Approximating Inference-enabled Federated SPARQL Queries on Multiple Endpoints

Yuji Yamagata and Naoki Fukuta

Graduate School of Informatics, Shizuoka University
Shizuoka, Japan
{gs14044@s,fukuta@cs}.inf.shizuoka.ac.jp

Abstract. Running inference-enabled SPARQL queries may sometimes require unexpectedly long execution time. Therefore, demand has increased to make them more usable by slightly changing their queries, which could produce an acceptable level of similar results. In this demonstration, we present our query-approximation system that can transform an inference-enabled federated SPARQL query into another one that can produce acceptably similar results without unexpectedly long runtimes to avoid timeout on executing inference-enabled federated SPARQL queries.

Keywords: SPARQL, inference, federated query, ontology mapping

1 Introduction

Reasoning on LODs allows queries to obtain unstated knowledge from a distinct one [1]. Techniques to utilize reasoning capability based on ontology have been developed to overcome several issues, such as higher complexity in the worst case [5][7][8][9].

When a query prepared by the client might require a long execution time, a standard SPARQL endpoint implementation will try to execute the query with lots of cost to return answers. If the endpoint receives lots of heavy queries, it might spend much time on their execution or, more severely, it might cause a server-down. This is especially important for endpoints that have inference engines to support OWL reasoning capability.

In this paper, we present an idea and its prototype implementation of a query-approximation system that can transform an inference-enabled federated SPARQL query into another one that can produce acceptably similar results without unexpectedly long runtimes to avoid timeout on executing inference-enabled federated SPARQL queries.[1]

2 Background

In [7], Kang et al. introduced a number of metrics that can be used to predict reasoning performance and evaluated various classifiers to know how accurately

[1] A demonstration is available at http://whitebear.cs.inf.shizuoka.ac.jp/Yaseki/

they predict classification time for an ontology based on its metric values. According to their evaluation results, they have prepared prediction models with accuracy of more than 80%, but there are still major difficulties in improving them.

In [5], it was introduced that reasoning tasks on ontologies constructed from an expressive description logic have a high worst-case complexity. It has been done by analyzing experimental results that divided each of several ontologies into four and eight random subsets of equal size and measuring classification times of these subsets as increments. They reported that some ontologies exhibit non-linear sensitivity on their inference performance. They also argued that there is no straightforward relationship between the performance of a subset of each isolated ontology and the contribution of each subset to the whole inference performance on the whole ontology, while they provided an algorithm that identifies an ontology's hot spots.

There are two possible approaches to managing long-running queries. One is to utilize parallel and distributed computing techniques to make those executions faster [10]. Another possible approach is rewriting a query that requires long execution time to a light-weight one. There are some query rewriting approaches to improve the quality of queries [2][3][4]. Also, there are some heuristic techniques to approximate inference-enabled queries by modifying some hotspots in the query that prevent faster execution [12]. However, since those hotspots are also dependent on their individual ontologies, such query modification should take into account both query-structure and characteristics of the ontologies used.

3 Outline and System Architecture

If a query seems not to be a time-consuming one, the endpoint executes the query. If the query execution is classified as time-consuming, the endpoint may have an option to reject the execution of the query or transform that query into an optimized one. To implement such behaviors in an endpoint, some extensions should be provided to allow a notification to the client that the received query has been transformed into another one, or the query has been rejected due to a heavy-load condition.

To realize the idea, we are implementing a preliminary system to classify whether a query execution is time-consuming or not, rewriting the query to a more light-weight one, and extending the protocol to notify the rejection of the query, the applied query-transformation for the query, and so on. We applied a pattern-based heuristic query rewriting technique that, for example, substitutes some named classes to subsets of their potential classes that are derived by the inference. Our prototype system has a unique proxy module called "Front-end EP" between the client and the endpoint (called "Back-end EP" in this paper). Figure 1 shows a brief overview of the query execution process mediated by a Front-end EP. Figure 2 shows the basic procedure of query processing on our system. Table 1 shows our preliminary evaluation on the heavy-query detection

on a single endpoint configuration shown in [12]. Here, we used Linklings ontology from the OAEI dataset in the preliminary experiment.

To prepare datasets to evaluate the performance sensitivity of ontology-level simplification techniques, we reduced Linklings ontology by cutting several relational descriptions and added 10 instances for each named class. As an experimental environment, we set up a SPARQL endpoint using Joseki (v3.4.4) in conjunction with a server-side reasoner using Pellet [11] to enable OWL-level inference capability on the endpoint. In this experiment, we used 100 ms as the threshold time. The evaluation data set was generated by queries to get the instances of a named class in the Linklings ontology. Here, we conducted an experiment for all 1,369 queries on N-fold cross validation. We used two classifiers: Bagged C4.5 and Boosted C4.5, implemented in Weka [6] with default parameters. Further evaluation of the performance on multiple-endpoint configurations remains as future work.

