SUITS4RDF: Incremental Query Construction for the Semantic Web

Enrico Minack Wolf Siberski Gideon Zenz
L3S Research Centre
Leibniz University Hanover
Appelstr. 9a, Hanover 30167, Germany
{minack,siberski,zenz}@L3S.de

Xuan Zhou
CSIRO ICT Centre
GPO Box 664
Canberra ACT 2601, Australia
xuan.zhou@csiro.au

ABSTRACT

With the advance of the Semantic Web technology, increasing data will be annotated with computer understandable structures (i.e. RDF and OWL), which allow us to use more expressive queries to improve our ability in information seeking. However, constructing a structured query is a laborious process, as a user has to master the query language as well as the underlying schema of the queried data. In this demo, we introduce SUITS4RDF, a novel interface for constructing structured queries for the Semantic Web. It allows users to start with arbitrary keyword queries and to enrich them incrementally with an arbitrary but valid structure, using computer suggested queries or query components. This interface allows querying the Semantic Web conveniently and efficiently, while enabling users to express their intent precisely.

1.INTRODUCTION

The vision of the Semantic Web is to equip the web with machine-processable and machine-understandable semantics, so that the Web can become a universal medium of information and knowledge exchange for both computer and human being. It can be foreseen that in the near future a lot of information on the Web will be described or annotated using these semantics, resulting in a huge knowledge base. How to utilize such rich semantics to improve the information seeking on the Web is becoming increasingly important.

Today's web search engines, such as Google, rely on a keyword search interface and a variety of statistical methods to catch users' information needs. This approach has been highly successful, as it is proven to be very intuitive and easy even for naïve web users. However, keyword search lacks the expressiveness to make use of the rich semantics in the Semantic Web. In contrast, structural query languages such as SPARQL possess sufficient expressiveness for exploiting the Semantic Web. However, its usability for end users is low. In order to construct a valid SPARQL query to retrieve desired information, a user has to not only grasp the query language, but also understand the ontology thoroughly so as to find the right concepts and structures to form the query. Such a query construction process is complex and laborious.

In [1], we have proposed SUITS, a query interface for relational database that smoothly integrates the intuitiveness of keyword search and the expressiveness of database queries. In this paper, we present SUITS4RDF, a query interface that applies the approach of SUITS to the Semantic Web. The interface of SUITS4RDF is as intuitive as keyword search, while it also allows users to utilize the available semantics to express their intent precisely. In addition, it is highly flexible, as users can choose to construct either completely or partially structured queries depending on the degree to which they want or are able to clarify their intent.

SUITS4RDF does not require users to master the SPARQL language or to know the ontology of the semantic data a priori. Instead, a user can issue arbitrary keyword queries in the way he uses a web search engine. Based on the keyword query, the system suggests some concepts or structures that the user can compose into more complex queries. Afterwards the system can suggest more complex concepts or structures, allowing the user to iteratively articulate the intent to the extend the user needs.

In Section 2, we describe the basic architecture of SUITS4RDF. In Section 3, we show how a structured query can be incrementally constructed through SUITS4RDF. Finally, in Section 4, we discuss the related work and summarize our contributions.

2.THE ARCHITECTURE

We have implemented the SUITS4RDF interface on a *Sesame Native Store*, which is enhanced with a *LuceneSail* layer [4]. The architecture of the system is shown in Figure 2, where processing steps can be split into an offline pre-computing phase and an online query phase. In the pre-computing phase, the RDF graph is indexed using the *LuceneSail* which will subsequently be used in both query generation and query execution. At the same time, SUITS4RDF generates query templates that can potentially be employed by users when forming SPARQL like queries.

The online query phase consists of three steps. In Step 1, the system receives the user's keyword query and passes it to the *LuceneSail* to check for occurrences of the query terms in all properties and concepts. In Step 2, it combines these term occurrences with the pre-computed query templates to generate meaningful SPARQL queries. In Step 3, the system ranks the SPARQL queries according to their likelihood of matching the user's intent (c.f. [1] for details), and returns the top-k queries with non-empty result-sets. When generating SPARQL queries in Step 2, the system also generates query construction options that the user can use later for incrementally constructing queries. These options are also ranked in Step 3 and returned to the user. If the desired query is not in the top-k queries, the user can select some of the query construction options, to iteratively refine the top-k queries (Step 2). This process can be repeated until the user finds the desired structured query.

