Department of Computer Science

# SEMANTIC WEB SEARCH BASED ON ONTOLOGICAL CONJUNCTIVE QUERIES

Bettina Fazzinga
Giorgio Gianforme
Georg Gottlob
Thomas Lukasiewicz

RR-11-08

Department of Computer Science, University of Oxford
Wolfson Building, Parks Road, Oxford OX1 3QD

DEPARTMENT OF COMPUTER SCIENCE RESEARCH REPORT

DEPARTMENT OF COMPUTER SCIENCE RESEARCH REPORT 11-08, OCTOBER 2011

# SEMANTIC WEB SEARCH BASED ON ONTOLOGICAL CONJUNCTIVE QUERIES

(PRELIMINARY VERSION, 12 OCTOBER 2011)

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 a novel approach to Semantic Web search, called *Serene*, which allows for a semantic processing of Web search queries, and for evaluating complex Web search queries that involve reasoning over the Web. More specifically, we first add ontological structure and semantics to Web pages, which then allows for both attaching a meaning to Web search queries and Web pages, and for formulating and processing ontology-based complex Web search queries (i.e., conjunctive queries) that involve reasoning over the Web. Here, we assume the existence of an underlying ontology (in a lightweight ontology language) relative to which Web pages are annotated and Web search queries are formulated. Depending on whether we use a general or a specialized ontology, we thus obtain a general or a vertical Semantic Web search interface, respectively. That is, we are actually mapping the Web into an ontological knowledge base, which then allows for Semantic Web search relative to the underlying ontology. The latter is then realized by reduction to standard Web search on standard Web pages and logically completed ontological annotations. That is, standard Web search engines are used as the main inference motor for ontology-based Semantic Web search. We develop the formal model behind this approach and also provide an implementation in desktop search. Furthermore, we report on extensive experiments, including an implemented Semantic Web search on the Internet Movie Database.

[1]Dipartimento di Elettronica, Informatica e Sistemistica, Università della Calabria, Italy; e-mail: bfazzinga@deis.unical.it.

[2]Dipartimento di Informatica e Automazione, Università Roma Tre, Italy; e-mail: giorgio.gianforme@gmail.com.

[3]Department of Computer Science and Oxford-Man Institute of Quantitative Finance, University of Oxford, UK; e-mail: georg.gottlob@cs.ox.ac.uk.

[4]Department of Computer Science, University of Oxford, UK; e-mail: thomas.lukasiewicz@cs.ox.ac.uk.

**Acknowledgements**: This paper is a significantly extended and revised version of a paper that appeared in: *Proceedings FoIKS-2010*, *LNCS 5956*, pp. 153–172. Springer, 2010 [27]. It is a preliminary version of a paper that will appear in: *J. Web Sem.*, 2011. This work was supported by the European Research Council under the EU's 7th Framework Programme (FP7/2007-2013)/ERC grant 246858 – DIADEM, by the German Research Foundation (DFG) under the Heisenberg Programme, and by a Yahoo! Research Fellowship. Georg Gottlob is a James Martin Senior Fellow. The work was carried out in the context of the James Martin Institute for the Future of Computing. We are very grateful to Giorgio Orsi for his generous help in providing the experimental results shown in Table 1. Many thanks also to the reviewers of the FoIKS-2010 abstract of this paper for their useful and constructive comments, which have helped to improve this work.

## Contents

## 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 [8]. For this reason, however, current standard Web search does not allow for a *semantic processing of Web search queries*, which analyzes both Web search queries and Web pages with respect to their meaning, and returns exactly the semantically relevant pages to a query. For the same reason, current standard Web search also does not allow for *evaluating complex Web search queries that involve reasoning over the Web*.

Many experts predict that the next huge step forward in Web information technology will be achieved by adding such structure and/or semantics to Web contents and exploiting them when processing Web search queries. Indeed, the *Semantic Web* [5, 6] as a vision of a more powerful future Web goes in this direction. It is a common framework that allows data to be shared and reused in different applications, enterprises, and communities. 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* [36, 54, 4], 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 vision of 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.

As an important example of an initiative towards adding structure and/or semantics to Web contents in practice (and thus ultimately also towards the Semantic Web), Google's *Rich Snippets*[1] highlight useful information from Web pages via structured data standards such as microformats and RDFa. As another example, *OpenCalais*[2] turns unstructured HTML into semantically marked up data, ordering data into groups such as "people", "places", "companies", and more. Other (less general) examples are *Freebase*[3], which is a semantically marked up database of structured information similar to Wikipedia, and *DBpedia*[4], which extracts structured information from Wikipedia and makes that data available on the Web.

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 (see Section 10). In particular, there is a fast growing number of commercial and academic Semantic Web search engines. There are essentially two main research directions. The first (and most common) one is to develop a new form of search for searching the pieces of data and knowledge that are encoded in the new representation formalisms of the Semantic Web, while the second (and nearly unexplored) direction is to use the formalisms of the Semantic Web in order to add some semantics to Web search. The second direction is also a first step towards Web search queries in (written or spoken) natural language.

In this paper, we follow the second line of research. We aim at adding ontological structure to Web pages, which then allows for both analyzing the meaning of Web search queries and Web pages, and for

[1]`http://knol.google.com/k/google-rich-snippets-tips-and-tricks`
[2]`http://www.opencalais.com/`
[3]`http://www.freebase.com/`
[4]`http://dbpedia.org/`

formulating and processing ontology-based complex Web search queries that involve reasoning over the Web. Intuitively, rather than being interpreted in a keyword-based syntactic fashion, the pieces of data on existing Web pages are connected to (and via) some ontological knowledge base and then interpreted relative to this knowledge base. That is, the pieces of data on Web pages are connected to (and via) a much more precise semantic and contextual meaning. This allows for answering Web search queries in a much more precise way, taking into account the meaning of Web search queries and pages, and it also allows for more complex ontology-based Web search queries that involve reasoning over the Web, which are also much closer to complex natural language search queries than current Boolean keyword-based search queries. The following are some examples of such Web search queries, which can be appropriately handled in our Semantic Web search, but not in current standard Web search:

- 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 that 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, that exist for at least 15 years, and that were 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 relative to some ontological knowledge.

- Suppose next that one is searching for "bus" (as a means of transportation for persons) on the Web. Then, one is looking for buses or synonyms / related concepts, but also for special kinds of buses that are not synonyms / related concepts, such as e.g. passenger vans. Ontological knowledge now allows for obtaining both a collection of contextually correct synonyms / related concepts and a collection of contextually correct special kinds of buses.

- Similarly, a Web search for "president of the USA" should also return Web pages that contain "George W. Bush" (who is one of the presidents of the USA according to some ontological knowledge). 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).

- On the other hand, when searching for Web pages about the first president of the USA, "Washington", ontological knowledge allows 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.

In our approach, an ontologically enriched Web along with complex ontology-based search on the Web are achieved on top of the existing Web and using existing Web search engines. Intuitively, standard Web pages are first connected to (and via) an ontological knowledge base, which then allows for formulating and processing complex ontology-based (conjunctive) search queries that involve reasoning over the data of the Web. The query processing step is based on new techniques (i) for pre-compiling the ontological knowledge using standard ontology reasoning techniques and (ii) for translating complex ontology-based Web queries into (sequences of) standard Web queries that are answered by standard Web search. That is, essential parts of ontological search on the Web are actually reduced to state-of-the-art search engines such as *Google search*. As important advantages, this approach can immediately be applied to the whole existing Web, and it can be done with existing Web search technology (and so does not require completely new technologies). Such a line of research aims at adding ontology-based structure and semantics (and thus in a sense also intelligence) to current search engines for the existing Web by combining existing Web pages and queries with ontological knowledge.

The ontological knowledge and annotations that are underlying our Semantic Web search can be classified according to its origin and contents. As for the origin, they may be either (a) explicitly defined by experts, or (b) automatically extracted from the Web, eventually coming along with existing pieces of ontological knowledge and annotations (e.g., from existing ontologies or ontology fragments, and/or from existing annotations of Web pages in microformats or RDFa). In the latter case, generating, maintaining, and updating the ontological knowledge and annotations is done automatically and much less cost-intensive than in the former case. As for the contents, (a) the ontological knowledge and annotations may either describe fully general knowledge (such as the knowledge encoded in Wikipedia) for general ontology-based search on the Web, or (b) they may describe some specific knowledge (such as biomedical knowledge) for vertical ontology-based search on the Web. The former results into a general ontology-based interface to the Web similar to Google, while the latter produces different vertical ontology-based interfaces.

The main contributions of this paper and the characteristic features of our approach to Semantic Web search can be briefly summarized as follows:

- We present a novel approach to Semantic Web search, called *Serene*, which allows for a semantic processing of Web search queries relative to an underlying ontology, and for evaluating ontology-based complex Web search queries that involve reasoning over the Web. We show how the approach can be implemented on top of standard Web search engines and ontological inference technologies.

- We develop the formal model behind this approach. In particular, we introduce Semantic Web knowledge bases and Semantic Web search queries to them. We also define the ObjectRank ranking for our Semantic Web search.

- We provide a technique for processing Semantic Web search queries, which consists of an offline inference and an online reduction to a collection of standard Web search queries. We prove that this way of processing Semantic Web search queries is always ontologically correct. Furthermore, we identify a large class of Semantic Web knowledge bases where it is also complete.

- The offline inference compiles terminological knowledge into so-called *completed* annotations. We prove that these have a polynomial size and can also be computed in polynomial time. Furthermore, experimental data show that they are also rather small in practice, especially since ontological hierarchies in practice are generally not that deep (a concept has at most a dozen superconcepts).

- We report on two prototype implementations of our approach in desktop search. Experiments with more than one million annotation facts show that the new methods are principally feasible and potentially scale to Web search (which is actually much faster than desktop search, even with a much larger search space).

- We also compare our most recent prototype with the Corese system [17], which is the Semantic Web search system in the state-of-the-art that is most closely related to our approach, showing that our system is 18 times quicker than Corese.

- Differently from conventional Boolean keyword-oriented Web search queries, the proposed Semantic Web search queries clearly empower the user to precisely describe her information need for certain kinds of queries, resulting in a very precise result set and a very high precision and recall for the query result.

- We show that our approach to Semantic Web search can be readily applied to existing Web pages, even if they are currently not (yet) semantically annotated, and that it can be used to perform a vertical ontology-based search. More concretely, we have used the approach to implement a Semantic Web search interface for the Web pages of the Internet Movie Database (IMDB)[5].

The rest of this paper is organized as follows. In Section 2, we give an overview of our approach to Semantic Web search. In Section 3, we introduce Semantic Web knowledge bases and Semantic Web search queries, and we define the ObjectRank ranking. Sections 4 to 6 describe how Semantic Web search queries are processed via offline inference and online reduction to standard Web search. In Sections 7 and 8, we report on two prototype implementations for semantic desktop search, along with extensive experimental results. Section 9 describes the implemented Semantic Web search on the IMDB. In Sections 10 and 11, we discuss related work, summarize our main results, and give an outlook on future research. The basics of the underlying tractable ontology language are recalled in Appendix A, and detailed proofs of all results are given in Appendix B.

## 2 System Overview

The overall architecture of our Semantic Web search system, called *Serene* (<u>Se</u>mantic Web sea<u>r</u>ch <u>en</u>gin<u>e</u>), is shown in Fig. 1. It consists of the *Interface*, the *Query Evaluator* (implemented on top of standard Web *Search Engines*), and the *Inference Engine* (blue parts). Standard *Web* pages and their objects are enriched by *Annotation* pages, based on an *Ontology*.

### 2.1 Ontology

Our approach to Semantic Web search is done relative to a fixed underlying *ontology*, which defines an alphabet of elementary ontological ingredients, as well as terminological relationships between these ingredients. The ontology may either describe fully general knowledge (such as the knowledge encoded in Wikipedia) for general ontology-based search on the Web, or it may describe some specific knowledge (such as biomedical knowledge) for vertical ontology-based search on the Web. The former results into a general ontology-based interface to the Web similar to Google, while the latter produces different vertical ontology-based interfaces to the Web. There are many existing ontologies that can be used, which have especially been developed in the context of the Semantic Web, but also in biomedical and technical areas. Such ontologies are generally created and updated by human experts in a knowledge engineering process. Recent research attempts are also directed towards an automatic generation of ontologies from text documents, eventually coming along with existing pieces of ontological knowledge [9, 24].

For example, an ontology may contain the knowledge 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*, and (v) *hasFirstAuthor* is a functional binary relationship, which is formalized by:

$$\begin{array}{l} \mathit{ConferencePaper} \sqsubseteq \mathit{Article},\ \mathit{JournalPaper} \sqsubseteq \mathit{Article}, \\ \mathit{ConferencePaper} \sqsubseteq \neg \mathit{JournalPaper}, \\ \exists \mathit{isAuthorOf} \sqsubseteq \mathit{Scientist},\ \exists \mathit{isAuthorOf}^- \sqsubseteq \mathit{Article}, \\ \mathit{isAuthorOf}^- \sqsubseteq \mathit{hasAuthor},\ \mathit{hasAuthor}^- \sqsubseteq \mathit{isAuthorOf}, \\ (\mathsf{funct}\ \mathit{hasFirstAuthor})\,. \end{array} \qquad (1)$$

[5]http://www.imdb.com

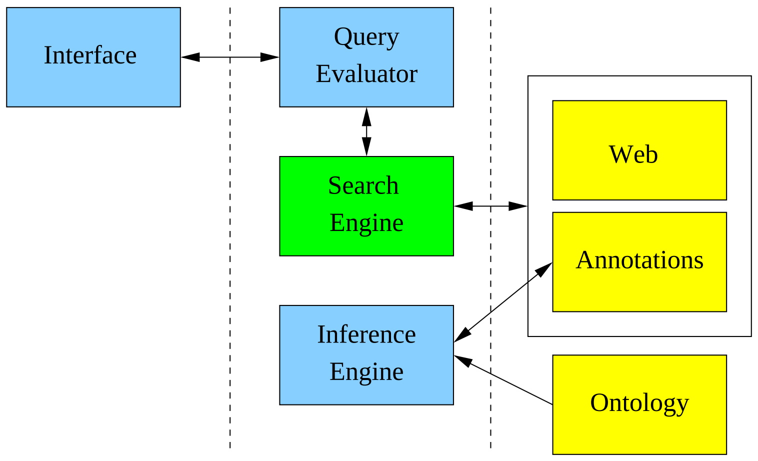


Figure 1: System architecture.

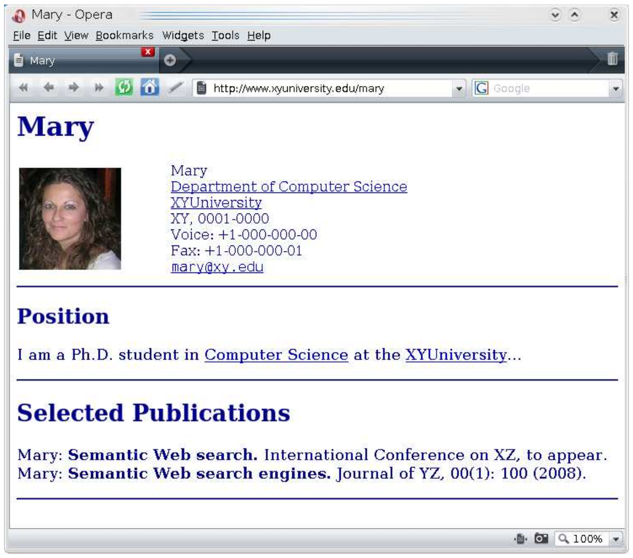
## 2.2 Annotations

As a second ingredient of our approach to Semantic Web search, we assume the existence of assertional pieces of knowledge about Web pages and their objects, also called *(semantic) annotations*, which are defined relative to the terminological relationships of the underlying ontology. Such annotations are starting to be widely available for a large class of Web resources, especially user-defined annotations with the Web 2.0. They may also be automatically learned from Web pages and their objects (see, e.g., [15]). As a midway between such fully user-defined and fully automatically generated annotations, one can also automatically extract annotations from Web pages using user-defined rules, as described in Section 9.

For example, in a very simple scenario relative to the ontology in Eq. 1, a Web page $i_1$ may contain information about a Ph.D. student $i_2$, called Mary, and two of her papers: a conference paper $i_3$ with title "*Semantic Web search*" and a journal paper $i_4$ entitled "*Semantic Web search engines*" and published in 2008. A simple HTML page representing this scenario is shown in Fig. 2.

