Predicting SPARQL Query Performance and Explaining Linked Data

Rakebul Hasan

Predicting SPARQL Query Performance and Explaining Linked Data*

Rakebul Hasan

INRIA Sophia Antipolis, Wimmics, 2004 route des Lucioles - B.P. 93,
06902 Sophia-Antipolis Cedex, France,
hasan.rakebul@inria.fr

Abstract. As the complexity of the Semantic Web increases, efficient ways to query the Semantic Web data is becoming increasingly important. Moreover, consumers of the Semantic Web data may need explanations for debugging or understanding the reasoning behind producing the data. In this paper, firstly we address the problem of SPARQL query performance prediction. Secondly we discuss how to explain Linked Data in a decentralized fashion. Finally we discuss how to summarize the explanations.

Keywords: Query performance, explanation, summarization

1 Introduction

As the complexity of the Semantic Web increases, it is becoming increasingly important to develop efficient ways to query the Semantic Web data [18]. Central to this problem is knowing how a query will behave prior to executing it [15]. Moreover, data publishers publish their data in an interlinked fashion using vocabularies defined in RDFS/OWL [7]. This presents opportunities for large-scale data integration and reasoning over cross-domain data. In such a distributed scenario, consumers of these data may need explanations for debugging or understanding the reasoning behind producing the data; they may need the possibility to transform long explanations into more understandable short explanations [4, 21].

In this paper, firstly we address the problem of SPARQL query performance prediction. Inspired by database research for accurate query performance prediction [2, 13, 14], we use machine learning techniques to predict SPARQL query execution time. Secondly we propose a decentralized solution to explanation for Linked Data. We discuss how to explain Linked Data in a decentralized fashion and how to summarize the explanations.

The structure of the rest of this paper is as follows: in section 2, we present the state of the art on related work. In section 3, we present the problems we address and our contributions. In section 4, we present our approach to the problems we address. In section 5 we present our preliminary results. In section 6 we present our future evaluation plan. Finally, we conclude in section 7.

* Advisors: Fabien Gandon and Pierre-Antoine Champin

2 State of the Art

Recent work on predicting database query performance [2, 13, 14] has argued that the analytical costs models used by the current generation query optimizers are good for comparing alternative query plans, but ineffective for predicting actual query performance metrics such as query execution time. Analytical cost models are unable to capture the complexities of modern database systems [2]. To address this, database researchers have experimented with machine learning techniques to learn query performance metrics. Ganapathi *et al.* [13] use Kernel Canonical Correlation Analysis (KCCA) to predict a set of performance metrics. For the individual query elapsed time performance metric, they were able to predict within 20% of the actual query elapsed time for 85% of the test queries. Gupta *et al.* [14] use machine learning for predicting query execution time ranges on a data warehouse and achieve an accuracy of 80%. Akdere *et al.* [2] study the effectiveness of machine learning techniques for predicting query latency of static and dynamic workload scenarios. They argue that query performance prediction using machine learning is both feasible and effective.

Related to the Semantic Web query processing, SPARQL query engines can be categorized into two categories: SQL-based and RDF native query engines [28]. SQL-based query engines rely on relational database systems storage and query optimization techniques to efficiently evaluate SPARQL queries. They suffer from the same problems mentioned above. Furthermore, due to the absence of schematic structure in RDF, cost-based approaches – successful in relational database systems – do not perform well in SPARQL query processing [28]. RDF native query engines typically use heuristics and statistics about the data for selecting efficient query execution plans [27]. Heuristics-based optimization techniques include exploiting syntactic and structural variations of triple patterns in a query [27], and rewriting a query using algebraic optimization techniques [12] and transformation rules [15]. Heuristics-based optimization techniques generally work without any knowledge of the underlying data. Stocker *et al.* [27] present optimization techniques with pre-computed statistics for reordering triple patterns in a SPARQL query for efficient query processing. However, in many use-cases involving querying Linked Data, statistics are often missing [28]. This makes these statistics-based approaches ineffective in the Linked Data scenario. Furthermore, as in the case of relation database systems, these existing approaches are unable to predict actual query performance metrics such as query execution time for a given configuration.

