

Seeking Knowledge with Falcons

Yuzhong Qu, Gong Cheng, Honghan Wu, Weiyi Ge, and Xiang Zhang

Institute of Web Science, School of Computer Science and Engineering
Southeast University, Nanjing 210096, P.R. China
{yzqu, gcheng, hhwu, wyge, xzhang}@seu.edu.cn

Abstract. The Semantic Web, as a Web of data, axiomatizes concepts and describes objects by using shared classes and properties. Seeking knowledge on the Web of data becomes a basic activity for Semantic Web application developers, and ordinary users also expect to be benefited from the capability of knowledge seeking brought by the RDF data model and Web ontologies. In this challenge, we present the object search service provided by our Falcons system, a keyword-based Semantic Web search engine. We illustrate how it facilitates knowledge seeking on the Web of data, briefly show its implementation, and give a preliminary evaluation of Web and Semantic Web search engines on knowledge seeking. Lessons learned are also reported.

1 Introduction

The RDF model and Web ontologies bring axioms and assertions to the Web world. Knowledge of concepts and objects is encoded in common formats and published online. Seeking knowledge naturally becomes a basic activity for both Semantic Web application developers and ordinary users. To satisfy these demands, we developed the Falcons system.¹

Falcons is a keyword-based Semantic Web search engine. It serves object search, which helps users to seek for objects as well as assertions on them, and serves concept search, which allows to seek concepts and related axioms. Falcons also provides knowledge seeking within single RDF documents. Limited by the space, in this paper we only show how the provided object search meets the challenge of knowledge seeking on the Web of data.

2 Scenarios

In this section, we illustrate how to seek various kinds of knowledge with Falcons by three scenarios.

¹ <http://iws.seu.edu.cn/services/falcons/>.

2.1 Scenario 1: Seeking Knowledge of a Subject

In many cases, people have something in mind and want to learn more about it. For example, to attend ISWC 2008, many researchers come to Karlsruhe for the first time. They want to know more about this city, maybe anything or maybe something in particular. Actually, such requirements exist widely in traditional Web search, covering over 60% of Web queries, reported by [2]. Traditional Web search engines provide webpages which contain the keywords in a query, e.g, “Karlsruhe”. The user obtains a series of webpages, but he/she lacks ways to get these webpages organized, or in other words, it is not easy for the user to specify a particular dimension of knowledge about the subject, except for resubmitting queries with different combinations of keywords to try his/her luck.

The screenshot shows the Falcons search interface. At the top, there is a search bar with the text "Karlsruhe" and a "Search Objects" button. Below the search bar, there are tabs for "Object", "Concept", and "Document". A note says "Separate keywords with a space, and put a phrase in double quotes." Below this, there is a section "Specify a type:" with a grid of categories: Agent, City, District, Document, Event, Event Series, Information Bearing Object, Landmark, Main, Ontology, Organization, Ph D Symposium, Social Group, **Subject**, and University. Below the grid, it says "Objects 1 - 10 of 1,773 for your search Karlsruhe (0.08 seconds)".

The second part of the screenshot shows the results for "All > Organization". It lists various types: Club, Company, Division, Educational Institution, Institute, Law Enforcement Agency, Manufacturer, Orchestra, Party, Railway, School, Team, and **University**. Below this, it says "Objects 1 - 10 of 46 for your search Karlsruhe (0.17 seconds)".

The third part of the screenshot shows the details for "University_of_Karlsruhe". It says "University_of_Karlsruhe is a Organization, Subject, University". It lists several links: "page: University of Karlsruhe", "isDefinedBy: http://ontoworld.org/wiki/Special:ExportRDF/University_of_Karlsruhe", and "label: University of Karlsruhe".

Fig. 1. Seeking knowledge of Karlsruhe.

The Semantic Web brings possibilities of classifying knowledge without using third-party algorithms. Generally, objects on the Semantic Web are with typing information, which can be naturally utilized to organize knowledge. In Falcons, after submitting a query “Karlsruhe”, the user is served with a list of objects as well as several types such as “Event”, “Landmark”, and “Organization”, as shown at the top of Fig. 1. Then the user can specify a type to focus on a particular dimension of knowledge. For example, organizations in Karlsruhe attract the user’s interest, and thus “Organization” is selected. As shown at the bottom of Fig. 1, the results are filtered to include only the objects of the type

“Organization”, such as “University of Karlsruhe”. Moreover, the type panel is updated to include “Club”, “Company”, “University”, etc., which are all subclasses of “Organization”. So actually, Falcons enables the user to navigate a class hierarchy to shift the focus from one type of knowledge to another.

