Poster Paper

Summarizing Linked Data RDF Graphs Using Approximate Graph Pattern Mining

Mussab Zneika
ETIS Lab,
ENSEA /University of Cergy-Pontoise /CNRS,
Cergy, France
mussab.zneika@ensea.fr

Claudio Lucchese
ISTI-CNR,
Pisa, Italy
claudio.lucchese@cnr.it

Dan Vodislav
ETIS Lab,
ENSEA /University of Cergy-Pontoise /CNRS,
Cergy, France
Dan.Vodislav@u-cergy.fr

Dimitris Kotzinos
ETIS Lab,
ENSEA /University of Cergy-Pontoise /CNRS,
Cergy, France
Dimitrios.Kotzinos@u-cergy.fr

ABSTRACT

The Linked Open Data (LOD) cloud brings together information described in RDF and stored on the web in (possibly distributed) RDF Knowledge Bases (KBs). The data in these KBs are not necessarily described by a known schema and many times it is extremely time consuming to query all the interlinked KBs in order to acquire the necessary information. To tackle this problem, we propose a method of summarizing large RDF KBs using approximate RDF graph patterns and calculating the number of instances covered by each pattern. Then we transform the patterns to an RDF schema that describes the contents of the KB. Thus we can then query the RDF graph summary to identify whether the necessary information is present and if so its size, before deciding to include it in a federated query result.

Keywords

Linked Open Data; RDF Summarization; Query Processing

1. INTRODUCTION

The amount of RDF (Resource Description Framework, www.w3.org/RDF/) data available on the semantic web is increasing fast both in size and complexity, e.g. more than 1000 datasets are now published as part of the Linked Open Data (LOD) cloud, which contains more than 62 billion RDF triples, forming big and complex RDF data graphs. It is also well established that the size and the complexity of the RDF data graph have a direct impact on the evaluation of the RDF queries we express against these data graphs. Especially on the LOD cloud, we observe that a query against a big and complex RDF Knowledge Base (KB) might retrieve no results at the end because either (a) the association between the different RDF KBs is weak (is based only on a few associative links) or (b) there is an association at the schema level that has never been instantiated at the actual data level. Thus we can conclude that having information on the content of a KB and statistical information on the number of instances described under various concepts will allow us to decide on whether or not to post a query based on the availability of the necessary information.

By creating summaries of the RDF KBs, we allow the user or the system to decide whether or not to post a query, since (s)he knows whether information is present or not. This would provide significant cost savings in processing time since we will substitute queries on complex RDF KBs with queries first on the summaries (on much simpler structures with no instances) and then with queries only towards the KBs that we know will produce significant results. We need to compute the summaries only once and update them only after significant changes to the KB. Given the (linked) nature of LOD this will speed up the processing of queries in both centralized and distributed settings.

Moreover, many RDF KBs are suffering from a total or partial absence of schema information. By applying RDF summarization techniques, we can extract, at least, a subset of the schema information and thus facilitate the query building for the end users with the additional benefit of categorizing the contents of the KB based on the summary. We can envision similar benefits when KBs are using mixed vocabularies to describe their content. In all these cases we can use the RDF summary to concisely describe the data in the RDF KB. Thus in this work we study the problem of LOD/RDF graph summarization that is: given an input RDF graph, find the summary graph which reduces its size, while preserving the original inherent structure and correctly categorizing the instances included in the KB.

Two main categories of graph summarization have been proposed to date: (1) aggregation and grouping approaches [3, 5], which are based on grouping the nodes of input RDF graph G into clusters/groups based on the similarity of attributes' values and neighborhood relationships associated with nodes of G and (2) structural extraction approaches [1, 2] which are based on extracting some kind of schema where the summary graph is obtained based on an equivalence relation on the RDF data graph G, where a node represents an equivalence class on nodes of G. To the best to our knowledge, few of these approaches are concentrating on RDF KBs and only one of them [1] is capable of producing a RDF schema as result, which would allow the use of RDF tools (e.g. SPARQL) to query the summary. Our approach provides comparable or better results in most cases.

In summary, our solution is responding to all the require-

ments by extracting the best approximate RDF graph patterns, construct a summary RDF schema out of them and thus concisely describe the RDF input data. We offer the following features: (1) The summary is a RDF graph itself, which allows us to post simplified queries towards the summarizations using the same techniques (e.g. SPARQL), (2) statistical information (number of class and property instances per pattern) is included in our summary graph, which allows us to estimate a query's expected results' size, (3) the summary is much smaller than the original RDF graph, contains all the important concepts and their relationships based on the number of instances, (4) schema independence: it summarizes an RDF graph regardless of having or not schema and RDFS triples and (5) heterogeneity independence: it summarizes an RDF graph regardless if it is hetero- or homo-geneous.

