I N F S Y S
R ESEARCH
R E P O R T

INSTITUT FÜR INFORMATIONSSYSTEME
ARBEITSBEREICH WISSENSBASIERTE SYSTEME

FROM WEB SEARCH TO SEMANTIC WEB SEARCH

BETTINA FAZZINGA GIORGIO GIANFORME
GEORG GOTTLOB THOMAS LUKASIEWICZ

INFSYS RESEARCH REPORT 1843-08-11
NOVEMBER 2008

Institut für Informationssysteme
AB Wissensbasierte Systeme
Technische Universität Wien
Favoritenstraße 9-11
A-1040 Wien, Austria
Tel: +43-1-58801-18405
Fax: +43-1-58801-18493
sek@kr.tuwien.ac.at
www.kr.tuwien.ac.at

TU WIEN

INFSYS RESEARCH REPORT

INFSYS RESEARCH REPORT 1843-08-11, NOVEMBER 2008

FROM WEB SEARCH TO SEMANTIC WEB SEARCH

NOVEMBER 5, 2008

Bettina Fazzinga [1] Giorgio Gianforme [2] Georg Gottlob [3]
Thomas Lukasiewicz [4]

Abstract. Many experts predict that the next huge step forward in Web information technology will be achieved by adding semantics to Web data, and will possibly consist of (some form of) the Semantic Web. In this paper, we present an approach to Semantic Web search, which combines standard Web search with ontological background knowledge. In fact, we show how standard Web search engines can be used as the main inference motor for ontology-based search. To make this possible, lightweight software clients are used for annotation and query decomposition. We develop the formal model behind this approach and also provide an implementation in desktop search. Experiments show that the implementation scales quite well to very large amounts of data.

---

[1]Dipartimento di Elettronica, Informatica e Sistemistica, Università della Calabria, Via P. Bucci, cubo 42 c, 87036 Rende, Italy; e-mail: bfazzinga@deis.unical.it.

[2]Dipartimento di Informatica e Automazione, Università Roma Tre, Via della Vasca Navale 79, 00146 Roma, Italy; e-mail: giorgio.gianforme@gmail.com.

[3]Computing Laboratory and Oxford-Man Institute of Quantitative Finance, University of Oxford, Wolfson Building, Parks Road, Oxford OX1 3QD, UK; e-mail: georg.gottlob@comlab.ox.ac.uk.

[4]Computing Laboratory, University of Oxford, Wolfson Building, Parks Road, Oxford OX1 3QD, UK; e-mail: thomas.lukasiewicz@comlab.ox.ac.uk. Institut für Informationssysteme, Technische Universität Wien, Favoritenstraße 9-11, 1040 Wien, Austria; e-mail: lukasiewicz@kr.tuwien.ac.at.

Acknowledgements: Georg Gottlob's work was supported by the EPSRC grant Number EP/E010865/1 "Schema Mappings and Automated Services for Data Integration." Georg Gottlob, whose work was partially carried out at the Oxford-Man Institute of Quantitative Finance, gratefully acknowledges support from the Royal Society as the holder of a Royal Society-Wolfson Research Merit Award. Thomas Lukasiewicz's work was supported by the German Research Foundation (DFG) under the Heisenberg Programme.

1 Introduction

Web search is a key technology of the Web, since it is the primary way to access content in the ocean of Web data. Current Web search technologies are essentially based on a combination of textual keyword search with an importance ranking of documents via the link structure of the Web [2].

Web search, however, is about to change radically with the development of a more powerful future Web, called the *Semantic Web*, which is a common framework that allows data to be shared and reused in different applications, enterprises, and communities [1]. The Semantic Web is an extension of the current 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. It consists of several hierarchical layers, where the *Ontology layer*, in form of the *OWL Web Ontology Language*, is the highest layer that has currently reached a sufficient maturity. Some important layers below the Ontology layer are the *RDF* and *RDF Schema layers* along with the *SPARQL* query language. For the higher *Rules*, *Logic*, and *Proof layers* of the Semantic Web, one has especially developed languages integrating rules and ontologies, and languages supporting more sophisticated forms of knowledge. During the recent decade, a huge amount of academic and commercial research activities has been spent towards realizing the Semantic Web. Hence, in addition to the traditional Web pages, future Web data are expected to be more and more organized in the new formalisms of the Semantic Web, and will thus also consist of RDF data along with ontological and rule-based knowledge. The development of a new search technology for the Semantic Web, called *Semantic Web search*, is currently an extremely hot topic, both in Web-related companies and in academic research. In particular, there is a fastly growing number of commercial and academic Semantic Web search engines. The following is a list of already existing or currently being developed Semantic Web search engines: Semantic Web Search Engine (SWSE)[1], Watson[2], Falcons[3], Semantic Web Search[4], Sindice[5], Yahoo! Microsearch[6], Swoogle[7], and Zitgist Search[8]. In parallel, the number of academic research papers on Semantic Web search is also increasing rapidly. The papers on the academic Semantic Web search engine Swoogle are among the earliest of such works [4, 5].

The main idea behind the present paper is to combine standard Web search with the power of Semantic Web formalisms and technologies. However, differently from the above-mentioned approaches to Semantic Web search, which are developing Semantic Web search as a completely new formal framework and technology, directed towards searching the Semantic Web as a future substitute for the current Web, the research proposed here is based on the central idea of realizing Semantic Web search by combining standard Web search with background knowledge. More

[1] http://swse.deri.org/
[2] http://watson.kmi.open.ac.uk/WatsonWUI/
[3] http://iws.seu.edu.cn/services/falcons/
[4] http://www.semanticwebsearch.com/query/
[5] http://www.sindice.com/
[6] http://www.yr-bcn.es/demos/microsearch/
[7] http://swoogle.umbc.edu/
[8] http://zitgist.com/

concretely, we show how standard Web search engines can be used as the main inference motor for Semantic Web search. To this end, lightweight user-site software clients are used for annotation and query decomposition. As important advantages of this approach, it can be applied to the whole existing Web (and not only to the future Semantic Web), it can be done immediately (and not only when the future Semantic Web is in place), and it can be done with existing Web search technology (and so does not require completely new technologies). This line of research aims at making current search engines for the existing Web more "semantic" (i.e., in a sense also more "intelligent") by combining the information on existing Web pages with background ontological knowledge. Intuitively, rather than being interpreted in a keyword-based syntactic fashion, the pieces of data on existing Web pages are interpreted by their connected ontology-based semantic background knowledge. That is, the pieces of data on Web pages are connected to a much more precise semantic and contextual meaning. This allows for more complex ontology-based Web search queries, and it also allows for answering Web search queries in a much more precise way, as the following simple examples show:

- As for complex Web search queries, when searching for a movie, one may be interested in movies that were produced by a US company before 1999 and had a French director. Similarly, when buying a house in a town, one may be interested in large house selling companies within 50 miles of that town, existing for at least 15 years, and not known to be blacklisted by a consumer organization in the last 5 years. Such queries are answered by connecting the information on existing Web pages with available background knowledge.
- Suppose next that one is searching for "laptop" on the Web. Then, one is looking for laptops or synonyms / related concepts (such as "notebook"), but also for special kinds of laptops that are not synonyms / related concepts, such as e.g. IBM/Lenovo ThinkPads. Semantic background knowledge now allows for obtaining both a collection of contextually correct synonyms / related concepts and a collection of contextually correct special kinds of laptops.
- Similarly, a Web search for "president of the USA" should also return Web pages that contain "George W. Bush" (who is/was one of the presidents of the USA according to some background ontology). Also, a Web search for "the president of the USA on September 11, 2001" should return Web pages mentioning "George W. Bush" (who was the president of the USA on September 11, 2001, according to some background ontology). On the other hand, when searching for Web pages about the first president of the USA, "Washington", semantic annotations and background knowledge allow us to restrict our search to Web pages that are actually about Washington as the name of the president, and so to ignore, e.g., Web pages about the state or town.

The above are examples of very simple Web search queries, which can be handled in our Semantic Web search (assuming suitable semantic annotations are available), but not appropriately in current Web search. The main contributions of this paper are summarized as follows:

- We present a novel approach to Semantic Web search, where standard Web search engines are combined with ontological background knowledge. We show how the approach can be implemented on top of standard Web search engines and ontological inference technologies,

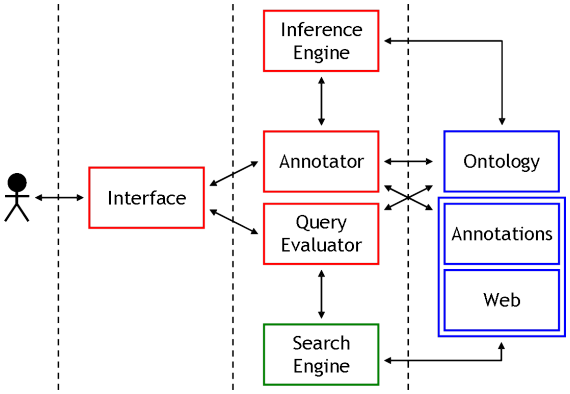

Figure 1: System architecture.

using lightweight user-site software clients for annotating Web pages and query decomposition.

- We develop the formal model behind this approach, which is based on tractable ontology languages. More specifically, we introduce Semantic Web knowledge bases and Semantic Web search queries to them. We also generalize the PageRank technique from standard Web search to our approach to Semantic Web search.
- We provide a technique for processing Semantic Web search queries, which consists of an offline inference (compiling terminological knowledge into annotations) and an online reduction to a collection of standard Web search queries. We prove that this way of processing Semantic Web search queries is ontologically correct.
- We report on an implementation of our approach in the framework of desktop search. We provide experimental results, which show that the approach scales quite well to very large amounts of data.

