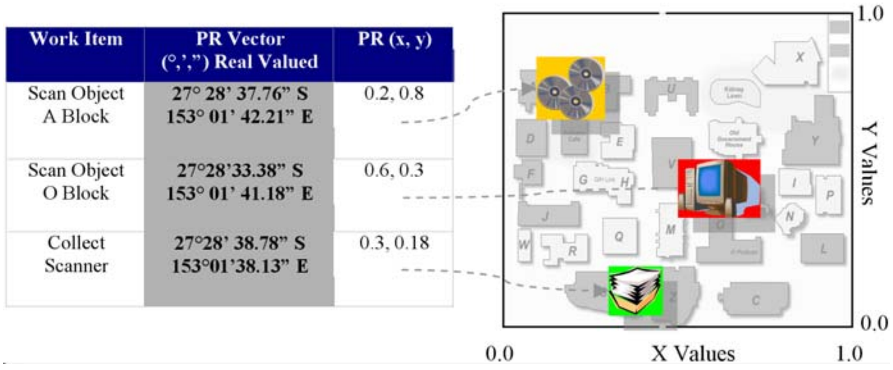
Each is a representation that can be used within a workflow system to decide about task choices, with regards to the relevant resources. They can be turned on and off by the designer of the workflow visualisation to allow or deny access to extra information regarding tasks. Each one can be modified to suit a particular application area, thus leaving room for development of novel visualisations tailor-made for different applications. Included are example mapping functions for the PR, to show formally how it is transformed into `(x,y)` visualisation locations.

5.3 Resource Mapping Examples

Let us return now to consider the stock take example scenario. For illustrative purposes, we have chosen a subset of the tasks in the case study. The example worklist contains the following three tasks (ie., work items).

1. Collect scanner from S Block;
2. Scan items in O Block;
3. Scan items in A Block.

We present a visualisation example for three resources we have chosen in the resource model: location, people and time, and apply them to the listed tasks in the stocktake case study.



Work Item	PR Vector (°,’,”) Real Valued	PR (x, y)
Scan Object A Block	27° 28’ 37.76” S 153° 01’ 42.21” E	0.2, 0.8
Scan Object O Block	27°28’33.38” S 153° 01’ 41.18” E	0.6, 0.3
Collect Scanner	27°28’ 38.78” S 153°01’38.13” E	0.3, 0.18

Fig. 7. Example Location Resource visualisation for the computer stock take example, mapping longitude and latitude values via a scale and translate to normalised window coordinates, overlaid onto a map of the QUT campus. A table of values is shown on the left, with the mapped locations on the right hand side as an example visualisation.

Example 1. **Location Resource (Figure 7)**

Task: Compare the spatial locations of tasks to be performed for logistical purposes.

Visualisation: Map detailing the arrangements of tasks in space, to aid the worker in identifying efficient ordering of the work.

Mapping Function: A variation of the coordinate mapping, $w_{dc} = M \times r_p$, where r_p consists of location resources information in specified units (eg. latitude and longitude) mapped to x_w, y_w values in pixels on visualisation device window, where M is the window transform (scale and translate) from world coordinates to device coordinates.

Example 2. **People Resource (Figure 8)**

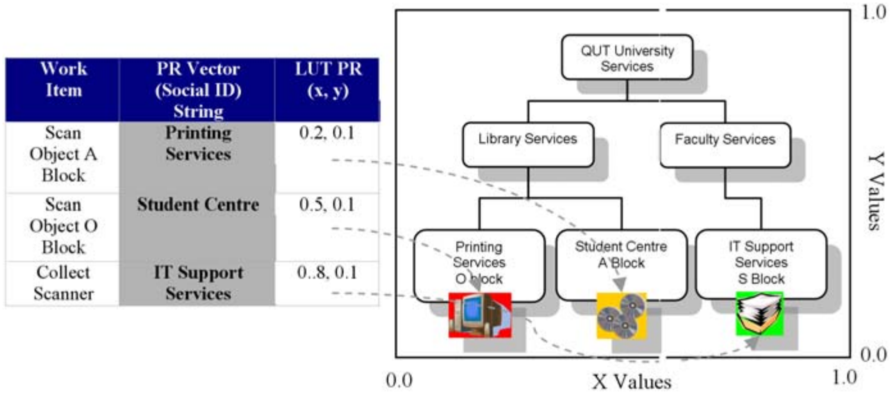
Task: Assess the capabilities of the people available for task.

