Survey of RDF Keyword Query Techniques

Jiang Dong-yang
Liaoning Mechatronics College, Dandong, P.R. China, 118009
linda164@163.com

Abstract. With the development of the Semantic Web and information extraction technologies, more and more available RDF data has emerged on the Web. It has become one of the research hot spots in semantic Web technology. In this paper, in terms of RDF data organization and storage, as well as query expression, query processing. The analysis and comparison of RDF keyword query approaches are presented deeply and future research challenges are put forward.

1. Introduction
In recent years, with the rapid development of World Wide Web, large-scale available RDF data is released. RDF has been used by many projects and institutions to represent their metadata, such as DBLP, Wikipedia, etc. IBM Smart Earth Research Center [1] uses RDF data description and integrated semantics; the meta-information of many fields such as sports and film is expressed by RDF, and the fields of biology, chemistry, biomedicine, geography and other fields build the domain ontology on RDF. In Fig. 1, a RDF graph is shown. According to the incomplete statistics from W3C's SWEO (Semantic Web Education and Outreach) research group, as of April 2018, the number of various RDF triples on the Internet (including Linked Open Data, YAGO[2], DBpedia [3] and other RDF datasets) have reached 82 billion. The Internet has shifted from a document web that contains only web pages and hyperlinks between web pages to a data web that can describe relationships between entities. Document retrieval in the traditional Internet usually takes HTML pages (documents) as an information carrier, and user's queries returns a number of documents containing these query keywords, while the semantic Web data search user hits a triplet Descriptive entity. The RDF keyword query has become an important research topic of today's semantic web technology.

Fig. 1 A RDF graph
2. Research Status and Analysis of RDF Keyword Search at Home and Abroad

In recent years, major search engine companies, including Google, Baidu, Sogou, etc., have built knowledge graphs by RDF data models to improve search quality. According to different storage methods for RDF data, it is divided into the keyword query centralized RDF data and keyword query distributed RDF data [4]. In this section the research status of RDF data keywords at home and abroad are introduced. In addition, the related work of querying RDF data by query translation is described.

2.1 Keyword Query for Centralized RDF data

In this paper three methods of querying RDF data keywords based on centralized storage are introduced: RDF data keyword query of relational database, RDF data keyword query based on graph, and RDF data keyword query by query translation.

2.1.1 RDF Keyword Query of Relational Database

RDF data keyword query of the relational database generally uses mature relational database technology to store and query RDF data. The RDF triples are organized according to the relational model. Currently, the three most common RDF data storage methods of the relational database are triple tables, attribute tables, and vertical storage respectively.

In the triple table all triples are stored in a relatively large triple table. Earlier RDF data management systems, such as 3-Store[6], Sesame[7], etc., all use R triple table for RDF data storage. However, performing a chained join and a star join on a triple table requires a large number of self-join operations, which greatly reduces the query performance. In order to improve query efficiency, many systems need to build a large number of indexes on triple tables. Oracle RDF [8] uses triplet tables to store and extract schema description information and stores them separately by the tables to build regular indexes. [9] uses relationships to store and build multiple indexes, including primary and secondary indexes. [10] proposes a horizontal partitioning method where each row in the table is a triple with the same subject.

In attribute tables triples are classified according to predicates. Each relation table stores a class. Each table uses the horizontal division technology. By classifying the predicates, avoiding too many columns in the horizontal division and reducing the storage space and the number of self-joins between the subjects can be effectively, thereby improving the query performance. However, in practice, there will be queries for unknown predicates, leading to the join or merge operation of multiple tables, which cannot avoid a large number of null values. Among them, Jena [11] used attribute table and uses SQL statements for query.

In the vertical storage the RDF triple table is divided vertically according to predicates. The triples of the same predicate are in a table. The predicate can be the table name. Only the subject (S) and object (O) columns are retained. And each table is sorted according to the subject, so that the query can be quickly joined. In [12], a vertical storage scheme is exploited. In order to speed up the query, only relevant columns were read. [13] uses a column storage scheme. However, because the subject has a corresponding row on each list, a large number of null values exist, which prevents the problem of sparse data. Table 1 shows the comparison of RDF keyword query techniques of relational database.

