International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 04 | Apr-2016 www.irjet.net p-ISSN: 2395-0072

A survey on: Keyword search and similarity using RDF schema

Surbhi Gujrani[1], Anupama Phakatkar[2]
[1] *Student, Department of computer engineering*
PICT, Pune, India.
[2] *Professor, Department of computer engineering*
PICT, Pune, India.

Abstract - *The increase in the world of internet and information has given rise to a lot of information stored on the web. All the information stored in the World Wide Web has semantics and relevance these days. Searching in this pool of information on the web is a very tedious task. Keyword search similarity is an important tool for exploring and searching large data repositories whose structure is either unknown, or constantly changing. The current existing systems define various techniques that work on searching the information semantically. These techniques have various limitations that give rise to many problems in the web. If the data is organized in a definite schema then the efficient results can be easily obtained. This paper focuses on various techniques that focus on searching of keywords using RDF schema and obtaining similarity in the web using different techniques. The main focus is on the systems that use partitioning and graph-structured techniques for searching.*

***Key Words*: Semantic web, Graph structure, Keyword search, RDF graph.**

1.INTRODUCTION

The RDF (Resource Description Framework) data sets are explored for searching various keywords using various tools. There are many techniques the explorations depend upon inclusive the construction of a distance matrix and comparing it with threshold for pruning and summary building from RDF graphs [12]. Many domains contain RDF data from different hundreds of sources which in turn contain triples associated to it. Keyword search is an important tool for exploring and searching large data repositories whose structure is either unknown, or constantly changing. The other basic solutions also have many disadvantages. They may perform well on data with a topological structure but are less efficient for unstructured or semi-structured databases [16]. The goal is to design scalable and exact solution that can handle tens of millions of triples. Basically, the RDF data set is considered as a triple which consists of subject, object and predicate. The triple is considered as a directed edge connecting the subject to the object. The directed edge it uses to connect is called as a predicate [17].

An RDF data is a graph which is made of different entities and relationships. The entities are represented by vertices and the edges represent the relationships between these entities called as predicates. Formally, we view an RDF data set as an RDF graph G = (V,E) where, V is the union of disjoint sets, V_E, V_T and V_W; V_E is the set of entity vertices, V_T is the set of type vertices, and V_W is a set of keyword vertices. E is the union of disjoint sets, E_R, E_A, and E_T; E_R is the set of entity-entity, E_A is the set of entity-keyword edges and E_T is the set entity-type edges. The following diagram shows a sample of RDF dataset containing different keywords.

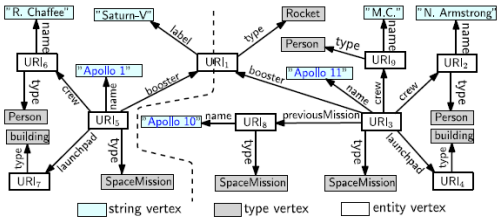

Fig-1: DBpedia dataset

1.1 Motivation

Query processing over graph-structured data has attracted much attention recently, as application from a variety of areas continue to produce large volumes of graph-structured data. In semantic web, two major standards, RDF and OWL, conform to node-labeled and edge-labeled graph models. There are various existing techniques that suffer from major searching limitations. The strongest ones being:
a) The result of the keyword search is incorrect.
b) The scalability problem which cannot handle millions of problems at a given time [9].

The main goal should be to overcome these problems that are the largest occurring ones in the keyword search of large RDF data. It is been shown that finding sub graphs rather than trees is more useful and informative for the users [2]. However, the current tree or graph based methods may produce answers in which some content nodes are not very close to each other. The basic motivation of this survey is to study systems that return correct searching results along with scalable and efficient answering of search queries [12]. It uses ranking function which works by partitioning and searching the RDF data. The effective system must lead to pruning of the unnecessary data by using correct methodology without sacrificing the soundness of the result. The ranking function is gaining interests of many applications because of two main reasons: first, user-friendly query interface does not require users to master complex query language or understand the underlying data schema. Second, many query languages are more suitable for well-structured schema. In this system, the RDF data is constrained on the basis of types and the summation of the structure is done using these types in RDF graphs and it is used to prove how it increases the search speed.

2. EXISTING SYSTEMS

The summary based evaluation proposed in [1] states that the RDF graph must be partitioned into type based sub graphs. These sub graphs should be used for query evaluation. The basic methodology proposed in this paper is very efficient but does not suit for large databases. In [2] and [8], the database is a large repository stored as graphs. The graph structure contains various vertices and edges represent the relationship between them. In [2] the concept of r-clique is used to obtain subgraphs of similar semantics. The problem arising in this approach is that either the relevant vertices are too large to handle or the results are erroneous. In [9], a distance matrix is maintained for all pairs of vertices of graph. If the data set is too large then this approach is very inefficient because the matrix maintenance becomes infeasible. [10] handles the typographical and orthographical errors by taking into consideration the user's query context. It does not relate the error to the dataset but compares it with the user's history to find the appropriate match. Other major problems taken into consideration in [3] are distance constraint, keyword constraint, search time constraint, index constraint and memory constraint. This approach is very efficient to obtain the keyword search with low memory consumption. The various heterogenous functions of graphs are taken into consideration in [11] which proposes a 3-in-1 approach to find rankings of all the subgraphs in RDF schema. When certain subgraphs are to joint to obtain the required results, pruning techniques need to be very efficient. In [12], scorebounds and thresholds are used for pruning. These pruning techniques give speedy results but less relevant. [13] and [14] propose searching methods for structured, unstructured and semi-structured databases. The results give high accuracy and better speed. Again storing the database as graphs becomes a problem because adjacency matrix is used as a storage data structure. Another method explored in the recent times is code-search method [6]. The search results obtained using this gives a set of execution paths each containing all the keywords. This method is not efficient in terms of minimal results. The proposed system is developed upon the summarization technique in [1]. It further enhances the partitioning scheme to give better results. Similar to summarization, another technique that gained popularity is proposed in [5] named as candidate network generation and evaluation. It extracts the frequent patterns and then uses a ranking function to obtain which of the keywords might be more similar. [4] deals with eradicating duplicate nodes even when the nodes are connected differently in different answers. The problem of finding duplication-free results is studied in this paper.