The rest of this paper is organized as follows. In Section 2, we give an overview of our approach to Semantic Web search. Section 3 recalls the basics of the underlying tractable ontology language. In Section 4, we introduce Semantic Web knowledge bases and Semantic Web search queries, including a generalized PageRank technique. Sections 5 and 6 describe how Semantic Web search queries are processed via offline inference and online reduction to standard Web search. In Section 7, we describe a first implementation for semantic desktop search on top of Windows Desktop Search, along with experimental results. Section 8 summarizes our main results and gives an outlook on future research.

2 System Overview

The overall architecture of our Semantic Web search system is shown in Fig. 1. It consists of the *Interface*, the *Annotator*, the *Inference Engine*, and the *Query Evaluator* (Fig. 1, red parts),

which are lightweight user-site software clients on top of standard Web *Search Engines*. Standard *Web* pages and their contained objects are enriched by *Annotation* pages, which are based on a background *Ontology*.

More concretely, the *Annotator* allows the user to add semantic annotations to standard Web pages and to objects on standard Web pages. For example, in a very simple scenario, a Web page i_1 may contain information about a Ph.D. student i_2, called Mary, and two of her papers, namely, a conference paper i_3 entitled "*Semantic Web search*" and a journal paper i_4 entitled "*Semantic Web search engines*" and published in 2008. Note that we assume that Web pages and their objects have unique identifiers (see Section 6.1). A simple HTML page representing this scenario is shown in Fig. 2. The user may now add one semantic annotation each for the Web page, the Ph.D. student Mary, the journal paper, and the conference paper. The annotation for the Web page may simply encode that it mentions Mary and the two papers, while the annotation for Mary may encode that she is a Ph.D. student with the name Mary and the author of the papers i_3 and i_4. The annotation for the paper i_3 may encode that i_3 is a conference paper and has the title "*Semantic Web search*", while the annotation for the paper i_4 may encode that i_4 is a journal paper, authored by Mary, has the title "*Semantic Web search engines*", was published in 2008, and has the keyword "RDF". The semantic annotations of i_1, i_2, i_3, and i_4 are formally expressed as the following sets of axioms $\mathcal{A}_{i_1}$, $\mathcal{A}_{i_2}$, $\mathcal{A}_{i_3}$, and $\mathcal{A}_{i_4}$, respectively:

$$\begin{aligned}
\mathcal{A}_{i_1} &= \{\mathit{contains}(i_1, i_2), \mathit{contains}(i_1, i_3), \mathit{contains}(i_1, i_4)\},\\
\mathcal{A}_{i_2} &= \{\mathit{PhDStudent}(i_2),\ \mathit{name}(i_2, \text{“}\mathit{mary}\text{”}), \mathit{isAuthorOf}(i_2, i_3),\\
&\qquad \mathit{isAuthorOf}(i_2, i_4)\},\\
\mathcal{A}_{i_3} &= \{\mathit{ConferencePaper}(i_3), \mathit{title}(i_3, \text{“}\mathit{Semantic\ Web\ search}\text{”})\},\\
\mathcal{A}_{i_4} &= \{\mathit{JournalPaper}(i_4),\ \mathit{hasAuthor}(i_4, i_2),\\
&\qquad \mathit{title}(i_4, \text{“}\mathit{Semantic\ Web\ search\ engines}\text{”}),\\
&\qquad \mathit{yearOfPublication}(i_4, 2008), \mathit{keyword}(i_4, \text{“RDF”})\}.
\end{aligned}$$

Using an ontology containing some background knowledge (e.g., from some Semantic Web repositories), these semantic annotations are then further enhanced in an offline inference step, where the *Inference Engine* adds all properties that can be deduced from the semantic annotations and the ontology. The resulting (*completed*) semantic annotations are then published as Web pages, so that they can be searched by standard Web search engines. E.g., an ontology may contain the knowledge that all journal and conference papers are also articles, that conference papers are not journal papers, and that "is author of" is the inverse relation to "has author", which is formally expressed by the following axioms:

$$\begin{aligned}
&\mathit{ConferencePaper} \sqsubseteq \mathit{Article},\ \mathit{JournalPaper} \sqsubseteq \mathit{Article},\\
&\mathit{ConferencePaper} \sqsubseteq \neg \mathit{JournalPaper},\\
&\mathit{isAuthorOf}^- \sqsubseteq \mathit{hasAuthor},\ \mathit{hasAuthor}^- \sqsubseteq \mathit{isAuthorOf}.
\end{aligned}$$

Using this ontological background knowledge, we can derive from the above annotations that the two papers i_3 and i_4 are also articles, and are both authored by Mary.

These searchable completed semantic annotations of (objects on) standard Web pages produced by the Annotator are published as HTML Web pages with pointers to the respective object pages.

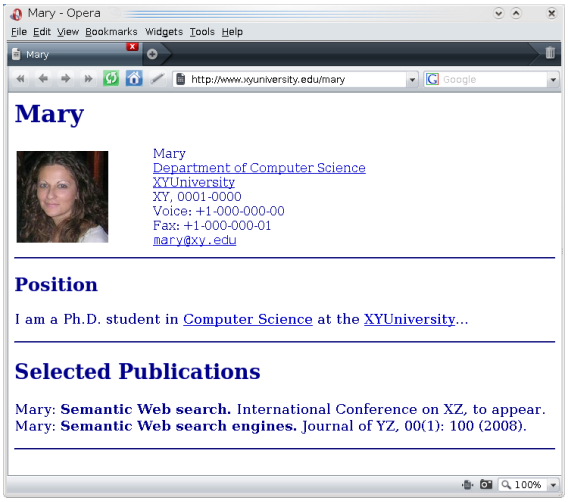

Figure 2: An HTML page p.

www.xyuniversity.edu/mary/an1.html

<html>
<body>
www.xyuniversity.edu/mary
WebPage i_1
contains i_2
contains i_3
contains i_4
</body>
</html>

www.xyuniversity.edu/mary/an2.html

<html>
<body>
www.xyuniversity.edu/mary
PhDStudent i_2
name mary
isAuthorOf i_3
isAuthorOf i_4
</body>
</html>

www.xyuniversity.edu/mary/an3.html

<html>
<body>
www.xyuniversity.edu/mary
Article i_3
ConferencePaper i_3
hasAuthor i_2
title Semantic Web search
</body>
</html>

www.xyuniversity.edu/mary/an4.html

<html>
<body>
www.xyuniversity.edu/mary
Article i_4
JournalPaper i_4
hasAuthor i_2
title Semantic Web search engines
yearOfPublication 2008
keyword RDF
</body>
</html>

Figure 3: HTML pages p_1, p_2, p_3, and p_4 encoding (completed) semantic annotations for p.

For example, the HTML pages for the completed semantic annotations of the above $\mathcal{A}_{i_1}$, $\mathcal{A}_{i_2}$, $\mathcal{A}_{i_3}$, and $\mathcal{A}_{i_4}$ are shown in Fig. 3. Note that these are not handwritten but comfortably obtained through an incentive ontology browsing and annotation step.

The *Query Evaluator* (see Fig. 1) reduces each Semantic Web search query of the user in an online step to a sequence of standard Web search queries on standard Web and annotation pages, which are then processed by a standard Web *Search Engine*, assuming standard Web and annotation pages are appropriately indexed. The Query Evaluator also collects the results and re-transforms them into a single answer which is returned to the user. As an example of a Semantic Web search query, one may ask for all Ph.D. students who have published an article in 2008 with RDF as a keyword, which is formally expressed as follows:

$$Q(x) = \exists y\,(\textit{PhDStudent}(x) \wedge \textit{isAuthorOf}(x,y) \wedge \textit{Article}(y) \wedge \textit{yearOfPublication}(y, 2008) \wedge \textit{keyword}(y, \text{“}RDF\text{”}))\,.$$

This query is transformed into the two queries $Q_1 =$ *PhDStudent* AND *isAuthorOf* and $Q_2 =$ *Article* AND *“yearOfPublication* 2008” AND *“keyword* RDF”, which can be both submitted to a standard Web search engine, such as Google. The result of the original query Q is then constructed from the results of the two queries Q_1 and Q_2.

More formally, the core of our approach reduces Semantic Web search to standard Web search via three transformations, τ_1, τ_2, and τ_3. Transformation τ_1 takes a semantic annotation $\mathcal{A}$ and transforms it (using an ontology) into its completed version $\tau_1(\mathcal{A})$, which can be appropriately indexed and searched via standard keyword search. These descriptions are published as Web pages. Transformation τ_2 takes an ontology-based Semantic Web query Q and translates it into a (usually short) sequence $\tau_2(Q) = \langle K_1, K_2, \ldots, K_n\rangle$ of standard keyword searches that are executed via classical search engines. Finally, transformation τ_3 assembles the answers $Ans(K_1)$, $Ans(K_2), \ldots, Ans(K_n)$, and aggregates and transforms them into a final result $R = \tau_3(\langle Ans(K_1), Ans(K_2), \ldots, Ans(K_n)\rangle)$ along with a ranking, which is returned to the user. The transformations are realized by efficient algorithms that are implemented in form of lightweight clients at the user-site or as Web services. The program implementing τ_1 corresponds to the Annotator (see Fig. 1). The Annotator uses in turn an inference engine that accesses the ontology. The Annotator publishes the results $\tau_1(\mathcal{A})$ via a local or remote Web server to which it is connected. Transformations τ_2 and τ_3 are both parts of the Query Evaluator (see Fig. 1). They communicate via APIs or Web interfaces with a search engine such as *Google Search*.