2. RDF-GRAPH PATTERNS COMPUTATION

We present in this section our approach of RDF graph summarization, based on mining a set of approximate graph patterns (an error-tolerant pattern mining technique). It aims at discovering the smallest set of k patterns that best describe the input dataset, where the quality of the description is measured by an information theoretic cost function. We use a modified version of the PaNDa+ algorithm [4], which uses a greedy strategy to identify the k patterns that best optimize the given cost function. Even if we do not fix the input parameter k, PaNDa+ can stop producing further patterns when the cost of a new pattern is more than the corresponding noise reduction. Our approach works in three independent steps that are described below and in Figure 1.

Binary Matrix Mapper: We transform the RDF graph into a binary matrix D, where the rows represent the subjects and the columns represent the predicates. We preserve the semantics of the information by capturing distinct types (if present), all attributes and properties and also reverse properties (so as to capture both subject and object of a property). We extend the RDF URI information by adding labels that represent the different predicates carrying this information into the patterns. *No* schema information is required for the algorithm to work adequately well.

$$D[i;j] = \begin{cases} 1, & \text{the i-th URI has j-typeof or is j-property's} \\ & \text{domain/range or is j-attribute's domain} \\ 0, & \text{otherwise} \end{cases}$$

Graph Pattern Identification: The binary matrix created in step 1 is used in a calibrated version of the PaNDa+ [4] algorithm, which allows us to experiment with different cost functions while retrieving the best approximate RDF graph patterns. Each extracted pattern identifies a set of subjects (rows) all having approximately the same properties (cols). The patterns are extracted so as to minimize errors and to maximize the coverage (i.e. provide a richer description) of the input data. A pattern thus encompasses a set of concepts (type, property, attribute) of the RDF dataset, holding at the same time information about the number of instances that support this set of concepts.

Constructing the RDF summary graph: We have implemented a process, which reconstructs the summary as a valid RDF graph using the extracted patterns. For each pattern in step 2 we generate a node labeled by a URI (minted from a hash function) and we add an attribute with the *bc:extent* label representing the number of instances for this pattern. Then we use the labels generated at step 1 to understand the type of predicate involved and to generate the proper links. The process exploits information already embedded in the binary matrix and tries to construct a valid RDF schema to represent the KB.

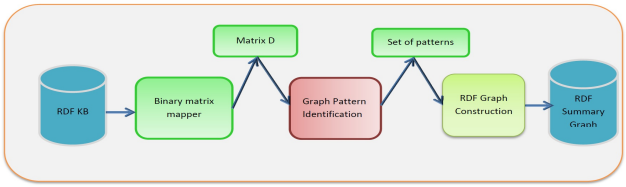

Figure 1: Our RDF Summarization Approach.

3. PRELIMINARY RESULTS

We evaluated so far our approach on variations (e.g. with or without any schema information) over two datasets. An artificial one, which consists of 2000 triples, classified under 8 classes and 9 properties. And a real one, called *Jamendo*, which consists of 11 classes and 25 properties and 1.05 M triples. In both cases 89% of the classes and properties is correctly identified(are the same with the original schema of the dataset even if the schema is not used as a part of the calculation) and the corresponding instances are correctly classified. We produce a summary which is still valid RDF/S and thus can be queried by the same tools.

4. CONCLUSIONS AND FUTURE WORK

In this work we apply an approximate graph pattern mining algorithm in order to extract a summary of an RDF KB. The summary is not necessarily the complete schema of the KB but it always remains a valid RDF/S graph. We plan to test our approach on more complex and bigger datasets (billion of triples); the results so far are promising. We plan to integrate additional RDF knowledge into the algorithm and allow for personalized summaries of RDF KBs.

5. REFERENCES

[1] S. Campinas, T. E. Perry, D. Ceccarelli, R. Delbru, and G. Tummarello. Introducing rdf graph summary with application to assisted sparql formulation. In *Database and Expert Systems Applications (DEXA), 2012 23rd International Workshop on*, pages 261–266. IEEE, 2012.

[2] S. Khatchadourian and M. Consens. Explod: Summary-based exploration of interlinking and rdf usage in the linked open data cloud. *The Semantic Web: Research and Applications*, pages 272–287, 2010.

[3] A. Louati, M.-A. Aufaure, Y. Lechevallier, and F. Chatenay-Malabry. Graph aggregation: Application to social networks. In *HDSDA*, pages 157–177, 2011.

[4] C. Lucchese, S. Orlando, and R. Perego. A unifying framework for mining approximate top-binary patterns. *Knowledge and Data Engineering, IEEE Transactions on*, 26(12):2900–2913, 2014.

[5] N. Zhang, Y. Tian, and J. M. Patel. Discovery-driven graph summarization. In *Data Engineering (ICDE), 2010 IEEE 26th International Conference on*, pages 880–891. IEEE, 2010.