Related to explanations for the Semantic Web, Inference Web [20, 21] explanation infrastructure addresses the explanation requirements of Semantic Web applications and exposes its explanation metadata in RDF using Proof Markup Language (PML) [26]. Inference Web provides a set of software tools for building, presenting, maintaining, and manipulating PML proofs. Inference Web provides a centralized registry based solution for publishing explanation metadata from distributed reasoners. The WIQA (Web Information Quality Assessment) framework [6] and KOIOS semantic search engine [11] provide explanations of their reasoning and expose their explanation metadata in RDF. However, they provide

application specific explanations which include process descriptions of specific algorithms. Although researchers [4, 21] highlighted the need for short explanations, previous work has not addressed the problem of providing short explanations. But researchers have studied ontology summarization. A notable work on ontology summarization is RDF sentence graph-based summarization [29]. This work extracts and summarizes RDF sentences based on centrality measures.

3 Problem Statement and Contributions

We address the problems of SPARQL query performance prediction and explaining reasoning over Linked Data. Our aim is to assisting users in querying and consuming Linked Data. In querying Linked Data, we provide performance related information to help understand how a query may behave. Users can use this information for query construction and refinement, workload management, and query scheduling. Also, SPARQL query optimizers can use our prediction models for query plan selection. In consuming Linked Data, we explain how a given piece of data was derived. Users can use such explanations to understand and debug Linked Data. In contrast to the previous work, we propose a decentralized solution to address explanations in the distributed setting of Linked Data. Our explanations are suitable for generic Linked Data scenarios; unlike WIQA and KOIOS. Finally, we address the problem of summarizing explanations. The main goal of summarizing explanations is twofold: (a) providing a brief overview of the background information used in the reasoning, (b) providing an entry point to the full explanation. Our approach is similar to sentence graph summarization. However, we define new measures for summarizing explanations. As an application of our research, we aim to apply our methodologies and strategies for processing and explaining distributed queries on Linked Data.

4 Research Methodology and Approach

4.1 Predicting SPARQL Query Performance

To predict query performance metrics prior to query execution, we apply machine learning techniques on the logs of executed queries. We work with query execution time as the query performance metric. We treat the SPARQL engine as a black box and learn query behaviors from the behaviors of already executed queries. This approach does not require any statistics of the underlying RDF data, which makes it ideal for the Linked Data scenario. A key challenge in applying machine learning for SPARQL query performance prediction is to represent SPARQL queries as feature vectors. We use the frequencies and the cardinalities of SPARQL algebra operators[1] of a query as its features. Additionally, to represent the query patterns as features, we first cluster the training queries based on the structural similarities between the graphs constructed from

[1] `http://www.w3.org/TR/sparql11-query/#sparqlQuery`

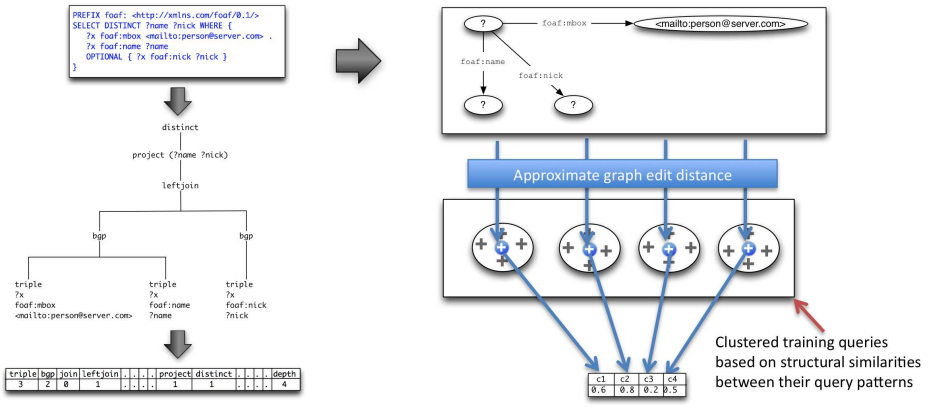

Fig. 1. Example of extracting SPARQL feature vector from a SPARQL query.