3 Description Logics

As underlying ontology language, we use the tractable description logic *DL-Lite*$_{\mathcal{A}}$ [7], which adds datatypes to a restricted combination of the tractable description logics *DL-Lite*$_{\mathcal{F}}$ (also called *DL-Lite*) and *DL-Lite*$_{\mathcal{R}}$. All these description logics belong to the *DL-Lite* family [3].

Intuitively, description logics model a domain of interest in terms of concepts and roles, which represent especially classes of individuals and binary relations between classes of individuals, respectively. A 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.

The *DL-Lite* description logics are a class of restricted description logics for which the main reasoning tasks are possible in polynomial time in general and some of them even in LOGSPACE in the data complexity. The *DL-Lite* description logics are fragments of OWL and the most common tractable ontology languages in the Semantic Web context. They are especially directed towards data-intensive applications.

We now recall the syntax and the semantics of *DL-Lite*$_\mathcal{A}$.

Syntax. As for the elementary ingredients of *DL-Lite*$_\mathcal{A}$, let $\mathbf{D}$ be a finite set of *atomic datatypes* d, which are associated with pairwise disjoint sets of *data values* $\mathbf{V}_d$. Let $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, and $\mathbf{I}$ be pairwise disjoint sets of *atomic concepts*, *atomic roles*, *atomic attributes*, and *individuals*, respectively, and let $\mathbf{V}$ denote the union of all $\mathbf{V}_d$ with $d \in \mathbf{D}$.

Roles, concepts, attributes, and datatypes are as follows:

- A *basic role* Q is either an atomic role $P \in \mathbf{R}_A$ or its inverse P^-. A *(general) role* R is either a basic role Q or the negation of a basic role $\neg Q$.
- A *basic concept* B is either an atomic concept $A \in \mathbf{A}$, or an existential restriction on a basic role Q, denoted $\exists Q$, or the domain of an atomic attribute U, denoted $\delta(U)$. A *(general) concept* C is either the *universal concept* $\top_C$, or a basic concept B, or the negation of a basic concept $\neg B$, or an existential restriction on a basic role Q of the form $\exists Q.C$, where C is a concept.
- A *(general) attribute* V is either an atomic attribute U or the negation of an atomic attribute $\neg U$.
- A *basic datatype* E is the range of an atomic attribute U, denoted $\rho(U)$. A *(general) datatype* F is either the *universal datatype* $\top_D$ or an atomic datatype.

An *axiom* is an expression of one of the following forms:

- $B \sqsubseteq C$ (*concept inclusion axiom*), where B is a basic concept, and C is a concept;
- $Q \sqsubseteq R$ (*role inclusion axiom*), where Q is a basic role, and R is a role;
- $U \sqsubseteq V$ (*attribute inclusion axiom*), where U is an atomic attribute, and V is an attribute;
- $E \sqsubseteq F$ (*datatype inclusion axiom*), where E is a basic datatype, and F is a datatype;
- $(\mathsf{funct}\ Q)$ (*role functionality axiom*), where Q is a basic role;
- $(\mathsf{funct}\ U)$ (*attribute functionality axiom*), where U is an atomic attribute;
- $A(a)$ (*concept membership axiom*), where A is an atomic concept and $a \in \mathbf{I}$;
- $P(a, b)$ (*role membership axiom*), where P is an atomic role and $a, b \in \mathbf{I}$;
- $U(a, v)$ (*attribute membership axiom*), where U is an atomic attribute, $a \in \mathbf{I}$, and $v \in \mathbf{V}$.

We next define knowledge bases, which consist of a restricted finite set of inclusion and functionality axioms, called TBox, and a finite set of membership axioms, called ABox. We also define queries to such knowledge bases.

We first define the restriction on inclusion and functionality axioms. A basic role Q (resp., atomic attribute U) is an *identifying property* in a set of axioms $\mathcal{S}$ iff $\mathcal{S}$ contains a functionality axiom $(\mathsf{funct}\ Q)$ (resp., $(\mathsf{funct}\ U)$). Given an inclusion axiom α of the form $X \sqsubseteq Y$ (resp., $X \sqsubseteq \neg Y$), a basic role (resp., atomic attribute) Y appears *positively* (resp., *negatively*) in the right-hand side of α. A basic role (resp., atomic attribute) is *primitive* in $\mathcal{S}$ iff it does not appear positively in the right-hand side of an inclusion axiom in $\mathcal{S}$ and it does not appear in an expression of the form $\exists Q.C$ in $\mathcal{S}$.

We can now define knowledge bases. A *TBox* is a finite set $\mathcal{T}$ of inclusion and functionality axioms such that every identifying property in $\mathcal{T}$ is primitive. Intuitively, identifying properties cannot be specialized in $\mathcal{T}$, i.e., they cannot appear positively in the right-hand side of inclusion axioms in $\mathcal{T}$. An *ABox* $\mathcal{A}$ is a finite set of membership axioms. A *knowledge base* $KB = (\mathcal{T}, \mathcal{A})$ consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$. A *query* ϕ is an open formula of first-order logic with equalities. A *conjunctive query* is of the form $\exists \mathbf{y}\, \phi(\mathbf{x}, \mathbf{y})$, where ϕ is a conjunction of atoms and equalities with free variables among $\mathbf{x}$ and $\mathbf{y}$. A *union of conjunctive queries* is of the form $\bigvee_{i=1}^{n} \exists \mathbf{y}_i\, \phi_i(\mathbf{x}, \mathbf{y}_i)$, where each ϕ_i is a conjunction of atoms and equalities with free variables among $\mathbf{x}$ and $\mathbf{y}_i$.

Example 1 *(Scientific Database).* We use a knowledge base $KB = (\mathcal{T}, \mathcal{A})$ in *DL-Lite*$_\mathcal{A}$ to specify some simple information about scientists and their publications. Consider the following sets of atomic concepts, atomic roles, atomic attributes, individuals, and data values:

$$\begin{aligned}
\mathbf{A} &= \{\textit{Scientist}, \textit{Article}, \textit{ConferencePaper}, \textit{JournalPaper}\},\\
\mathbf{R}_A &= \{\textit{hasAuthor}, \textit{isAuthorOf}, \textit{hasFirstAuthor}\},\\
\mathbf{R}_D &= \{\textit{name}, \textit{title}, \textit{yearOfPublication}\},\\
\mathbf{I} &= \{i_1, i_2\},\\
\mathbf{V} &= \{\text{“}\textit{mary}\text{”}, \text{“}\textit{Semantic Web search}\text{”}, 2008\}.
\end{aligned}$$

The TBox $\mathcal{T}$ contains the subsequent axioms, which informally express that (i) conference and journal papers are articles, (ii) conference papers are not journal papers, (iii) *isAuthorOf* relates scientists and articles, (iv) *isAuthorOf* is the inverse of *hasAuthor*, i.e., (*scientist*, *article*) belongs to *isAuthorOf* iff (*article*, *scientist*) belongs to *hasAuthor*, and (v) *hasFirstAuthor* is a functional binary relationship:

$$\begin{aligned}
&\textit{ConferencePaper} \sqsubseteq \textit{Article},\ \textit{JournalPaper} \sqsubseteq \textit{Article},\\
&\textit{ConferencePaper} \sqsubseteq \neg \textit{JournalPaper},\\
&\exists \textit{isAuthorOf} \sqsubseteq \textit{Scientist},\ \exists \textit{isAuthorOf}^- \sqsubseteq \textit{Article},\\
&\textit{isAuthorOf}^- \sqsubseteq \textit{hasAuthor},\ \textit{hasAuthor}^- \sqsubseteq \textit{isAuthorOf},\\
&(\mathsf{funct}\ \textit{hasFirstAuthor}).
\end{aligned}$$

The ABox $\mathcal{A}$ contains the following axioms, which express that the individual i_1 is a scientist whose name is “*mary*” and who is the author of article i_2, which is entitled “*Semantic Web search*”

and has been published in the year 2008:

$$\textit{Scientist}(i_1),\ \textit{name}(i_1, \text{“}\textit{mary}\text{”}), \textit{isAuthorOf}(i_1, i_2),\\ \textit{Article}(i_2),\ \textit{title}(i_2, \text{“}\textit{Semantic Web search}\text{”}),\\ \textit{yearOfPublication}(i_2, 2008).$$

Querying for all scientists who published an article in 2008 can be expressed by the following conjunctive query:

$$Q(x) = \exists y\,(\textit{Scientist}(x) \wedge \textit{isAuthorOf}(x, y) \wedge \\ \textit{Article}(y) \wedge \textit{yearOfPublication}(y, 2008)).$$

Semantics. The semantics of *DL-Lite*$_\mathcal{A}$ is defined in terms of standard first-order interpretations as usual. An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ consists of (i) a nonempty domain $\Delta^\mathcal{I} = (\Delta^\mathcal{I}_O, \Delta^\mathcal{I}_V)$, which is the disjoint union of the *domain of objects* $\Delta^\mathcal{I}_O$ and the *domain of values* $\Delta^\mathcal{I}_V = \bigcup_{d \in \mathbf{D}} \Delta^\mathcal{I}_d$, where the $\Delta^\mathcal{I}_d$'s are pairwise disjoint domains of values for the datatypes $d \in \mathbf{D}$, and (ii) a mapping $\cdot^\mathcal{I}$ that assigns to each datatype $d \in \mathbf{D}$ its domain of values $\Delta^\mathcal{I}_d$, to each data value $v \in \mathbf{V}_d$ an element of $\Delta^\mathcal{I}_d$ (such that $v \neq w$ implies $v^\mathcal{I} \neq w^\mathcal{I}$), to each atomic concept $A \in \mathbf{A}$ a subset of $\Delta^\mathcal{I}_O$, to each atomic role $P \in \mathbf{R}_A$ a subset of $\Delta^\mathcal{I}_O \times \Delta^\mathcal{I}_O$, to each atomic attribute $P \in \mathbf{R}_D$ a subset of $\Delta^\mathcal{I}_O \times \Delta^\mathcal{I}_V$, to each individual $a \in \mathbf{I}$ an element of $\Delta^\mathcal{I}_O$ (such that $a \neq b$ implies $a^\mathcal{I} \neq b^\mathcal{I}$). Note that different data values (resp., individuals) are associated with different elements of $\Delta^\mathcal{I}_V$ (resp., $\Delta^\mathcal{I}_O$) (*unique name assumption*). The extension of $\cdot^\mathcal{I}$ to all concepts, roles, attributes, and datatypes, and the *satisfaction* of an axiom α in $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, denoted $\mathcal{I} \models \alpha$, are defined as follows:

