SWiPE: Searching Wikipedia by Example

Maurizio Atzori *
Math/CS Department
University of Cagliari
09124 - Cagliari, Italy
atzori@unica.it

Carlo Zaniolo †
Computer Science Department
University of California
Los Angeles, CA 90095, USA
zaniolo@cs.ucla.edu

ABSTRACT

A novel method is demonstrated that allows semantic and well-structured knowledge bases (such as DBpedia) to be easily queried directly from Wikipedia's pages. Using *Swipe*, naive users with no knowledge of RDF triples and SPARQL can easily query DBpedia with powerful questions such as: "Who are the U.S. presidents who took office when they were 55-year old or younger, during the last 60 years", or "Find the town in California with less than 10 thousand people". This is accomplished by a novel *Search by Example* (SBE) approach where a user can enter the query conditions directly on the Infobox of a Wikipedia page. In fact, *Swipe* activates various fields of Wikipedia to allow users to enter query conditions, and then uses these conditions to generate equivalent SPARQL queries and execute them on DBpedia. Finally, *Swipe* returns the query results in a form that is conducive to query refinements and further explorations. *Swipe*'s SBE approach makes semi-structured documents queryable in an intuitive and user-friendly way and, through Wikipedia, delivers the benefits of querying and exploring large knowledge bases to all Web users.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces

General Terms

Algorithms, Design, Experimentation, Human Factors.

Keywords

Structured query interface, visual query language, semi-structured data querying

*Work founded in part by RAS Project CRP-17615 *DENIS: Dataspaces Enhancing Next Internet in Sardinia*.

†Work performed in part under the aegis of the Visiting Professor Program at the University of Cagliari.

WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

1. INTRODUCTION

There has been much recent interest in querying the massive knowledge bases, such as DBpedia [1] and Yago [8], that were harvested from Wikipedia and other Web sources. Indeed, DBpedia contains more than one billion pieces of information organized as RDF triples. Using systems such as Virtuoso this public-domain database can be searched using powerful SPARQL [11, 13] queries. The realization that these new capabilities can deliver major benefits to users and applications has recently stimulated much research interest [12, 6, 7]. Indeed this will take us from Web-search engines and their keyword-oriented searches toward the much more powerful query capabilities of database management systems, whereby the following queries will be supported:

Q1 Who are the politicians that studied in the same university as Nicolas Sarkozy? Among them, who belongs to a political party that is different from his?

Q2 Which Italian singers/songwriters belong to the same genre as Madonna's music genres?

Q3 What is the average population of California cities with less than 10 thousand people, and what is the largest of those cities and its population?

The excitement of having these powerful queries available on such a large knowledge base is moderated by the realization that there is no simple user-friendly way to pose such queries: the power of DBpedia and Virtuoso is only available to those who, besides SPARQL, know the internal DBpedia names of entities and attributes (i.e., names like: `foaf:givenName` and `dbpprop:populationTotal`).

This situation has motivated many interesting approaches to improve the access to DBpedia for casual Web users, including the two discussed next.

Exploratory Browsing. This approach allows users to navigate through the triplets of the SPARQL graph by starting from an entity (node) and then, by clicking on a property (edge), move to another related entity[1]. Although the user does not need to know the exact names of properties in advance, this approach can only be used effectively for exploring the graph in the vicinity of the original entity. No close integration with Wikipedia is provided in this approach.

Faceted Search. An interesting user interface that supports a top-down search on DBpedia triplet graph is proposed in [6]. This approach deserves many praises insofar as

[1]Available at `http://dbpedia.org/fct`

(a) a Wiki page as example

(b) mouse move on field "Birth name"

(c) user types, introducing a constraint

(d) other constraints, then click "Swipe" button

Figure 1: The swipe interface. In this example, the user is asking for all "musician" people named "Mike" that doesn't play "Pop-rock".

it seeks to provide a user-friendly interactive way to query DBpedia[2] for users who are unfamiliar with either the internal representations used by DBpedia, or SPARQL. On the other hand, this approach only supports queries that can be expressed via a cascade of filters and, even for those, the query formulation process can be laborious requiring several iterations when many properties are present. Unfortunately, this is often the case, and the problem is compounded by the fact that property names often leave room for ambiguity.