Visualisation: Overlays of people available to meet task with encoding of match between people and the tasks colours/textures, including hierarchical views, social network views. In this example, the work items are mapped to a hierarchy list, to show which bureaucratic section the example is executed in, in order to see who needs to be contacted for task execution purposes.

Mapping Function: A form of the region mapping, $w_{dc} = LUT(w_{soc})$, where LUT is a window coordinate lookup table function indexed by the id tag w_{soc} for the social group in which the worklist item belongs.

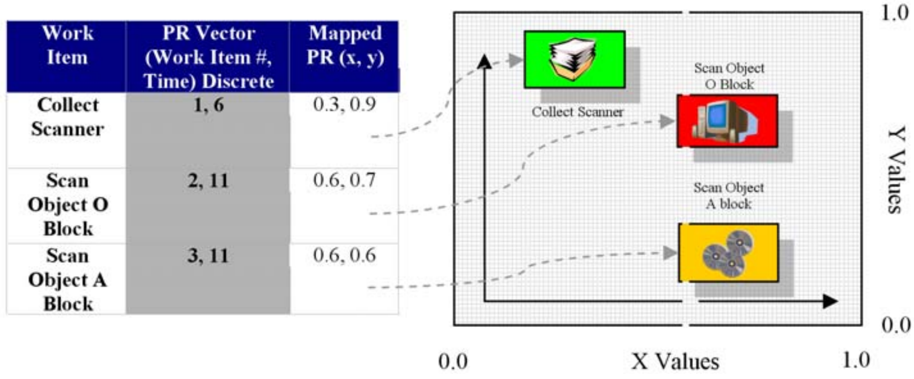
Example 3. **Time Resource (Figure 9)**

Task: Compare the relative start and finish times for each task and insert it into the worklist at appropriate moments if time resources are available, either



Work Item	PR Vector (Social ID) String	LUT PR (x, y)
Scan Object A Block	Printing Services	0.2, 0.1
Scan Object O Block	Student Centre	0.5, 0.1
Collect Scanner	IT Support Services	0..8, 0.1

Fig. 8. Example People resource visualisation as a QUT section hierarchy, showing the region mapping from hierarchy ID to visualisation position via a lookup table. A table of values is shown on the left, with the mapped locations on the right hand side as an example visualisation.



Work Item	PR Vector (Work Item #, Time) Discrete	Mapped PR (x, y)
Collect Scanner	1, 6	0.3, 0.9
Scan Object O Block	2, 11	0.6, 0.7
Scan Object A Block	3, 11	0.6, 0.6

Fig. 9. Example time resource visualisation for the stock take example. Each PR is a tuple of work item ID and time in hours. A table of values is shown on the left, with the mapped locations on the right hand side as an example visualisation. Depending on the intended workflow application, the other resources: Space, Materials, Equipment, Services and Active Worklist, can be visualised using a similar mapping process as the principal resource on the overlay plane. We now define an interaction framework for the previously mapped resource visualisations.

by leaving the task as whole, or dividing it into smaller components for insertion into small time gaps.

Visualisation: Gantt Chart showing all available tasks on a time line in stacked manner to identify insertion points for the worklist items.

Mapping Function: A fixed grid mapping, $w_{dc} = f(w_{id}, t)$, where w_{id} is the work item identification number, t is the time and w_{dc} the final visualisation window device coordinates.