There may now exist 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 one 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 $i_3$ may encode that $i_3$ is a conference paper and has the title "*Semantic Web search*", while the one for $i_4$ 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 then formally expressed as the following sets of ontological axioms $\mathcal{A}_{i_1}$, $\mathcal{A}_{i_2}$, $\mathcal{A}_{i_3}$, and $\mathcal{A}_{i_4}$, respectively:

$$\begin{aligned}
\mathcal{A}_{i_1} &= \{\mathit{contains}(i_1, i_2),\ \mathit{contains}(i_1, i_3), \mathit{contains}(i_1, i_4)\},\\
\mathcal{A}_{i_2} &= \{\mathit{PhDStudent}(i_2),\ \mathit{name}(i_2, \text{"\textit{mary}"}),\\
&\quad\ \mathit{isAuthorOf}(i_2, i_3),\ \mathit{isAuthorOf}(i_2, i_4)\},\\
\mathcal{A}_{i_3} &= \{\mathit{ConferencePaper}(i_3),\ \mathit{title}(i_3, \text{"\textit{Semantic Web search}"})\},\\
\mathcal{A}_{i_4} &= \{\mathit{JournalPaper}(i_4),\ \mathit{hasAuthor}(i_4, i_2),\\
&\quad\ \mathit{title}(i_4, \text{"\textit{Semantic Web search engines}"}),\\
&\quad\ \mathit{yearOfPublication}(i_4, 2008),\ \mathit{keyword}(i_4, \text{"RDF"})\}.
\end{aligned} \tag{2}$$


Mary - Opera
File Edit View Bookmarks Widgets Tools Help
Mary
http://www.xyuniversity.edu/mary
Google
100%

Mary

Mary
Department of Computer Science
XYUniversity
XY, 0001-0000
Voice: +1-000-000-00
Fax: +1-000-000-01
mary@xy.edu

Position

I am a Ph.D. student in Computer Science at the XYUniversity...

Selected Publications

Mary: **Semantic Web search.** International Conference on XZ, to appear.
Mary: **Semantic Web search engines.** Journal of YZ, 00(1): 100 (2008).

Figure 2: HTML page.

## 2.3 Inference Engine

Differently from the ontology, the semantic annotations can be directly published on the Web and searched via standard Web search engines. In order to also make it visible to standard Web search engines, the ontology is compiled into the semantic annotations. More specifically, all semantic annotations are completed in an offline ontology compilation step, where the *Inference Engine* adds all properties (i.e., ground atoms) that can be deduced from the ontology and the semantic annotations. The resulting (*completed*) semantic annotations are then published as Web pages, so that they can be searched by standard Web search engines.

For example, considering again the running scenario, using the ontology in Eq. 1, in particular, we can derive from the semantic annotations in Eq. 2 that the two papers $i_3$ and $i_4$ are also articles, and both authored by Mary.

## 2.4 HTML Encoding of Annotations

The above searchable (completed) semantic annotations of (objects on) standard Web pages are published as HTML Web pages with pointers to the respective object pages, so that they (in addition to the standard Web pages) can be searched by standard search engines. 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. We here use the HTML address of the Web page/object's annotation page as an identifier for that Web page/object. The plain textual representation of the completed semantic annotations allows their processing by existing standard search engines for the Web. It is important to point out that this textual representation is simply a list of properties, each eventually along with an identifier or a data value as attribute value, and it can thus immediately be encoded as a list of RDF triples. Similarly, the completed semantic annotations can be easily encoded in RDFa or microformats.

$i_1$ : www.xyuni.edu/mary/an1.html

```
<html>
<body>
www.xyuni.edu/mary <br>
WebPage <br>
contains i2 <br>
contains i3 <br>
contains i4 <br>
</body>
</html>
```

$i_3$ : www.xyuni.edu/mary/an3.html

```
<html>
<body>
www.xyuni.edu/mary <br>
Article <br>
ConferencePaper <br>
hasAuthor i2 <br>
title Semantic Web search <br>
</body>
</html>
```

$i_2$ : www.xyuni.edu/mary/an2.html

```
<html>
<body>
www.xyuni.edu/mary <br>
PhDStudent <br>
name mary <br>
isAuthorOf i3 <br>
isAuthorOf i4 <br>
</body>
</html>
```

$i_4$ : www.xyuni.edu/mary/an4.html

```
<html>
<body>
www.xyuni.edu/mary <br>
Article <br>
JournalPaper <br>
hasAuthor i2 <br>
title Semantic Web search engines <br>
yearOfPublication 2008 <br>
keyword RDF <br>
</body>
</html>
```

Figure 3: Four HTML pages encoding the (completed) semantic annotations for the HTML page in Fig. 2 and the three objects on it.

## 2.5 Query Evaluator

The *Query Evaluator* reduces each Semantic Web search query of the user in an online query processing 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*. 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:

$$\begin{aligned} Q(x) = \exists y\, (&\mathit{PhDStudent}(x) \wedge \mathit{isAuthorOf}(x, y) \wedge \mathit{Article}(y) \wedge \\ &\mathit{yearOfPublication}(y, 2008) \wedge \mathit{keyword}(y, \text{“}RDF\text{”}))\,. \end{aligned} \tag{3}$$

This query is transformed into the two queries $Q_1 = \mathit{PhDStudent}$ AND *isAuthorOf* and $Q_2 = \mathit{Article}$ AND *“yearOfPublication* 2008” AND *“keyword* RDF”, which can both be submitted to a standard Web search engine. The result of the original query $Q$ is then built from the results of the two queries $Q_1$ and $Q_2$. Note that a graphical user interface, such as the one of Google's advanced search, and ultimately a natural language interface (for queries in written or spoken natural language) can help to hide the conceptual complexity of ontological queries to the user.

## 3 Semantic Web Search

In this section, we introduce Semantic Web knowledge bases, and we define the syntax and semantics of Semantic Web search queries to such knowledge bases. We then introduce a ranking for individuals in our approach, called *ObjectRank*, which generalizes the standard PageRank ranking of Web pages. Although we implicitly assume the tractable description logic *DL-Lite*$_\mathcal{A}$ (cf. Appendix A) as underlying ontology language for Semantic Web knowledge bases and search queries, any other ontology languages may be used as well.

### 3.1 Semantic Web Knowledge Bases

Informally, 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.

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. Let $\mathbf{I}$ be the disjoint union of two sets $\mathbf{P}$ and $\mathbf{O}$ of *Web pages* and *Web objects*, respectively. Informally, every Web page $p \in \mathbf{P}$ is an identifier for a concrete Web page, while every Web object $o \in \mathbf{O}$ 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 objects on concrete Web pages.

**Definition 1** 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 $U(a, v)$ (which all have the Web page or object $a$ as first argument), 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 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 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 1** *(Scientific Database cont'd).* Continuing the running example of Section 2, a Semantic Web knowledge base may specify some simple information about scientists and their publications. The sets of atomic concepts, atomic roles, atomic attributes, individuals, and data values are given as follows:

$$\begin{aligned}
\mathbf{A} &= \{\mathit{Scientist}, \mathit{PhDStudent}, \mathit{Article}, \mathit{ConferencePaper}, \mathit{JournalPaper}\},\\
\mathbf{R}_A &= \{\mathit{hasAuthor}, \mathit{isAuthorOf}, \mathit{hasFirstAuthor}, \mathit{contains}\},\\
\mathbf{R}_D &= \{\mathit{name}, \mathit{title}, \mathit{yearOfPublication}, \mathit{keyword}\},\\
\mathbf{I} &= \{i_1, i_2, i_3, i_4\},\\
\mathbf{V} &= \{\text{“}\mathit{mary}\text{”}, \text{“}\mathit{Semantic\ Web\ search}\text{”}, 2008,\\
&\qquad \text{“}\mathit{Semantic\ Web\ search\ engines}\text{”}, \text{“}\mathit{RDF}\text{”}\}\,.
\end{aligned}$$

The set $\mathbf{I}$ is partitioned into the set $\mathbf{P} = \{i_1\}$ of Web pages and the set $\mathbf{O} = \{i_2, i_3, i_4\}$ of Web objects on $i_1$. Then, a Semantic Web knowledge base is given by $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$, where the TBox $\mathcal{T}$ is given by the axioms in Eq. 1, and the semantic annotations $\mathcal{A}_a$ of the individuals $a \in \mathbf{P} \cup \mathbf{O}$ are the ones in Eq. 2.

## 3.2 Semantic Web Search Queries

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

### Syntax

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

We first give some preparative definitions as follows. 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*) $\alpha$ has 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* is of 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 $\alpha$ and equalities $=(t, t')$, which have free variables among $\mathbf{x}$ and $\mathbf{y}$. We are now ready to define the notion of a Semantic Web search query as follows.

**Definition 2** 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 $\phi_i$ with $i \in \{1, \ldots, n\}$ is a conjunction of atoms $\alpha$ (also called *positive atoms*), conjunctive formulas $\psi$, negated conjunctive formulas $\mathit{not}\ \psi$, and equalities $=(t, t')$, which have free variables among $\mathbf{x}$ and $\mathbf{y}_i$.

**Example 2** *(Scientific Database cont'd).* The query $Q(x)$ of Eq. 3 is a Semantic Web search query. Two other Semantic Web search queries are:

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

Informally, $Q_1(x)$ asks for all 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 $Q_2(x)$ asks for all 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 TBox $\mathcal{T}$ of the underlying Semantic Web knowledge base $KB$), such as Ph.D. students or computer scientists.

## Semantics of Positive Search Queries

We define the semantics of positive Semantic Web search queries, which are negation-free, in terms of ground substitutions via the notion of logical consequence.

We first give some preliminary definitions. A Semantic Web search query $Q(\mathbf{x})$ is *positive* iff it contains no negated conjunctive subqueries. A *(variable) substitution* $\theta$ maps variables from $\mathbf{X}$ to terms. A substitution $\theta$ 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 $\phi$ is a *logical consequence* of a 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 $\phi$.

**Definition 3** Given a Semantic Web knowledge base $KB$ and a positive Semantic Web search query $Q(\mathbf{x})$ with free variables $\mathbf{x}$, an *answer* for $Q(\mathbf{x})$ to $KB$ is a ground substitution $\theta$ for the variables $\mathbf{x}$ such that $KB \models Q(\mathbf{x}\theta)$.

**Example 3** *(Scientific Database cont'd).* Consider again the Semantic Web knowledge base $KB$ of Example 1. The Semantic Web search query $Q(x)$ of Eq. 3 is positive, and an answer for $Q(x)$ to $KB$ is given by $\theta = \{x/i_2\}$.

## Semantics of General Search Queries

We next define the semantics of general 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 $\psi$. That is, negations are interpreted by a closed-world semantics on top of the open-world semantics of description logics.

**Definition 4** Given a Semantic Web knowledge base $KB$ and search query

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

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

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

Informally, a ground substitution $\theta$ is an answer for the search query $Q(\mathbf{x})$ to $KB$ iff (i) $\theta$ is an answer for $Q^+(\mathbf{x})$ to $KB$, and (ii) $\theta$ 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.

**Example 4** *(Scientific Database cont'd).* Consider again the Semantic Web knowledge base $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ of Example 1 and the following general Semantic Web search query, asking for Mary's unpublished non-journal papers:

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

Then, 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*”. Note that the membership axioms $\mathit{Article}(i_3)$ and $\mathit{hasAuthor}(i_2, i_3)$ are not in the semantic annotations $\mathcal{A}_a$, $a \in \mathbf{P} \cup \mathbf{O}$, but they can be inferred from them using the ontology $\mathcal{T}$.

### 3.3 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 generalize the PageRank technique [8]: 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. More concretely, we define the *ObjectRank* of a Web page or object $a$ as follows:

$$R(a) = d \cdot \textstyle\sum_{b \in B_a} R(b) \, / \, N_b + (1-d) \cdot E(a)\,, \tag{4}$$

where (i) $B_a$ is the set of all Web pages and objects $o$ that relate to $a$ (i.e., $o$ relates to $a$ via some role), (ii) $N_b$ is the number of Web pages and objects $o$ that relate from $b$ (i.e., $b$ relates to $o$ via some role), (iii) $d$ is a damping factor, and (iv) $E$ associates with every Web page and object an initial value, called *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 objects, and on the relationships between Web objects, where the user fixes the roles to be considered. Note that in some cases, only a subset of all relationships may be used for specifying ObjectRank. For example, in the Scientific Database, the relationship "cites" alone between articles produces a very useful ranking on articles.

The ranking on Web pages and objects is then naturally extended to answers (i.e., tuples of Web pages, Web objects, and values) for Semantic Web search queries to Semantic Web knowledge bases. For example, the answers can be ordered lexicographically, or the rank of an answer can be defined as the minimum (or maximum) of the ranks of its Web pages and objects, and then ordered as usual.

## 4 Realizing Semantic Web Search

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 when performing 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 (being represented on the Web in addition to the standard Web pages) is then searched online by a collection of standard Web search queries depending on $Q$. So, processing Semantic Web search queries $Q$ is divided into

- an offline ontology reasoning step, where all semantic annotations of Web pages and objects are completed by membership axioms entailed from $KB$, and
- an online reduction to standard Web search, where $Q$ is transformed into standard Web search queries whose answers are used to construct the answer for $Q$.

Observe that the compilation of the TBox $\mathcal{T}$ via an offline ontology reasoning step into the ABox $\mathcal{A}$ of $KB$ has several important advantages over the compilation of $\mathcal{T}$ into the query $Q$. Note that the latter technique is also applied in the inference and query processing algorithms of *DL-Lite*; it implies that ontology reasoning is done online during the query processing step. As a first advantage of the former, ontology reasoning is done offline (i.e., before and independently from query processing), and thus its computation time does not appear in the query processing time. Second, ontology reasoning is done only once, and then used in processing different queries, while the compilation of $\mathcal{T}$ into $Q$ requires to newly perform ontology reasoning for every query, and thus results into many repeated computations. The above two advantages are especially important for very large knowledge bases, as it is the case in Semantic Web search. Third, online query processing on the data resulting from an offline ontology inference step is very close to current Web search techniques, which also include the offline construction of a search index, which is then used for rather efficiently performing online query processing. In a sense, offline ontology inference can be considered as the offline construction of an ontological index, in addition to the standard index for Web search. Fourth, our approach reuses existing Web search techniques and can easily be integrated with them, which is generally not possible for the compilation of $\mathcal{T}$ into $Q$, as this requires a logically correct handling of Boolean operators in search queries; current standard Web search engines, however, are generally lacking such a handling. Fifth, as another advantage, the compilation of $\mathcal{T}$ into $\mathcal{A}$ actually also makes our approach independent from the underlying ontology language. A disadvantage of our approach to offline reasoning compared to the compilation of $\mathcal{T}$ into $Q$ is that its result must be updated whenever the TBox $\mathcal{T}$ or the ABox $\mathcal{A}$ change, which is, however, less an issue especially when $\mathcal{T}$ and $\mathcal{A}$ change only rarely.

In the offline ontology reasoning step, we check whether the Semantic Web knowledge base is satisfiable, and we compute the completion of all semantic annotations, i.e., we augment the semantic annotations with all concept, role, and attribute membership axioms that can be deduced from the semantic annotations and the ontology. We suggest to use only the so-called simple completion of all semantic annotations, which is sufficient for a large class of Semantic Web knowledge bases and search queries. It is important to point out that since ontology reasoning is done offline (like the construction of an index structure for Web search), its running time does not contribute to the running time of the actual online processing of Semantic Web search queries. Thus, the running time used for ontology reasoning can be fully neglected. Nonetheless, in tractable ontology languages such as $\mathit{DL\text{-}Lite}_{\mathcal{A}}$, checking whether a Semantic Web knowledge base $\mathit{KB}$ is satisfiable, and checking whether a membership axiom is a logical consequence of $\mathit{KB}$ for computing the simple completion of $\mathit{KB}$ can both be done in LOGSPACE in the data complexity, and one can use existing systems such as QuOnto [12].

In the online reduction to standard Web search, we decompose a given Semantic Web search query $Q$ into a collection of standard Web search queries, of which the answers are then used to construct the answer for $Q$. The standard Web search queries are processed with existing search engines on the Web. Publishing the completed semantic annotations as standard Web pages, as we propose here, this standard Web search can be done immediately with existing standard Web search engines (see Section 7 for an implementation of semantic desktop search on top of standard desktop search). Alternatively, we may also keep the completed semantic annotations in a virtual way only and use them for the construction of the index structure for Web search only. In that case, the offline ontology reasoning step can be combined with the construction of the index structure for Web search.

Note that the terms “online” and “offline” are here used in a computational sense. In the following, we describe the offline ontology reasoning step in Section 5 and the online reduction to standard Web search in Section 6. We finally describe the implementation of a semantic desktop search engine in Section 7.