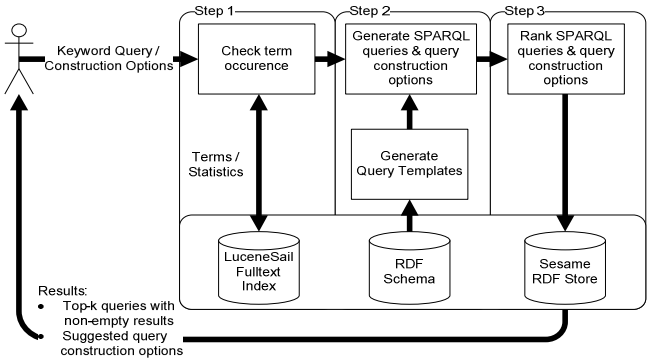

Figure 1: Architecture of SUITS4RDF

The efficiency of the architecture is determined by the following two constructs: (1) the mechanism for inferring SPARQL queries from keywords and query construction options, (2) the algorithm for ranking queries and query construction options so that users can obtain desired queries as fast as possible.

3.QUERY CONSTRUCTION & RANKING

A structured query for the Semantic Web is composed of multiple concepts, properties and literals. The construction process of the structured query can be modeled as a hierarchy of query components, as illustrated in Figure 2. At the bottom of the hierarchy are the smallest components, where each is comprised of a single concept, a single property and a single keyword. The higher in the hierarchy the more complex the query components become. SUITS4RDF lets users start with the smallest query components, and gradually evolve them into larger query components by climbing up the query hierarchy.

For example, to search for the movie "Random Hearts", a user might issue a keyword query "random crash alcee". For each of the terms, SUITS4RDF provides a list of term-property combinations. For example, the user can specify whether "alcee" should appear in the actor name, character name or movie title. After the user specifies some basic query components, the system offers larger components that contain the selected smaller ones. For instance, after the user specifies the character name "alcee" and movie title "random", the system can suggest the query component that connects these two term-property combination using the *actsin* property, as shown in the middle left of Figure 6. Afterwards, the user specifies that "crash" should appear in the plot-text, and the system can suggest the query component at the top, which is already the complete structured query required by the user. Usually, a user does not need to go through the complete construction process, as she can find the desired query among the top-k queries before he reaches the top of the hierarchy.

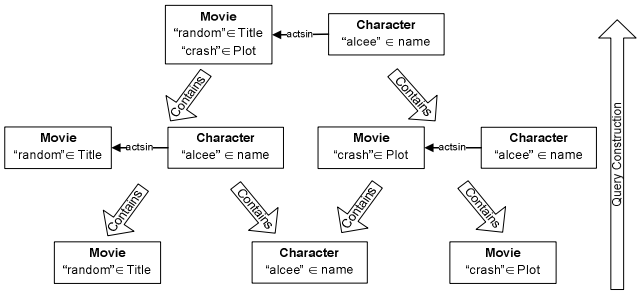

Figure 2: Hierarchy of Query Components

To accelerate query construction, it necessary to rank the possible queries and query components based on their likelihood of matching the user's intent. In SUITS4RDF, we use the following ranking function:

$$Score\ (Q) = SEL\ (Q)^{ps} \cdot PC\ (Q)^{pa} \quad (5)$$

SEL(Q) denotes the selectivity of a query or a query component Q, which measures how many percent of data instances matching Q's template can be selected by Q. Normally, the more selective a query or query component is, the more likely it is chosen by users, as typical users intend their queries to be sufficiently concise and descriptive. PC(Q) denotes the property completeness of Q, which measures how completely each literal property of Q is covered by terms of the keyword query. The intuition behind this factor is if a user describes a property entirely it is more probably meant (i.e. full actor name). Additionally, longer properties are weighted lower as they are generally more likely to match keywords. *ps* and *pa* are two tuning parameters. Our experiments showed this formula is highly effective in ranking queries and query components.

4.RELATED WORK AND CONCLUSION

Keyword search on structured data has been extensively investigated in recent years. In [2] and [3], the authors proposed mechanisms for conducting keyword search on the Semantic Web. Their approaches aim to predict structured queries that best match users' intents behind their keyword queries. However, as the number of possible structured queries grows exponentially with the size of RDF schema and the size of keyword queries, their approaches are only applicable to small datasets. SUITS4RDF go beyond the state of the art, by using an incremental query construction process, so that it can be used on much larger data sources. In SUITS4RDF, we also devised efficient scheme for query optimization, so that query construction and query processing can be performed in reasonable response time.

5.REFERENCES

[1] X. Zhou, G. Zenz, E. Demidova, W. Nejdl: SUITS – Constructing Structured Data from Keywords. Technical report, L3S Research Center, 2007

[2] T. Tran, P. Cimiano, S. Rudolph, R. Studer: Ontology-Based Interpretation of Keywords for Semantic Search. ISWC-2007

[3] Q. Zhou, C. Wang, M. Xiong, H. Wang, Y. Yu: SPARK: Adapting Keyword Query to Semantic Search. ISWC/ASWC, 2007

[4] E. Minack , L. Sauermann , G. Grimnes , C. Fluit , J. Broekstra: The Sesame LuceneSail: RDF Queries with Full-text Search. NEPOMUK Technical Report 2008-1, Feb. 2008