- $(\top_D)^\mathcal{I} = \Delta^\mathcal{I}_V$ and $(\top_C)^\mathcal{I} = \Delta^\mathcal{I}_O$,
- $(\neg U)^\mathcal{I} = (\Delta^\mathcal{I}_O \times \Delta^\mathcal{I}_V) \setminus U^\mathcal{I}$,
- $(\neg Q)^\mathcal{I} = (\Delta^\mathcal{I}_O \times \Delta^\mathcal{I}_O) \setminus Q^\mathcal{I}$,
- $(\rho(U))^\mathcal{I} = \{v \in \Delta^\mathcal{I}_V \mid \exists o\colon (o, v) \in U^\mathcal{I}\}$,
- $(\delta(U))^\mathcal{I} = \{o \in \Delta^\mathcal{I}_O \mid \exists v\colon (o, v) \in U^\mathcal{I}\}$,
- $(P^-)^\mathcal{I} = \{(o, o') \in \Delta^\mathcal{I}_O \times \Delta^\mathcal{I}_O \mid (o', o) \in P^\mathcal{I}\}$,
- $(\exists P)^\mathcal{I} = \{o \in \Delta^\mathcal{I}_O \mid \exists o'\colon (o, o') \in P^\mathcal{I}\}$,
- $(\exists Q.C)^\mathcal{I} = \{o \in \Delta^\mathcal{I}_O \mid \exists o'\colon (o, o') \in Q^\mathcal{I}, o' \in C^\mathcal{I}\}$,
- $(\neg B)^\mathcal{I} = \Delta^\mathcal{I}_O \setminus B^\mathcal{I}$.

The *satisfaction* of an axiom α in the interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, denoted $\mathcal{I} \models \alpha$, is defined by:

- $\mathcal{I} \models B \sqsubseteq C$ iff $B^\mathcal{I} \subseteq C^\mathcal{I}$,
- $\mathcal{I} \models Q \sqsubseteq R$ iff $Q^\mathcal{I} \subseteq R^\mathcal{I}$,
- $\mathcal{I} \models E \sqsubseteq F$ iff $E^\mathcal{I} \subseteq F^\mathcal{I}$,
- $\mathcal{I} \models U \sqsubseteq V$ iff $U^\mathcal{I} \subseteq V^\mathcal{I}$,

- $\mathcal{I} \models (\mathsf{funct}\ Q)$ iff $(o,q),(o,q') \in Q^{\mathcal{I}}$ implies $q = q'$,
- $\mathcal{I} \models (\mathsf{funct}\ U)$ iff $(o,v),(o,v') \in U^{\mathcal{I}}$ implies $v = v'$,
- $\mathcal{I} \models A(a)$ iff $a^{\mathcal{I}} \in A^{\mathcal{I}}$,
- $\mathcal{I} \models P(a,b)$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P^{\mathcal{I}}$,
- $\mathcal{I} \models U(a,v)$ iff $(a^{\mathcal{I}}, v^{\mathcal{I}}) \in U^{\mathcal{I}}$.

The interpretation $\mathcal{I}$ *satisfies* the axiom α, or $\mathcal{I}$ is a *model* of α, iff $\mathcal{I} \models \alpha$. The interpretation $\mathcal{I}$ *satisfies* a knowledge base $KB = (\mathcal{T}, \mathcal{A})$, or $\mathcal{I}$ is a *model* of KB, denoted $\mathcal{I} \models KB$, iff $\mathcal{I} \models \alpha$ for all $\alpha \in \mathcal{T} \cup \mathcal{A}$. We say KB is *satisfiable* (resp., *unsatisfiable*) iff KB has a (resp., no) model. An axiom α is a *logical consequence* of KB, denoted $KB \models \alpha$, iff every model of KB satisfies α. An *answer* for a query ϕ to KB is a ground substitution θ for all free variables in ϕ such that $\phi\theta$ is a logical consequence of KB.

As shown in [7], in particular, deciding the satisfiability of knowledge bases in *DL-Lite*$_{\mathcal{A}}$ and deciding logical consequences of membership axioms from knowledge bases in *DL-Lite*$_{\mathcal{A}}$ can both be done in LOGSPACE in the size of the ABox in the data complexity.

Example 2 *(Scientific Database cont'd).*The knowledge base $KB = (\mathcal{T}, \mathcal{A})$ of Example 1 is satisfiable, and *JournalPaper* $\sqsubseteq \neg$*ConferencePaper* and *hasAuthor*(i_2, i_1) are logical consequences of KB. Furthermore, the ground substitution $\theta = \{x/i_1\}$ is an answer for the conjunctive quer $Q(x)$ of Example 1. Informally, *mary* published an article in 2008.

4 Semantic Web Search

In this section, we first introduce the notion of a Semantic Web knowledge base and the syntax and the semantics of Semantic Web search queries to such knowledge bases. We then generalize the PageRank technique to our approach.

4.1 Semantic Web Knowledge Bases

Intuitively, a Semantic Web knowledge base consists of a background TBox and a collection of ABoxes, one for every concrete Web page and for every object on a Web page. For example, the homepage of a scientist may be such a concrete Web page and be associated with an ABox, while the publications on the homepage may be such objects, which are also associated with one ABox each.

As in Section 3, we assume pairwise disjoint sets $\mathbf{D}$, $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, $\mathbf{I}$, and $\mathbf{V}$ of atomic datatypes, atomic concepts, atomic roles, atomic attributes, individuals, and data values, respectively. We assume that these sets are all finite. Let $\mathbf{I}$ be the disjoint union of two sets $\mathbf{P}$ and $\mathbf{O}$ of *Web pages* and *Web objects*, respectively. Informally, every $p \in \mathbf{P}$ is an identifier for a concrete Web page, while every $o \in \mathbf{O}$ is an identifier for a concrete object on a concrete Web page. We assume the atomic roles *links_to* between Web pages and *contains* between Web pages and Web objects. The former represents the link structure between concrete Web pages, while the latter encodes the occurrences of concrete Web objects on concrete Web pages.

Definition 3 A *semantic annotation* $\mathcal{A}_a$ for a Web page or object $a \in \mathbf{P} \cup \mathbf{O}$ is a finite set of concept membership axioms $A(a)$, role membership axioms $P(a, b)$, and attribute membership axioms $U(a, v)$, where $A \in \mathbf{A}$, $P \in \mathbf{R}_A$, $U \in \mathbf{R}_D$, $b \in \mathbf{I}$, and $v \in \mathbf{V}$. A *Semantic Web knowledge base* $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ consists of a TBox $\mathcal{T}$ and one semantic annotation $\mathcal{A}_a$ for every Web page and object $a \in \mathbf{P} \cup \mathbf{O}$.

Informally, a Semantic Web knowledge base consists of some background terminological knowledge and some assertional knowledge for every concrete Web page and for every concrete object on a Web page. The background terminological knowledge may be an ontology from some global Semantic Web repository or an ontology defined locally by the user site. In contrast to the background terminological knowledge, the assertional knowledge will be directly stored on the Web (on annotation pages like the described standard Web pages) and is thus accessible via Web search engines.

Example 4 *(Scientific Database cont'd). Let* $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, *and* $\mathbf{V}$ *be as in Example 1 except that we add contains and "Semantic Web search engines" to* $\mathbf{R}_A$ *and* $\mathbf{V}$*, respectively. Let* $\mathbf{I} = \mathbf{P} \cup \mathbf{O}$ *be the set of individuals, where* $\mathbf{P} = \{i_1\}$ *is the set of Web pages, and* $\mathbf{O} = \{i_2, i_3, i_4\}$ *is the set of Web objects on the Web page* i_1*. Let the TBox* $\mathcal{T}$ *be as in Example 1. Then, a Semantic Web knowledge base is given by* $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$*, where the semantic annotations of the individuals in* $\mathbf{P} \cup \mathbf{O}$ *are the following:*

$$
\begin{aligned}
\mathcal{A}_{i_1} &= \{\textit{contains}(i_1, i_2), \textit{contains}(i_1, i_3), \textit{contains}(i_1, i_4)\},\\
\mathcal{A}_{i_2} &= \{\textit{PhDStudent}(i_2),\ \textit{name}(i_2, \text{“}\textit{mary}\text{”}), \textit{isAuthorOf}(i_2, i_3),\\
&\qquad \textit{isAuthorOf}(i_2, i_4)\},\\
\mathcal{A}_{i_3} &= \{\textit{ConferencePaper}(i_3), \textit{title}(i_3, \text{“}\textit{Semantic Web search}\text{”})\},\\
\mathcal{A}_{i_4} &= \{\textit{JournalPaper}(i_4),\ \textit{hasAuthor}(i_4, i_2),\\
&\qquad \textit{title}(i_4, \text{“}\textit{Semantic Web search engines}\text{”}),\\
&\qquad \textit{yearOfPublication}(i_4, 2008), \textit{keyword}(i_4, \textit{“RDF”})\}.
\end{aligned}
$$

4.2 Semantic Web Search Queries

We use unions of conjunctive queries with negated conjunctive subqueries as Semantic Web search queries to Semantic Web knowledge bases. We now first define the syntax of Semantic Web search queries and then the semantics of positive and general such queries.

