# Data Linking with Ontology Alignment

Zhengjie Fan

INRIA & LIG
655, avenue de l'Europe, Montbonnot Saint Martin, 38334 Saint-Ismier, France
zhengjie.fan@inria.fr

**Abstract.** It is a trend to publish RDF data on the web, so that users can share information semantically. Then, linking isolated data sets together is highly needed. I would like to reduce the comparison scale by isolating the types of resources to be compared, so that it enhances the accuracy of the linking process. I propose a data linking method for linked data on the web. Such a method can interlink linked data automatically by referring to an ontology alignment between linked data sets. Alignments can provide them entities to compare.

**Keywords:** Data Linking, Ontology Alignment, Linked Data.

## 1 Motivation and Research Questions

Nowadays, countless linked data sets are published on the web. They are written with respect to different ontologies. Linking resources in these various data sets is the key for achieving a web of data. However, it is impossible for people to interlink them manually. Thus, many methods are proposed to link these data sets together. Here, I propose a data linking method based on ontology matching, which can automatically link data sets from different domains. Formally, data linking is an operation whose input are two collections of data. Its output is a collection of links between entities from both collections, in which there are binary relations on entities corresponding semantically to each other [4]. So, the research problem which will be tackled here is: given two RDF data sets, try to find out all possible "owl:sameAs" links between them automatically and correctly. My work is part of the Datalift[1] project, which aims to build a platform for data publishing. It is made of several modules, such as vocabulary selection, format conversion, interconnection and the infrastructure to host linked data sets. My work is to build the interconnection module, the last step of Datalift, that is, linking RDF data sets.

The paper is organized as follows. First, the state of the art on data linking is briefly analyzed in Section 2. Then in Section 3, a data linking method is introduced. Finally, Section 4 outlines the planned research methodology.

[1] http://datalift.org/

E. Simperl et al. (Eds.): ESWC 2012, LNCS 7295, pp. 854–858, 2012.

## 2 State of the Art

Data linking is the process of linking data, it can be done according to the results from comparing property values of instances in source classes with the ones of instances in target classes. Suppose there are $m$ instances in the source class. There are $n$ instances in the target class. So there should be $m * n$ comparison pairs. Thus, several techniques are proposed to reduce such comparison scale, while enhancing the precision and recall of the linking process.

One fundamental strategy is finding the keys of data sets. For key is used to identify and distinguish instances within the data set. The linking method only need to compare the property values within keys, then it can decide whether two instances are similar or not. So ideally, it can reduce the comparison scale of linking process. However, there are limitations for such strategy. On one hand, some key property cannot be used to compare, because they are meaningless out of the data source, such as the code or id. Usually different data sets have different coding formats. For these codes are not meaningful for human being, it is hard to find a transformation function. On the other hand, if the properties within a key do not have corresponding properties within one key of another data set. It is impossible to compare those keys for judging whether two instances are "owl:sameAs" or not. Thus, it is common to combine the key with other techniques such as machine learning [5,11].

Machine learning is a widely used strategy for data linking. It is used to find out the potential comparison pairs, as shown in [5,6,7]. That is, from which classes and properties values should be compared. For example, Hu et al. (2011) uses machine learning to enlarge the key property set for matching instances. The linking method in [5] considers not only key properties, but also uses machine learning to search frequently linked properties to find out similar instances using machine learning. There are two kinds of machine learning methods: supervised methods and unsupervised methods. Supervised methods use a training data set to find out the most suitable comparison pairs [5,8]. While more work focus on unsupervised methods for interlinking instances, for it saves time on collecting training sets [1]. Besides machine learning, graph structure and statistical technique are also used for finding out comparison pairs and reducing comparison scale [8]. Usually, these strategies are combined to fulfill the data linking process.

Above all, these linking strategies need to find out the linking pattern between two data sets. That is to say, which comparison pairs are more likely to contain similar instances, which pairs are not. Such linking pattern can be found out more easily with certain heuristics, such as ontology alignment. Ontology Alignment contains correspondences which may be used as pairs of entities from which to find instance to compare. It could be the corresponding classes, or corresponding properties, or corresponding class and property with restrictions. Thus, it can be used to automate the data linking process.

There are several research problems on using ontology matching for data linking. First, what kind of correspondence can be used for data linking? what cannot? Second, which correspondence or group of correspondences can efficiently help comparing data? How far ontology matching can enhance the linking speed and accuracy? What its main advantage over other techniques? What are its limitations? In which case, it cannot efficiently help linking RDF data sets?

## 3 Proposed Approach

As a well-known data linking tool, SILK [2] is designed to execute the data linking process, following manually written scripts specifying from which class and property the value should be compared, as well as which comparison method is used. So, I plan to realize the data linking process by transforming alignments into SILK scripts. Then the data linking process can be triggered on SILK afterwards.