the query patterns of the training queries. Each dimension of the query pattern feature vector for a query is the structural similarity score between the graph represented by the query pattern belonging to a cluster center query and the graph represented by the query pattern belonging to the query. We use a polynomial time suboptimal solution of approximate graph edit distance problem [24] to compute the structural similarities between the graphs represented by query patterns. We use the k-mediods [19] clustering algorithm with approximate graph edit distance as the distance function to cluster the training queries. Figure 1 shows an example of extracting the feature vector from a SPARQL query. We experiment on predicting query execution times using the support vector machine regression (SVR) [25] with the SPARQL algebra features, using SVR with SPARQL algebra and query pattern features, using multiple SVRs for queries with different execution time ranges with SPARQL algebra and query pattern features, and finally a single k-nearest neighbors regression (k-NN) [3] with SPARQL algebra and query pattern features. We describe the results of our experiments in section 5.1.

4.2 Generating and Summarizing Explanations

We follow the Linked Data principles [5] to publish explanation metadata. We describe these metadata using our proposed vocabulary *Ratio4TA*[2]. We generate explanations by retrieving the explanation metatada by following their dereferenceable URIs and presenting them in a human understandable form. We define *Ratio4TA* as an extension of the W3C PROV Ontology[3]. This promotes interoperability by enabling data consumers to process explanation metadata

[2] `http://ns.inria.fr/ratio4ta/`
[3] `http://www.w3.org/TR/prov-o/`

according to W3C PROV standards. *Ratio4TA* allows describing data, reasoning processes, results, data derivations, rules, and software applications. We use the named graph mechanism [8] to make statements about RDF triples. Using named graph allows us to associate explanation metadata for data with different levels of granularity – explanation metadata for a triple or a graph containing more than one triple. Furthermore, we use named graphs to group together explanation metadata and make the metadata for an explanation referenceable by a single URI. We opt for our own vocabulary because the prominent previous work PML has limitations with respect to Linked Data common practices. PML uses RDF container concepts. RDF containers use blank nodes to connect a sequence of items [1]. However, as a common practice, blank nodes are avoided while publishing Linked Data [17].

We define five measures to summarize explanations: salience (S_{SL}), similarity (S_{SM}), abstractness (S_{AB}), salience with respect to proof tree (S_{ST}), and coherence (S_{CO}). We compute salience of an RDF statement by combining the normalized degree centrality scores of the subject and the object of the statement. We use the measures salience, similarity, and abstractness for ranking. For the similarity measure, users can specify a set of concepts as their explanation filtering criteria. We rank the more similar statements to the concepts given in filtering criteria higher. We use the approximate query solving feature of Corese [10] for similarity computations. For abstractness, we consider a statement that is close to the root in corresponding proof tree is more abstract than a statement that is far from the root. We use salience with respect to proof tree and coherence measures to re-rank already ranked statements. We compute salience with respect to proof tree for an RDF statement by taking the average score (computed using combinations of ranking measures) of all statements of the tree that the statement roots in the corresponding proof tree. Finally we consider an RDF statement to be coherent to an RDF statement if the first statement is directly derived from the second statement. We summarize the RDF statements in an explanation using combinations of these measures.