6 Interacting with the Visualised Worklist

In the visualised worklist, each task is represented by a coloured icon. An aggregated icon represents multiple instances of the same task. Figure 10 shows an example of a task with multiple instances. An aggregated icon is shown with four icons with numeric information regarding the number of instances and their

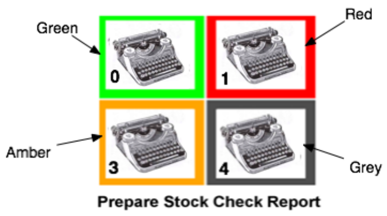


Fig. 10. Illustration of an aggregated icon made up of single task icons. The example shows a task titled "Prepare Stock Check Report" with zero checked in, one checked out, three available and one task unavailable. The colour of each segment is annotated beside the icon for clarity.

status within the system. The state of any delivered task at one time may be the following: inactive, available, checked out and suspended, and included is the colour we have mapped to the state using the traffic light metaphor of red, green and amber:

– Inactive: unavailable to the worker (grey);
– Available: available to the worker to check out (amber);
– Checked Out: has been checked out by the worker (green);
– Checked In: has been checked in and completed by the worker (red);
– Suspended: has been checked out by the worker, is still incomplete, but checked in to the user (amber dashed).

After executing the visual worklist handler, and linking with a work flow server, the user interacts with the work item icons, by clicking on the icons to check out available tasks, and by clicking on checked out icons to check in

Table 2. List of worklist visualisation handler user interactions

User Interaction	Description
Check Out	The user right clicks on an Amber icon component within a visualisation, and chooses an instance of the work item in order to take responsibility for the task.
Check In	The user right clicks on the Green icon component and chooses an instance of the work item to check in, to notify the workflow system of the completion of the task.
Change View	To change visualisation type, the user clicks on the window tab listing the desired visualisation.

completed tasks. Whenever appropriate, a form will be presented by the workflow system, to obtain data from the worker. The visual worklist handler user interactions are listed in Table 2.

7 Implementation

A major test of any workflow visualisation approach is its ability to be incorporated into a modern client server-based workflow system. We have built a prototype of the proposed visualisation framework, and interfaced it with the workflow system YAWL. This section discusses the system architecture and implementation in detail.

7.1 The YAWL Environment

Our implementation is based on the open source workflow environment named YAWL (Yet Another Workflow Language), which is a research initiative at Queensland University of Technology [23]. YAWL is based on a set of workflow patterns developed via analysis and comparison of a number of commercial workflow systems. It provides a powerful and formal workflow description language, as well as an execution environment.

To understand the architecture of our visualisation framework, we first present the overall architecture of YAWL. Workflow specifications are created in the YAWL designer which is a graphical editor, and deployed to the YAWL engine. The engine performs verification of the specifications and stores them in the YAWL repository. The specification can be loaded and launched for execution via the YAWL manager, and is hereafter referred to as a schema. The execution itself is managed by the YAWL engine. The YAWL engine interacts with the components labelled as YAWL services through *Interface B*. The YAWL services (worklist handler, web services broker, interoperability broker and custom YAWL services) are based on the web services paradigm and all are abstracted as services in YAWL.

How the engine communicates with the YAWL worklist handler is of particular interest in our work. The worklist handler is the component that is responsible for dispatching tasks to the workers. Through the worklist handler, the workers accept tasks and mark their completions. In conventional workflow systems, the worklist handler is part of the workflow engine. However, in the YAWL environment it is a separate component that interacts with the engine through Interface B. Through the interface, a custom service or application can be developed to extract worklist information for display in whatever manner is required.

7.2 Worklist Visualisation Architecture

Based on the existing YAWL architecture, we have developed a new type of YAWL worklist handler which interacts with the engine through Interface B. The overview architecture is shown in Fig. 11. It has capabilities to (i) display the visualised resources and (ii) dispatch tasks like a normal worklist handler. The