The data linking method proposed here is illustrated in Fig. 1. Suppose there are two data sets to be linked. Firstly, their vocabularies, namespaces or ontology URIs are sent to an Alignment Server, which is an alignment storage [3], so as to check whether there is an ontology alignment available. If there is an alignment and it is written in EDOAL[2] [10], which is an expressive language for expressing correspondences between entities from different ontologies, or not. It cannot only express correspondences between classes and attributes, but also express complex correspondences with restrictions. Then I directly produce a SILK script. If it is not written in EDOAL, then each concept's keys are computed according to the TANE algorithm [12], which is for finding out functional and approximate dependencies between properties of data sets, or coverage and discriminating rate of the properties. If the Alignment Server does not contain any alignment, then the vocabularies are searched. And an ontology matcher is used to generate an alignment between the data sets' ontologies. So, the linking method introduced here has limitations when there is no correspondences or vocabularies available. It will take extra time to compute the data set's ontology and ontology alignment.

At the early stage of my PHD research, I simplify my data linking method as "extracting correspondences information from alignment written in EDOAL to generate SILK script". After it is successfully done, the key will be taken into consideration to complete the picture.

## 4 Planned Research Methodology

i Methods for data collection and presentation
For the work is part of the Datalift project, geographical RDF data sets provided by Datalift will be tested with my data linking method. Furthermore, data from the Linked Data Cloud will also be tested. An interface for the interconnection module will be built, which not only shows the owl:sameAs links, but also shows the details of instances.

[2] http://alignapi.gforge.inria.fr/edoal.html

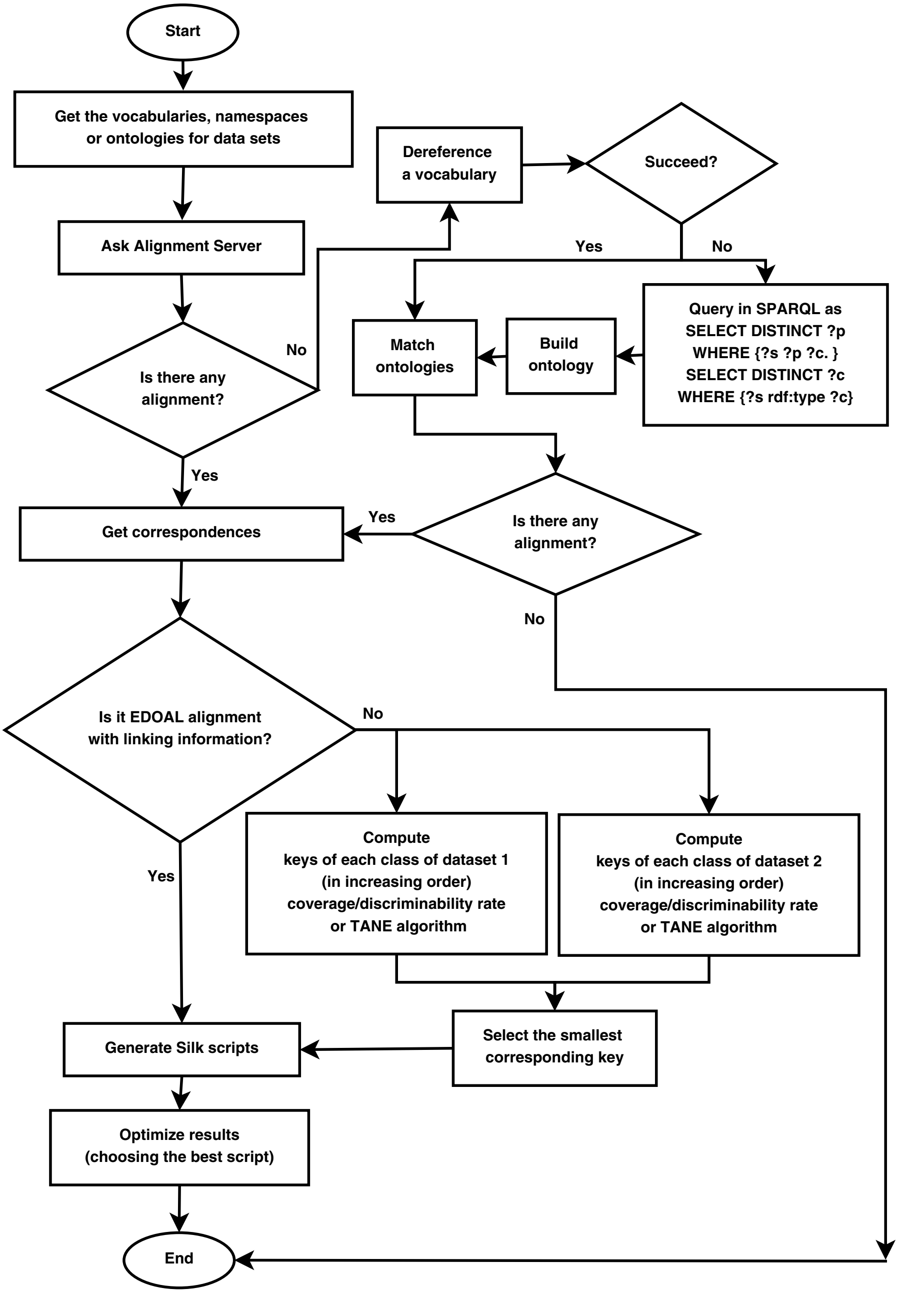
Start
Get the vocabularies, namespaces or ontologies for data sets
Ask Alignment Server
Is there any alignment?
No
Yes
Dereference a vocabulary
Succeed?
Yes
No
Query in SPARQL as SELECT DISTINCT ?p WHERE {?s ?p ?c. } SELECT DISTINCT ?c WHERE {?s rdf:type ?c}
Build ontology
Match ontologies
Is there any alignment?
Yes
No
Get correspondences
Is it EDOAL alignment with linking information?
No
Yes
Compute keys of each class of dataset 1 (in increasing order) coverage/discriminability rate or TANE algorithm
Compute keys of each class of dataset 2 (in increasing order) coverage/discriminability rate or TANE algorithm
Select the smallest corresponding key
Generate Silk scripts
Optimize results (choosing the best script)
End