2.2 Scenario 2: Seeking Objects with Properties

In some cases, people seek objects with one or more particular properties, e.g., “find out people that know Peter Mika”. The user has some knowledge about the targets but wants to know more. In traditional Web search, it is called a resource query [2], which covers over 20% of Web queries. In this case, the user knows that the target people know Peter Mika, and wants to obtain their names. In traditional Web search engines, the user may submit a query with the phrases “knows” and “Peter Mika”, but he/she is likely to find a lot of webpages that contain both phrases but these pages do not answer the query well. Even some useful webpages are returned, the user still has to open those pages and mine related knowledge from them, which costs a lot of time.



```

Michael Hausenblas is a Person, Ontology, Subject
☆ name: Michael Hausenblas - From semanticweb.org >
☆ label: Michael Hausenblas - From semanticweb.org >
☆ knows: Peter Mika - From semanticweb.org >
http://semanticweb.org/id/Michael\_Hausenblas - Described in 84 documents

FOAF Description for Frank van Harmelen is a PersonalProfileDocument
☆ primaryTopic: [ knows: [ name: Peter Mika ] ] - From www.cs.vu.nl >
☆ generatorAgent: foaf-O-matic/ - From www.cs.vu.nl >
☆ errorReportsTo: bortoli@dit.unitn.it - From www.cs.vu.nl >
http://www.cs.vu.nl/~frankh/foaf.rdf - Described in 23 documents

```

Fig. 2. Seeking who knows Peter Mika.

On the Semantic Web, objects are defined with RDF triples, which are naturally property-value pairs. It creates conditions to improve the precision of retrieval. In Falcons, after submitting a query with the phrases “knows” and “Peter Mika”, the user obtains a list of objects, as shown in Fig. 2. The user can immediately find that Michael Hausenblas and Frank van Harmelen know Peter Mika, and the demand is satisfied, even before clicking on any links in the results page. It is because, for each object in the results, we provide a snippet of knowledge of this object, as the form of property-value pairs. It is important that the snippet is query-dependent, which may directly answer the user’s question in mind. If the user wants to know more about each resulting object, he/she can click on it to obtain a comprehensive knowledge of this object, which is integrated from all over the Semantic Web.

2.3 Scenario 3: Seeking Relations

On traditional Web search engines, in order to seek relations between objects, e.g., between Peter Mika and Jim Hendler, the user submits a query with their names. A webpage is returned only because its content contains both names, although in many cases it is too difficult to find out any relations between these two phrases in the page due to the long distance in text.



Fig. 3. Seeking relations between Peter Mika and Jim Hendler.

Comparatively, the RDF model used by the Semantic Web exactly describes relations between resources. The scenario in the previous subsection has already showed how Falcons enables to seek direct relations. Falcons also enables to seek indirect relations. As shown in Fig. 3, the user submits a query with the phrases “Peter Mika” and “Jim Hendler”, and obtains a list of objects. The first one is a person that knows both Peter Mika and Jim Hendler, and the second one is a conference of which both Peter Mika and Jim Hendler are organization committee members. The user immediately gets answers and does not have to spend any more time in other activities like reading webpages.

3 Technical Features

3.1 Architecture

Figure 4 presents the architecture of Falcons Object Search. Quadruples (RDF triples and provenance) are parsed from crawled RDF documents by using Jena and stored in a MySQL database. Newly discovered URIs in the parsing process are submitted to the crawler for further dereferencing.

On one hand, class-inclusion reasoning is performed based on explicitly stated class inclusion/equivalence axioms, which is then combined with explicitly stated instantiation relation from objects to classes to establish a mapping between objects and their classes (including both explicitly specified and inferred ones). On