5 Preliminary Results

5.1 Query Performance Prediction Results

We randomly select 6000 queries from DBPSB [22] DBpedia[4] query log dataset. Then we run them on a locally loaded DBpedia 3.8 into a Jena TDB triple store[5] to record their execution times. We split the 6000 queries into 60% training, 20% validation, and 20% test splits. We use R^2 (coefficient of determination) values to evaluate our regression predictions. R^2 measures how well the regression approximates the real data points. An R^2 value of 1 means that the regression perfectly fits the data. Figure 2 shows log-log plots of the predicted and actual query execution times for the test queries for different learning methods we experimented with. Our first experiment using SVR with only SPARQL algebra

[4] `http://dbpedia.org`
[5] Jena TDB: `http://jena.apache.org/documentation/tdb`

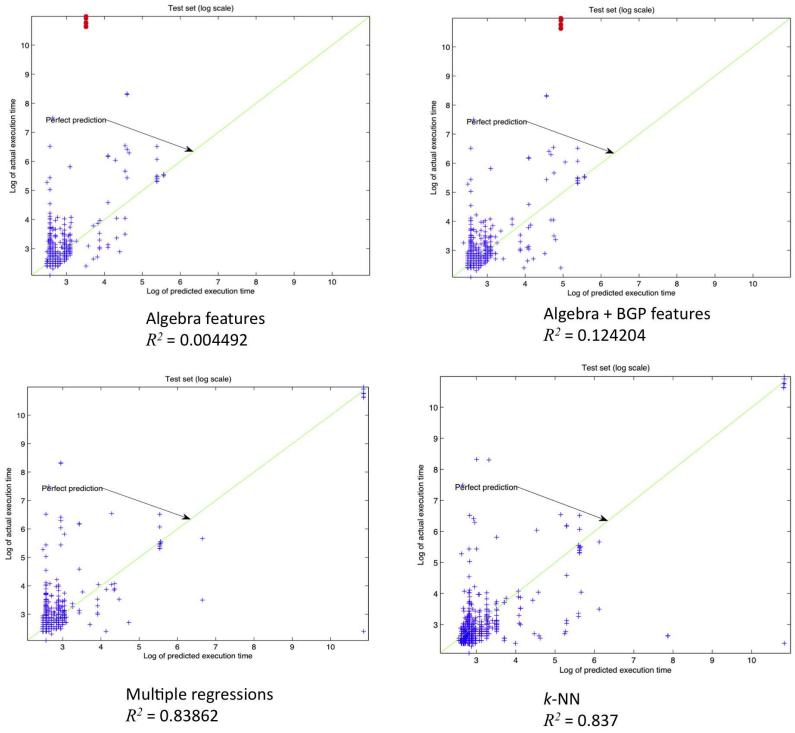

Fig. 2. Comparison of learning methods.

features performs poorly with a low R^2 value of 0.004492. We experiment with another SVR using SPARQL algebra features and query pattern features. The R^2 value improves to 0.124204 which is still very low with outliers – highlighted in red – far from the perfect prediction line. A possible reason for this is the fact that our training dataset has queries with various different time ranges. Fitting a curve in such irregular data points is often inaccurate. To address this, we first split our training data according to execution time ranges, then we train different regressions for different time ranges. We cluster the training data into X clusters based on the execution times using x-means [23] clustering algorithm. We use x-means because it automatically chooses the number of clusters. We then train X number of SVM regressions with training queries corresponding to cluster for each regression. As features, we use SPARQL algebra and query pattern features. Then we train a Support Vector Machine (SVM) classifier [9] with a training dataset containing all the training queries and the cluster number of each query as the label for the queries. For an unseen query, we first predict the cluster for the query using the SVM classifier, then we predict the execution time using the SVM regression that corresponds to the predicted cluster for that query. The accuracy of the SVM classifier on our test dataset is 96.0833%. This means that we can accurately predict the execution time ranges of unseen queries. The overall R^2 value on our test dataset with this approach jumps to 0.83862. This is demonstrated by the long running queries moving very close to the perfect

prediction line. Also more queries moved towards the perfect prediction line than before. In our final experiment, we train the regression variant of k-NN algorithm with SPARQL algebra and query pattern features. We achieve an R^2 value of 0.837 on the test dataset. The result of k-NN and multiple regressions are almost same. However, the complexity of training the k-NN regression is less. Also the concentration of the short running queries near the perfect prediction line is more for k-NN.