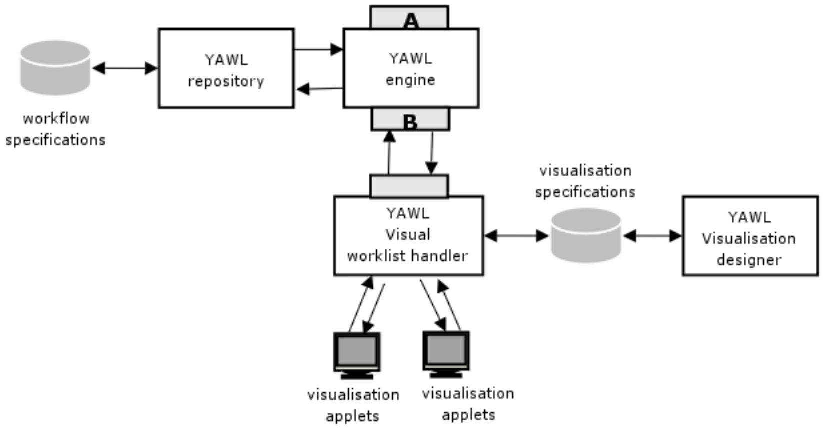


Fig. 11. YAWL Visualisation Framework: Overall architecture

architecture consists of two components which have been designed and partially implemented: a visual worklist handler and a visualisation designer.

The visual worklist handler can view multiple cases of running workflows, with multiple resource-centric views matched to the requirements devised by the YAWL schema designer. The worker loads the cases and is presented with a list of tasks, and a tabbed view list to switch between difference representations of the worklists. In the following two sections we describe the two components, and illustrate them with developed examples.

7.3 YAWL Visualisation Designer

The designer application is the most complete at this stage. It is designed around the structure of the visualisation approach we have developed, and is implemented in Java, as is the rest of the YAWL implementation. The visualisation designer allows the user to load Scalable Vector Graphics (SVG) files as backgrounds and icons for the overlay planes. This allows easy modification of images via other drawing tools. The SVG component of the designer is managed by the Batik Java package [24]. The designer application is an implementation of the work item coordinate scheme we detailed earlier. This designer allows the easy outlaying of tasks as icons across the background in the program. The process of designing a visualisation view for a schema is as follows:

First decide on the background and overlay images, editing them in a separate tool and saving them as SVG files. Decide on the spatial arrangement of the tasks to be displayed according to the resources that need to be analysed, for example a map for logistics on QUT campus that will help a worker to decide where to perform their tasks. Until there is an adequate implementation of a resource model in YAWL, the icons are located by hand for the purposes of proof of concept. Load the workflow schema into the editor to obtain the tasks in the system, which appear in a mouse menu on a right click at the chosen location on the background (refer to Fig. 12). Load the background image. Set the current

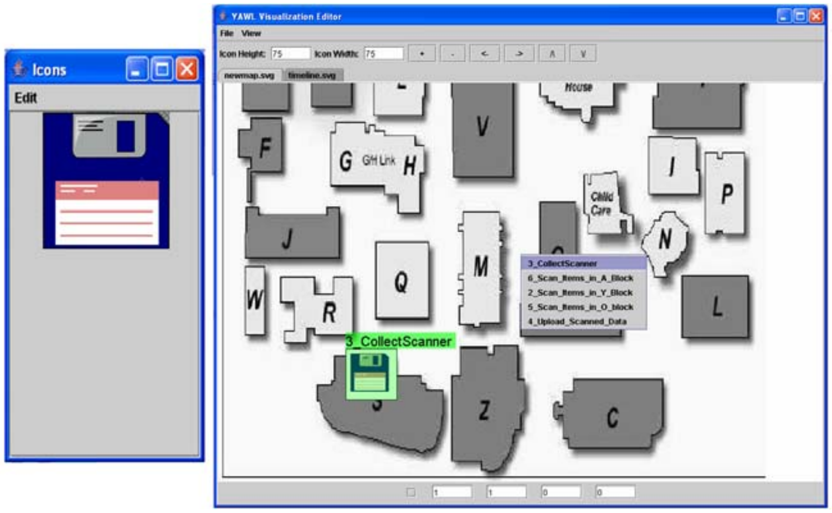


Fig. 12. Yawl Visualisation Designer: main components. The left window is the list of active icons, the right window is the main editing window, with the collect scanner work item being placed on the diagram.

icon to be used by choosing from the list in a dialog (refer to Fig. 12). Move pointer to the location of the worklist item and right click to choose a task, and icon, repeating for all worklist items.