Syntax. Let $\mathbf{X}$ be a finite set of variables. A *term* is either a Web page $p \in \mathbf{P}$, a Web object $o \in \mathbf{O}$, a data value $v \in \mathbf{V}$, or a variable $x \in \mathbf{X}$. An *atomic formula* (or *atom*) α is of one of the following forms: (i) $d(t)$, where d is an atomic datatype, and t is a term; (ii) $A(t)$, where A is an atomic concept, and t is a term; (iii) $P(t, t')$, where P is an atomic role, and t, t' are terms; and (iv) $U(t, t')$, where U is an atomic attribute, and t, t' are terms. An *equality* has the form $=(t, t')$, where t and t' are terms. A *conjunctive formula* $\exists \mathbf{y}\, \phi(\mathbf{x}, \mathbf{y})$ is an existentially quantified conjunction of atoms α and equalities $=(t, t')$, which have free variables among $\mathbf{x}$ and $\mathbf{y}$.

Definition 5 A *Semantic Web search query* $Q(\mathbf{x})$ is an expression of the form $\bigvee_{i=1}^{n} \exists \mathbf{y}_i \, \phi_i(\mathbf{x}, \mathbf{y}_i)$, where each ϕ_i with $i \in \{1, \ldots, n\}$ is a conjunction of atoms α (also called *positive atoms*), negated conjunctive formulas $not\, \psi$, and equalities $=(t, t')$, which have free variables among $\mathbf{x}$ and $\mathbf{y}_i$.

Intuitively, Semantic Web search queries are unions of conjunctive queries, which may contain negated conjunctive queries in addition to atoms and equalities as conjuncts.

Example 6 *(Scientific Database cont'd).*Two Semantic Web search queries are given as follows:

$$\begin{aligned} Q_1(x) = {} & (\textit{Scientist}(x) \wedge not\ \textit{doctoralDegree}(x, \text{“}\textit{oxford university}\text{”}) \wedge \textit{worksFor}(x, \text{“}\textit{oxford university}\text{”})) \vee \\ & (\textit{Scientist}(x) \wedge \textit{doctoralDegree}(x, \text{“}\textit{oxford university}\text{”}) \wedge not\ \textit{worksFor}(x, \text{“}\textit{oxford university}\text{”})); \\ Q_2(x) = {} & \exists y\, (\textit{Scientist}(x) \wedge \textit{worksFor}(x, \text{“}\textit{oxford university}\text{”}) \\ & \wedge \textit{isAuthorOf}(x, y) \wedge not\ \textit{ConferencePaper}(y) \wedge \\ & not\ \exists z\, \textit{yearOfPublication}(y, z)). \end{aligned}$$

Informally, $Q_1(x)$ asks for scientists who are either working for *oxford university* and did not receive their Ph.D. from that university, or who received their Ph.D. from *oxford university* but do not work for it. Whereas query $Q_2(x)$ asks for scientists of *oxford university* who are authors of at least one unpublished non-conference paper. Note that when searching for scientists, the system automatically searches for all subconcepts (known according to the background ontology), such as e.g. Ph.D. students or computer scientists.

Semantics of Positive Search Queries. We now define the semantics of positive Semantic Web search queries, which are free of negations, in terms of ground substitutions via the notion of logical consequence.

A Semantic Web search query $Q(\mathbf{x})$ is *positive* iff it does not contain negated conjunctive subqueries. A *(variable) substitution* θ maps variables from $\mathbf{X}$ to terms. A substitution θ is *ground* iff it maps to Web pages $p \in \mathbf{P}$, Web objects $o \in \mathbf{O}$, and data values $v \in \mathbf{V}$. A closed first-order formula ϕ is a *logical consequence* of a Semantic Web knowledge base $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$, denoted $KB \models \phi$, iff every first-order model $\mathcal{I}$ of $\mathcal{T} \cup \bigcup_{a \in \mathbf{P} \cup \mathbf{O}} \mathcal{A}_a$ also satisfies ϕ.

Definition 7 Given a Semantic Web knowledge base KB and a positive Semantic Web search query $Q(\mathbf{x})$, an *answer* for $Q(\mathbf{x})$ to KB is a ground substitution θ for the variables $\mathbf{x}$ such that $KB \models Q(\mathbf{x}\theta)$.

Example 8 *(Scientific Database cont'd).*Consider the Semantic Web knowledge base KB of Example 4 and the following positive Semantic Web search query, asking for all scientists who author at least one published journal paper:

$$\begin{aligned} Q(x) = {} & \exists y\, (\textit{Scientist}(x) \wedge \textit{isAuthorOf}(x, y) \wedge \\ & \textit{JournalPaper}(y) \wedge \exists z\, \textit{yearOfPublication}(y, z)). \end{aligned}$$

An answer for $Q(x)$ to KB is given by $\theta = \{x/i_2\}$. Recall that i_2 represents the scientist *mary*.

Semantics of General Search Queries. We next define the semantics of general Semantic Web search queries by reduction to the semantics of positive ones, interpreting negated conjunctive subqueries $not\,\psi$ as the lack of evidence about the truth of ψ. That is, to interpret negations, we define a closed-world semantics on top of the open-world semantics of description logics.

Definition 9 Given a Semantic Web knowledge base KB and a (general) Semantic Web search query

$$Q(\mathbf{x}) = \bigvee_{i=1}^{n} \exists \mathbf{y}_i\, \phi_{i,1}(\mathbf{x}, \mathbf{y}_i) \wedge \cdots \wedge \phi_{i,l}(\mathbf{x}, \mathbf{y}_i) \wedge \\ not\, \phi_{i,l+1}(\mathbf{x}, \mathbf{y}_i) \wedge \cdots \wedge not\, \phi_{i,m}(\mathbf{x}, \mathbf{y}_i)\,,$$

an *answer* for $Q(\mathbf{x})$ to KB is a ground substitution θ for the variables $\mathbf{x}$ such that $KB \models Q^+(\mathbf{x}\theta)$ and $KB \not\models Q^-(\mathbf{x}\theta)$, where $Q^+(\mathbf{x})$ and $Q^-(\mathbf{x})$ are defined as follows:

$$Q^+(\mathbf{x}) = \bigvee_{i=1}^{n} \exists \mathbf{y}_i\, \phi_{i,1}(\mathbf{x}, \mathbf{y}_i) \wedge \cdots \wedge \phi_{i,l}(\mathbf{x}, \mathbf{y}_i) \text{ and}$$
$$Q^-(\mathbf{x}) = \bigvee_{i=1}^{n} \exists \mathbf{y}_i\, \phi_{i,1}(\mathbf{x}, \mathbf{y}_i) \wedge \cdots \wedge \phi_{i,l}(\mathbf{x}, \mathbf{y}_i) \wedge \\ (\phi_{i,l+1}(\mathbf{x}, \mathbf{y}_i) \vee \cdots \vee \phi_{i,m}(\mathbf{x}, \mathbf{y}_i))\,.$$

Roughly, a ground substitution θ is an answer for $Q(\mathbf{x})$ to KB iff (i) θ is an answer for $Q^+(\mathbf{x})$ to KB, and (ii) θ is not an answer for $Q^-(\mathbf{x})$ to KB, where $Q^+(\mathbf{x})$ is the positive part of $Q(\mathbf{x})$, while $Q^-(\mathbf{x})$ is the positive part of $Q(\mathbf{x})$ combined with the complement of the negative one. Observe that both $Q^+(\mathbf{x})$ and $Q^-(\mathbf{x})$ are positive Semantic Web search queries.

Example 10 *(Scientific Database cont'd).*Consider the Semantic Web knowledge base $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ of Example 4 and the following general Semantic Web search query, asking for *mary*'s unpublished non-journal papers:

$$Q(x) = \exists y\, (\textit{Article}(x) \wedge \textit{hasAuthor}(x, y) \wedge \\ \textit{name}(y, \text{“}\textit{mary}\text{”}) \wedge not\, \textit{JournalPaper}(x) \wedge \\ not\, \exists z\, \textit{yearOfPublication}(x, z)).$$

An answer for $Q(x)$ to KB is given by $\theta = \{\mathbf{x}/i_3\}$. Recall that i_3 represents an unpublished conference paper entitled "*Semantic Web search*". Observe that the membership axioms $\textit{Article}(i_3)$ and $\textit{hasAuthor}(i_2, i_3)$ do not appear in the semantic annotations $\mathcal{A}_a$ with $a \in \mathbf{P} \cup \mathbf{O}$, but they can be inferred from them using the background ontology $\mathcal{T}$.

4.3 Ranking Answers

As for the ranking of all answers for a Semantic Web search query Q to a Semantic Web knowledge base KB (i.e., ground substitutions for all free variables in Q, which correspond to tuples of Web pages, Web objects, and data values), we use a generalization of the PageRank technique: rather than considering only Web pages and the link structure between Web pages (expressed through the role *links_to* here), we also consider Web objects, which may occur on Web pages (expressed through the role *contains*), and which may also be related to other Web objects via other roles.

We first recall the standard PageRank technique for ranking Web pages, which is based on the analysis of the link structure between the Web pages. For example, Web pages generally contain links to other Web pages. Similarly, pieces of literature generally cite other pieces of literature.