For instance, a user searching for "cities in California" will have to start by supplying an item-type (e.g., "City"). This reduces the search space to cities, whereby the user can now specify a number of other type-depending filters, such as a latitude or `is-city-of`. If the user specifies `is-city-of = "California"`, the search returns no results, since this property in DBpedia only applies to rivers on which the city is a riparian settlement (e.g., the Hudson River, for New York) rather than the U.S. State to which the city belongs. Finally, queries requiring complex boolean expressions or aggregates are not supported in this approach. For instance, there is no simple way to express queries Q2 and Q3 above.

2. SWIPE

Our *Swipe* system[3] seeks to maximize both ease of use and query power by letting users work directly on the Wikipedia pages displayed in their browsers; there users specify their queries by marking up the information-box shown in the page–Fig. 1(b). The approach is inspired by the very successful Query By Example (QBE) interface of relational query languages and, along with ease-of-use offers significant

[2]Available at `http://dbpedia.neofonie.de/`

[3]*Swipe* is an acronym for *Searching WIkiPedia by Example.*

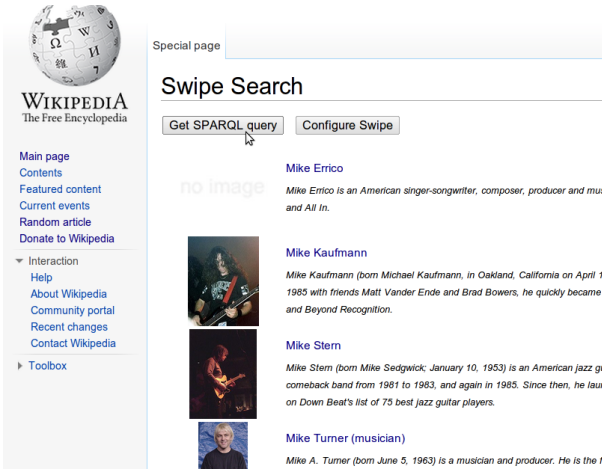

Figure 2: Results shown by the SBE query in Fig 1.

power and flexibility, whereby all the previously mentioned queries are easily expressed.

The *Swipe* system enables users to perform a *Search by Example* (SBE) on DBpedia by the following three steps:

1. A user starts by loading an "example" page in the browser, i.e., a page that represents the general kind of entities the user is interested in. For instance, users interested in cities might retrieve the Wikipedia page for "Boston" but any large city would do. This initial step provides the starting point for the actual query.

2. The example page looks like the original Wikipedia page. However, the Infobox of the page is now an active form that can be used to enter the search conditions defining the query.

3. The user specifies the query by typing into selected value fields in the Infobox, and then issues the query by a simple click. In response to that, *Swipe* returns a description of the Wikipedia pages that satisfy the query conditions entered by the user.

An example of the above interaction is the sequence shown in Fig. 1, where our user wants to find musicians whose birth name is "Mike" and "Pop-rock" is not among their genres. Once the mouse is over a particular field, the box changes to yellow denoting that it is ready to accept the input condition. Thus, our user can move the pointer to the box next to "Birth name" and type `Mike`, and then move it to the box next to occupation and enter `musician`. Finally, the user can type `"un(pop-rock)"`in order to exclude `pop-rock` from the acceptable music genres. Having thus specified conditions on three property values (the specification order is immaterial) the user can hit the *Swipe* button, whereby *Swipe* generates the SPARQL query that searches DBpedia using these conditions. The query results are then returned to the user as shown in Fig. 2 (this default output format is easily modified via the control buttons provided by *Swipe*).

2.1 The Query language

When the user moves the mouse over a field, this becomes editable whereby the user can enter various constraints, including complex conditions (via regular expressions) and