A Declarative Framework for Semantic Link Discovery over Relational Data

Oktie Hassanzadeh* University of Toronto oktie@cs.toronto.edu

Lipyeow Lim IBM Research liplim@us.ibm.com

Anastasios Kementsietsidis IBM Research akement@us.ibm.com

Min Wang IBM Research min@us.ibm.com

ABSTRACT

In this paper, we present a framework for online discovery of semantic links from relational data. Our framework is based on declarative specification of the linkage requirements by the user, that allows matching data items in many real-world scenarios. These requirements are translated to queries that can run over the relational data source, potentially using the semantic knowledge to enhance the accuracy of link discovery. Our framework lets data publishers to easily find and publish high-quality links to other data sources, and therefore could significantly enhance the value of the data in the next generation of web.

Categories and Subject Descriptors

H.2.5 [**Information Systems**]: Heterogeneous Databases; H.2.4 [**Information Systems**]: Systems; H.2.3 [**Information Systems**]: Languages

General Terms

Algorithms, Design, Languages

Keywords

Link Discovery, Record Linkage, Linked Data, Semantic Web

1. INTRODUCTION

While publishing online data is now easier than ever, attempts to establish semantic relationships between different published data sources has been less successful. So, from the users' perspective, online sources resemble *islands of data* (or *data silos*), where each island maintains only part of the data necessary to satisfy a user's information needs. Penetrating these silos to both understand their contents and understand potential semantic connections is a daunting task. What web users need is automated support for creating referential links between data that reside in different sources and that are semantically related. Of course, finding such links requires the use of both approximate matching (to overcome syntactic representational differences and errors) and semantic matching (to find specific semantic relationships). Furthermore, both types of matching must be tightly integrated to accommodate for the tremendous heterogeneity found in web repositories.

*Work done while at IBM T.J. Watson Research Center

WWW 2009, April 20–24, 2009, Madrid, Spain.
ACM 978-1-60558-487-4/09/04.

In the context of the Linking Open Data (LOD) community project at W3C, a number of tools and frameworks have been developed that allow the generation and publication of linked data from relational databases. Examples of such frameworks include D2RQ [2] and OpenLink's Virtuoso [1]. Although these frameworks significantly simplify the process of publishing linked data, and are a main reason behind the recent growth of the linked data sources, they do not provide tools for *discovering* links between the data sources. The number of links between existing LOD data sources is still three orders of magnitude less than the number of base data triples (as of January 2009).

For typical users, this means they must experiment with a myriad of different link discovery methods to find one that suits their needs. In this paper, we seek to develop a generic and extensible framework for integrating link discovery methods. Our goal is to facilitate experimentation and help users find and tune the link discovery methods that will work best for their domain and application. Our framework permits the discovery of links between autonomous relational sources.

2. MOTIVATING EXAMPLE

As our motivating example, consider the sample relations and data shown in Figure 1, including a database of clinical trials (the CT relation), a source of electronic medical records (EMR) for patients (relation PV) and a web source extracted from DBpedia (or Wikipedia) which stores information about drugs and diseases (the $DBPD$ and $DBPG$ relations). Starting from these relations, we now describe briefly the types of links data publishers would like to discover between them. Patient visit "VID770" with diagnosis "Thalassaemia" in the PV relation should be linked to the trial "NCT00579111" with condition "Hematologic Diseases" since "Thalass*a*emia" is a different representation of "Thalassemia" and according to a medical thesaurus (such as NCI) "Thalassemia" *is a type of* "Hematologic Diseases". The *prescr*iption column of both patient visits in the PV relation should link to trial "NCT00336362" based on the fact that "Hydroxycarbamide", "Hydroxyura" and "Hydroxyurea" all refer to the same drug. Other interesting correlations include linking "Westchester Med. Ctr" from visit "VID777" to "Columbia University" based on additional geographical information that both locations are in the same state (New York), and finding other related trials based on a specific criteria, e.g., based on the similarity of the title and authors of the trials' corresponding publications. The online trials data source can link the condition "Hemato-

trial	*cond*	*inter*	*loc*	*city*	*pub*
NCT00336362	Beta-Thalassemia	Hydroxyurea	Columbia University	New York	14988152
NCT00579111	Hematologic Diseases	Campath	Texas Children's Hospital	Austin	3058228

(a) Clinical trials (*CT*)

visitid	*diag*	*prescr*	*location*
VID770	Thalassaemia	Hydroxyura	Texas Hospital
VID777	PCV	Hydroxycarbamide	Westchester Med. Ctr