## 5 Offline Ontology Compilation

The offline ontology reasoning step compiles the implicit pieces of 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 and objects, so that they (in addition to the standard Web pages) can be searched by standard Web search engines. This compilation is always correct, also for other underlying ontology languages different from *DL-Lite*$_{\mathcal{A}}$. To also obtain completeness under *DL-Lite*$_{\mathcal{A}}$, (a) in the case of quantifier-free search queries, (b) when the TBox is equivalent to a Datalog program, and (c) more generally (relative to both) when the existentially quantified variables in search queries occur only in safe positions, it is sufficient to add all logically entailed membership axioms constructed from Web pages, Web objects, and data values. Note that this completeness result assumes that Semantic Web knowledge bases are defined relative to *DL-Lite*$_{\mathcal{A}}$.

## 5.1 Simple Completion

We now introduce the notion of simple completion for Semantic Web knowledge bases, which formalizes the compilation of TBox into ABox knowledge. Informally, for every Web page and object $a$, all deducible ground membership axioms are collected in a completed semantic annotation of $a$. Observe here that the notion of simple completion of a Semantic Web knowledge base depends only on its sets of atomic concepts, atomic roles, atomic attributes, individuals, and data values, and is otherwise independent from the underlying ontology language.

**Definition 5** Let $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ be a satisfiable Semantic Web knowledge base. The *simple completion* of $KB$ is the knowledge base $KB' = (\emptyset, (\mathcal{A}_a{}')_{a \in \mathbf{P} \cup \mathbf{O}})$ where each $\mathcal{A}_a{}'$ is the set of all concept membership axioms $A(a)$, role membership axioms $P(a,b)$, and attribute membership axioms $U(a,v)$ logically implied by $\mathcal{T} \cup \bigcup_{a \in \mathbf{P} \cup \mathbf{O}} \mathcal{A}_a$, where $A \in \mathbf{A}$, $P \in \mathbf{R}_A$, $U \in \mathbf{R}_D$, $b \in \mathbf{I}$, and $v \in \mathbf{V}$.

**Example 5** *(Scientific Database cont'd).* Consider again the Semantic Web knowledge base $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ of Example 1. Then, the simple completion of $KB$ contains in particular the three new axioms $\mathit{Article}(i_3)$, $\mathit{hasAuthor}(i_3, i_2)$, and $\mathit{Article}(i_4)$. The first two are added to $A_{i_3}$ and the last one to $A_{i_4}$.

In general, for any underlying ontology language of a Semantic Web knowledge base, the simple completion allows for correctly but not for completely evaluating Semantic Web search queries (i.e., all answers are correct, but some answers may be missing). This is because existentially quantified variables in the search query may refer to incompletely specified existentially quantified entries in the Semantic Web knowledge base, which thus may not be connected to concrete individuals and values. Hence, one case where we easily obtain completeness is when there are no existential quantifiers in the search query. Towards this result, the following theorem shows that positive quantifier-free search queries to a knowledge base $KB$ over *DL-Lite*$_{\mathcal{A}}$ can be evaluated on the simple completion of $KB$ (which contains only compiled but no explicit TBox knowledge anymore).

**Theorem 1** *Let $KB$ be a satisfiable Semantic Web knowledge base over DL-Lite$_{\mathcal{A}}$, let $Q(\mathbf{x})$ be a positive Semantic Web search query without existential quantifiers, and let $\theta$ be a ground substitution for $\mathbf{x}$. Then, $\theta$ is an answer for $Q(\mathbf{x})$ to $KB$ iff $\theta$ is an answer for $Q(\mathbf{x})$ to the simple completion of $KB$.*

As an immediate consequence, general quantifier-free search queries to a Semantic Web knowledge base $KB$ over *DL-Lite*$_{\mathcal{A}}$ can also be evaluated on the simple completion of $KB$, which is expressed by the following corollary.

**Corollary 2** *Let $KB$ be a satisfiable Semantic Web knowledge base over DL-Lite$_{\mathcal{A}}$, $Q(\mathbf{x})$ be a (general) Semantic Web search query without existential quantifiers, and $\theta$ be a ground substitution for $\mathbf{x}$. Then, $\theta$*

*is an answer for $Q(\mathbf{x})$ to $KB$ iff $\theta$ is an answer for $Q^{+}(\mathbf{x})$ but not an answer for $Q^{-}(\mathbf{x})$ to the simple completion of $KB$.*

Another case where we easily obtain completeness is when there are no incomplete existentially quantified entries in the Semantic Web knowledge base, which requires to disallow some concept inclusion axioms, and which actually means that the knowledge base is equivalent to a Datalog program. This is expressed by the following theorem for positive search queries. Observe that the theorem does not exclude existentially quantified variables to occur in such queries.

**Theorem 3** *Let $KB$ be a satisfiable Semantic Web knowledge base over DL-Lite$_{\mathcal{A}}$ such that none of the concept inclusion axioms in $KB$ has one of the forms $B \sqsubseteq \exists P$, $B \sqsubseteq \exists P^{-}$, $B \sqsubseteq \delta(U)$, $B \sqsubseteq \exists P.C$, and $B \sqsubseteq \exists P^{-}.C$. Let $Q(\mathbf{x})$ be a positive Semantic Web search query, and let $\theta$ be a ground substitution for $\mathbf{x}$. Then, $\theta$ is an answer for $Q(\mathbf{x})$ to $KB$ iff $\theta$ is an answer for $Q(\mathbf{x})$ to the simple completion of $KB$.*

It follows immediately that fully general search queries to a Semantic Web knowledge base $KB$ over *DL-Lite*$_{\mathcal{A}}$, where the same concept inclusion axioms are disallowed, can also be evaluated on the simple completion of $KB$.

**Corollary 4** *Let $KB$ be a satisfiable Semantic Web knowledge base over DL-Lite$_{\mathcal{A}}$ such that none of the concept inclusion axioms in $KB$ has one of the forms $B \sqsubseteq \exists P$, $B \sqsubseteq \exists P^{-}$, $B \sqsubseteq \delta(U)$, $B \sqsubseteq \exists P.C$, and $B \sqsubseteq \exists P^{-}.C$. Let $Q(\mathbf{x})$ be a (general) Semantic Web search query, and let $\theta$ be a ground substitution for $\mathbf{x}$. Then, $\theta$ is an answer for $Q(\mathbf{x})$ to $KB$ iff $\theta$ is an answer for $Q^{+}(\mathbf{x})$ but not an answer for $Q^{-}(\mathbf{x})$ to the simple completion of $KB$.*

More generally, we also obtain completeness when all existentially quantified variables in search queries occur only in positions that do not carry any incomplete existentially quantified entry in the Semantic Web knowledge base. To formalize this result, we first define the notion of positions and their safeness as follows. A *position* $F[i]$ consists of an atomic concept, atomic role, or attribute $F$ and an argument position $i = 1$, if $F$ is an atomic concept, and $i \in \{1, 2\}$, if $F$ is an atomic role or attribute. A position $F[i]$ is *safe* relative to a Semantic Web knowledge base $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ iff only Web pages $p \in \mathbf{P}$, Web objects $o \in \mathbf{O}$, and values $v \in \mathbf{V}$ occur in that position in atoms in any universal model of the Datalog$^{\pm}$ translation of $\mathcal{T} \cup \bigcup_{a \in \mathbf{P} \cup \mathbf{O}} \mathcal{A}_a$ [11]. The following theorem then formalizes the above idea for the case of positive search queries; it generalizes Theorems 1 and 3.

**Theorem 5** *Let $KB$ be a satisfiable Semantic Web knowledge base over DL-Lite$_{\mathcal{A}}$. Let $Q(\mathbf{x})$ be a positive Semantic Web search query such that all existentially quantified variables occur only in safe positions, and let $\theta$ be a ground substitution for $\mathbf{x}$. Then, $\theta$ is an answer for $Q(\mathbf{x})$ to $KB$ iff $\theta$ is an answer for $Q(\mathbf{x})$ to the simple completion of $KB$.*

As an immediate consequence, this result also carries over to the case of general search queries to a Semantic Web knowledge base $KB$ over *DL-Lite*$_{\mathcal{A}}$. This is expressed by the following corollary, which generalizes Corollaries 2 and 4.

**Corollary 6** *Let $KB$ be a satisfiable Semantic Web knowledge base over DL-Lite$_{\mathcal{A}}$. Let $Q(\mathbf{x})$ be a (general) Semantic Web search query such that all existentially quantified variables occur only in safe positions, and let $\theta$ be a ground substitution for $\mathbf{x}$. Then, $\theta$ is an answer for $Q(\mathbf{x})$ to $KB$ iff $\theta$ is an answer for $Q^{+}(\mathbf{x})$ but not an answer for $Q^{-}(\mathbf{x})$ to the simple completion of $KB$.*

To obtain a sufficient syntactic condition for the safeness of positions relative to $KB$, we now define the notion of unsafe positions, which is based on modeling the propagation of existentially quantified entries in the Datalog$^{\pm}$ encoding of $KB$. Given a Semantic Web knowledge base $KB$ over *DL-Lite*$_{\mathcal{A}}$, we define the set of all *unsafe* positions relative to $KB$, denoted $U_{KB}$, inductively as follows:

- $U_{KB}$ contains (1) $P[2]$, (2) $P[1]$, (3) $U[2]$, (4) $P[2]$ and $A[1]$, and (5) $P[1]$ and $A[1]$ for every concept inclusion axiom $B \sqsubseteq C$ in $KB$ such that $C$ has the form (1) $\exists P$, (2) $\exists P^{-}$, (3) $\delta(U)$, (4) $\exists P.A$, and (5) $\exists P^{-}.A$, respectively;

- if $U_{KB}$ contains (1) $A_1[1]$, (2) $P_1[1]$, (3) $P_1[2]$, and (4) $U_1[1]$ for $B$ of the form (1) $A_1$, (2) $\exists P_1$, (3) $\exists P_1^{-}$, and (4) $\delta(U_1)$, respectively, then $U_{KB}$ also contains (5) $A_2[1]$, (6) $P_2[1]$, (7) $P_2[2]$, (8) $U_2[1]$, (9) $P_2[1]$, and (10) $P_2[2]$ for every concept inclusion axiom $B \sqsubseteq C$ in $KB$ such that $C$ has the form (5) $A_2$, (6) $\exists P_2$, (7) $\exists P_2^{-}$, (8) $\delta(U_2)$, (9) $\exists P_2.A_2$, and (10) $\exists P_2^{-}.A_2$, respectively;

- if $U_{KB}$ contains (1) $P_1[1]$ and (2) $P_1[2]$ for $Q$ of the form (1) $P_1$ and (2) $P_1^{-}$, respectively, then $U_{KB}$ also contains (3) $P_2[1]$ and (4) $P_2[2]$ for every role inclusion axiom $Q \sqsubseteq R$ in $KB$ such that $R$ has the form (3) $P_2$ and (4) $P_2^{-}$, respectively;

- if $U_{KB}$ contains (1) $P_1[1]$ and (2) $P_1[2]$ for $Q$ of the form (1) $P_1^{-}$ and (2) $P_1$, respectively, then $U_{KB}$ also contains (3) $P_2[2]$ and (4) $P_2[1]$ for every role inclusion axiom $Q \sqsubseteq R$ in $KB$ such that $R$ has the form (3) $P_2$ and (4) $P_2^{-}$, respectively;

- if $U_{KB}$ contains (1) $U_1[1]$ and (2) $U_1[2]$ then $U_{KB}$ also contains (1) $U_2[1]$ and (2) $U_2[2]$, respectively, for every role inclusion axiom $U_1 \sqsubseteq U_2$ in $KB$.

Informally, the set of all unsafe positions is a superset for the set of all positions that are not safe, since some of the assumed propagations may not actually occur, because the corresponding rule in the Datalog$^{\pm}$ program may be inactive due to missing data. This result is formally expressed by the following theorem.

**Theorem 7** *Let $KB$ be a satisfiable Semantic Web knowledge base over DL-Lite$_{\mathcal{A}}$. Then, any position that is not unsafe relative to $KB$ is safe relative to $KB$.*

The following result shows that deciding the satisfiability of Semantic Web knowledge bases and the logical consequence of ground atoms can be done in polynomial time in general and in LOGSPACE in the data complexity.

**Theorem 8** *Given a Semantic Web knowledge base $KB$ over DL-Lite$_{\mathcal{A}}$, deciding (a) whether $KB$ is satisfiable and (b) whether a given ground atom is in the simple completion of $KB$ can both be done in polynomial time in general and in* LOGSPACE *in the size of the ABox of $KB$ in the data complexity.*

The next result says that the size of the simple completion of every semantic annotation in a Semantic Web knowledge base $KB$ is quadratic in the size of $KB$ in general and linear in the size of the ABox of $KB$ in the data complexity.

**Theorem 9** *Let $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ be a satisfiable Semantic Web knowledge base over DL-Lite$_{\mathcal{A}}$, let $\mathbf{A}'$, $\mathbf{R}'_A$, $\mathbf{R}'_D$, $\mathbf{P}$, $\mathbf{O}$, and $\mathbf{V}'$ denote the sets of all atomic concepts, atomic roles, atomic attributes, Web pages, Web objects, and values that occur in $KB$, respectively, and let $KB' = (\emptyset, (\mathcal{A}'_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ be the simple completion of $KB$. Then, the size of every $\mathcal{A}'_a$ with $a \in \mathbf{P} \cup \mathbf{O}$ is in $O(|\mathbf{A}'| + |\mathbf{R}'_A| \cdot |\mathbf{P} \cup \mathbf{O}| + |\mathbf{R}'_D| \cdot |\mathbf{V}'|)$, i.e., it is quadratic in the size of $KB$ in general and linear in the size of the ABox of $KB$ in the data complexity.*

It thus follows immediately from the above two theorems that the simple completion of a Semantic Web knowledge base $KB$ has a cubic size in the size of $KB$ in general and a quadratic size in the size of the ABox of $KB$ in the data complexity, and that it can be computed in polynomial time in the size of $KB$. These results are more formally expressed by the following corollary.

**Corollary 10** *Let $KB$ be a Semantic Web knowledge base over DL-Lite$_{\mathcal{A}}$. Then, (a) the size of the simple completion $KB'$ of $KB$ is cubic in the size of $KB$ in general and quadratic in the size of the ABox of $KB$ in the data complexity, and (b) computing $KB'$ can be done in polynomial time in the size of $KB$.*

In summary, the simple completion of a Semantic Web knowledge base $KB$ has a cubic size in the size of $KB$ in general and a quadratic size in the size of the ABox of $KB$ in the data complexity, and it can be computed in polynomial time. Furthermore, the simple completion assures always a correct query processing, and also guarantees a complete query processing for a large class of Semantic Web search queries. Intuitively, query processing under the simple completion essentially corresponds to ignoring existentially quantified entries in the Semantic Web knowledge base that cannot be concretely instantiated by individuals or values, i.e., such query processing actually only results into a slightly different semantics of answers. For these reasons, and since completeness of query processing is actually not that much an issue in an inherently incomplete environment like the Web, we propose to use the simple completion as the basis of our Semantic Web search.

## 5.2 HTML Encoding

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 $a$.

Observe that this HTML encoding can be done in a way such that the atomic concepts, atomic roles, atomic attributes, individuals, and data values do not mix up with strings that occur on standard Web pages, e.g., by marking the HTML pages that are representing semantic annotations as such, and considering only such marked HTML pages during the online processing of Semantic Web search queries. Alternatively, one can also use a unique identifier (which does not occur elsewhere on the Web) for every ontology as a prefix in the encoding of atomic concepts, roles, and attributes, as well as individuals and data values.

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 $Q(x) = \mathit{Article}(x) \wedge \mathit{yearOfPublication}(x, 2008) \wedge \mathit{keyword}(x, \text{“}\mathit{RDF}\text{”})$ is translated into *Article* AND “*yearOf-Publication* 2008” AND “*keyword* RDF”. In this form, the query can be evaluated by standard Web search engines, since it is only a conjunction of a keyword and a phrase.

We rely on the assumption that each Web page and object $a \in \mathbf{P} \cup \mathbf{O}$ is associated with an identifier, which uniquely characterizes the individual. Here, we use the HTML address of the Web page's and object's annotation page as identifier. We employ the identifiers to evaluate complex queries involving more than one atomic concept, thus involving several annotations. For example, consider the query $Q(x)$ of Section 2 and the standard queries $Q_1 =$ *PhDStudent* AND *isAuthorOf* and $Q_2 =$ *Article* AND “*yearOfPublication* 2008” 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, which are the sets of annotation pages $\{i_2\}$ and $\{i_4\}$, respectively. We return the annotation page $p$ belonging to $r_1$ if there exists an annotation page in $r_2$ that occurs beside *isAuthorOf* on $p$. Since $i_4$ occurs beside *isAuthorOf* on the annotation page $i_2$, we thus return $i_2$ as overall query result.