5.2 Explanation Summarization Results

We evaluate our summarization approach by comparing the summarized explanations generated by our approach and ground truth summarized explanations generated by humans. We obtained our ground truths by surveying 24 people from different backgrounds. We used three test cases – three queries with their results along with the explanations for the results. Each query result is an inferred statement by our reasoner. Each test case has two scenarios: without filtering criteria FL, and with filtering criteria FL. Each participant answered questions for one test case. We randomly assigned a test case to a participant. We ask the participants to rate, from a scale of 1 to 5, the need for each of the statements in the explanation. For, the scenario with filtering criteria FL, we give the query,

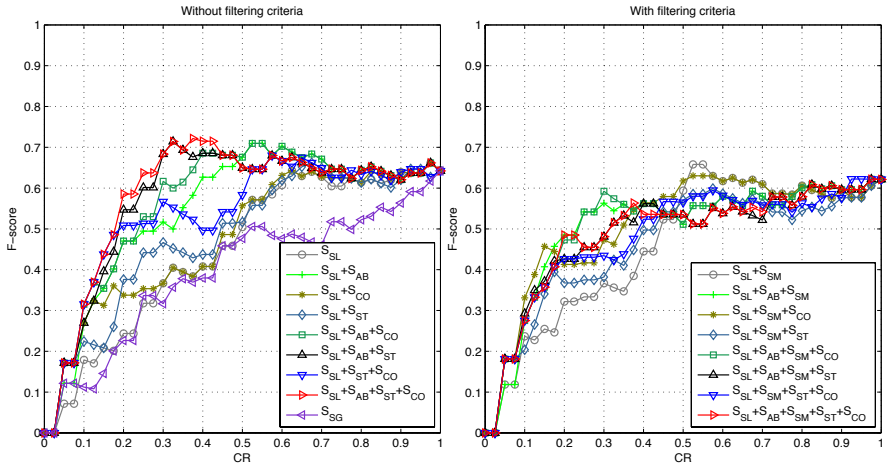

Fig. 3. Compression ratio (CR) vs *F-score*.

the answer, and the explanation but with a user's filtering criteria class taken from the schemata used in the reasoning process. The explanations, the questionnaires, the responses, and the results of the evaluation are publicly available online[6]. We evaluate different combinations of the summarization measures we define. For the scenario without FL, we also compare our summaries to sentence graph summarization – denoted as S_{SG}. We evaluate evaluate summaries of different sizes by measuring *F-score* for summarized explanations with different

[6] http://ns.inria.fr/ratio4ta/sm/

compression ratios, CR. To generate the ground truth summarized explanation for an explanation, we include a statement in the ground truth summarized explanation if its rating is greater than or equal to the average rating of all the statements in the original explanation. *F-scores* reflects the accuracy of automatically generated summaries with respect to the ground truth summary. A desirable situation would be a summarized explanation with high *F-score* and low CR. Figure 3 shows the average *F-scores* for different measure combinations for summaries with different sizes for the three test cases. The *x-axis* represents compression ratio CR. The *y-axis* represents *F-scores*. For the scenario without FL, the best *F-score* is 0.72 when CR value is 0.33 by the measure combinations $S_{SL} + S_{AB} + S_{ST}$ and $S_{SL} + S_{AB} + S_{ST} + S_{CO}$. This is a desirable situation with a high *F-score* and low CR. The sentence graph summarization performs poorly with a best *F-score* value of 0.34 in the CR interval 0.05 to 0.3. For the scenario with FL, the best *F-score* is 0.66 at CR values 0.53 and 0.55 by the measure combination$S_{SL} + S_{SM}$. However, the *F-score* 0.6 at CR value 0.3 by the measure combination $S_{SL} + S_{AB} + S_{SM} + S_{CO}$ is more desirable because the size of the summary is smaller. As expected, our summarization approach perform worse in the scenario with FL where we use S_{SM}. This is due to the fact that the survey participants had to consider the highly subjective factor of similarity. An overview of our work on generating and summarizing explanations for Linked Data is available in [16].