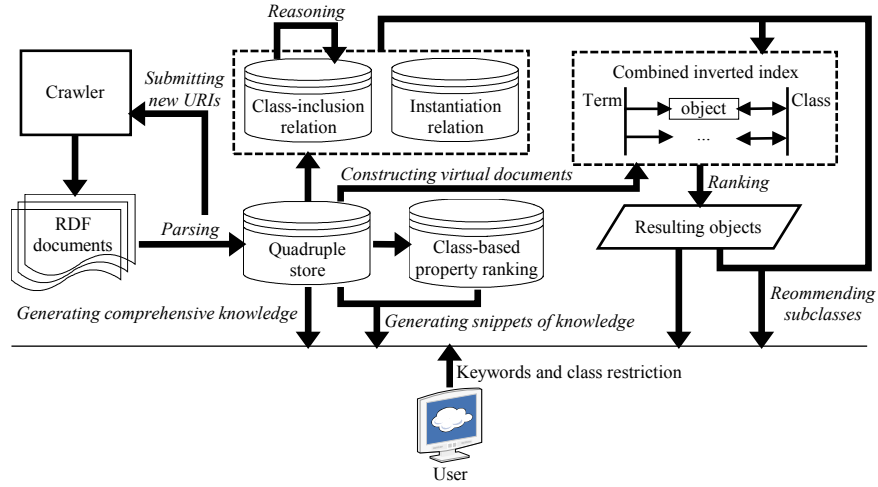


Fig. 4. Architecture of Falcons Object Search.

the other hand, for each object, its local name, associated literals, and labels of its neighboring objects in RDF graphs are collected and weighted to construct a virtual document, based on which an inverted index from terms to objects is built. In this way, the mapping from classes to objects and the mapping from terms to objects form a combined inverted index, which can answer keyword queries with class restrictions for filtering results. Meanwhile, based on the mapping from resulting objects to classes and the class-inclusion relation, several subclasses of the current class restriction are recommended as candidates for further refinement.

Accompanied with each resulting object, a query-dependent snippet of knowledge is provided. Properties in a snippet is selected by considering not only query keywords but also the salience of these properties w.r.t. the class of the resulting object. A comprehensive knowledge of an object is also accessible from the quadruple store.

3.2 Data Set

Seeds are collected from the Linking Open Data project, Ping the Semantic Web.com, etc. We also extract phrases from the Open Directory Project and randomly combine them into keyword queries and send them to Swoogle and Google (for “filetype:rdf” and “filetype:owl”) to retrieve potential RDF documents. At the time of writing, we have crawled and processed over 11 million RDF documents, containing about 600 million RDF triples, in which 73 million objects have been discovered.

3.3 Technical Details

Limited by the space, we only briefly describe important components. For more implementation details, please refer to [1].

Virtual document. With RDF, to describe knowledge of an object, we can associate it with a literal, connect it to a named resource identified by a URI, or connect it to a blank node. To cover all these cases, we construct and index its virtual document that includes not only its local name and associated literals but also the labels and local names of all other neighboring resources in all the discovered RDF documents.

Class-inclusion reasoning. We perform reasoning offline and use more disk space to store consequential data. Class inclusion/equivalence axioms from the dereference documents of involved classes, which are considered to be reliable, are submitted to the transitive reasoning engine to compute all the superclasses of each class. At indexing time, for each object we index not only its explicitly specified classes but also inferred ones.

Ranking. Objects are ranked based on a combination of their relevance to the query and their popularity. The relevance score is calculated based on the cosine similarity measure between the query and the virtual documents of objects, and the popularity score is evaluated according to the number of RDF documents that objects are described by.

Knowledge snippet. A snippet of knowledge of a resulting object contains the top three property-value pairs, and three factors dominate the ranking algorithm. Firstly, a property-value pair that contains more keywords in the query is ranked higher. Secondly, for each class, we compute its most salient properties by counting all the associated properties of all its instances, and then weighting these counts by the corresponding inverse class frequency of each property, the principle of which is similar to the traditional inverse document frequency technique used in information retrieval. It means the ranking of property-value pairs of an object depends on its class. Thirdly, property-value pairs from dereference documents are ranked higher. In addition, redundant knowledge is removed.

Comprehensive knowledge. A comprehensive knowledge of an object is provided after it is clicked. It collects property-value pairs from all the RDF documents that the object is described by, and removes redundancy. Property-value pairs are clustered according to the vocabulary that the property comes from. Pictures are shown directly, and some third-party multimedia applications such as Google Maps are integrated.

4 Evaluation

Because it lacks a golden standard, we performed a task-oriented evaluation and set eight simple tasks to evaluate how much the Semantic Web facilitates knowledge seeking, shown in Table 1. Considering that different search engines are based on different data sets, beforehand we tried several keywords related to each task on all the search engines to be evaluated and had verified that at least

something useful could be found. Then, twenty-six postgraduates, all of whom almost knew nothing about the Semantic Web, were randomly assigned three systems, including one traditional Web search engine (Google or Yahoo) and two Semantic Web search engines (Falcons, Sindice, Swoogle, SWSE, or Watson), to perform all the tasks. For each task, we calculated its average completion quality and average number of clicks. The quality, ranging from 0 (the worst) to 1 (the best), was blindly evaluated by two human experts who set these tasks. The results are summarized in Table 2.