## 6 Online Query Processing

We now define simple and safe Semantic Web search queries and describe how they can be reduced to collections of standard Web search queries, assuming that each completed semantic annotation of a Web page or object $a \in \mathbf{P} \cup \mathbf{O}$ is stored on an HTML page on the Web. We also show how the computation of the ObjectRank ranking can be reduced to the computation of the standard PageRank ranking.

### 6.1 Simple Search Queries

Semantic Web search queries that contain no equalities and only one free variable, which is the first argument in every atom, are called simple search queries.

**Definition 6** A Semantic Web search query is *simple* iff it has the form $Q(x) = \bigvee_{i=1}^{n} Q_{i,0}(x) \wedge \bigwedge_{j=1}^{n_i} \mathit{not}\; Q_{i,j}(x)$, where $x$ is a single variable from $\mathbf{X}$, and every $Q_{i,j}(x) = \bigwedge_{k=1}^{m_j} \phi_{i,j}^{k}$ with $i \in \{1, \ldots, n\}$ and $j \in \{0, \ldots, n_i\}$ is an equality-free conjunctive formula with $x$ as first argument in all atoms, i.e., either $\phi_{i,j}^{k} = p_{i,j}^{k}(x)$ or $\phi_{i,j}^{k} = p_{i,j}^{k}(x, t_{i,j}^{k})$, for all $i \in \{1, \ldots, n\}$ and $j \in \{0, \ldots, n_i\}$.

**Example 6** *(Scientific Database cont'd).* Query $Q_1(x)$ of Example 2 is simple.

Simple Semantic Web search queries can immediately be translated into exactly one variable-free Boolean keyword-based standard Web search query.

**Theorem 11** *Let $KB = (\emptyset, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ be a Semantic Web knowledge base. Let $Q(x)$ be a simple Semantic Web search query as in Definition 6. Then, the set of all answers for $Q(x)$ to KB is given by $\{\theta = \{x/a\} \mid a \in \bigcup_{i=1}^{n} I_{i,0} \setminus (\bigcup_{j=1}^{n_i} I_{i,j})\}$, where $I_{i,j} = \{a \in \mathbf{P} \cup \mathbf{O} \mid \forall k\colon p_{i,j}^{k}(a) \in \mathcal{A}_a$ or $p_{i,j}^{k}(a, t_{i,j}^{k}) \in \mathcal{A}_a\}$.*

**Example 7** *(Scientific Database cont'd).* Query $Q_1(x)$ of Example 2 can be translated into the following variable-free Boolean keyword-based Web search query:

$$\begin{aligned}&(\mathit{Scientist} \wedge \mathit{not}\ \mathit{doctoralDegree}(\text{“}\mathit{oxford\ university}\text{”}) \wedge \\ &\mathit{worksFor}(\text{“}\mathit{oxford\ university}\text{”})) \vee \\ &(\mathit{Scientist} \wedge \mathit{doctoralDegree}(\text{“}\mathit{oxford\ university}\text{”}) \wedge \\ &\mathit{not}\ \mathit{worksFor}(\text{“}\mathit{oxford\ university}\text{”}))\,.\end{aligned}$$

### 6.2 Safe Search Queries

Search queries where all free variables in negated conjunctive formulas and in equalities also occur in positive atoms are safe queries. That is, we connect the use of negation to a safeness condition, as usual in databases. They are reduced to collections of standard 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 7** 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 $\phi_i$ and freely in a negated conjunctive formula also occurs in a positive atom in $\phi_i$.

```
Algorithm SWSearch
Input: Semantic Web knowledge base $KB = (\emptyset, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$; safe (=-free) Semantic
  Web search query $Q(\mathbf{x}) = \bigvee_{i=1}^{n} \exists \mathbf{y}_i \, Q_i(\mathbf{x}, \mathbf{y}_i)$, where $Q_i(\mathbf{x}, \mathbf{y}_i) = Q_{i,0}(\mathbf{x}, \mathbf{y}_i) \wedge$
  $\bigwedge_{j=1}^{n_i} \mathit{not}\, Q_{i,j}(\mathbf{x}, \mathbf{y}_i)$ and the free variables of $Q_i(\mathbf{x}, \mathbf{y}_i)$ are among $\mathbf{x}, \mathbf{y}_i$.
Output: set $\Theta$ of all answers $\theta$ for $Q(\mathbf{x})$ to $KB$.
 1. $R := \emptyset$;
 2. for $i := 1$ to $n$ do begin
 3.   $R_{i,0} := \mathbf{PositiveSWSearch}(KB, Q_{i,0}(\mathbf{x}, \mathbf{y}_i))$;
 4.   for $j := 1$ to $n_i$ do begin
 5.     $R_{i,j} := \mathbf{PositiveSWSearch}(KB, Q_{i,j}(\mathbf{x}, \mathbf{y}_i))$;
 6.     $R_{i,0} := \{t \in R_{i,0} \mid \forall t_{i,j} \in R_{i,j} : t[R_{i,j}] \neq t_{i,j}\}$
 7.   end;
 8.   $R := R \cup \pi_{\mathbf{x}}(R_{i,0})$
 9. end;
10. return $R$.
```

Figure 4: Algorithm **SWSearch**.

**Example 8** *(Scientific Database cont'd). The following Semantic Web search queries ask for all students who do not attend at least one existing course (resp., event):*

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

*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 $Q_2(x)$.*

We now describe an algorithm for the online reduction of safe Semantic Web search queries $Q$ to standard Web search queries. Since such $Q$'s with equalities can easily be reduced to those without (via variable substitutions, if possible at all), we assume w.l.o.g. that $Q$ is equality-free. Furthermore, we assume w.l.o.g. that $Q$ contains no conjunctive subqueries. So, the algorithm reduces fully general but safe (and equality-free) Semantic Web search queries (without conjunctive subqueries) to several standard Web search queries.

Algorithm **SWSearch** in Fig. 4 takes as input a Semantic Web knowledge base $KB = (\emptyset, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ and a safe (equality-free) Semantic Web search query $Q(\mathbf{x})$, and it returns as output the set $\Theta$ of all answers $\theta$ for $Q(\mathbf{x})$ to $KB$. The main ideas behind it are informally described as follows. We first decompose the query $Q(\mathbf{x})$ into the positive subqueries $Q_{i,j}(\mathbf{x}, \mathbf{y}_i)$ with $i \in \{1, \ldots, n\}$ and $j \in \{0, \ldots, n_i\}$ whose free variables are among $\mathbf{x}$ and $\mathbf{y}_i$. Here, $Q_{i,0}(\mathbf{x}, \mathbf{y}_i)$ stands for the positive part of the $i$-th disjunct of $Q(\mathbf{x})$, while the $Q_{i,j}(\mathbf{x}, \mathbf{y}_i)$'s with $j > 0$ stand for the negative parts of the $i$-th disjunct of $Q(\mathbf{x})$. We then compute the answers for the positive subqueries $Q_{i,j}(\mathbf{x}, \mathbf{y}_i)$ via Algorithm **PositiveSWSearch** in Fig. 5 (lines 3 and 5). Thereafter, the result for the $i$-th disjunct of $Q(\mathbf{x})$ is computed by removing from the set of all answers for the positive part all tuples matching with an answer for one of the negative parts (line 6). Here, $t[R_{i,j}]$ denotes the restriction of the tuple $t$ to the attributes of the tuples in $R_{i,j}$. Finally, the overall result is computed by projecting the results for all disjuncts onto the free variables $\mathbf{x}$ of $Q(\mathbf{x})$ and unifying the resulting answer sets (line 8).

Algorithm **PositiveSWSearch** in Fig. 5 computes the set of all answers for positive Semantic Web search queries. It takes as input a Semantic Web knowledge base $KB = (\emptyset, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ and a positive (equality-free) Semantic Web search query $Q(\mathbf{x})$, and it returns as output the set $\Theta$ of all answers $\theta$ for $Q(\mathbf{x})$ to

```
Algorithm PositiveSWSearch

Input: Semantic Web knowledge base $KB = (\emptyset, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$; positive (=-free)
  Semantic Web search query $Q(\mathbf{x}) = \exists \mathbf{y} \bigwedge_{i=1}^{n} Q_i(\mathbf{x}, \mathbf{y})$, where $Q_i(\mathbf{x}, \mathbf{y}) = \bigwedge_{j=1}^{n_i} \phi_{i,j}$,
  for all $i \in \{1, \ldots, n\}$, with $\phi_{i,j} = p_{i,j}(t_i)$ or $\phi_{i,j} = p_{i,j}(t_i, t_{i,j})$, and the free
  variables of $Q_i(\mathbf{x}, \mathbf{y})$ are among $\mathbf{x}, \mathbf{y}$.

Output: set $\Theta$ of all answers $\theta$ for $Q(\mathbf{x})$ to $KB$.

1. for $i := 1$ to $n$ do begin
2.   if $t_i \in \mathbf{P} \cup \mathbf{O}$ then $I_i := \{t_i\}$
3.     else $I_i := \{a \in \mathbf{P} \cup \mathbf{O} \mid \exists \theta \forall j\colon p_{i,j}(a) \in \mathcal{A}_a$ or $p_{i,j}(a, t_{i,j}\theta) \in \mathcal{A}_a\}$;
4.   for each $a \in I_i$ do
5.     for $j := 1$ to $n_i$ do
6.       $R_{i,j}[t_i, t_{i,j}] := \{(a, t_{i,j}\theta) \mid p_{i,j}(a) \in \mathcal{A}_a$ or $p_{i,j}(a, t_{i,j}\theta) \in \mathcal{A}_a\}$
7. end;
8. return $\pi_{\mathbf{x}}(\bowtie_{i=1}^{n} \bowtie_{j=1}^{n_i} R_{i,j})$.
```

Figure 5: Algorithm **PositiveSWSearch**.

$KB$. We first decompose the query $Q(\mathbf{x})$ into the subqueries $Q_i(\mathbf{x}, \mathbf{y}) = \bigwedge_{j=1}^{n_i} \phi_{i,j}$, $i \in \{1, \ldots, n\}$, where $\phi_{i,j} = p_{i,j}(t_i)$ or $\phi_{i,j} = p_{i,j}(t_i, t_{i,j})$, whose free variables are among $\mathbf{x}$ and $\mathbf{y}$. Note that all atoms in such queries have the same term $t_i$ as first argument. We then collect the set $I_i$ of all matching Web pages and objects in $KB$ for $t_i$ as follows. If $t_i$ is already a Web page or object, then $I_i = \{t_i\}$ (line 2), and if $t_i$ is a variable, then we collect in $I_i$ all Web pages and objects $a$ with a matching semantic annotation $\mathcal{A}_a$ in $KB$ (line 3). These matching Web pages and objects in $I_i$ are then used in a look-up step on $KB$ to fill all the matching identifiers, identifier-value pairs, and identifier-identifier pairs from the semantic annotations $\mathcal{A}_a$ in $KB$ for all atomic concepts $A(t_i)$, attributes $U(t_i, t_{i,j})$, and roles $P(t_i, t_{i,j})$ in $Q_i(\mathbf{x}, \mathbf{y})$ into collections of unary and/or binary relations $A[t_i]$, $U[t_i, t_{i,j}]$, and $P[t_i, t_{i,j}]$, respectively (line 6). These relations are then joined via common variables, individuals, and values in $Q_i(\mathbf{x}, \mathbf{y})$, and finally projected to all free variables $\mathbf{x}$ in $Q(\mathbf{x})$ (line 8).

The operations in lines 3 and 6 of Algorithm **PositiveSWSearch** are realized by a standard Web search and by a look-up on the Web, respectively. In detail, recall that the semantic annotation $\mathcal{A}_a$ for every Web page and 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 $a$'s standard Web page, if $a$ is a Web page, and all standard Web pages mentioning $a$, if $a$ is a Web object. In addition, it contains 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 $KB \models P(a, b)$. Hence, the search in line 3 can be realized by searching for all the URIs whose pages contain all atomic concepts "$A$", attributes "$U$" (resp., "$U\ t_{i,j}$", if $t_{i,j}$ is a value), and roles "$P$" (resp., "$P\ t_{i,j}$", if $t_{i,j}$ is an identifier) such that $A(t_i)$, $U(t_i, t_{i,j})$, and $P(t_i, t_{i,j})$, respectively, occur in $Q_i(\mathbf{x}, \mathbf{y})$, while the operations in line 6 can be realized by look-ups under the given URIs, collecting all the matching data.

The following theorem shows that **SWSearch** and **PositiveSWSearch** in Figs. 4 and 5, respectively, are correct, i.e., they return the set of all answers for safe (and equality-free) general and positive Semantic Web search queries, respectively, to Semantic Web knowledge bases $KB = (\mathcal{T}, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ with $\mathcal{T} = \emptyset$. The theorem holds by the above textual explanations of the two algorithms.

**Theorem 12** *Let $KB = (\emptyset, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ be a Semantic Web knowledge base, and let $Q(\mathbf{x})$ be a safe (and* *equality-free) Semantic Web search query. Then, Algorithm* **SWSearch** *on $KB$ and $Q(\mathbf{x})$ returns the set of all answers for $Q(\mathbf{x})$ to $KB$.*

### 6.3 Ranking Answers

The following theorem shows that computing the ObjectRank ranking can be reduced to computing the PageRank ranking. That is, using the encoding of semantic annotations as HTML pages on the Web, the ObjectRank of all Web pages and objects is given by the PageRank of their HTML pages on the Web.

**Theorem 13** *Let $KB = (\emptyset, (\mathcal{A}_a)_{a \in \mathbf{P} \cup \mathbf{O}})$ be a Semantic Web knowledge base, let $E$ be a source of rank, and let $d$ be a damping factor. Let the directed graph $G_{KB} = (V, E)$ be defined by $V = \mathbf{P} \cup \mathbf{O}$ and $(u, v) \in E$ iff $P(u, v) \in \mathcal{A}_u$. Then, for every $u \in \mathbf{P} \cup \mathbf{O}$, the ObjectRank of $u$ relative to $KB$ is the PageRank of $u$ relative to $G_{KB}$.*

## 7 Implementation

We have implemented two prototypes (the one described in [27] and a new one) of Serene for a semantic desktop search engine (in desktop search, it was possible to quickly index 500,000 facts at once). In the following, we report on the new prototype implementation in the context of this paper. The implementation is based on the above offline inference technique and a (slightly more sophisticated) desktop version of the above online Semantic Web search (by reduction to standard Web search). The former uses the deductive database system DLV [41], while the latter is written in C# (nearly 2 000 lines of code) and uses Microsoft Windows Desktop Search 3.0 (WDS) as external desktop search engine. More concretely, it uses the search index created by WDS, which is queried using an OLE DB connection and an SQL-like syntax; the template used for an index query is as follows (as described in the Windows Search 3x SDK released by Microsoft):

```
SELECT System.ItemName
FROM SystemIndex
WHERE freetext('⟨positivePart⟩') [AND NOT freetext('⟨negativePart⟩')]⋆
```

Here, as usual, the "$\star$" means that the portion enclosed in squared brackets is optional and may be repeated as needed.

The prototype uses a slightly more sophisticated version of the two algorithms in Section 6, yielding improved performances. In fact, computationally, the most expensive operations are index queries and the look-up phase. The former are queries over the index generated by WDS, aimed at finding all the resources containing a set of keywords, while the look-up phase is the scanning (i.e., parsing) of annotations, needed to search for repeated variables in different annotations to verify the (roles of the) query. Algorithm **SWSearch**, in general, recalls **PositiveSWSearch** a huge and unnecessary number of times without a precise order, which causes a huge and unnecessary number of index queries. Furthermore, **PositiveSWSearch**, after querying the index, during the look-up phase, performs an exhaustive search of the found annotations, scanning a huge and unnecessary number of annotations. To avoid such a wasting of time, we slightly changed the two algorithms.

The implemented algorithm first decomposes a given query $Q(\mathbf{x})$ into the $n$ subqueries $Q_i(\mathbf{x}, \mathbf{y}) = \bigwedge_{j=1}^{n_i} \phi_{i,j}$, $i \in \{1, \ldots, n\}$, where $\phi_{i,j} = p_{i,j}(t_i)$ or $\phi_{i,j} = p_{i,j}(t_i, t_{i,j})$, whose free variables are among $\mathbf{x}$ and $\mathbf{y}$, in the same way as in Algorithm **PositiveSWSearch**. Note that also here, all atoms in such queries have the same term $t_i$ as first argument. Then, it divides these subqueries into $Q_a$, containing the subqueries about concepts for which may be specified attributes but no roles, and $Q_b$, containing the other subqueries, i.e., those about concepts for which at least one role is specified. Formally, $Q_a$ is the set of all $Q_i$ such that $i \in \{1, \ldots, n\}$ and no $p_{i,j}$ with $j \in \{1, \ldots, n_i\}$ is an atomic role, and $Q_b = \{Q_1, \ldots, Q_n\} \setminus Q_a$.