6 Evaluation Plan

Our future plan includes evaluating three more aspects. First, we would like to evaluate our prediction methods using SPARQL benchmark queries. DBPSB includes 25 query templates for evaluating SPARQL engines with DBPedia dataset. Our aim would be to generate training, validation, and test datasets from these query templates and evaluating our approach using them. Second, we would like to evaluate our prediction methods for query plan selection for SPARQL query processing over Linked Data. A possible direction for this would to use our query performance prediction approach for selecting efficient query plans in federated SPARQL query processing. Third, we will evaluate the impact of explanations and summarized explanations on end-users for a selected domain.

7 Conclusion

In this paper, firstly we study the techniques to predict SPARQL query performance. We learn query execution times from query history using machine learning techniques. This approach does not require any statistics of the underlying RDF data, which makes it ideal for the Linked Data scenario. We achieved high accuracy ($R^2 = 0.84$) for predicting query execution time.

Secondly we discuss how to generate and summarize explanations for Linked Data. We present an ontology to describe explanation metadata and discuss

publishing explanation metadata as Linked Data. In addition, we presented five summarization measures to summarize explanations. We evaluate different combinations of these measures. The evaluation shows that our approach produces high quality rankings for summarizing explanation statements. Our summarized explanations are also highly accurate with *F-score* values ranging from 0.6 to 0.72 for small summaries. Our approach outperforms the sentence graph based ontology summarization approach.

Acknowledgments: This work is supported by the ANR CONTINT program under the Kolflow project (ANR-2010-CORD-021-02).

References

1. RDF semantics. W3C recommendation (2004)
2. Akdere, M., Cetintemel, U., Riondato, M., Upfal, E., Zdonik, S.: Learning-based query performance modeling and prediction. In: Data Engineering (ICDE), 2012 IEEE 28th International Conference on. pp. 390–401 (2012)
3. Altman, N.: An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician 46(3), 175–185 (1992)
4. Angele, J., Moench, E., Oppermann, H., Staab, S., Wenke, D.: Ontology-based query and answering in chemistry: Ontonova project halo. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) The Semantic Web - ISWC 2003, LNCS, vol. 2870, pp. 913–928. Springer Berlin / Heidelberg (2003)
5. Berners-Lee, T.: Linked Data. W3C Design Issues `http://www.w3.org/DesignIssues/LinkedData.html` (2006)
6. Bizer, C.: Quality-Driven Information Filtering in the Context of Web-Based Information Systems. Ph.D. thesis, Freie Universität Berlin, Universitätsbibliothek (2007)
7. Bonatti, P., Hogan, A., Polleres, A., Sauro, L.: Robust and scalable linked data reasoning incorporating provenance and trust annotations. Web Semantics: Science, Services and Agents on the World Wide Web 9(2), 165–201 (2011)
8. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. In: Proceedings of the 14th international conference on World Wide Web. pp. 613–622. WWW '05, ACM, New York, NY, USA (2005)
9. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011), software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
10. Corby, O., Dieng-Kuntz, R., Gandon, F., Faron-Zucker, C.: Searching the Semantic Web: approximate query processing based on ontologies. Intelligent Systems, IEEE 21(1), 20–27 (2006)
11. Forcher, B., Sintek, M., Roth-Berghofer, T., Dengel, A.: Explanation-aware system design of the semantic search engine koios. In: Proc. of the the 5th Int'l. Workshop on Explanation-aware Computing (2010)
12. Frasincar, F., Houben, G.J., Vdovjak, R., Barna, P.: RAL: An algebra for querying RDF. World Wide Web 7(1), 83–109 (2004)
13. Ganapathi, A., Kuno, H., Dayal, U., Wiener, J.L., Fox, A., Jordan, M., Patterson, D.: Predicting multiple metrics for queries: Better decisions enabled by machine