In addition, a basic triple SPO index is used. The SPO index is an index of the subject, predicate and object. According to the respective order of the SPO, the indexes are sequentially built on S, P, and O. RDF-3X[8] and Hexastore[14] both use this exhaustive method to build a complete index of SPO, including indexes of SPO, SOP, PSO, POS, OSP, and OPS. Based on the relational database schema, SPARQK[15], and SQAK[16] firstly set up a schema graph, then, by the relationship between the foreign key or primary key between the tables, find the join tree containing the key nodes, and finally infer the query schema structure, such as SQL queries.

| Storage types  | Methods    | Index types                  |
|---------------|------------|------------------------------|
| The triple table | 3-Store[6] | Jena[11]                  |

Table 1. Comparison of RDF keyword query techniques of relational database
2.1.2 RDF Keyword Query for Graph-based RDF data

RDF data can be represented as directed graphs with labeled vertices and edges. Thus, RDF data directly is stored to be the RDF graph in many studies, and the graph traversal algorithm is used to perform keyword queries on the RDF data. The RDF data keyword query is transformed into a subgraph matching on large graphs. In the RDF graph, the label information of the vertex, and the out-edge list of the vertex. Most RDF data keyword query based on graphs reduce the search space through effective indexes, limit the search scope to smaller sub-graphs, and then find the corresponding query results on the subgraphs. GRIN first uses the clustering method to re-divide all the nodes in the RDF data graph. By clustering the subjects, the vertices in the graph are respectively clustered into C disjoint sets, and then for each the center points and radii of the clusters are calculated, and a tree-like index is established based on these clustering results. The divided the data, and then established a tree index-PIG (parameterizable index graph).

[21] designed query schemes by grouping Steiner trees. The problem of keyword search is defined as finding a result tree T with the shortest path between nodes and containing the most keyword information on a data graph G with edge weight or no weight. Although the group Steiner tree problem is an NP-complete problem, it can be solved using an approximate algorithm. BANKS, BANKS II, and BLINKS are all group Steiner tree problems based on approximation algorithms. BLINKS divides the graph into subgraphs or blocks, and then establishes the index between blocks and the shortest path index between nodes within the block. The search starts from a certain block containing at least one keyword entered by the user, and uses a backward search algorithm to search the RDF graph. Each search selects the incoming edge of the node that has just been visited to access the source node, and the source node and the shortest path to the source node are added to the result tree until the root nodes of all keyword nodes are found.

2.1.3 RDF Keyword Search by Query Translation

The RDF keyword query by query translation has received more and more attention. Generally, firstly the RDF pattern information from the basic RDF data is extracted. Then, a summary index for query translation is set up, and by the summary index it is found relationships between the entities containing the keywords. The subgraph with the relationships can be converted into a formal query language, and finally, results are sorted and returned back to the user. The user selects a satisfactory query to execute by the existing query search engine and the final results are obtained. Because the scale of the summary index for query translation is much smaller than the RDF data graph, it is high that the efficiency of constructing a query algorithm on the summary index. In addition, the converted queries can be processed through existing query search engine optimization techniques, so that better query performance can be achieved.

At present, there are not many researches on keyword query of RDF data based on the translation from keyword query to SPARQL query. The typical literatures are. [25] extracts RDF graph structure information from RDF data, constructs structural indexes for searching, and constructs joint queries through these indexes. In [26], a ranking model of probabilistic queries is used to select the most likely SPARQL query. Like most statistical methods, the query results are affected by the statistical model built. In [28], in order to better express the user's query intent, users were provided with an incremental step options to construct SPARQL queries. However, this method requires users to have certain domain knowledge. [29] translates the query of keywords into structured queries. This method needs to manually label the query log, use the labeled query log to train a semantic word classifier, and also obtain some structured queries through the labeled query log template. Therefore,
this method is not practical for large-scale open RDF data sets.

2.2 RDF Keyword Query of Distributed Storage
Nowadays RDF data on the World Wide Web has also shown a massive and dynamic growth trend. This requires RDF data keyword query technology can also be applied to dynamic large-scale RDF data sets. Query processing on such a data set requires the query technique to be scalable in order to perform real-time processing on the continuously updated data set. Although the RDF data keyword technology used on the centralized data set cannot be directly applied to the distributed data set in most cases, the keyword query of the centralized data can still be used to store RDF data at each node of the distribution. So the research of centralized RDF keyword query method has very important research value.