Now we can start with the subqueries of $Q_a$, executing index queries that require a computationally cheap look-up phase: in fact, these queries can be realized by searching for all the annotations containing all atomic concepts "$A$" and attributes "$U$" (resp., "$U\ v$", if $v$ is a value) such that $A(t_i)$ and $U(t_i, v)$, respectively, occur in a $Q_i(\mathbf{x}, \mathbf{y})$, thus obtaining one set of resulting annotations for each executed query. Then, we sort this set on the basis of the sizes of its elements, and we scan it: for each set element, from resulting annotations, we extract the URI of the corresponding Web page, updating, for each annotation the sets of positive and negative URIs, depending on $A$ (i.e., if it is positive or not). At the end of this stage, we thus have a partial result with a candidate resulting set of URIs (i.e., Web pages) and/or forbidden URIs (i.e., those extracted from resulting annotations corresponding to negated concepts of $Q_a$).

Regarding the subqueries of $Q_b$, to avoid useless scanning of annotations, we define the directed graph of $Q_b$ as $(V, E)$, where $V$ is the set of all concepts that are in $Q_b$, and $(id_1, id_2) \in E$ iff $Q_b$ contains some $p(id_1, id_2)$ such that $p$ is an atomic role, and $id_1$ and $id_2$ are concept identifiers. Here, we assume that the above directed graph is acyclic, but all the algorithms can be easily extended to the cyclic case by adding a preprocessing step, removing some roles and thus the cycles in the query, and a verification step at the end. Then, exploiting the graph, we make a new decomposition of the queries of $Q_b$ as follows: component $Q_{b,1}$ contains the nodes without ingoing edges, and every component $Q_{b,i}$ with $i \geq 2$ contains the nodes with ingoing edges starting from nodes of $Q_{b,i-1}$.

First, we execute index queries for $Q_{b,1}$ as described above for the subqueries of $Q_a$, with the only difference that now the annotations must also contain roles "$P$" such that $P(id_1, id_2)$ occur in $Q_{b,1}$. Again, we obtain a set of resulting annotations, which we can sort on the basis of the sizes of its elements. We scan this set and for each set element and resulting annotation, we extract the URI of the corresponding Web page and store it for the next step iff such URI is between the candidate ones (and/or not between the forbidden ones); if it is the case, we look up the resulting annotations to find the aforementioned $id_2$'s of the $P(id_1, id_2)$'s of $Q_{b,1}$.

Then, we update the sets of candidate URIs and forbidden ones, on the basis of the resulting annotations found, execute index queries for $Q_{b,2}$ and look up the resulting annotations in order to find identifiers matching with the $id_2$'s. If a resulting annotation has an identifier that matches with one of the $id_2$'s, we store it for the next step and update the sets of candidate URIs and forbidden ones. We proceed, in turn, in a similar way for the other partitions $Q_{b,i}$.

In this way, we improve the performance of the two algorithms, because:

- we execute a reduced set of index queries;
- we execute a complete scan of an annotation iff it can lead to a resulting Web page or to a forbidden one;
- we prune the search space whenever possible, updating the sets of candidate URIs and forbidden ones at each step.

## 8 Experimental Results

In this section, we report on our experimental results with the two prototype implementations for a semantic desktop search engine (the prototype implementation described in [27] and the new one), namely, on the size of completed annotations, the running time of the online query processing step, and the precision and the recall of our approach to Semantic Web search compared to Google.

## 8.1 Size of Completed Annotations

By Theorem 9, given a Semantic Web knowledge base $KB$, the (worst-case) size of every generated completed semantic annotation of $KB$ is in $O(|\mathbf{A}'| + |\mathbf{R}'_A| \cdot |\mathbf{P} \cup \mathbf{O}| + |\mathbf{R}'_D| \cdot |\mathbf{V}'|)$ (where $\mathbf{A}'$, $\mathbf{R}'_A$, $\mathbf{R}'_D$, $\mathbf{P}$, $\mathbf{O}$, and $\mathbf{V}'$ denote the sets of all atomic concepts, atomic roles, atomic attributes, Web pages, Web objects, and values that occur in $KB$, respectively), i.e., it is quadratic in the size of $KB$ in general and linear in the size of the ABox of $KB$ in the data complexity.

In practice, since ontological hierarchies are generally not that deep (intuitively, a concept/role/attribute has generally at most a dozen superconcepts/-roles/-attributes), the generated completed semantic annotations are generally even much smaller. To prove this experimentally, we have measured the size of the generated completed annotations for some commonly used and standard benchmark ontologies, namely, for the Adolena, Buildings & Places, Cell, DOLCE-$Lite$, Pathway, Pizza, and Zebrafish ontologies from the TONES Repository[6], for the Finite-State-Machine (FSM), New-Testament-Names (NTN), Science, and Surface-Water-Model (SWM) ontologies from the Protégé Ontology Library[7], for the Stock Exchange and Vicodi ontologies [47], for the Finance ontology[8], for the Lehigh University Benchmark (LUBM) ontology[9], for the SWETO ontology[10], for the Uniprot (core) ontology[11], and for the University Ontology Benchmark (UOBM) ontology [44]. We have computed the completed annotations either for individuals that are already included in the above ontologies, where such (rather realistic) individuals are available (in the majority of cases), or for artificially created individuals, otherwise. Indeed, the experimental results in Table 1 show that the maximal and average sizes of completed annotations (i.e., the maximal and average numbers of all ABox axioms, denoted *Max Comp* and *Avg Comp*, respectively) are rather small. Table 1 also provides for every ontology the DL expressivity (i.e., the underlying description logic), the size (i.e., the number of all TBox and ABox axioms), the maximal and average depths of property (i.e., role or attribute) hierarchies (denoted *Max DPH* and *Avg DPH*, respectively), as well as the maximal and average depths of concept hierarchies (denoted *Max DCH* and *Avg DCH*, respectively).

## 8.2 Efficiency of Online Query Processing

Experiments with our two implemented semantic desktop search engines (the implementation described in [27] and the new one) show the principle feasibility of our approach, and that it scales quite well to very large collections of standard pages, annotation pages, and background ontologies. The results are summarized in Table 2, which shows in bold the total time (in ms) used by our new system (including the WDS calls) on a standard laptop for processing ten different search queries ($Q_1, \ldots, Q_{10}$) on a randomly generated knowledge base (in the context of the running Scientific Database), consisting of 5 000 annotations with 590 027 facts. Notice that this total time (for the decomposition of the query, for processing all subqueries via WDS, and for the composition of the query results) is very small (all below one second). Table 2 also shows the different numbers of returned pages and objects. Observe that our new prototype is on the average

[6] http://owl.cs.manchester.ac.uk/repository/

[7] http://protegewiki.stanford.edu/index.php/Protege_Ontology_Library

[8] http://www.cs.put.poznan.pl/alawrynowicz/financial.owl

[9] http://swat.cse.lehigh.edu/projects/lubm/

[10] http://knoesis.wright.edu/library/ontologies/sweto/

[11] http://dev.isb-sib.ch/projects/uniprot-rdf/owl/

Table 1: Maximal and average sizes of completed annotations for different ontologies.

| Ontology | DL Expressivity | Size | Max DPH | Avg DPH | Max DCH | Avg DCH | Max Comp | Avg Comp |
|---|---|---|---|---|---|---|---|---|
| Adolena | $\mathcal{SHI}$(**D**) | 418 | 1 | 0.12 | 9 | 5.1 | 12 | 5.34 |
| Buildings & Places | $\mathcal{ALCHIO}$(**D**) | 3635 | 1 | 0.05 | 6 | 2.6 | 213 | 9.37 |
| Cell | $\mathcal{EL}^{++}$ | 4184 | 0 | 0 | 15 | 9.3 | 42 | 4.84 |
| DOLCE-$Lite$ | $\mathcal{SHIF}$ | 538 | 4 | 2.24 | 6 | 4.7 | 10 | 3.47 |
| Finance | $\mathcal{ALCHIF}$ | 3204 | 3 | 1.83 | 10 | 4.2 | 40 | 6.70 |
| FSM | $\mathcal{SF}$(**D**) | 294 | 0 | 0 | 4 | 2.7 | 1 | 0.29 |
| LUBM | $\mathcal{ALEHI}+$ | 246 | 2 | 0.17 | 5 | 3 | 3 | 1.74 |
| NTN | $\mathcal{SHIF}$(**D**) | 416 | 1 | 0.35 | 6 | 4 | 2 | 1.50 |
| Pathway | $\mathcal{EL}$ | 2500 | 0 | 0 | 8 | 4.8 | 2 | 1.06 |
| Pizza | $\mathcal{SHOIN}$ | 1006 | 1 | 0.66 | 8 | 5.3 | 18 | 5.24 |
| Science | $\mathcal{ALCIF}$ | 2217 | 0 | 0 | 5 | 2.7 | 5 | 2.09 |
| Stock Exchange | $\mathcal{ALCI}$ | 81 | 0 | 0 | 4 | 2.1 | 6 | 3.3 |
| SWETO | $\mathcal{ALH}$(**D**) | 736 | 2 | 0.02 | 4 | 3.5 | 2 | 1.38 |
| SWM | $\mathcal{ALCOF}$(**D**) | 647 | 0 | 0 | 5 | 2.2 | 4 | 1.71 |
| Uniprot (core) | $\mathcal{ALCHIOF}$(**D**) | 1293 | 1 | 0.12 | 4 | 2.7 | 8 | 2.53 |
| UOBM | $\mathcal{SHIN}$(**D**) | 501 | 2 | 0.26 | 5 | 3.1 | 4 | 1.54 |
| Vicodi | RDFS(DL) | 426 | 2 | 1.57 | 8 | 5.8 | 2 | 1.05 |
| Zebrafish | $\mathcal{S}$ | 19302 | 0 | 0 | 11 | 4.0 | 181 | 50.59 |

*more than 130 times quicker* than our previous one described in [27]. This performance increase of the new over the old prototype is due to several code optimizations, including the direct use of the API of WDS for querying the search index created by WDS, rather than a cmdlet script in Microsoft Powershell 1.0. Further dramatic reductions (even with much larger datasets) can be achieved by employing a Web search engine (such as Google) rather than a desktop search engine (since Web search is actually much faster, even with a much larger search space, especially because it uses a huge number of processors, differently from desktop search).

The ten search queries $Q_1, \ldots, Q_{10}$ 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 $c_{12}$:

$$Q_1(x) = \mathit{Professor}(x) \wedge \mathit{teacherOf}(x, c_{12})\,;$$

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

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

Table 2: Total time used (in ms) and number of returned URIs for processing the ten Semantic Web search queries $Q_1, \ldots, Q_{10}$ on a randomly generated Semantic Web knowledge base with 5000 semantic annotations and 590270 facts.

| Query | Total Time (ms) | | No. URIs |
|---|---|---|---|
| | FoIKS-2010 Prototype | New Prototype | |
| $Q_1(x)$ | 12123 | **204** | 613 |
| $Q_2(x)$ | 5893 | **27** | 116 |
| $Q_3(x)$ | 20858 | **153** | 582 |
| $Q_4(x)$ | 14592 | **91** | 529 |
| $Q_5(x)$ | 23001 | **521** | 679 |
| $Q_6(x)$ | 16264 | **220** | 204 |
| $Q_7(x)$ | 43847 | **976** | 687 |
| $Q_8(x)$ | 4979 | **10** | 20 |
| $Q_9(x)$ | 38971 | **870** | 687 |
| $Q_{10}(x)$ | 54403 | **884** | 671 |

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

$$\begin{aligned} Q_3(x) = {} & (\mathit{Scientist}(x) \wedge \mathit{worksFor}(x, o_{12}) \wedge \mathit{hasWritten}(x, a_4)) \vee \\ & (\mathit{Scientist}(x) \wedge \mathit{worksFor}(x, o_3) \wedge \mathit{hasWritten}(x, a_{25}))\,; \end{aligned}$$

(4) scientists working for $u$ but not having a doctoral degree from $u$, or scientists having a doctoral degree from $u$ but not working for $u$:

$$\begin{aligned} Q_4(x) = {} & (\mathit{Scientist}(x) \wedge \mathit{worksFor}(x, u) \wedge \mathit{not\ doctoralDegree}(x, u)) \vee \\ & (\mathit{Scientist}(x) \wedge \mathit{doctoralDegree}(x, u) \wedge \mathit{not\ worksFor}(x, u))\,; \end{aligned}$$

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

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

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

$$\begin{aligned} Q_6(x) = {} & \exists y\, (\mathit{Article}(x) \wedge \mathit{yearOfPublication}(x, 2007) \wedge \mathit{hasWritten}(y, x) \wedge \\ & \mathit{Scientist}(y) \wedge \mathit{nationality}(y, \mathit{italian}))\,; \end{aligned}$$

(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:

$$\begin{aligned} Q_7(x) = {} & \exists y, z\, (\mathit{Scientist}(x) \wedge \mathit{hasWritten}(x, y) \wedge \mathit{JournalPaper}(y) \wedge \\ & \mathit{yearOfPublication}(y, 2007) \wedge \mathit{hasWritten}(x, z) \wedge \mathit{ConferencePaper}(z) \wedge \\ & \mathit{yearOfPublication}(z, 2007)) \vee \exists y\, (\mathit{Scientist}(x) \wedge \mathit{hasWritten}(x, y) \wedge \\ & \mathit{Book}(y) \wedge \mathit{yearOfPublication}(y, 2007))\,; \end{aligned}$$

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

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

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

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

(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:

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

## 8.3 Efficiency Comparison to the Corese System

We now compare the running time of query processing in our new prototype with the running time of query processing in the Corese system [17], which is the Semantic Web search system in the literature that is most closely related to our approach. It turns out that our new prototype is on the average nearly 18 times quicker than Corese. Note that this difference in the query processing time is partially due to the fact that Corese's ontological inference is performed online (i.e., at query processing time), while the ontological inference in our approach is done offline. The detailed results are summarized in Table 3, which shows the total time (in ms) used by Corese and by our new prototype on a standard laptop for processing ten different search queries ($Q_1, \ldots, Q_{10}$) on a randomly generated knowledge base (in the context of the running Scientific Database), consisting of 5580 annotations with 33519 facts. In detail, the ten search queries $Q_1, \ldots, Q_{10}$ are given as follows:

(1) female employees:

$$Q_1(x) = \mathit{Woman}(x) \wedge \mathit{work}(x, \mathit{employee})\ ;$$

(2) female physicians who are married:

$$Q_2(x) = \exists y\, (\mathit{Woman}(x) \wedge \mathit{work}(x, \mathit{physician}) \wedge \mathit{hasSpouse}(x, y))\ ;$$

(3) persons who are not married and have no friends

$$Q_3(x) = \mathit{Person}(x) \wedge \mathit{not}\ \exists y\, \mathit{hasSpouse}(x, y) \wedge \mathit{not}\ \exists y\, \mathit{hasFriend}(x, y)\ ;$$

(4) employees or teachers:

$$Q_4(x) = (\mathit{Person}(x) \wedge \mathit{work}(x, \mathit{employee})) \vee (\mathit{Person}(x) \wedge \mathit{work}(x, \mathit{teacher}))\ ;$$

Table 3: Total time used (in ms) by Corese and by our new prototype, along with the number of returned URIs, for processing the Semantic Web search queries $Q_1, \ldots, Q_{10}$ on a randomly generated Semantic Web knowledge base with 5580 semantic annotations and 33519 facts.

| Query | Total Time (ms) | | No. URIs |
|---|---|---|---|
| | Corese | New Prototype | |
| $Q_1(x)$ | 531 | **115** | 946 |
| $Q_2(x)$ | 420 | **43** | 313 |
| $Q_3(x)$ | 581 | **226** | 1942 |
| $Q_4(x)$ | 395 | **225** | 1896 |
| $Q_5(x)$ | 402 | **76** | 613 |
| $Q_6(x)$ | 391 | **45** | 335 |
| $Q_7(x)$ | 336 | **4** | 7 |
| $Q_8(x)$ | 556 | **209** | 1252 |
| $Q_9(x)$ | 521 | **10** | 32 |
| $Q_{10}(x)$ | 557 | **155** | 970 |

(5) non-married female employees:

$$Q_5(x) = \mathit{Woman}(x) \wedge \mathit{work}(x, \mathit{employee}) \wedge \mathit{not}\ \exists y\, \mathit{hasSpouse}(x, y)\,;$$

(6) female employees without friends:

$$Q_6(x) = \mathit{Woman}(x) \wedge \mathit{work}(x, \mathit{employee}) \wedge \mathit{not}\ \exists y\, \mathit{hasFriend}(x, y)\,;$$