Figure 12 illustrates the major components of the visualisation designer user interface via the stocktaking example on the campus map. The large window is the main window for visualisation design, and the smaller window shows a list of potential icons to be placed at locations on the visualisation. Each view is placed into a tabbed list, just as they are to be displayed in the visualisation agent. The menu is displayed using a mouse right click, showing the tasks defined in the schema. The icon can be placed at the location of the right click of the mouse, or using actual coordinates in the text entry boxes at the bottom of the screen. The icon at the bottom left of the image is the current task icon, "CollectScanner" and is shown using a disk icon.

This visualisation design information is stored in an XML file (see Fig. 13) that defines an arbitrary number of views per schema, and the task icons, gained from the number of tasks within the YAWL schema. This file is then read by the Visual Worklist Handler to form the visualisation structure for communication to the YAWL engine. The following is a snippet from a visualisation specification. A specification may have a number of views `<view>`, and each view may have a number of tasks `<task>`. A view is associated with a background representing a resource. Each task is assigned a color for the description, coordinates, and an icon.

We have implemented the beginnings of a visualisation editor and visualisation viewer, which we show in this paper. In a final implementation, additional resource information will be selected from the resource view of the YAWL schema

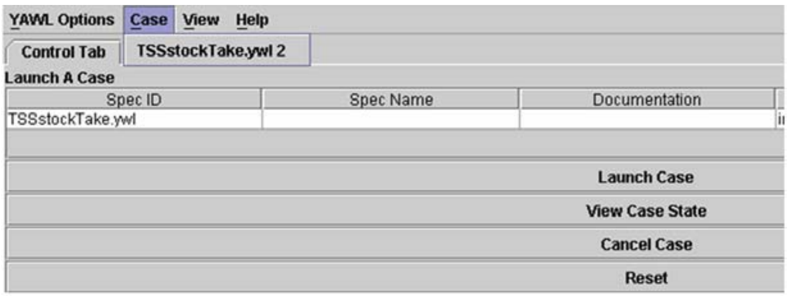
```
<specification id = "TSSstockTake.ywl"
uri = "file:/D:/Yawlstuff/batik/demo/TssStockTake.xml">
<view id = "file:/D:/Yawlstuff/batik/demo/map-1/newmap.svg">
<task id = "3_CollectScanner">
<color> -16777216 </color>
<coordX> 240 </coordX>
<coordY> 760 </coordY>
<icon width="75" height="75">
    file:/D:/Yawlstuff/demo/floppy.svg
</icon>
</task>
</view>
</specification>
```

Fig. 13. Example of visualisation specification file

as it is running. For now we are able to design worklists arranged according to grid, spatial and time arrangements.