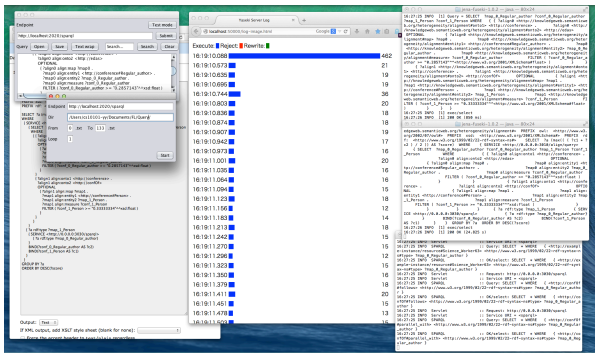

Fig. 1. Overview of Our System

Table 1. Classification Performance on Our Approach (N-fold Cross Validation)[12]

Classifier	Recall	Precision	F-Measure
Bagged C4.5	0.959	0.977	0.964
Boosted C4.5	0.999	0.999	0.999

References

1. Baader, F., Suntisrivaraporn, B.: Debugging SNOMED CT Using Axiom Pinpointing in the Description Logic $\mathcal{EL}^+$. In: Cornet, R., Spackman, K. (eds.) Representing and sharing knowledge using SNOMED. Proceedings of the 3rd International Conference on Knowledge Representation in Medicine KR-MED 2008, vol. 410, pp. 1–7. CEUR-WS (2008)
2. Bischof, S., Polleres, A.: RDFS with Attribute Equations via SPARQL Rewriting. In: Cimiano, P., Corcho, O., Presutti, V., Hollink, L., Rudolph, S. (eds.) The Semantic Web: Semantics and Big Data. LNCS, vol. 7882, pp. 335–350. Springer-Verlag (2013)
3. Fujino, T., Fukuta, N.: SPARQLoid - a Querying System using Own Ontology and Ontology Mappings with Reliability. In: Proc. of the 11th International Semantic Web Conference (Poster & Demos) (ISWC 2012) (2012)

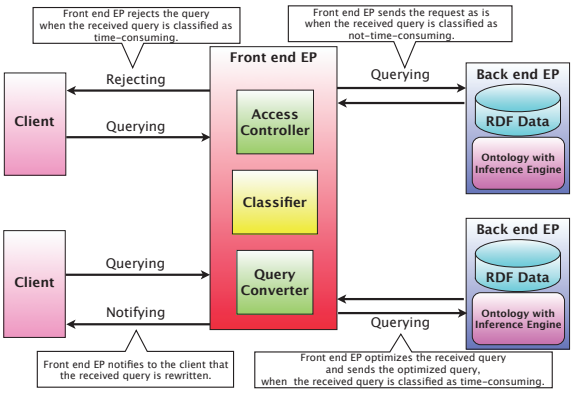

Fig. 2. Basic Query Processing Procedure

4. Fujino, T., Fukuta, N.: Utilizing Weighted Ontology Mappings on Federated SPARQL Querying. In: Kim, W., Ding, Y., Kim, H.G. (eds.) The 3rd Joint International Semantic Technology Conference (JIST2013). LNCS, vol. 8388, pp. 331–347. Springer International Publishing (2013)
5. Gonçalves, R.S., Parsia, B., Sattler, U.: Performance Heterogeneity and Approximate Reasoning in Description Logic Ontologies. In: Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J.X., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E. (eds.) The Semantic Web–ISWC 2012 Part I. LNCS, vol. 7649, pp. 82–98. Springer-Verlag (2012)
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. ACM SIGKDD explorations newsletter 11(1), 10–18 (2009)
7. Kang, Y.B., Li, Y.F., Krishnaswamy, S.: Predicting Reasoning Performance Using Ontology Metrics. In: Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J.X., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E. (eds.) The Semantic Web–ISWC 2012 Part I. LNCS, vol. 7649, pp. 198–214. Springer-Verlag (2012)
8. Motik, B., Shearer, R., Horrocks, I.: Hypertableau Reasoning for Description Logics. Journal of Artificial Intelligence Research 36, 165–228 (2009)
9. Romero, A.A., Grau, B.C., Horrocks, I.: MORe: Modular Combination of OWL Reasoners for Ontology Classification. In: Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J.X., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E. (eds.) The Semantic Web–ISWC 2012 Part I. LNCS, vol. 7649, pp. 1–16. Springer (2012)
10. Schätzle, A., Przyjaciel-Zablocki, M., Hornung, T., Lausen, G.: PigSPARQL: A SPARQL Query Processing Baseline for Big Data. In: Proc. of the 12th International Semantic Web Conference (Poster & Demos) (ISWC 2013) (2013)
11. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl reasoner. Journal of Web Semantics 5(2), 51 – 53 (2007)
12. Yamagata, Y., Fukuta, N.: A Dynamic Query Optimization on a SPARQL Endpoint by Approximate Inference Processing . In: Proc. of 5th International Conference on E-Service and Knowledge Management (ESKM 2014). pp. 161–166 (2014)