(7) Bettina's friends:

$$Q_7(x) = \mathit{Person}(x) \wedge \mathit{hasFriend}(x, \mathit{Bettina})\,;$$

(8) married persons who have friends and work, but not as employees:

$$Q_8(x) = \exists y, z, z'\ (\mathit{Person}(x) \wedge \mathit{hasSpouse}(x, y) \wedge \mathit{hasFriend}(x, z) \wedge \mathit{work}(y, z') \wedge \mathit{not}\ \mathit{work}(y, \mathit{employee}))\,;$$

(9) married men who are 40 years old:

$$Q_9(x) = \exists y\ (\mathit{Man}(x) \wedge \mathit{age}(x, 40) \wedge \mathit{hasSpouse}(x, y))\,;$$

(10) men who work and have no friends:

$$Q_{10}(x) = \exists y\ (\mathit{Man}(x) \wedge \mathit{work}(x, y) \wedge \mathit{not}\ \exists z\, \mathit{hasFriend}(x, z))\,.$$

Note that Corese has its own query syntax, which is based on RDF(S). For example, the search query $Q_1(x)$ is expressed by the following query in Corese:

```
select ?x display xml
where {?x rdf:type animals:Woman .
```

```
  ?x animals:work ?y
  FILTER (?y='employee')}
```

## 8.4 Precision and Recall of Semantic Web Search

Differently from conventional Boolean keyword-oriented Web search, the proposed Semantic Web search clearly empowers the user to precisely describe her information need for certain kinds of queries, resulting in a very precise result set and a very high precision and recall [2] for the query result. In particular, in many cases, Semantic Web search queries exactly describe the desired answer sets, resulting into a precision and a recall of 1. Some examples of such Semantic Web search queries (addressed to the CIA World Fact Book[12] relative to the WORLD-FACT-BOOK ontology[13]), which have a precision and a recall of 1 in our approach to Semantic Web search, are shown below, along with corresponding Google queries:

(1) countries having a common border with Austria:
$Q_1(x) = \mathit{Country}(x) \wedge \mathit{borderCountries}(x, \mathit{Austria})$,
″border countries″ Austria ;

(2) countries having Bulgaria as exports partners:
$Q_2(x) = \mathit{Country}(x) \wedge \mathit{exportsPartners}(x, \mathit{Bulgaria})$,
″exports - partners″ Bulgaria ;

(3) countries in which Italian is spoken:
$Q_3(x) = \mathit{Country}(x) \wedge \mathit{languages}(x, \mathit{Italian})$,
languages Italian ;

(4) countries importing tobacco:
$Q_4(x) = \mathit{Country}(x) \wedge \mathit{importsCommodities}(x, \mathit{tobacco})$,
″imports - commodities″ tobacco ;

(5) countries exporting tobacco and in which French is spoken:
$Q_5(x) = \mathit{Country}(x) \wedge \mathit{exportsCommodities}(x, \mathit{tobacco}) \wedge \mathit{languages}(x, \mathit{French})$,
″exports - commodities″ tobacco languages French ;

(6) countries in which Italian is not spoken:
$Q_6(x) = \mathit{Country}(x) \wedge \mathit{not}\ \mathit{languages}(x, \mathit{Italian})$,
languages -Italian ;

(7) countries in which Arabic is spoken:
$Q_7(x) = \mathit{Country}(x) \wedge \mathit{languages}(x, \mathit{Arabic})$,
languages Arabic ;

[12]http://www.cia.gov/library/publications/the-world-factbook/
[13]http://www.ontoknowledge.org/oil/case-studies/

(8) countries in which Arabic is spoken and not English:

$Q_8(x) = \mathit{Country}(x) \wedge \mathit{languages}(x, \mathit{Arabic}) \wedge \mathit{not}\ \mathit{languages}(x, \mathit{English})$,
languages Arabic -English ;

(9) countries importing tobacco and food:

$Q_9(x) = \mathit{Country}(x) \wedge \mathit{importsCommodities}(x, \mathit{tobacco}) \wedge$
$\mathit{importsCommodities}(x, \mathit{food})$,
$''$imports - commodities$''$ tobacco food ;

(10) countries importing tobacco and not food:

$Q_{10}(x) = \mathit{Country}(x) \wedge \mathit{importsCommodities}(x, \mathit{tobacco}) \wedge$
$\mathit{not}\ \mathit{importsCommodities}(x, \mathit{food})$,
$''$imports - commodities$''$ tobacco -food .

The precision and the recall of the above ten Google queries compared to their Semantic Web search queries are shown in Table 4. Observe that the Google queries often cannot that precisely describe the desired answer sets, and are thus often resulting into a precision and a recall much below 1. Note that this lower precision and recall is due to the limited expressivity of Google queries, and not due to some incomplete indexing; this is especially obvious for queries that use the negation: Google's recall is always 1 in positive queries, and it is less than 1 in queries containing at least one negated predicate having a keyword as value; the lower recall in the latter case is because Google discards all the pages containing the keyword, including those where the keyword does not refer to the specified predicate, e.g., when processing the query "languages -Italian", Google discards all the pages containing "Italian", including those where "Italian" does not refer to "languages"; however, Google's precision is often 1 for such queries with negated predicates, since all the returned pages are in general also answers to the queries.

## 9 Semantic Web Search on the Internet Movie Database

In this section, we show that our approach to Semantic Web search can be readily applied to existing Web pages, even if they are currently not yet semantically annotated. More specifically, we show how our approach can be used to perform a vertical ontology-based search on the Web pages of the Internet Movie Database (IMDB)[14]. To this end, the necessary semantic annotations are automatically constructed from the IMDB Web pages. That is, we are actually mapping the IMDB Web pages into an ontological knowledge base, which then allows for processing Semantic Web search queries in the query language of the underlying ontology. Intuitively, such an ontological knowledge base can be considered as an ontological index over the IMDB, against which ontological conjunctive search queries on the IMDB can be answered. So, our vertical search on the IMDB works on an ontologically structured copy of the IMDB without actually changing it.

We considered a sample set of more than 60 000 movies and actors of the IMDB. To avoid a manual annotation of Web pages, we automatically extracted the annotations for all the movies and actors in our sample set via the wrapper *SCRAP* [26].

SCRAP is able to extract pieces of information from HTML pages and to reorganize them into new XML documents. To accomplish this task, SCRAP uses a set of *extraction rules* and an *extraction schema*. The extraction rules are XPath expressions identifying the portion of the HTML pages to be extracted, while the

[14]http://www.imdb.com

Table 4: Precision and recall of Google vs. Semantic Web search (SWS).

| **Query** | Results Google | Correct Results | Correct Results Google | **Precision Google** | **Recall Google** | **Precision SWS** | **Recall SWS** |
|---|---|---|---|---|---|---|---|
| $Q_1(x)$ | 17 | 8 | 8 | 0.47 | 1 | 1 | 1 |
| $Q_2(x)$ | 19 | 5 | 5 | 0.26 | 1 | 1 | 1 |
| $Q_3(x)$ | 21 | 13 | 13 | 0.62 | 1 | 1 | 1 |
| $Q_4(x)$ | 51 | 10 | 10 | 0.2 | 1 | 1 | 1 |
| $Q_5(x)$ | 24 | 4 | 4 | 0.17 | 1 | 1 | 1 |
| $Q_6(x)$ | 229 | 253 | 229 | 1 | 0.91 | 1 | 1 |
| $Q_7(x)$ | 33 | 32 | 32 | 0.97 | 1 | 1 | 1 |
| $Q_8(x)$ | 11 | 13 | 11 | 1 | 0.85 | 1 | 1 |
| $Q_9(x)$ | 45 | 7 | 7 | 0.16 | 1 | 1 | 1 |
| $Q_{10}(x)$ | 6 | 3 | 1 | 0.17 | 0.33 | 1 | 1 |

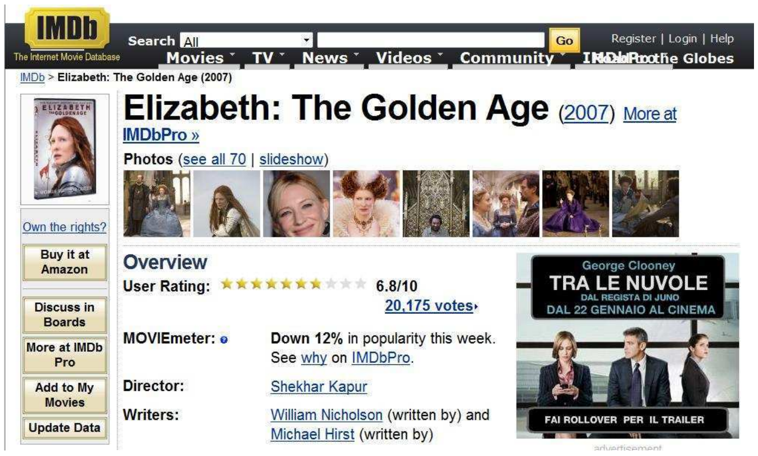
extraction schema is a document type definition (DTD) specifying the structure of the output XML document and associating each element type with an extraction rule. Thus, an extraction rule is used to define the path in the input HTML page that locates the text to be returned as the content of an XML element. In our context, this feature will be exploited for the purpose of annotation by using XML tag names that describe the semantics of the pieces of information extracted.

The process of generating the annotations was divided into the following two phases. In the first phase, by using SCRAP, we extracted the information of the HTML pages to be inserted in the annotations (such as movie titles, actor names, etc.), and returned it in the form of XML documents conforming to the specified DTD. In the second phase, we then further processed these XML documents and translated them into documents conforming to our annotation syntax.

In more detail, in the first phase, we used a visual tool provided by SCRAP to define the schemas for extracting the information about movies and actors. In particular, the extraction schema for movies consisted of the following DTD:

```
<!ELEMENT doc (Movie)>
<!ELEMENT Movie (title,director*,creator*,writer*,genre*,
  language*,country?,releaseDate?,awards?,star*)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT director (#PCDATA)>
<!ELEMENT creator (#PCDATA)>
<!ELEMENT genre (#PCDATA)>
<!ELEMENT writer (#PCDATA)>
<!ELEMENT star (#PCDATA)>
<!ELEMENT country (#PCDATA)>
<!ELEMENT releaseDate (#PCDATA)>
<!ELEMENT awards (#PCDATA)>
<!ELEMENT language (#PCDATA)>
```

The semantics of this extraction schema is that, for every HTML page describing a movie, an XML



Elizabeth: The Golden Age (2007) More at IMDbPro »

Photos (see all 70 | slideshow)

Overview

User Rating: 6.8/10 20,175 votes

MOVIEmeter: Down 12% in popularity this week. See why on IMDbPro.

Director: Shekhar Kapur

Writers: William Nicholson (written by) and Michael Hirst (written by)

Figure 6: An excerpt of a movie page.

document must be generated containing the title of the movie, the director, the country, the names of all the actors of the movie, etc. By means of an analogous extraction schema, suitable for every HTML page describing an actor, we specified that an XML document must be generated which contains the name of the actor, the year of birth, the city of birth, the titles of all the movies starring her, etc. The tag names and the structure of the extraction schema were defined according to the IMDB ontology[15]. Exploiting SCRAP facilities, we also associated each element type of the extraction schemas with an appropriate extraction rule. For instance, we associated the element type `title` of the above-reported DTD with the following XPath extraction rule: `html/head/title/text()`, which captures the text contained in the specified path of the HTML pages.

After defining the extraction schemas and the associated set of extraction rules, we ran SCRAP on the sample set from the IMDB, thus obtaining a set of XML documents, each containing the desired information about a movie or an actor. For instance, consider the HTML page shown in Fig. 6. The XML document obtained by running SCRAP on it is shown in Fig. 7.

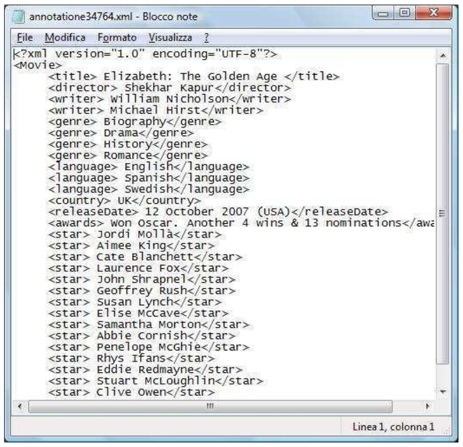
In the second phase, we took each XML document returned by SCRAP in the first phase, and automatically transformed it into a final annotation, conforming to our syntax. The final annotation obtained from the HTML of Fig. 6 is shown in Fig. 8.

We point out that, since all the HTML pages describing movies in the IMDB site share the same structure, the set of extraction rules was defined looking only at a sample page, and then used to support the extraction from all the other pages. The same strategy was used for actor pages. This way, we dramatically reduced the overall human effort needed for the annotation task. In fact, the only human work to be done was the definition of extraction schemas and rules, thus not requiring any user to be called for annotating HTML pages one by one.

The following are some Semantic Web search queries, which we processed in our Semantic Web search interface for the IMDB:

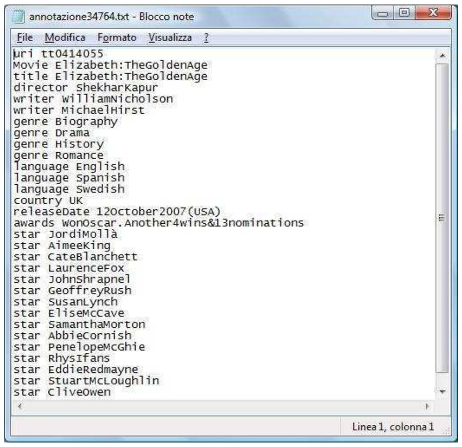
(1) comedy movies in English that were nominated for the Oscar:

[15]http://www.csd.abdn.ac.uk/∼ggrimnes/dev/imdb/IMDB.rdfs


annotatione34764.xml - Blocco note
File Modifica Formato Visualizza ?
Linea 1, colonna 1

```
<?xml version="1.0" encoding="UTF-8"?>
<Movie>
     <title> Elizabeth: The Golden Age </title>
     <director> Shekhar Kapur</director>
     <writer> William Nicholson</writer>
     <writer> Michael Hirst</writer>
     <genre> Biography</genre>
     <genre> Drama</genre>
     <genre> History</genre>
     <genre> Romance</genre>
     <language> English</language>
     <language> Spanish</language>
     <language> Swedish</language>
     <country> UK</country>
     <releaseDate> 12 October 2007 (USA)</releaseDate>
     <awards> Won Oscar. Another 4 wins & 13 nominations</awa
     <star> Jordi Mollà</star>
     <star> Aimee King</star>
     <star> Cate Blanchett</star>
     <star> Laurence Fox</star>
     <star> John Shrapnel</star>
     <star> Geoffrey Rush</star>
     <star> Susan Lynch</star>
     <star> Elise McCave</star>
     <star> Samantha Morton</star>
     <star> Abbie Cornish</star>
     <star> Penelope McGhie</star>
     <star> Rhys Ifans</star>
     <star> Eddie Redmayne</star>
     <star> Stuart McLoughlin</star>
     <star> Clive Owen</star>
```

Figure 7: XML document output of SCRAP.


annotazione34764.txt - Blocco note
File Modifica Formato Visualizza ?
Linea 1, colonna 1

```
uri tt0414055
Movie Elizabeth:TheGoldenAge
title Elizabeth:TheGoldenAge
director ShekharKapur
writer WilliamNicholson
writer MichaelHirst
genre Biography
genre Drama
genre History
genre Romance
language English
language Spanish
language Swedish
country UK
releaseDate 12October2007(USA)
awards WonOscar.Another4wins&13nominations
star JordiMollà
star AimeeKing
star CateBlanchett
star LaurenceFox
star JohnShrapnel
star GeoffreyRush
star SusanLynch
star EliseMcCave
star SamanthaMorton
star AbbieCornish
star PenelopeMcGhie
star RhysIfans
star EddieRedmayne
star StuartMcLoughlin
star CliveOwen
```

Figure 8: Final annotation.

$$Q_1(x) = \mathit{Movie}(x) \wedge \mathit{genre}(x, \mathit{Comedy}) \wedge \mathit{language}(x, \mathit{English}) \wedge \\ \mathit{awards}(x, \mathit{NominatedforOscar})\,;$$

(2) comedy movies directed either by Frank Capra or Woody Allen:

$$Q_2(x) = \mathit{Movie}(x) \wedge \mathit{genre}(x, \mathit{Comedy}) \wedge (\mathit{director}(x, \mathit{FrankCapra}) \vee \\ \mathit{director}(x, \mathit{WoodyAllen}))\,;$$