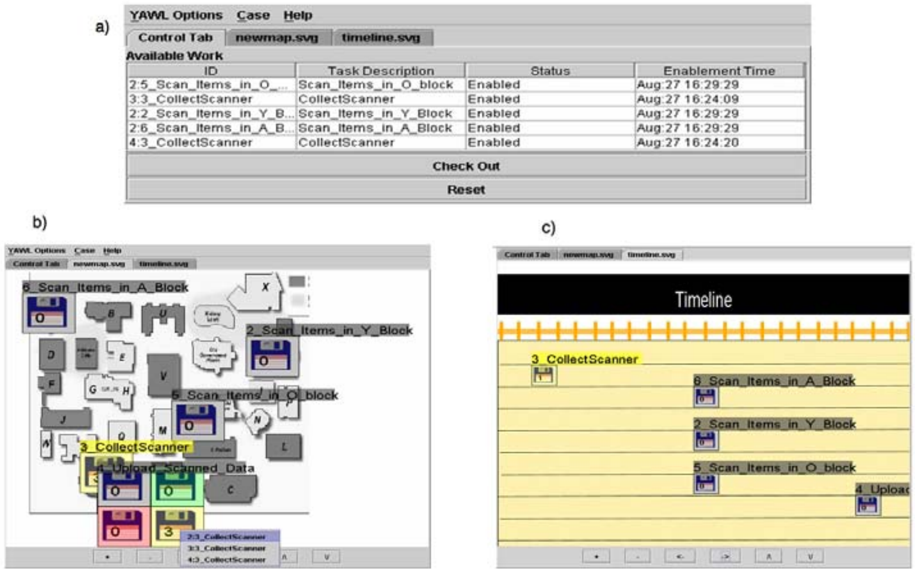
7.4 YAWL Visual Worklist Handler

Worklists are disseminated in YAWL via the default worklist handler as simple dialogs containing lists of tasks, with no other resource information being displayed. We have begun implementing a visual worklist handler that is an extension of the default handler. The YAWL workflow implementation is structured around a component architecture that communicates via XML formatted commands. Thus the worklist handler is able to utilise the B interface to the running YAWL case in the same manner as the default worklist handler. The visual worklist handler is able to execute the visualisation developed with the designer that is stored in a file (see Fig. 11). The new worklist handler allows a more intuitive mapping of task coordinates to the check in and check out process. The user is able to check items in and out by simply clicking on the potential worklist item in its location on a map or hierarchy diagram. A running version of this design is shown in Fig. 14 and Fig. 15.



Spec ID	Spec Name	Documentation
TSSstockTake.ywl		

Fig. 14. The administration screen for the visualisation program, showing the stock take work flow schema being loaded and executed



Available Work

ID	Task Description	Status	Enablement Time
2:5_Scan_Items_in_O_...	Scan_Items_in_O_block	Enabled	Aug:27 16:29:29
3:3_CollectScanner	CollectScanner	Enabled	Aug:27 16:24:09
2:2_Scan_Items_in_Y_B...	Scan_Items_in_Y_Block	Enabled	Aug:27 16:29:29
2:6_Scan_Items_in_A_B...	Scan_Items_in_A_Block	Enabled	Aug:27 16:29:29
4:3_CollectScanner	CollectScanner	Enabled	Aug:27 16:24:20

Fig. 15. Screen dump of a running visualisation handler, showing a) default simple work item list, b) campus map visualisation, and c) same worklist viewed from the timeline perspective

With a varied spatial organisations to the tasks, the person doing this stock take process can evaluate the task, using the map to make a decision about the acceptance of the worklist item in consideration of the location, time and potentially other resources.

8 Conclusion and Ongoing Work

We have described the beginnings of a thorough analysis of workflow visualisation; its theoretical basis, resource centric approach and appropriate visualisation techniques. Analysis of these sections revealed how to use these techniques within a typical workflow system. The task coordinate approach was described, showing how this can be generalised across a number of visualisations using a background and overlay approach. We have also begun the development of a visualisation development environment, with an editor and visualisation agent that uses SVG files and is easily integrated into the YAWL workflow system created by the BPM group at QUT. We have therefore shown supporting evidence that this visualisation approach can be used within a fully featured workflow environment.

Further analysis will continue to refine the visualisation mappings to produce a knowledge base for development of visualisations within workflow applications. Firstly, there is the need to investigate the dependencies and relationships between separate resources, so that these relationships can be represented properly in the worklist visualisations, to enhance the decision making capabilities of the worker regarding multiple resources. In addition, there will be refinement of the broad categories of resources into more fine grained categories to derive a rule-base for an intelligent design agent to be incorporated into the visualisation designer. Evaluation experiments will be performed within a case study in order to ascertain the effectiveness of the resource centric visualisation approach with users of workflow tools.

In addition, we are currently working on generalising the idea of worklist, which is produced from a workflow engine, to *tasklist*. Our tasklists can be (i) automatically populated from the work coming from various resources including a workflow engine, email correspondence, or calendar events, etc. and (ii) presented (i.e., visualised) in a coherent way that aids completion of a given task as well as providing a significant control over all the facet of task visualisation. The project is inspired by the fact that the data and knowledge we consume for everyday tasks are more and more distributed (e.g., Internet, Intranet, email clients, mobile devices, calendar clients), improving productivity in workplace means improving the way people manage such knowledge. We plan to, first, exploit the latest resource view developments that are being implemented within YAWL, to enable the run time specification of resources and data associated with a task.