[30] store RDF data distributed, and in each node relational database technology is used to store data. This is applicable in a distributed environment. With the rapid development of cloud computing technology, the RDF keywords query technology based on cloud platforms has also gradually developed. [31] uses distributed file system (HDFS) to organize and store data under cloud platforms. And in order to distribute the data at each node, the graph structure is used for division. [32] according to the structural characteristics of RDF graphs, the nodes close to each other in the graph are divided into the same machine node; then according to the predicate of the triples, the document divides the data into individual Different files, and then divide the predicate file into smaller files by object type.

3. Conclusion
Since the research on unstructured data and big data in recent years, the management of RDF data, especially query processing, has been a research hot. From the perspective of dataset's storage, the existing RDF query processing technologies are analyzed and compared, including query expression, query processing technology, and query optimization. However, due to space limitations, only a part of the current research on RDF data query is analyzed and discussed, and some work (such as workshops in some conferences) has not been mentioned. Whether it is a static data set or a query plan on a dynamic data set, most of them require a large number of indexing mechanisms and redundant storage. It is also worthwhile to build lightweight indexes on a big data to reduce data redundancy, and provide efficient query processing. Graph processing technology under the parallel framework of BSP programming model, with its more efficient support for complex diagram iterative algorithm, could be an important research topic of massive RDF data query processing under the cloud environment.

References
[1] IBM smart planet[EB/OL]. http://www.ibm.com/developerworks/cn/web/wa-aj-smartweb/index.html
[2] Mahdisoltani F, Biega J, and Suchanek F. Yago3: A knowledge base from multilingual wikipedias[C]. In CIDR, 2014.
[3] Lehmann J, Isele R, Jakob M, Jentzsch A, Kontokostas D, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from Wikipedia[J]. Semantic Web, 6(2):167-195, 2015.
[4] Du F, Chen YG, Du XY. Survey of RDF Query Processing Techniques, Journal of Software,2013,24(6):1222-1242[J].
[5] Zou L, Özsu M T, Chen L, Shen X, Huang R, and Zhao D. gstore: a graph-based sparql query engine[C]. VLDB J, 2014, 23(4):565-590.
[6] Harris S, Gibbins N. 3store: Efficient bulk RDF storage, In Proceedings of the 1st Int’l Workshop on Practical and Scalable Semantic Systems[C]. Sanibel Island: CEUR-WS Verlag, 2003.
[7] Broekstra J, Kampman A, van Harmelen F. Sesame: A generic architecture for storing and querying RDF and RDF schema[C]. In Proceedings of the 1st Int’l Semantic Web Conf. (ISWC 2002), LNCS 2342, Sardinia: Springer-Verlag, 2002, 54-68.
[8] Murray C, Alexander N, Das S, Eadon G, Ravada S. Oracle spatial resource description framework
(RDF), 10g Release 2, 2005.

[9] Lu J, Ma L, Zhang L, Brunner JS, Wang C, Pan Y, Yu Y. SOR: A practical system for ontology storage, reasoning and search[C]. In Proceedings of the 33rd Int’l Conf. on Very Large Data Bases (VLDB 2007), Austria: ACM Press, 2007, 1402-1405.

[10] Chu E, Baid A, Chen T, Doan AH, Naughton J. A relational approach to incrementally extracting and querying structure in unstructured data[C]. In Proceedings of the 33rd Int’l Conf. on Very Large Data Bases (VLDB 2007), Austria: ACM Press, 2007, 1045-1056.

[11] Carroll JJ, Dickinson I, Dollin C, Reynolds D, Seaborne A, Wilkinson K. Jena: Implementing the semantic Web recommendations[C]. In Proceedings of the 13th Int’l World Wide Web Conf. on Alternate Track Papers & Posters (WWW 2004), New York: ACM Press, 2004, 74-83.