(3) actors born in 1964 or in New York who are still alive:

$$Q_3(x) = \mathit{Actor}(x) \wedge (\mathit{yearOfBirth}(x, 1964) \vee \mathit{cityOfBirth}(x, \mathit{NewYorkCity})) \wedge \\ \mathit{not}\ \exists y\, (\mathit{yearOfDeath}(x, y))\,;$$

(4) crime movies in English with an award and Nicolas Cage as a star:

$$Q_4(x) = \mathit{Movie}(x) \wedge \mathit{genre}(x, \mathit{Crime}) \wedge \mathit{language}(x, \mathit{English}) \wedge \\ \exists y\, \mathit{awards}(x, y) \wedge \mathit{star}(x, \mathit{NicolasCage})\,;$$

(5) American movies with Julia Roberts and not Clive Owen as a star:

$$Q_5(x) = \mathit{Movie}(x) \wedge \mathit{country}(x, \mathit{USA}) \wedge \mathit{star}(x, \mathit{JuliaRoberts}) \wedge \\ \mathit{not}\ \mathit{star}(x, \mathit{CliveOwen})\,;$$

(6) movies not directed by Julien Temple and with Monica Bellucci as a star:

$$Q_6(x) = \mathit{Movie}(x) \wedge \mathit{not}\ \mathit{director}(x, \mathit{JulienTemple}) \wedge \mathit{star}(x, \mathit{MonicaBellucci})\,;$$

(7) actors who played in at least one comedy and no western movie:

$$Q_7(x) = \exists y\, (\mathit{Actor}(x) \wedge \mathit{film}(x, y) \wedge \mathit{Movie}(y) \wedge \mathit{genre}(y, \mathit{Comedy}) \wedge \\ \mathit{not}\ \exists z\, (\mathit{Movie}(z) \wedge \mathit{genre}(z, \mathit{Western}) \wedge \mathit{film}(x, z)))\,;$$

Table 5: Total time used (in ms) and number of returned URIs for processing the Semantic Web search queries $Q_1, \dots, Q_{10}$ on a sample set of Web pages extracted from the IMDB, which included more than 60 000 movies and actors, resulting into more than one million facts.

| Query | Total Time (ms) | No. URIs |
|---|---|---|
| $Q_1(x)$ | 128 | 143 |
| $Q_2(x)$ | 43 | 24 |
| $Q_3(x)$ | 160 | 407 |
| $Q_4(x)$ | 119 | 5 |
| $Q_5(x)$ | 24 | 22 |
| $Q_6(x)$ | 14 | 12 |
| $Q_7(x)$ | 5915 | 3271 |
| $Q_8(x)$ | 1987 | 981 |
| $Q_9(x)$ | 2196 | 11867 |
| $Q_{10}(x)$ | 1298 | 10522 |

(8) actors who played in at least one movie in Italian:

$$Q_8(x) = \exists y\,(\mathit{Actor}(x) \wedge \mathit{film}(x,y) \wedge \mathit{Movie}(y) \wedge \mathit{language}(y, \mathit{Italian}))\,;$$

(9) actors who played in no movie in French:

$$Q_9(x) = \mathit{Actor}(x) \wedge \mathit{not}\; \exists z\,(\mathit{Movie}(z) \wedge \mathit{language}(z, \mathit{French}) \wedge \mathit{film}(x,z))\,;$$

(10) drama movies without thrillers:

$$Q_{10}(x) = \mathit{Movie}(x) \wedge \mathit{genre}(x, \mathit{Drama}) \wedge \mathit{not}\; \mathit{genre}(x, \mathit{Thriller})\,.$$

The total processing times (used by the new prototype implementation described in Section 7 on a standard laptop) and the numbers of returned URIs for the above Semantic Web search queries are shown in Table 5. Note that most of the search queries had a total processing time below one second; only those with large outputs were taking slightly more time (some few up to six seconds).

It is also important to point out that the precision and the recall of our approach to Semantic Web search for the above ten Semantic Web search queries are both 1, as the search queries describe exactly their natural-language descriptions. That is, in the IMDB application scenario, our approach to Semantic Web search is both sound and complete, and a precision and recall different from 1 for the above ten search queries would mean that the prototype implementation contains some errors.

## 10 Related Work

We now discuss related work on Semantic Web search (see especially [28] for a recent survey), which can roughly be divided into (1) approaches that are based on structured query languages, such as [17, 30, 35, 38, 45, 46, 50], and (2) approaches for *naive* users, requiring no familiarity with structured query languages. In this category, we distinguish keyword-based approaches, such as [13, 33, 34, 40, 51, 52, 55], where queries consist of lists of keywords, and natural-language-based approaches, such as [16, 23, 29, 32, 42, 43], where users can express queries in natural language. Finally, we also discuss related work on ranking techniques for the Semantic Web.

In order to evaluate user queries on Semantic Web documents, both keyword-based and natural-language-based approaches need a reformulation phase, where user queries are transformed into "semantic" queries. In keyword-based approaches, query processing generally starts with the assignment of a semantic meaning to the keywords, i.e., each keyword is mapped to an ontological concept (property, entity, class, etc.). Since each keyword can match a class, a property, or an instance, several combinations of semantic matchings of the keywords are considered, and, in some cases, the user is asked for choosing the right assignment. Similarly, natural-language-based approaches focus mainly on the translation of queries from natural language to structured languages, by directly mapping query terms to ontological concepts or by using some ad-hoc translation techniques.

In the following, we discuss some approaches based on structured query languages, which are most closely related to ours. We first focus on some more general approaches [17, 35, 38] closest in spirit to ours in that they aim at providing general semantic search facilities. We then discuss some proposals [30, 50, 45, 46] that address some specific aspects of semantic search or that are targeted at specific domains, so that they cannot be strictly viewed as semantic search engines.

The Corese system [17] is an ontology-based search engine for the Semantic Web, which retrieves Web resources that are annotated in RDF(S) via a query language based on RDF(S). It is the system that is perhaps closest in spirit to our approach. In a first phase, Corese translates annotations into conceptual graphs, it then applies proper inference rules to augment the information contained in the graphs, and finally evaluates a user query by projecting it onto the annotation graphs. The Corese query language is based on RDF, and it allows variables and operators.

SHOE [35] is one of the first attempts to semantically query the Web. It provides the following: a tool for annotating Web pages, allowing users to add SHOE markup to a page by selecting ontologies, classes, and properties from a list; a Web crawler, which searches for Web pages with SHOE markup and stores the information in a knowledge base (KB); an inference engine, which provides new markups by means of inference rules (basically, Horn clauses); and several query tools, which allow users to pose structured queries against an ontology. One of the query tools allows users to draw a graph in which nodes represent constant or variable instances, and arcs represent relations. To answer the query, the system retrieves subgraphs matching the user graph. The SHOE search tool allows users to pose queries by first choosing an ontology from a drop-down list and next choosing classes and properties from another list. Finally, the system builds a conjunctive query, issues the query to the KB, and presents the results in a tabular form.

NAGA [38] provides a graph-based query language to query the underlying KB encoded as a graph. The KB is built automatically by a tool that extends the approach proposed in [49] and extracts knowledge from three Web sources: Wordnet, Wikipedia, and IMDB. The nodes and edges in the knowledge graph represent entities and relationships between entities, respectively. The query language is based on SPARQL, and adds the possibility of formulating graph queries with regular expressions on edge labels, but the language does not allow queries with negation. Answers to a query are subgraphs of the knowledge graph matching the query graph and are ranked using a specific scoring model for weighted labeled graphs.

Comparing the above three approaches to ours, in addition to the differences in the adopted query languages (in particular, SHOE and NAGA do not allow complex queries with negation) and underlying ontology languages, there is a strong difference in the query-processing strategy. Indeed, Corese, SHOE, and NAGA all rely on building a unique KB, which collects the information disseminated among the data sources, and which is suitably organized for query processing via the adopted query language. However, this has a strong limitations. First, representing the whole information spread across the Web in a unique KB and efficiently processing each user query on the thus obtained huge amount of data is a rather challenging task. This makes these approaches more suitable for specific domains, where the amount of data to be dealt with is usually much smaller. In contrast, our approach allows the query processing task to be supported by well-established Web search technologies. In fact, we do not evaluate user queries on a single KB, but we represent the information implied by the annotations on different Web pages, and evaluate queries in a distributed way. Specifically, user queries are processed as Web searches over completed annotations. We thus realize Semantic Web search by using standard Web search technologies as well-established solutions to the problem of querying huge amounts of data. Second, a closely related limitation of query processing in Corese, SHOE, and NAGA is its tight connection to the underlying ontology language, while our approach is actually independent from the ontology language and works in the same way for other underlying ontology languages.

Swoogle [30] is a crawler-based system for discovering, indexing, and querying RDF documents. Swoogle mainly provides a search for Semantic Web documents and terms (i.e., the URIs of classes and properties). It allows users to search for all instance data about a specified class, or on a specified subject, and to specify queries containing conditions on the document-level metadata (i.e., queries asking for documents having `.rdf` as the file extension), but it does not allow complex queries. Retrieved documents are ranked according to a ranking algorithm measuring the documents' importance on the Semantic Web.

ONTOSEARCH2 [50] is a search and query engine for ontologies on the Semantic Web. It stores a copy of the ontologies in a tractable description logic and allows SPARQL queries to be evaluated on both the structures and instances of ontologies. The Coraal system [45] is a knowledge-based search engine for biomedical literature. Coraal uses NLP-based heuristics to process texts and build RDF triples from them. These RDF triples are integrated with existing domain knowledge and all the collected information can be queried by the user via a specific query language.

The aim of the approach proposed in [46] is approximately querying RDF datasets with SPARQL [53]. To this end, a SPARQL query is encoded as a set of triple constraints with variables, and an approximate answer is a substitution of the variables with data that may not satisfy all the constraints. The proposed strategy refines the accuracy of the answers progressively, so that the algorithm searching for the answers can be stopped at any time producing a meaningful result.

There is a plethora of ranking techniques for the Semantic Web; here, we discuss only some most closely related ones. Also the ObjectRank approach in [3] adds link semantics to the PageRank technique, but it requires weights for different forms of links before its application. Such weights are assigned by experts and influence the random walk of users. Beagle$^{++}$ [14] extends the Beagle desktop search engine, applying ObjectRank to RDF metadata about desktop objects for an improved ranking in desktop search. TripleRank [31] also considers the semantics of links, but it does not rely on an expert assignment of link weights, and is based on the HITS technique [39] instead of PageRank. Our ObjectRank technique, in contrast, is conceptually simpler, and it can be easily reduced to the standard PageRank technique in the context of our approach to Semantic Web search (cf. Theorem 13).

## 11 Conclusion

We have presented a novel approach to Semantic Web search, which allows for a semantic processing of Web search queries relative to an underlying ontology, and for evaluating ontology-based complex Web search queries that involve reasoning over the Web. We have shown how the approach can be implemented on top of standard Web search engines and ontological inference technologies. We have developed the formal model behind this approach, and we have also generalized the PageRank technique to our approach. We have then 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 (which can be implemented using efficient relational database technology), and we have proved it ontologically correct (and in many cases also ontologically complete). The offline inference compiles terminological knowledge into completed annotations, which have a polynomial size, can be computed in polynomial time, and are also rather small in practice. We have reported on two prototype implementations in desktop search, and provided very positive experimental results on the running time of the online query processing step, and the precision and the recall of our approach to Semantic Web search. We have also shown that our Semantic Web search can be readily applied to existing Web pages without annotations. More specifically, we have implemented a Semantic Web search interface for the Internet Movie Database.

In a companion work [20, 21, 22], we have explored a variant of our Semantic Web search, which uses inductive reasoning techniques (based on similarity search for retrieving the resources that likely have a query property), rather than deductive ones in the offline ontological inference step. This adds an increased robustness, as it allows for handling inconsistencies, noise, and incompleteness in Semantic Web knowledge bases. Furthermore, inductive reasoning allows to infer new (not logically deducible) knowledge (from training individuals). The main idea behind the inductive approach in [20, 21, 22] is also closely related to the idea of using probabilistic ontologies to increase the precision and the recall of querying databases and of information retrieval in general. However, rather than learning probabilistic ontologies from data, representing them, and reasoning with them, the inductive approach directly uses the data in the inductive inference step.

In the future, we aim especially at extending the desktop implementation to a real Web implementation, using existing Web search engines, and to more deeply investigate the properties of the ObjectRank technique. Another interesting topic for future research is to explore how search expressions that are formulated as plain natural language sentences can be translated into the ontological conjunctive queries of our approach. It would also be interesting to investigate whether our approach to Semantic Web search can be combined with top-k query techniques from databases for a further improved performance. Finally, another interesting topic is to explore the use of probabilistic ontologies rather than classical ones.

## A Appendix: Description Logics

As underlying ontology language, we use the tractable description logic *DL-Lite*$_{\mathcal{A}}$ [48], 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 [12], which are a class of restricted description logics for which the main reasoning tasks are feasible 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. In particular, *DL-Lite*$_{\mathcal{R}}$ provides the logical underpinning for the OWL 2 profile QL, and it is also able to fully capture the (DL fragment of) RDF Schema [7], the vocabulary description language for RDF; see [25] for a translation. The *DL-Lite* description logics are especially directed towards data-intensive applications, and they can all be translated into Datalog$_0^{\pm}$ [11, 10]. We now briefly recall the syntax and the semantics of *DL-Lite*$_{\mathcal{A}}$. For more details on description logics in general, we refer the reader to [1].

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 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. As an important ingredient, *DL-Lite*$_{\mathcal{A}}$ also allows for datatypes in such pieces of knowledge.

## 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 $\mathbf{V} = \bigcup_{d \in \mathbf{D}} \mathbf{V}_d$. From these, roles, concepts, attributes, and datatypes are then constructed 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 $U \in \mathbf{R}_D$, 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 $U \in \mathbf{R}_D$ or the negation of an atomic attribute $\neg U$.
- A *basic datatype* $E$ is the range of an atomic attribute $U \in \mathbf{R}_D$, 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: (1) $B \sqsubseteq C$ (*concept inclusion axiom*), where $B$ is a basic concept, and $C$ is a concept; (2) $Q \sqsubseteq R$ (*role inclusion axiom*), where $Q$ is a basic role, and $R$ is a role; (3) $U \sqsubseteq V$ (*attribute inclusion axiom*), where $U$ is an atomic attribute, and $V$ is an attribute; (4) $E \sqsubseteq F$ (*datatype inclusion axiom*), where $E$ is a basic datatype, and $F$ is a datatype; (5) $(\mathsf{funct}\ Q)$ (*role functionality axiom*), where $Q$ is a basic role; (6) $(\mathsf{funct}\ U)$ (*attribute functionality axiom*), where $U$ is an atomic attribute; (7) $A(a)$ (*concept membership axiom*), where $A$ is an atomic concept and $a \in \mathbf{I}$, (8) $P(a,b)$ (*role membership axiom*), where $P$ is an atomic role and $a, b \in \mathbf{I}$; (9) $U(a,v)$ (*attribute membership axiom*), where $U$ is an atomic attribute, $a \in \mathbf{I}$, and $v \in \mathbf{V}$.

Note that concept inclusion axioms of the form $B \sqsubseteq \top_C$ and datatype inclusion axioms of the form $\rho(U) \sqsubseteq \top_D$ can be safely ignored, and that concept inclusion axioms of the form $B \sqsubseteq \exists Q.C$ can be expressed by the two concept inclusion axioms $B \sqsubseteq \exists Q.A$ and $A \sqsubseteq C$, where $A$ is a fresh atomic concept.

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. Formally, a *TBox* is a finite set $\mathcal{T}$ of inclusion and functionality axioms such that every identifying property in $\mathcal{T}$ is primitive (see [48] for a definition of *primitive identifying properties*). 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* $\phi$ 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 $\phi$ 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 $\phi_i$ is a conjunction of atoms and equalities with free variables among $\mathbf{x}$ and $\mathbf{y}_i$.

**Example 9** *(Scientific Database).* Continuing the running example of Section 2, 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. The sets of atomic concepts, atomic roles, atomic attributes, individuals, and data values are defined as in Example 9. Then, a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$ are given by the axioms in Eqs. 1 and 2, respectively, while the query $Q(x)$ of Eq. 3 is actually a conjunctive query.

## Semantics

The semantics of *DL-Lite*$_{\mathcal{A}}$ is defined in terms of standard first-order interpretations as follows. 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 $\alpha$ in $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, denoted $\mathcal{I} \models \alpha$, are defined as usual [48]. The interpretation $\mathcal{I}$ *satisfies* the axiom $\alpha$, or $\mathcal{I}$ is a *model* of $\alpha$, 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 $\alpha$ is a *logical consequence* of $KB$, denoted $KB \models \alpha$, iff every model of $KB$ satisfies $\alpha$. An *answer* for a query $\phi$ to $KB$ is a ground substitution $\theta$ for all free variables in $\phi$ such that $\phi\theta$ is a logical consequence of $KB$.