**Fig. 1.** Workflow of Data Linking Method Based on Ontology Alignment

ii Methods for data analysis and evaluation
The linking method will take part in IM@OAEI. I will test my linking method on OAEI data sets. The precision, recall and duration of the method will be computed. If there are several SILK script produced, the result will be analyzed and improved as below:
   i Compute the duration, precision and recall of producing and running each script. Find out the wrong linkings and the missing linkings of each script.
   ii Compute the average duration, precision and recall. Pick out the script $S_a$ whose duration, precision and recall near the average values. Pick out the scripts $S_d$, whose duration is the smallest, $S_p$, whose precision is the highest, $S_r$, whose recall is the highest.
   iii Find out which linkings cost more time to be found by comparing the linking set of $S_d$ with $S_p$, $S_d$ with $S_r$. What correspondences they belong to. What kind of wrong links tend to be produced by comparing the linking set of $S_a$ with the linking set of $S_p$. How to adjust the script to increase the precision and recall for less wrong linkings and missing linkings.

## References

1. Araújo, S., Hidders, J., Schwabe, D., de Vries, A.P.: SERIMI - Resource Description Similarity, RDF Instance Matching and Interlinking. CoRR. abs/1107.1104 (2011)
2. Bizer, C., Volz, J., Kobilarov, G., Gaedke, M.: Silk - A Link Discovery Framework for the Web of Data. CEUR Workshop Proceedings, vol. 538, pp. 1–6 (2009)
3. David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The Alignment API 4.0. Semantic Web Journal 2(1), 3–10 (2011)
4. Ferrara, A., Nikolov, A., Scharffe, F.: Data linking for the Semantic Web. International Journal of Semantic Web in Information Systems 7(3), 46–76 (2011)
5. Hu, W., Chen, J., Qu, Y.: A Self-Training Approach for Resolving Object Coreference on the Semantic Web. In: Proceedings of WWW 2011, pp. 87–96. ACM (2011)
6. Isele, R., Bizer, C.: Learning Linkage Rules using Genetic Programming. In: OM 2011. CEUR Workshop Proceedings, vol. 814 (2011)
7. Ngonga Ngomo, A.-C., Lehmann, J., Auer, S., Höffner, K.: RAVEN - Active Learning of Link Specifications. In: OM 2011. CEUR Workshop Proceedings, vol. 814 (2011)
8. Nikolov, A., Uren, V.S., Motta, E., Roeck, A.N.D.: Handling Instance Coreferencing in the KnoFuss Architecture. In: IRSW 2008. CEUR Workshop Proceedings, vol. 422, pp. 265–274 (2008)
9. Raimond, Y., Sutton, C., Sandler, M.: Automatic Interlinking of Music Datasets on the Semantic Web. In: LDOW 2008. CEUR Workshop Proceedings, vol. 369 (2008)
10. Scharffe, F.: Correspondence Patterns Representation. PhD thesis, University of Innsbruck (2009)
11. Song, D., Heflin, J.: Automatically Generating Data Linkages Using a Domain-Independent Candidate Selection Approach. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 649–664. Springer, Heidelberg (2011)
12. Huhtala, Y., Kärkkäinen, J., Porkka, P., Toivonen, H.: TANE: An Efficient Algorithm for Discovering Functional and Approximate Dependencies. The Computer Journal 42(2), 100–111 (1999)