[12] Abadi DJ, Marcus A, Madden SR, Hollenbach K. Sw-Store: A vertically partitioned database system for semantic Web data management[C]. The VLDB Journal, 2009, 18(2):385-406.

[13] Sidirourgos L, Goncalves R, Kersten M, Nes N, Manegold S. Column-Store support for RDF data management: Not all swans are white[C]. In Proceedings of the VLDB Endowments, 2008, 1(2):1553-1563.

[14] Weiss C, Karras P, Bernstein A. Hexastore: Sextuple indexing for semantic Web data management[C]. In Proceedings of the VLDB Endowments, 2008, 1(1):1008-1019.

[15] Luo Y, Lin X, Wang W, and Zhou X. Spark: top-k keyword query in relational databases[C]. In SIGMOD, 2007, 115-126.

[16] Tata S and Lohman G M. Sqak: doing more with keywords[C]. In SIGMOD, 2008, 889-902.

[17] Zou L, Özsu M T, Chen L, Shen X, Huang R, and Zhao D. gstore: a graph-based sparql query engine[C]. VLDB J, 2014, 23(4):565-590.

[18] Udrea O, Pugliese A, Subrahmanian VS. Grin: A graph based RDF index[C]. In Proceedings of the 23rd AAAI Conf. on Artificial Intelligence, Vancouver: AAAI Press, 2007, 1465-1470.

[19] Tran T, Ladwig G. Structure index for RDF data[C]. In Proceedings of the Workshop on Semantic Data Management, Singapore: CEUR-WS Press, 2010, 20-25.

[20] He H, Wang HX, Yang J, Yu PS. Blinks: Ranked keyword searches on graphs[C]. In Proceedings of ACM SIGMOD international conference on Management of Data, Beijing: ACM Press, 2007, 305-316.

[21] Li HY, Qu YZ. A Keyword query approach on RDF data[J]. Journal of Southeast University (Natural Science Edition), 2012, 40(2):270-274.

[22] Chekufi C, Even G Kortsarz G A greedy approximation algorithm for the group Steiner problem[J]. Discrete Applied Mathematics, 2006, 154(1): 15-34.

[23] Bhalotia G, Hulgeri A, Nakhe C, Chakrabarti S, and Sudarshan S. Keyword searching and browsing in databases using banks[C]. In ICDE, 2002, 431-440.

[24] Gkirtzou K, Papastefanatos G, and Dalamagas T. RDF keyword search based on keywords-to-SPARQL translation[C]. In the First International Workshop on Novel Web Search Interface and Systems (NWSearch’15), 2015, 3-5.

[25] Ladwig G and Tran T. Combining Query Translation with Query Answering for Efficient Keyword Search[C]. International Conference on the Semantic Web: Research and Applications, 2010, 288-303.

[26] Zhou Q, Wang C, Xiong M, Wang H, Yu Y. SPARK: Adapting Keyword Query to Semantic Search[C]. International the Semantic Web and Asian Conference on Asian Semantic Web Conference, 2007, 4825:694-707.

[27] Tran T, Wang H, Rudolph S, and Cimiano P. Top-k exploration of query candidates for efficient keyword search on graph-shaped (RDF) data[C]. In IEEE International Conference on Data Engineering, 2009, 405-416.

[28] Zenz G, Zhou X, Wolf Bierskens E, and Nejdl W. From keywords to semantic queries-incremental query construction on the semantic web[J]. Journal of Web Semantics, 2009, 7(3): 166-176.

[29] Pound J, Hudek A K, Ilyas I F, and Weddell G. Interpreting keyword queries over web knowledge bases[C]. In CIKM, 2012, 305-314.
[30] Stuckenschmidt H, Vdovjak R, Houben GJ, Broekstra J. Index structures and algorithms for querying distributed RDF repositories[C]. In Proceedings of the 13th Int’l Conf. on World Wide Web, New York: ACM Press, 2004, 631-639.

[31] Kaoudi Z, Manolescu I. RDF in the clouds: a survey[J]. VLDB Journal, 2015, 24(1):67-91.

[32] Huang JW, Abadi DJ, Ren K. Scalable SPARQL querying of large RDF graphs[C]. In Proceedings of the VLDB Endowment, 2011, 4(11): 1123-1134.