As shown in [48], 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 (where only the ABox is variable, but the rest is fixed).

**Example 10** *(Scientific Database cont'd).* It is not difficult to verify that the knowledge base $KB = (\mathcal{T}, \mathcal{A})$ of Example 9 is satisfiable, and that the two axioms *Jour- nalPaper* $\sqsubseteq \neg$*ConferencePaper* and *hasAuthor*$(i_3, i_2)$ are logical consequences of $KB$. Informally, $KB$ implies that no journal paper is a conference paper, and that $i_3$ has the author $i_2$. Furthermore, the ground substitution $\theta = \{x/i_2\}$ is an answer for the conjunctive query $Q(x)$ of Example 9. Informally, *mary* is a Ph.D. student who has published an article in 2008 with RDF as a keyword

# B Appendix: Proofs

**Proof of Theorem 1.** Recall that the ground substitution $\theta$ is an answer for $Q(\mathbf{x}) = q_1(\mathbf{x}) \wedge \cdots \wedge q_m(\mathbf{x})$ to $KB$ iff $Q(\mathbf{x}\theta)$ is a logical consequence of $KB$. The latter is equivalent to all $q_i(\mathbf{x}\theta)$ with $i \in \{1, \ldots, m\}$ being a logical consequence of $KB$, which in turn is equivalent to all $q_i(\mathbf{x}\theta)$ with $i \in \{1, \ldots, m\}$ being in the simple completion of $KB$. That is, $Q(\mathbf{x}\theta)$ is a logical consequence of the simple completion of $KB$. That is, $\theta$ is an answer for $Q(\mathbf{x})$ to the simple completion of $KB$. $\Box$

**Proof of Theorem 3.** As shown in [11], every knowledge base $KB$ in *DL-Lite*$_{\mathcal{A}}$ can be translated into a program $P_{KB}$ in Datalog$^{\pm}$. Suppose that no concept inclusion axiom in $KB$ has one of the forms $B \sqsubseteq \exists P$, $B \sqsubseteq \exists P^{-}$, $B \sqsubseteq \delta(U)$, $B \sqsubseteq \exists P.C$, and $B \sqsubseteq \exists P^{-}.C$. Then, the resulting $P_{KB}$ is simply a Datalog program, and any universal model for evaluating the conjunctive query $Q(\mathbf{x})$ relative to $P_{KB}$ (i.e., also relative to $KB$) is given by the simple completion of $KB$. That is, $\theta$ is an answer for $Q(\mathbf{x})$ to $KB$ iff $\theta$ is an answer for $Q(\mathbf{x})$ to the simple completion of $KB$. $\Box$

**Proof of Theorem 5.** Consider the Datalog$^{\pm}$ encoding $P_{KB}$ of $KB$ [11]. Then, (i) $\theta$ is an answer for $Q(\mathbf{x})$ to $KB$ iff $\theta$ is an answer for $Q(\mathbf{x})$ to $P_{KB}$, (ii) deciding whether $\theta$ is an answer for $Q(\mathbf{x})$ to $P_{KB}$ can be evaluated on any universal model of $P_{KB}$, and (iii) only Web pages $p \in \mathbf{P}$, Web objects $o \in \mathbf{O}$, and values $v \in \mathbf{V}$ occur in safe positions in facts in any universal model of $P_{KB}$. Hence, deciding whether $\theta$ is an answer for $Q(\mathbf{x})$ to $P_{KB}$ (and thus to $KB$) can actually already be evaluated on the simple completion of $KB$, which is a subset of every universal model of $P_{KB}$. $\Box$

**Proof of Theorem 7 (sketch).** A universal model of the Datalog$^{\pm}$ encoding $P_{KB}$ of $KB$ [11] can be constructed via the chase, which follows the rules of $P_{KB}$, and generates and propagates existentially quantified entries in the same way as described above in the definition of unsafe positions relative to $KB$. $\Box$

**Proof of Theorem 8.** Immediate by the observation that $KB$ corresponds to a knowledge base in *DL-Lite*$_{\mathcal{A}}$, and that deciding satisfiability and logical consequences of ground atoms for such knowledge bases can both be done in polynomial time in general and in LOGSPACE in the size of the ABox in the data complexity. $\Box$

**Proof of Theorem 9.** The Datalog$^{\pm}$ encoding $P_{KB}$ of $KB$ [11] is safe in that the Web pages $p \in \mathbf{P}$, Web objects $o \in \mathbf{O}$, and values $v \in \mathbf{V}$ in any universal model must already occur in $P_{KB}$, i.e., in the ABox of $KB$. Hence, every $\mathcal{A}'_a$ in the simple completion of $KB$ has at most $|\mathbf{A}'| + |\mathbf{R}'_A| \cdot |\mathbf{P} \cup \mathbf{O}| + |\mathbf{R}'_D| \cdot |\mathbf{V}'|$ elements. $\Box$

**Proof of Theorem 11.** Recall that the ground substitution $\theta = \{x/a\}$ with $a \in \mathbf{P} \cup \mathbf{O}$ is an answer for $Q(x)$ to $KB$ iff $\theta$ is an answer for $\bigvee_{i=1}^{n} Q_{i,0}(x)$ to $KB$ but not an answer for $\bigvee_{i=1}^{n} Q_{i,0}(x) \wedge \bigvee_{j=1}^{n_i} Q_{i,j}(x)$ to $KB$. Since the TBox of $KB$ is empty, the latter is equivalent to $a \in \bigcup_{i=1}^{n} I_{i,0}$ but not $a \in \bigcup_{i=1}^{n} I_{i,0} \cap (\bigcup_{j=1}^{n_i} I_{i,j})$, where $I_{i,j} = \{a \in \mathbf{P} \cup \mathbf{O} \mid \forall k\colon p^k_{i,j}(a) \in \mathcal{A}_a \text{ or } p^k_{i,j}(a, t^k_{i,j}) \in \mathcal{A}_a\}$. That is, $a \in \bigcup_{i=1}^{n} I_{i,0} \setminus (\bigcup_{j=1}^{n_i} I_{i,j})$. Observe also that any ground substitution $\theta = \{x/v\}$ with $v \in \mathbf{V}$ cannot be an answer for $Q(x)$ to $KB$, since $x$ is the first argument in all atoms of $Q(x)$. $\Box$

**Proof of Theorem 13.** Recall that the PageRank $R(u)$ of a node $u \in V$ relative to $G_{KB}$ is defined as follows:

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

where (i) $B_u$ is the set of nodes that point to $u$, and (ii) $N_v$ is the number of links from $v$ [8]. This already shows that the PageRank of $u$ coincides with the ObjectRank of $u$ (see Eq. 4), for all Web pages and objects $u \in \mathbf{P} \cup \mathbf{O}$. $\Box$

## References

[1] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

[3] A. Balmin, V. Hristidis, and Y. Papakonstantinou. ObjectRank: Authority-based keyword search in databases. In *Proceedings VLDB-2004*, pp. 564–575. Morgan Kaufmann, 2004.

[4] J. Bao, E. F. Kendall, D. L. McGuinness, and E. K. Wallace. OWL2 Web ontology language: Quick reference guide, 2008. www.w3.org/TR/owl2-quick-reference/.

[5] T. Berners-Lee. *Weaving the Web*. Harper, San Francisco, CA, 1999.

[6] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Sci. Amer.*, 284:34–43, 2001.

[7] D. Brickley and R. V. Guha. RDF vocabulary description language 1.0: RDF Schema, 2004. W3C Recommendation.

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

[9] P. Buitelaar and P. Cimiano. *Ontology Learning and Population: Bridging the Gap Between Text and Knowledge.* IOS Press, 2008.

[10] A. Calì, G. Gottlob, and T. Lukasiewicz. Datalog$^{\pm}$: A unified approach to ontologies and integrity constraints. In *Proceedings ICDT-2009*, *ACM International Conference Proceeding Series* 361, pp. 14–30. ACM Press, 2009.

[11] A. Calì, G. Gottlob, and T. Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. In *Proceedings PODS-2009*, pp. 77–86. ACM Press, 2009.

[12] 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.

[13] G. Cheng, W. Ge, and Y. Qu. Falcons: Searching and browsing entities on the Semantic Web. In *Proceedings WWW-2008*, pp. 1101–1102. ACM Press, 2008.

[14] P.-A. Chirita, S. Costache, W. Nejdl, and R. Paiu. Beagle$^{++}$: Semantically enhanced searching and ranking on the desktop. In *Proceedings ESWC-2006*, *LNCS* 4011, pp. 348–362. Springer, 2006.

[15] P.-A. Chirita, S. Costache, W. Nejdl, and S. Handschuh. P-TAG: Large scale automatic generation of personalized annotation tags for the Web. In *Proceedings WWW-2007*, pp. 845–854. ACM Press, 2007.

[16] P. Cimiano, P. Haase, J. Heizmann, M. Mantel, and R. Studer. Towards portable natural language interfaces to knowledge bases — The case of the ORAKEL system. *Data Knowl. Eng.*, 65(2):325–354, 2008.

[17] O. Corby, R. Dieng-Kuntz, and C. Faron-Zucker. Querying the Semantic Web with Corese search engine. In *Proceedings ECAI-2004*, pp. 705–709. IOS Press, 2004.

[18] L. Ding and T. Finin. Characterizing the Semantic Web on the Web. In *Proceedings ISWC-2006*, *LNCS* 4273, pp. 242–257. Springer, 2006.

[19] L. Ding, T. W. Finin, A. Joshi, Y. Peng, R. Pan, and P. Reddivari. Search on the Semantic Web. *IEEE Computer*, 38(10):62–69, 2005.

[20] C. d'Amato, F. Esposito, N. Fanizzi, B. Fazzinga, G. Gottlob, and T. Lukasiewicz. Inductive reasoning and Semantic Web search. In *Proceedings SAC-2010*, pp. 1446–1447. ACM Press, 2010.

[21] C. d'Amato, N. Fanizzi, B. Fazzinga, G. Gottlob, and T. Lukasiewicz. Combining Semantic Web search with the power of inductive reasoning. In *Proceedings URSW-2009*, *CEUR Workshop Proceedings* 527. CEUR-WS.org, 2009.

[22] C. d'Amato, N. Fanizzi, B. Fazzinga, G. Gottlob, and T. Lukasiewicz. Combining Semantic Web search with the power of inductive reasoning. In *Proceedings SUM-2010*, *LNCS* 6379, pp. 137–150. Springer, 2010.

[23] D. Damljanovic, M. Agatonovic, and H. Cunningham. Natural language interface to ontologies: Combining syntactic analysis and ontology-based lookup through the user interaction. In *Proceedings ESWC-2010*, Part I, *LNCS* 6088, pp. 106–120. Springer, 2010.

[24] N. Fanizzi, C. d'Amato, and F. Esposito. Metric-based stochastic conceptual clustering for ontologies. *Inf. Syst.*, 34(8):792–806, 2009.

[25] J. de Bruijn and S. Heymans. Logical foundations of (e)RDF(S): Complexity and reasoning. In *Proceedings ISWC-2007*, *LNCS* 4825, pp. 86–99. Springer, 2007.

[26] B. Fazzinga, S. Flesca, and A. Tagarelli. Schema-based Web wrapping. *Knowl. Inf. Syst.*, 26(1):127–173, 2011.

[27] B. Fazzinga, G. Gianforme, G. Gottlob, and T. Lukasiewicz. Semantic Web search based on ontological conjunctive queries. In *Proceedings FoIKS-2010*, *LNCS 5956*, pp. 153–172. Springer, 2010.

[28] B. Fazzinga and T. Lukasiewicz. Semantic search on the Web. *Semantic Web*, 1(1/2):89–96, 2010.

[29] M. Fernández, V. Lopez, M. Sabou, V. S. Uren, D. Vallet, E. Motta, and P. Castells. Semantic search meets the Web. In *Proceedings ICSC-2008*, pp. 253–260. IEEE Computer Society, 2008.

[30] 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 *Proceedings AAAI-2005*, pp. 1682–1683. AAAI Press / MIT Press, 2005.

[31] T. Franz, A. Schultz, S. Sizov, and S. Staab. TripleRank: Ranking Semantic Web data by tensor decomposition. In *Proceedings ISWC-2009*, *LNCS* 5823, pp. 213–228. Springer, 2009.

[32] Google. `http://www.google.com`.

[33] R. V. Guha, R. McCool, and E. Miller. Semantic search. In *Proceedings WWW-2003*, pp. 700–709. ACM Press, 2003.

[34] A. Harth, A. Hogan, R. Delbru, J. Umbrich, S. O'Riain, and S. Decker. SWSE: Answers before links! In *Proceedings Semantic Web Challenge 2007*, *CEUR Workshop Proceedings* 295. CEUR-WS.org, 2007.

[35] J. Heflin, J. A. Hendler, and S. Luke. SHOE: A blueprint for the Semantic Web. In D. Fensel, W. Wahlster, and H. Lieberman, editors, *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*, pp. 29–63. MIT Press, 2003.

[36] I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a Web ontology language. *J. Web Sem.*, 1(1):7–26, 2003.

[37] U. Hustadt, B. Motik, and U. Sattler. Data complexity of reasoning in very expressive description logics. In *Proceedings IJCAI-2005*, pp. 466–471. Professional Book Center, 2005.

[38] G. Kasneci, F. M. Suchanek, G. Ifrim, M. Ramanath, and G. Weikum. NAGA: Searching and ranking knowledge. In *Proceedings ICDE-2008*, pp. 953–962. IEEE Computer Society, 2008.

[39] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[40] Y. Lei, V. S. Uren, and E. Motta. SemSearch: A search engine for the Semantic Web. In *Proceedings EKAW-2006*, *LNCS* 4248, pp. 238–245. Springer, 2006.

[41] 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.

[42] V. Lopez, M. Pasin, and E. Motta. AquaLog: An ontology-portable question answering system for the Semantic Web. In *Proceedings ESWC-2005*, *LNCS* 3532, pp. 546–562. Springer, 2005.

[43] V. Lopez, M. Sabou, and E. Motta. PowerMap: Mapping the real Semantic Web on the fly. In *Proceedings ISWC-2006*, *LNCS* 4273, pp. 414–427. Springer, 2006.

[44] L. Ma, Y. Yang, Z. Qiu, G. T. Xie, Y. Pan, and S. Liu. Towards a complete OWL ontology benchmark. In *Proceedings ESWC-2006*, *LNCS* 4011, pp. 125–139. Springer, 2006.

[45] V. Novácek, T. Groza, and S. Handschuh. CORAAL — Towards deep exploitation of textual resources in life sciences. In *Proceedings AIME-2009*, *LNCS* 5651, pp. 206–215. Springer, 2009.

[46] E. Oren, C. Guéret, and S. Schlobach. Anytime query answering in RDF through evolutionary algorithms. In *Proceedings ISWC-2008*, *LNCS* 5318, pp. 98–113. Springer, 2008.

[47] H. Pérez-Urbina, I. Horrocks, and B. Motik. Efficient query answering for OWL 2. In *Proceedings ISWC-2009*, *LNCS* 5823, pp. 489–504. Springer, 2009.

[48] 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.

[49] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A core of semantic knowledge. In *Proceedings WWW-2007*, pp. 697–706. ACM Press, 2007.

[50] E. Thomas, J. Z. Pan, and D. H. Sleeman. ONTOSEARCH2: Searching ontologies semantically. In *Proceedings OWLED-2007*, *CEUR Workshop Proceedings* 258. CEUR-WS.org, 2007.

[51] T. Tran, P. Cimiano, S. Rudolph, and R. Studer. Ontology-based interpretation of keywords for semantic search. In *Proceedings ISWC/ASWC-2007*, *LNCS* 4825, pp. 523–536. Springer, 2007.

[52] G. Tummarello, R. Cyganiak, M. Catasta, S. Danielczyk, R. Delbru, and S. Decker. Sig.ma: Live views on the Web of data. In *Proceedings WWW-2010*, pp. 1301–1304. ACM Press, 2010.

[53] W3C. SPARQL Query Language for RDF, 2008. W3C Recommendation (15 January 2008). `http://www.w3.org/TR/rdf-sparql-query/`.

[54] W3C. OWL web ontology language overview, 2004. W3C Recommendation (10 Feb- ruary 2004). `www.w3.org/TR/2004/REC-owl-features-20040210/`.

[55] G. Zenz, X. Zhou, E. Minack, W. Siberski, and W. Nejdl. From keywords to semantic queries — Incremental query construction on the Semantic Web. *J. Web Sem.*, 7(3):166–176, 2009.