The PageRank technique, which stands behind the Web search engine Google [2], is one of the most prominent ways of ranking objects based on the link structure between the objects. The PageRank of a Web page u is defined as

$$R(u) = d \cdot \sum_{v \in B_u} R(v) \,/\, N_v + (1 - d) \cdot E(u)\,,$$

where (i) B_u is the set of pages that point to u, (ii) N_v is the number of links from v, (iii) d is a damping factor, and (iv) E associates with every Web page a source of rank. Informally, the more Web pages with high rank point to a Web page, the higher is the rank of this Web page. The PageRank ranking thus extracts the importance of a Web page from the link structure between the Web pages.

In the context of Semantic Web knowledge bases, we define the PageRank of a Web page or an object a as follows:

$$R(a) = d \cdot \sum_{b \in B_a} R(b) \,/\, N_b + (1 - d) \cdot E(a)\,,$$

where (i) B_a is the set of all Web pages and Web objects that relate to a, (ii) N_b is the number of Web pages and Web objects that relate from b, (iii) d is a damping factor, and (iv) E associates with every Web page and every Web object a source of rank. So, rather than depending only on the link structure between Web pages, the new ranking depends also on the relationships between Web pages and Web objects, and on the relationships between Web objects, where the user fixes the roles to be considered.

This ranking on Web pages and Web objects is then extended to a lexicographic order on answers for Semantic Web search queries to Semantic Web knowledge bases.

5 Processing Search Queries

The main idea behind processing Semantic Web search queries Q to a knowledge base KB is to reduce them to standard Web search queries. To this end, the TBox $\mathcal{T}$ of KB must be considered during standard Web search. There are two main ways to do so. The first is to compile $\mathcal{T}$ into Q, yielding a new standard Web search query Q' on the ABox $\mathcal{A}$ of KB. The second, which we adopt here, is to compile $\mathcal{T}$ via offline ontology reasoning into the ABox $\mathcal{A}$ of KB, yielding a completed ABox $\mathcal{A}'$, which is then searched by a collection of standard Web search queries depending on Q.

So, processing Semantic Web search queries Q is divided into (1) an offline ontology reasoning step, where roughly all semantic annotations of Web pages / objects are completed by logically entailed membership axioms, and (2) an online reduction to standard Web search, where Q is transformed into a collection of standard Web search queries of which the answers are used to construct the answer for Q. In this section, we first describe the offline ontology reasoning step and then the online reduction to standard Web search.

5.1 Offline Ontology Reasoning

The offline ontology reasoning step transforms the implicit terminological knowledge in the TBox of a Semantic Web knowledge base into explicit membership axioms in the ABox, i.e., in the semantic annotations of Web pages / objects, so that it can be searched by standard Web search engines. In the case of quantifier-free search queries, and when the TBox is equivalent to a Datalog program, it is sufficient to add all logically entailed membership axioms constructed from Web pages, Web objects, and data values. In the case of general search queries, one also has to add logically entailed membership axioms with new constants.

Quantifier-Free Search Queries. The compilation of TBox knowledge into ABox knowledge is formalized as follows. Given a satisfiable Semantic Web knowledge base $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$, the *simple completion* of KB is the Semantic Web knowledge base $KB' = (\emptyset, (\mathcal{A}_a{}')_{a \in \mathbf{P} \cup \mathbf{O}})$ such that every $\mathcal{A}_a{}'$ is the set of all concept membership axioms $A(a)$, role membership axioms $P(a, b)$, and attribute membership axioms $U(a, v)$, where $A \in \mathbf{A}$, $P \in \mathbf{R}_A$, $U \in \mathbf{R}_D$, $b \in \mathbf{I}$, and $v \in \mathbf{V}$, that logically follow from $\mathcal{T} \cup \bigcup_{a \in \mathbf{P} \cup \mathbf{O}} \mathcal{A}_a$.

Example 11 Consider again the TBox $\mathcal{T}$ of Example 1 and the semantic annotations $(\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}}$ of Example 4. The simple completion contains in particular the new axioms *Article*(i_3), *hasAuthor*(i_3, i_2), and *Article*(i_4). The first two axioms are added to A_{i_3} and the last one to A_{i_4}.

The following theorem shows that positive quantifier-free search queries to a Semantic Web knowledge base KB can be evaluated on the simple completion of KB (which contains only compiled but no explicit TBox knowledge anymore).

Theorem 12 *Let KB be a satisfiable Semantic Web knowledge base, let $Q(\mathbf{x})$ be a positive Semantic Web search query without existential quantifiers, and let θ be a ground substitution for $\mathbf{x}$. Then, θ is an answer for $Q(\mathbf{x})$ to KB iff θ is an answer for $Q(\mathbf{x})$ to the simple completion of KB.*

As an immediate consequence, we obtain that general quantifier-free search queries to a Semantic Web knowledge base KB can also be evaluated on the simple completion of KB, which is expressed by the next theorem.

Corollary 13 *Let KB be a satisfiable Semantic Web knowledge base, let $Q(\mathbf{x})$ be a (general) Semantic Web search query without existential quantifiers, and let θ be a ground substitution for $\mathbf{x}$. Then, θ is an answer for $Q(\mathbf{x})$ to KB iff θ is an answer for $Q^+(\mathbf{x})$ but not an answer for $Q^-(\mathbf{x})$ to the simple completion of KB.*

Similar results hold when the TBox of KB is equivalent to a Datalog program, and the query $Q(\mathbf{x})$ is fully general.

General Search Queries. The *k-completion* of a satisfiable Semantic Web knowledge base $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ is the Semantic Web knowledge base $KB' = (\emptyset, (\mathcal{A}_a{}')_{a \in \mathbf{P} \cup \mathbf{O}})$ such that every $\mathcal{A}_a{}'$ is the set of all concept membership axioms $A(a)$, role membership axioms $P(a, b)$, and attribute membership axioms $U(a, v)$ in $finChase_k^A(\mathcal{T}, \bigcup_{a \in \mathbf{P} \cup \mathbf{O}} \mathcal{A}_a)$, which is the *finite chase of degree* k of $(\mathcal{T}, \bigcup_{a \in \mathbf{P} \cup \mathbf{O}} \mathcal{A}_a)$ [8].

The following theorem shows that (not necessarily quantifier-free) positive search queries to a Semantic Web knowledge base KB can be evaluated on the k-completion of KB.

Theorem 14 *Let KB be a satisfiable Semantic Web knowledge base, let $Q(\mathbf{x})$ be a positive Semantic Web search query of size k, and let θ be a ground substitution for $\mathbf{x}$. Then, θ is an answer for $Q(\mathbf{x})$ to KB iff θ is an answer for $Q(\mathbf{x})$ to the k-completion of KB.*

Consequently, general search queries to a Semantic Web knowledge base KB can also be evaluated on the k-completion of KB, which is expressed by the next theorem.

Corollary 15 *Let KB be a satisfiable Semantic Web knowledge base, let $Q(\mathbf{x})$ be a (general) Semantic Web search query of size k, and let θ be a ground substitution for $\mathbf{x}$. Then, θ is an answer for $Q(\mathbf{x})$ to KB iff θ is an answer for $Q^+(\mathbf{x})$ but not for $Q^-(\mathbf{x})$ to the k-completion of KB.*

5.2 Online Reduction to Web Search

Each semantic annotation of a Web page / object $a \in \mathbf{P} \cup \mathbf{O}$ is stored on an HTML page in the Web. We now define simple and safe search queries, and describe how they are answered by reduction to standard Web search queries. In detail, simple search queries are directly reduced to variable-free Boolean Web search queries, while safe search queries are reduced to collections of atomic Web search queries.

Simple Search Queries. Search queries that contain only one variable and no equalities are called simple search queries. Such search queries can immediately be reduced to variable-free Boolean keyword-based Web search queries.

Definition 16 A Semantic Web search query is *simple* iff it has the form $Q(x) = \bigvee_{i=1}^{n} \phi_i(x)$, where x is a single variable from $\mathbf{X}$, and each ϕ_i with $i \in \{1, \ldots, n\}$ is a conjunction of atoms and negated conjunctive formulas without quantifiers, which may contain the singleton x as only free variable.

Example 17 *(Scientific Database cont'd).*Query $Q_1(x)$ of Example 6 is a simple Semantic Web search query. It can immediately be reduced to the following variable-free Boolean keyword-based Web search query:

(*Scientist* $\wedge$ not *doctoralDegree*(“*oxford university*”) $\wedge$
worksFor(“*oxford university*”)) $\vee$
(*Scientist* $\wedge$ *doctoralDegree*(“*oxford university*”) $\wedge$
not *worksFor*(“*oxford university*”)).

Safe Search Queries. Search queries where all free variables in negated conjunctive formulas and in equalities also occur in positive atoms are safe queries. They are reduced to collections of atomic Web search queries, one collection for the positive part, and one for every negative subquery. Due to the safeness, we retain all results of the positive part that are not matching with any result of a negative subquery.

Definition 18 A Semantic Web search query $Q(\mathbf{x}) = \bigvee_{i=1}^{n} \exists \mathbf{y}_i \, \phi_i(\mathbf{x}, \mathbf{y}_i)$ is *safe* iff, for every $i \in \{1, \ldots, n\}$, each variable that occurs in an equality in ϕ_i and freely in a negated conjunctive formula also occurs in a positive atom in ϕ_i.