Table 1. Tasks for evaluation

T1	Give a brief description of Tim Berners-Lee.
T2	Find out the homepage and phone number of the person with email chris@bizer.de.
T3	Find out three people that know both the two people in T1 and T2.
T4	Find out three kinds of relations between the two people in T1 and T2.
T5	Give three conferences on Artificial Intelligence as well as their homepages.
T6	Give five popular buildings in Berlin.
T7	What is Lehman Brothers? What happened to it in 2008 and why?
T8	Find out a song titled Forget-me-nots and its singer, genre, and year. Also describe a kind of plant called Forget-me-nots.

Table 2. Evaluation results

	Google & Yahoo		Falcons		Other SW Search Engines	
	Quality	Click	Quality	Click	Quality	Click
T1	1.00	1.90	0.90	6.33	0.65	8.46
T2	0.93	6.00	0.93	5.27	0.65	8.24
T3	0.87	6.05	0.80	4.80	0.37	8.05
T4	0.67	9.80	0.58	6.87	0.19	11.92
T5	1.00	7.00	0.87	6.07	0.42	7.84
T6	0.97	5.05	0.96	5.67	0.31	7.92
T7	0.85	7.85	0.60	8.07	0.31	6.22
T8	0.88	5.70	0.87	3.93	0.30	4.46

T1 is a typical subject-oriented knowledge seeking task. Falcons performed closely to traditional Web search engines, although it cost more clicks, maybe because performers were not familiar with Falcons at the beginning. T2 asked for knowledge of an object specified by an inverse functional property, in which Falcons and traditional Web search engines showed no difference. In T3 and T4, performers were asked to seek relations. They obtained slightly better results on traditional Web search engines, but paid far more clicks, compared to the one on Falcons. T5 and T6 are typical resource seeking tasks, in which Falcons

and traditional Web search engines differed not much. However, traditional Web search engines dominated news seeking, illustrated by T7. T8 is about two distinct objects with exactly the same name. Clearly, performers paid less clicks on Falcons, mostly due to the function of class-based query refinement.

We also collected performers' experience. We found that, out of the 26 performers: 16 ones thought traditional Web search engines are easy to use and are likely to have indexed far more data; 16 ones complained about the response time of Semantic Web search engines; 11 ones felt that Falcons is good at seeking relations; and 3 ones were impressed by the high precision of Falcons.

5 Lessons Learned

By the preliminary evaluation, we have demonstrated that the capability of knowledge seeking provided by Falcons is promising and has opened a door for end users to perform high-precision retrieval tasks on the Web of data. However, existing knowledge seeking on the Semantic Web cannot replace traditional webpage search. Firstly, end users are more familiar with the interfaces and features of traditional Web search engines. Falcons and many other Semantic Web search engines have introduced several new features, which also confuse many end users. Secondly, most existing Semantic Web data sources are transformed from databases and lack updating, causing that a lot of data are somewhat not up-to-date enough. If competing with traditional Web search engines is a goal to pursue, more up-to-date and primitive Semantic Web data are greatly in demand. Thirdly, we integrate all the discovered ontologies into a universal class hierarchy to serve query refinement. During this process, we have confirmed its difficulty because a lot of open problems need to be solved, including ontology alignment, semantic mapping, trust, etc. Actually, the heterogeneity problem at the semantic level brings many problems. For example, we have found dozens of properties that are widely used to describe the name of an object, which burdens the display module of our system. In addition, we even have not addressed ourselves to coreference resolution for object identifies, which is our major future work.

Acknowledgments. The work is supported in part by the NSFC under Grant 60773106, and in part by the 973 Program of China under Grant 2003CB317004. We are grateful to Dr. ChengXiang Zhai for his suggestions on the system.

References

1. Cheng, G., Ge, W., Wu, H., Qu, Y.: Searching Semantic Web Objects Based on Class Hierarchies. In: Bizer, C., Heath, T., Idehen, K., Berners-Lee, T. (eds.) LDOW 2008. CEUR-WS, vol. 369. CEUR-WS.org (2008)
2. Rose, D.E., Levinson, D.: Understanding User Goals in Web Search. In: 13th International World Wide Web Conference, pp. 13–19. ACM Press, New York (2004)