learning. In: Proceedings of the 2009 IEEE International Conference on Data Engineering. pp. 592–603. ICDE '09, IEEE Computer Society, Washington, DC, USA (2009)
14. Gupta, C., Mehta, A., Dayal, U.: PQR: Predicting query execution times for autonomous workload management. In: Proceedings of the 2008 International Conference on Autonomic Computing. pp. 13–22. ICAC '08, IEEE Computer Society, Washington, DC, USA (2008)
15. Hartig, O., Heese, R.: The sparql query graph model for query optimization. In: Proceedings of the 4th European Conference on The Semantic Web: Research and Applications. pp. 564–578. ESWC '07, Springer-Verlag, Berlin, Heidelberg (2007)
16. Hasan, R.: Generating and summarizing explanations for linked data. In: Proc. of the 11th Extended Semantic Web Conference (2014), (to appear)
17. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Morgan & Claypool, 1st edn. (2011), http://linkeddatabook.com/
18. Huang, J., Abadi, D.J., Ren, K.: Scalable SPARQL querying of large RDF graphs. Proceedings of the VLDB Endowment 4(11), 1123–1134 (2011)
19. Kaufman, L., Rousseeuw, P.: Clustering by means of medoids. In: Dodge, Y. (ed.) Statistical Data Analysis based on the L1 Norm, p. 405416 (1987)
20. McGuinness, D., Furtado, V., Pinheiro da Silva, P., Ding, L., Glass, A., Chang, C.: Explaining semantic web applications. In: Semantic Web Engineering in the Knowledge Society (2008)
21. McGuinness, D., Pinheiro da Silva, P.: Explaining answers from the semantic web: the inference web approach. Web Semantics: Science, Services and Agents on the World Wide Web 1(4), 397 – 413 (2004)
22. Morsey, M., Lehmann, J., Auer, S., Ngonga Ngomo, A.C.: Dbpedia SPARQL benchmark performance assessment with real queries on real data. In: Aroyo, L., et al. (eds.) The Semantic Web ISWC 2011, LNCS, vol. 7031, pp. 454–469. Springer Berlin Heidelberg (2011)
23. Pelleg, D., Moore, A.W.: X-means: Extending K-means with efficient estimation of the number of clusters. In: Proceedings of the Seventeenth International Conference on Machine Learning. pp. 727–734. ICML '00, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2000)
24. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. Image Vision Comput. 27(7), 950–959 (Jun 2009)
25. Shevade, S.K., Keerthi, S.S., Bhattacharyya, C., Murthy, K.R.K.: Improvements to the SMO algorithm for SVM regression. Neural Networks, IEEE Transactions on 11(5), 1188–1193 (2000)
26. Pinheiro da Silva, P., McGuinness, D., Fikes, R.: A proof markup language for semantic web services. Information Systems 31(4-5), 381–395 (2006)
27. Stocker, M., Seaborne, A., Bernstein, A., Kiefer, C., Reynolds, D.: SPARQL basic graph pattern optimization using selectivity estimation. In: Proceedings of the 17th International Conference on World Wide Web. pp. 595–604. WWW '08, ACM, New York, NY, USA (2008)
28. Tsialiamanis, P., Sidirourgos, L., Fundulaki, I., Christophides, V., Boncz, P.: Heuristics-based query optimisation for SPARQL. In: Proceedings of the 15th International Conference on Extending Database Technology. pp. 324–335. EDBT '12, ACM, New York, NY, USA (2012)
29. Zhang, X., Cheng, G., Qu, Y.: Ontology summarization based on RDF sentence graph. In: Proceedings of the 16th international conference on World Wide Web. pp. 707–716. WWW '07, ACM, New York, NY, USA (2007)