Example 19 *(Scientific Database cont'd).*The following two Semantic Web search queries ask for all students who do not attend at least one existing course (resp., event):

$$Q_1(x) = \exists y \, (\textit{Student}(x) \wedge \mathit{not}\ \textit{attends}(x, y) \wedge \textit{Course}(y)),$$
$$Q_2(x) = \exists y \, (\textit{Student}(x) \wedge \mathit{not}\ \textit{attends}(x, y)).$$

Observe that query $Q_1(x)$ is safe, whereas $Q_2(x)$ is not, since the variable y does not occur in any positive atom of $Q_2(x)$.

6 Computation

In this section, we give some further details on the computational aspects behind our approach.

6.1 Offline Ontology Reasoning

We first describe the offline inference technique and how the resulting completed semantic annotations are encoded as HTML pages to make them searchable by means of Web search engines such as Google. We also discuss the aspect of object identifiers in Web and annotation pages.

In the offline inference technique, we have to check whether the Semantic Web knowledge base is satisfiable, and we have to compute the completion of all semantic annotations, i.e., to augment the semantic annotations with all concept, role, and attribute membership axioms that can be deduced from the semantic annotations and the background ontology as described in Section 5.1. We suggest to use only the simple completion of all semantic annotations, which can be computed efficiently and which is complete for a large class of Semantic Web knowledge bases and search queries.

Checking whether a Semantic Web knowledge base is satisfiable and computing its simple completion can both be done efficiently in the data complexity for knowledge bases in *DL-Lite*$_{\mathcal{A}}$ (see Section 3), and one can use existing systems such as QuOnto [3]. An alternative way of performing offline inference (for the case that the TBox is equivalent to a Datalog program) is based on a reduction to deductive databases. We first transform all the axioms in the TBox and in the semantic annotations into a set of Datalog rules R and a set of facts F, respectively. We then run a deductive database system on $R \cup F$, thus checking satisfiability and obtaining the set of all deducible concept, attribute, and role membership axioms A in the simple completion.

Example 20 It is not difficult to see that the TBox T of Example 1 is equivalent to the following Datalog rules R:

$\{$*Article(x)* $\leftarrow$ *ConferencePaper(x)*,
Article(x) $\leftarrow$ *JournalPaper(x)*,
$\bot \leftarrow$ *ConferencePaper(x)*, *JournalPaper(x)*,
Scientist(x) $\leftarrow$ *isAuthorOf(x,_)*,
Article(x) $\leftarrow$ *isAuthorOf(_,x)*,
hasAuthor(x,y) $\leftarrow$ *isAuthorOf(y,x)*,
isAuthorOf(x,y) $\leftarrow$ *hasAuthor(y,x)*,
$\bot \leftarrow$ *hasFirstAuthor(z,x)*, *hasFirstAuthor(z,y)*, $x \neq y\}$.

The semantic annotations $(\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}}$ of Example 4 yield a set of facts F. We then use a deductive database system to check whether $R \cup F$ is satisfiable and to deduce new axioms, such as *Article*(i_3), *hasAuthor*(i_3, i_2), and *Article*(i_4), which are added to A_{i_3}, A_{i_3}, and A_{i_4} of Example 4, respectively.

Once the completed semantic annotations are computed, we encode them as HTML pages, so that they are searchable via standard keyword search. We build one HTML page for the semantic annotation $\mathcal{A}_a$ of each individual $a \in \mathbf{P} \cup \mathbf{O}$. That is, for each individual a, we build a page p containing all the atomic concepts whose argument is a and all the atomic roles/attributes where the first argument is a. Roughly, this rewriting of the axioms in $\mathcal{A}_a$ consists in removing all the brackets and the first argument of binary roles.

After rewriting the annotations, also search queries are rewritten to deal with the new syntax of the annotations. Specifically, we remove all the variables and the brackets. For example, the query $Q(x) =$ *Article*$(x) \wedge$ *yearOfPublication*$(x, 2008) \wedge$ *keyword*$(x, \text{“}RDF\text{”})$ is translated into *Article* AND “*yearOfPublication* 2008” AND *“keyword* RDF”. In this form, the query can be evaluated by Web search engines, since it is a simple query consisting of a conjunction of a keyword and a phrase (see Section 6.2).

We rely on the assumption that each Web page / object $a \in \mathbf{P} \cup \mathbf{O}$ is associated with an identifier, which uniquely characterizes the individual. Practically, this identifier may simply be the HTML address of the Web page's / object's annotation page. On the HTML page of each individual, the identifier is located beside the atomic concept below the row specifying the URIs. For example, considering the HTML pages of Fig. 3, the individual described by p_4 is i_4, and the one described by p_2 is i_2. We employ these identifiers to evaluate complex queries involving more than one atomic concept, thus involving several annotations. Consider the query $Q(x)$ of Section 2 and the standard queries Q_1 and Q_2 obtained from it. To evaluate $Q(x)$, we submit Q_1 and Q_2 to a Web search engine, and we collect the results r_1 and r_2 of the two queries. We return the identifier of a page p belonging to r_1 if there exists a page in r_2 such that the identifier of the described individual matches with the one of the article identifiers contained in p beside *isAuthorOf*. Considering the HTML pages of Fig. 3, we compare the identifier i_4 of page p_4 with the identifiers of the articles contained beside *isAuthorOf* on page p_2. Since i_4 matches with one of the article identifiers in p_2, we return the identifier of p_2.

6.2 Online Reduction to Web Search

We next describe our algorithm for the online reduction of Semantic Web search queries to standard Web search queries. The algorithm reduces fully general but safe Semantic Web search queries to a collection of standard Web search queries. Recall that the former are unions of conjunctive queries, which may contain negated conjunctive queries in addition to atoms and equalities as conjuncts. Here, we illustrate the algorithm only for a single conjunction. To process a Semantic Web search query with two and more conjunctions, it suffices to apply this technique for each conjunction and then unify the results. Without loss of generality, we also ignore equalities here.

Let $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ be a Semantic Web knowledge base. We recall that the semantic annotation for every Web page / object $a \in \mathbf{P} \cup \mathbf{O}$ is stored on the Web as an HTML annotation page. The annotation page for a contains a collection of URIs, namely, the HTML address of a, if a is a Web page, and all Web pages mentioning a, if a is a Web object. In addition, it contains the identifier a, all atomic concepts "A" such that $KB \models A(a)$, all atomic-attribute-value pairs "$U\ v$" such that $KB \models U(a, v)$, and all atomic-role-identifier pairs "$P\ b$" such that $KB \models P(a, b)$. Thus, the annotation pages may be linked via roles.

The algorithm for the online reduction of Semantic Web search to standard Web search is in Fig. 4. The main ideas behind the algorithm are informally described as follows.

Consider first a positive Semantic Web search query Q. To answer such a query, we first transform the query Q into a collection of standard Web search queries $Q^{x_1}, \ldots, Q^{x_n}$, one for each variable x_i (for an individual) in Q (line 1). Note that some of these queries may result from simple subqueries of Q. We then send each Web search query Q^{x_i} to a Web search engine, which is in fact searching for all atomic concepts A, attributes U (eventually with values), and roles P (eventually with identifiers) such that $A(x_i)$, $U(x_i, t)$, and $P(x_i, t')$ occur in Q, respectively, collecting the identifiers of all matching annotation pages (line 2). These identifiers are then used to fill all the matching identifiers, identifier-value pairs, and identifier-identifier pairs from the annotation pages for these atomic concepts $A(x_i)$, attributes $U(x_i, t)$, and roles $P(x_i, t')$, respectively, into collections of unary and/or binary relations $A[x_i]$, $U[x_i, t]$ (or $U[x_i]$ when t is a value), and $P[x_i, t']$ (or $P[x_i]$ when t' is an identifier or $t' = x_i$), respectively (line 3). These relations are then joined via common variables in Q, and finally projected to all free variables in Q (line 4).

Consider next a safe general Semantic Web search query. To answer such a query, we first split Q into its positive part Q_0 and all the negative subqueries $Q_1, \ldots, Q_n$ (line 5). We then process the positive part of Q, and every negative subquery of Q in the same way as positive Semantic Web search queries above (lines 6 and 7), but after the join for the positive part of Q, we remove all tuples matching with the result of one negative subquery, and then we project the result to all free variables in Q (line 8). Note that the algorithm can be easily implemented and optimized using standard relational database operations and techniques.

7 Implementation and Experiments

We have implemented a prototype for a semantic desktop search engine. Note that the implementation requires a complete indexing of all annotation pages, which is easily possible via existing

POSITIVESEMANTICWEBSEARCHQUERY(Q)

1 $(Q^{x_1},\ldots,Q^{x_n}) \leftarrow$ POSITIVEPARSE(Q);
2 FOR $i = 1$ TO n DO $I_i \leftarrow$ WEBSEARCHQUERY(Q^{x_i});
3 FOR $i = 1$ TO n DO $(R_j)_{j\in J_i} \leftarrow$ FILLRELATIONS(I_i);
4 RETURN $\pi_{\text{FREE}(Q)}(\bowtie_{i=1}^{n}\bowtie_{j\in J_i} R_j)$.

SEMANTICWEBSEARCHQUERY(Q)

5 $(Q_0,Q_1,\ldots,Q_m) \leftarrow$ PARSE(Q);
6 FOR $i = 0$ TO m DO
7 $R_i \leftarrow$ POSITIVESEMANTICWEBSEARCHQUERY(Q_i);
8 RETURN $\pi_{\text{FREE}(Q)}(\{t{\in}R_0 \mid \forall 1{\leq}i{\leq}m\, \forall t_i{\in}R_i\colon t[R_i]{\neq}t_i\})$.

Figure 4: Online reduction to Web search.

desktop search engines (but less easily via existing Web search engines, which perform only an indexing of a random selection of annotation pages).