Acknowledgement

This project is partially supported by a QUT Faculty of Information Technology collaborative grant. We acknowledge the programming assistance provided by Tore Fjellheim and Guy Redding, who programmed the visualisation editor and agent applets and integrated them into the YAWL workflow system. Their dedication and hard work towards implementing this project have been greatly appreciated.

References

1. Keller, P., Keller, M.: Visual Cues. IEEE Press, Piscataway (1993)
2. Tufte, E.: The Visual Display of Quantitative Information. Graphics Press, Cheshire (1983)
3. Luttighuis, P., Lankhorst, M., van der Wetering, R., Bal, R., van Berg, H.: Visualising Business Processes. Computer Languages, 39–59 (2001)
4. Bobrik, R., Reichert, M., Bauer, T.: View-based process visualization. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 88–95. Springer, Heidelberg (2007)
5. Streit, A., Pham, B., Brown, R.: Visualization support for managing large business process specifications. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 205–219. Springer, Heidelberg (2005)
6. UNISYS: 3D Visible Enterprise (2004), `http://www.3dvisibleenterprise.com/3dv/`

7. Schonhage, B., van Ballegooij, A., Elliens, A.: 3D gadgets for business process visualization:a case study. In: Symposium on Virtual Reality Modeling Language, Monterey, California, pp. 131–138. ACM Press, New York (2000)
8. Systems, I.S.: Interactive Software (2004), http://www.interactive-software.de/
9. Beeri, C., Eyal, A., Kamenkovich, S., Milo, T.: Querying business processes with bp-ql. In: Proceedings of the 31st international conference on Very large data bases, VLDB Endowment, pp. 1255–1258 (2005)
10. Freire, J., Silva, C.: Towards enabling social analysis of scientific data. In: Social Data Analysis Workshop (in conjunction with CHI 2008 (2008)
11. Jones, W., Bruce, H.: A report on the nsf-sponsored workshop on personal information management (2005), http://pim.ischool.washington.edu/
12. Karger, D., Bakshi, K., Huynh, D., Quan, D., Sinha, V.: Haystack: A general purpose information management tool for end users of semistructured data, pp. 13–26 (2005)
13. Dittrich, J., Salles, M.V.: idm: A unified and versatile data model for personal dataspace management. In: VLDB, pp. 367–378 (2006)
14. Latva-Koivisto, A.: User interface design for business process modelling and visualisation. Technical report, Department of Computer Science, Helsinki University of Technology, Helsinki, Masters Thesis (2001)
15. Leroux, H., Exton, C.: COOPE: a tool for representing concurrent object-oriented program execution through visualisation. In: Proc. of 9th Euromicro Workshop Parallel and Distributed Processing, pp. 71–76 (2001)
16. Jennings, N., Norman, T., Faratin, P.: ADEPT: An Agent-based Approach to Business Process Management. ACM SIGMOD Record 27, 29–32 (1998)
17. Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M., Edmond, D.: Workflow Resource Patterns: Identification, Representation and Tool Support. In: Pastor, Ó., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 216–232. Springer, Heidelberg (2005)
18. Curtis, B., Kelner, M., Over, J.: Process modelling. Communication of the ACM 35, 75–90 (1992)
19. Heijens, S.: Support for workflow administration and monitoring in the yawl environment. Master's thesis, Vrije Universiteit Amsterdam (August 2005), http://www.yawl.fit.qut.edu.au/yawldocs/
20. Fadel, F.G.: A Resource Ontology for Enterprise Modelling. Technical report, Enterprise Integration Laboratory, Toronto, M.A.Sc. (1994)
21. Shreiner, D., Woo, M., Neider, J., Davis, T.: OpenGL(R), Programming Guide: The Official Guide to Learning OpenGL(R), 5th edn. Addison-Wesley Professional, New York (2005)
22. Schneiderman, B.: Designing the User Interface, 3rd edn. Addison-Wesley, Reading (1997)
23. van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: Yet another workflow language. Information Systems 30, 245–275 (2005)
24. Batik: Batik SVG Toolkit (2005), http://xml.apache.org/batik

Author Index