(b) Patient visit (*PV*)

name
Thalassemia
Blood_Disorders

(c) DBpedia Disease (*DBPD*)

name
Alemtuzumab
Hydroxyurea

(d) DBpedia Drug (*DBPG*)

Figure 1: Sample relations

```
linkspec_stmt:= CREATE LINKSPEC linkspec_name ( col1, col2 )
                AS link_method opt_args opt_limit;

link_method:= native_link | link_clause_expr | UDF;

native_link:= synonym | hyponym | stringMatch;

link_clause_expr:= link_clause AND link_clause_expr
                 | link_clause OR link_clause_expr
                 | link_clause;

link_clause:= LINK col3 WITH col4
              USING link_terminal opt_limit;

link_terminal:= native_link | UDF | linkspec_name

opt_limit:= LINKLIMIT number;
```

Figure 2: The LinQL grammar

logic Diseases" to DBpedia resource (or Wikipedia page) on "Blood_Disorders", and link the intervention "Campath" to DBpedia resource "Alemtuzumab" using the semantic knowledge that "Campath" is the brand name for the chemical name "Alemtuzumab".

3. THE LINQL LANGUAGE

We introduce LinQL, an extension of SQL that integrates querying with link discovery methods. A query written in LinQL identifies (a) the subset of the *local* data that must be linked; (b) a source containing *external* data that are to be linked with the local data from (a); and (c) the methods(s) that must be used to find links between the local and external data. Our framework includes a variety of *native* (or *built-in*) methods including synonym, hyponym, and a variety of string similarity functions. Such native methods can be used as such or they can be customized by setting their parameters and combined to allow both semantic and syntactic matching. This allows users to interleave declarative queries with interesting combinations of link discovery requests. The native support for methods also permits customization where domain knowledge is available that can greatly enhance the performance. The framework is also extensible to additional methods, written as "black-box" user-defined functions (UDF).

Although the linkage specification elements of LinQL are expressible in a number of different languages and notations (e.g., RDF/XML, N3, NTriples), in our framework we choose for consistency an SQL-like syntax. A *link specification*, or *linkspec* for short, defines the conditions that two given values must satisfy before a link can be established between them. In more detail, a linkspec is defined using the grammar of Figure 2 (we omit here the full grammar and only present its main components). As shown in the figure, a `CREATE LINKSPEC` statement defines a new linkspec and accepts as parameters the name of the defined linkspec and the names of the relation columns whose values need to be linked.

A link specification can also be defined in terms of *link clause expressions*. Link clause expressions are boolean combinations of link clauses, where each link clause is semantically a boolean condition on two columns and is specified using either (a) a native method; (b) a user-defined function (UDF); or (c) a previously defined linkspec. Note the mutual recursive nature of linkspecs and link clauses. A linkspec can be defined using link clauses while a link clause can be specified using previously defined linkspecs. The `LINKLIMIT` allows the users to limit the number of links and only consider k results, or the *top-k* where ordering is possible.

4. FROM LINQL TO SQL

Irrespectively of the language used, a declarative specification must be translated into some form of a program that finds the links between the local and external data. Often, such programs are implemented using programming languages like Java or C by third-party developers, and are automatically invoked with arguments defined through the declarative specification. As such, for the data publishers these programs act as *black-boxes* that sit outside the data publishing framework and whose modification requires the help of these developers. We propose an approach to address these shortcomings by providing a native SQL implementation for the link discovery algorithms. This approach has several advantages including the ability to (a) easily implement this framework on existing relational data sources with minimum effort and without any need for externally written program code; (b) take advantage of the underlying DBMS optimizations in the query engine while evaluating the SQL implementations of the link finding algorithms; and (c) use specific efficiency and functionality enhancements to improve the efficiency of these algorithms.

5. CONCLUSION

In this paper, we outlined a declarative extensible framework for link discovery from relational data. We have implemented the framework on a commercial database engine and validated the effectiveness of our approach by finding links between a real clinical trials data source and several other data sources[1]. We believe that our framework can greatly enhance the process of publishing a high-quality data source with links to other data sources on the web. Furthermore, our framework combined with an existing popular declarative approach for generating RDF data on the web such as [2], can lead to a quick and simple way of publishing an online linked data source with high-quality links.

6. REFERENCES

[1] Openlink Virtuoso: Universal server platform for the real-time enterprise, http://virtuoso.openlinksw.com/.

[2] C. Bizer and A. Seaborne. D2RQ - Treating non-RDF databases as virtual RDF graphs. In *ISWC2004 (posters)*.

[1]A subset of the links found are released at http://linkedct.org.