The implementation is based on the above offline inference technique and a (simplified) desktop version of the above online Semantic Web search (by reduction to standard Web search). The former uses the deductive database system DLV [6], while the latter is written in Java (nearly 2 000 lines of code) and uses Microsoft Windows Desktop Search 3.0 (WDS) as external desktop search engine; in detail, it uses the search index created by WDS, which is queried by a cmdlet script in Microsoft Powershell 1.0.

First experiments with our implemented semantic desktop search engine show that the online desktop search procedure scales quite well to very large collections of standard pages, annotation pages, and background ontologies. Our experimental results are summarized in Table 1, which shows in bold the net time (in ms) used by our system (without the WDS calls) for processing ten different search queries ($Q_1, Q_2, \ldots, Q_{10}$) on four different randomly generated knowledge bases (in the context of the running Scientific Database), consisting of up to 5 000 annotations with up to 590 027 facts. Notice that this net system time (for the decomposition of the query and the composition of the query results) is very small (at most 6 seconds in the worst case). Table 1 also shows the time used for calling WDS for processing all subqueries, as well as the different numbers of returned pages and objects. The biggest part of the total running time is used for these WDS calls, which is due to the fact that our current implementation is based on a file interface to WDS. We expect that this time can be dramatically reduced by using an API, and also by using a more efficient Web search engine (such as Google) rather than WDS.

The ten search queries $Q_1, Q_2, \ldots, Q_{10}$ are more concretely given as follows (where the a_i's, c_i's, o_i's, and u_i's are either individuals or values); they ask for all the following individuals (so also yielding the Web pages containing them):

(1) professors giving the course c_{12}:

$$Q_1(x) = \textit{Professor}(x) \wedge \textit{teacherOf}(x, c_{12})\,.$$

Table 1: WDS and system time (in ms) used for processing the search queries $Q_1, Q_2, \ldots, Q_{10}$ on four different random knowledge bases.

	625	1250	2500	5000	no. annotations
	69283	142565	292559	590270	no. facts
Q_1	5370	6324	7222	9665	WDS time
	338	**689**	**1251**	**2458**	**system time**
	95	199	373	613	no. URIs
Q_2	4865	5171	5224	5549	WDS time
	20	**59**	**165**	**344**	**system time**
	5	17	55	116	no. URIs
Q_3	10365	11419	13545	17149	WDS time
	402	**899**	**1856**	**3709**	**system time**
	105	214	406	646	no. URIs
Q_4	9498	10002	10888	12732	WDS time
	218	**479**	**944**	**1860**	**system time**
	73	162	304	529	no. URIs
Q_5	10914	12176	14748	19847	WDS time
	490	**904**	**1632**	**3154**	**system time**
	95	228	420	679	no. URIs
Q_6	10331	11096	12446	15532	WDS time
	138	**191**	**359**	**732**	**system time**
	23	48	95	204	no. URIs
Q_7	24735	26580	30352	37857	WDS time
	748	**1523**	**2996**	**5990**	**system time**
	112	235	431	687	no. URIs
Q_8	4777	4882	4878	4920	WDS time
	9	**30**	**45**	**59**	**system time**
	1	8	14	20	no. URIs
Q_9	16892	19524	24798	34218	WDS time
	593	**1179**	**2297**	**4753**	**system time**
	53	225	431	687	no. URIs
Q_{10}	31455	33966	39040	48686	WDS time
	690	**1399**	**2833**	**5717**	**system time**
	6	47	89	171	no. URIs

(2) professors giving the course c_{12} but not the course c_{20}:

$$Q_2(x) = \textit{Professor}(x) \wedge \textit{teacherOf}(x, c_{12}) \wedge \\ not\ \textit{teacherOf}(x, c_{20})\ .$$

(3) scientists working for o_{12} and authoring a_4, or scientists working for o_3 and authoring a_{25}:

$$Q_3(x) = (\textit{Scientist}(x) \wedge \textit{worksFor}(x, o_{12}) \wedge \\ \textit{hasWritten}(x, a_4)) \vee (\textit{Scientist}(x) \wedge \\ \textit{worksFor}(x, o_3) \wedge \textit{hasWritten}(x, a_{25}))\ .$$

(4) scientists working for u_{11} but not having a doctoral degree from u_{11}, or scientists having a doctoral degree from u_{11} but not working for u_{11}:

$$Q_4(x) = (\textit{Scientist}(x) \wedge \textit{worksFor}(x, u_{11}) \wedge \\ not\ \textit{doctoralDegree}(x, u_{11})) \vee (\textit{Scientist}(x) \wedge \\ \textit{doctoralDegree}(x, u_{11}) \wedge not\ \textit{worksFor}(x, u_{11}))\ .$$

(5) professors who are also the head of a department:

$$Q_5(x) = \exists y\ (\textit{Professor}(x) \wedge \textit{headOf}(x, y) \wedge \\ \textit{Department}(y))\ .$$

(6) articles with an Italian author and published in 2007:

$$Q_6(x) = \exists y\ (\textit{Article}(x) \wedge \textit{yearOfPublication}(x, 2007) \wedge \\ \textit{hasWritten}(y, x) \wedge \textit{Scientist}(y) \wedge \\ \textit{nationality}(y, \textit{italian}))\ .$$

(7) scientists who are the authors of a journal and a conference paper published in 2007, or scientists who are the authors of a book published in 2007:

$$Q_7(x) = \exists y, z\ (\textit{Scientist}(x) \wedge \textit{hasWritten}(x, y) \wedge \\ \textit{JournalPaper}(y) \wedge \textit{yearOfPublication}(y, 2007) \wedge \\ \textit{hasWritten}(x, z) \wedge \textit{ConferencePaper}(z) \wedge \\ \textit{yearOfPublication}(z, 2007)) \vee \\ \exists y\ (\textit{Scientist}(x) \wedge \textit{hasWritten}(x, y) \wedge \\ \textit{Book}(y) \wedge \textit{yearOfPublication}(y, 2007))\ .$$

(8) Italian professors who are not heading any department:

$$Q_8(x) = \textit{Professor}(x) \wedge \textit{nationality}(x, \textit{italian}) \wedge \\ not\ \exists y\ (\textit{headOf}(x, y) \wedge \textit{Department}(y))\ .$$

(9) scientists who work for a university, but for no university from which they have the doctoral degree:

$$Q_9(x) = \exists z\,(\textit{Scientist}(x) \wedge \textit{worksFor}(x,z) \wedge \textit{University}(z) \wedge \mathit{not}\ \exists y\,(\textit{doctoralDegree}(x,y) \wedge \textit{worksFor}(x,y) \wedge \textit{University}(y)))\,.$$

(10) Italian scientists who have a non-Italian doctoral degree and work for an Italian organization, or non-Italian scientists who have an Italian doctoral degree and work for a non-Italian organization:

$$\begin{aligned} Q_{10}(x) = {} & \exists y, z\,(\textit{Scientist}(x) \wedge \textit{nationality}(x, \textit{italian}) \wedge \\ & \textit{University}(y) \wedge \mathit{not}\ \textit{state}(y, \textit{italy}) \wedge \\ & \textit{doctoralDegree}(x,y) \wedge \textit{worksFor}(x,z) \wedge \\ & \textit{Organization}(z) \wedge \textit{state}(z, \textit{italy})) \vee \\ & \exists y, z\,(\textit{Scientist}(x) \wedge \mathit{not}\ \textit{nationality}(x, \\ & \textit{italian}) \wedge \textit{University}(y) \wedge \textit{state}(y, \textit{italy}) \wedge \\ & \textit{doctoralDegree}(x,y) \wedge \textit{worksFor}(x,z) \wedge \\ & \textit{Organization}(z) \wedge \mathit{not}\ \textit{state}(z, \textit{italy}))\,. \end{aligned}$$

8 Summary and Outlook

We have presented a novel approach to Semantic Web search, where standard Web search is combined with ontological background knowledge. We have shown how the approach can be implemented on top of standard Web search engines and ontological inference technologies, using lightweight user-site software clients for annotation and query decomposition. We have developed the formal model behind this approach, which is based on tractable ontology languages. We have also generalized the PageRank technique to this approach. We have provided a technique for processing Semantic Web search queries, which consists of an offline ontological inference step and an online reduction to standard Web search queries, and we have proved that it is ontologically correct. We have reported on an implementation of our approach in desktop search, and we have provided experimental results, which show that the approach scales quite well to very large amounts of data.

In the future, we aim especially at extending the desktop implementation to a real Web implementation, using Google as Web search engine; we intend to cooperate with Google for a reliable indexing. Another interesting topic is to explore how plain natural language search strings can be transformed into the presented Semantic Web search queries. Furthermore, it would be interesting to investigate the use of probabilistic ontologies rather than classical ones.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284:34–43, 2001.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks*, 30(1/7):107–117, 1998.

[3] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning*, 39(3):385–429, 2007.

[4] L. Ding, R. Pan, T. W. Finin, A. Joshi, Y. Peng, and P. Kolari. Finding and ranking knowledge on the Semantic Web. In *Proc. ISWC-2005*, *LNCS* 3729, pp. 156–170.

[5] T. W. Finin, L. Ding, R. Pan, A. Joshi, P. Kolari, A. Java, and Y. Peng. Swoogle: Searching for knowledge on the Semantic Web. In *Proc. AAAI-2005*, pp. 1682–1683.

[6] N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello. The DLV system for knowledge representation and reasoning. *ACM Trans. Comput. Log.*, 7(3):499–562, 2006.

[7] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati. Linking data to ontologies. *J. Data Semantics*, 10:133–173, 2008.

[8] R. Rosati. Finite model reasoning in *DL-Lite*. In *Proc. ESWC-2008*, *LNCS* 5021, pp. 215–229.