When semantic search comes into picture, the method of obtaining semantic similarity using terms and their frequency is very popular. One such method is proposed in [7] which is based on similarity graph that contains the degree of semantic similarity between terms. Keyword search basically gained its popularity on databases. Keyword search on relational database is already been studied in [15], [16], [17], [18]. The basic idea used for searching in databases relies upon top-k query processing. The pruning methods used here are required to be very effective. The answers to the query in databases give out results for top-k most relevant results which is not satisfactory in all situations. Another issue encountered in searching on databases is that the pruning methods are unable to capture the interesting relationships that are hidden in the databases.

3. PROPOSED SYSTEM

In the survey done, many disadvantages of the backward search methods have led to the conclusion that this method is not efficient for scalable keyword search. The techniques used can consider backward search as a paradigm but not as a full proof approach. Here, type-based summarization is idealized which states that partitioning need to be performed for each type and then individually we can use the backward search method on each partition [1]. The idea is to induce partition on the whole RDF graph G. The keywords being queried will be concatenated by each

partition and then will be further generalized to form the actual result. The most important factor to be considered here is the partitioning factor and how pruning of partitions is to be performed when a query is fired [1], [15].

4. CONCLUSION

The problem of scalable keyword search on large RDF data assumes a lot of issues including the size of the RDF data and the incorrectness of problems. The paper studies and compares various methods that define different approaches which include summarization and partitioning techniques. The most efficient way of partitioning the data obtained from this is by searching space and then formulating SPARQL queries. The query taken in the proposed system is simply a list of keywords and does not contain any relation between the attributes present inside.

REFERENCES

[1] Anastasios Kementsietsidis, "Scalable Keyword Search on Large RDF Data", *IEEE Transactions on Knowledge & Data Engineering*, , no. 1, pp. 1, PrePrints PrePrints, 2014.

[2] M. Kargar and A. An, "Keyword Search in Graphs: Finding Rcliques," *Proc. VLDB Endowment, vol. 4, pp. 681-692, 2011.*

[3] Yue Wang; Ke Wang; Fu, A.W.-C.; Wong, R.C.-W., "KeyLabel algorithms for keyword search in large graphs," in *Big Data (Big Data), 2015 IEEE International Conference,2015.*

[4] Kargar, M.; Aijun An; Xiaohui Yu, "Efficient Duplication Free and Minimal Keyword Search in Graphs," in *Knowledge and Data Engineering, IEEE Transactions on* , vol.26, no.7, pp.1657-1669, July 2014.

[5] Jing Zhou; Xiaohui Yu; Yang Liu; Ziqiang Yu, "Ranking Keyword Search Results with Query Logs," in *Big Data (BigData Congress), 2014 IEEE International Congress in* Bigdata.

[6] Kamiya, T., "An algorithm for keyword search on an execution path," in *Software Maintenance, Reengineering and Reverse Engineering (CSMR-WCRE), 2014 Software Evolution Week - IEEE Conference on* , vol., no., pp.328-332, 3-6 Feb. 2014.

[7] Stanchev, L., "Semantic search using a similarity graph," in *Semantic Computing (ICSC), 2015 IEEE International Conference on* , vol., no., pp.93-100, 7-9 Feb. 2015.

[8] K. Golenberg, B. Kimelfeld, and Y. Sagiv, "Keyword Proximity Search in Complex Data Graphs," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD)*, 2008.

[9] H. He, H. Wang, J. Yang, and P.S. Yu, "Blinks: Ranked Keyword Searches on Graphs," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '07),* 2007.

[10] H. Fu and K. Anyanwu, "Effectively Interpreting Keyword Queries on RDF Databases with a Rear View," *Proc. 10th Int'l Conf. Semantic Web (ISWC),* 2011.

[11] Y. Luo, W. Wang, and X. Lin, "SPARK: A Keyword Search Engine on Relational Databases," *Proc. IEEE 24th Int'l Conf. Data Eng. (ICDE)*, 2008.

[12] Xiang Lian, Lei Chen, Member, IEEE, and Zi Huang, Member, IEEE "Keyword Search Over Probabilistic RDF Graphs" in *Knowledge and Data Engineering, IEEE Transactions on* , vol.27, no.5, pp.1246-1260, May 1 2015.

[13] G. Li, B.C. Ooi, J. Feng, J. Wang, and L. Zhou, "Ease: Efficient and Adaptive Keyword Search on Unstructured, Semi-structured and Structured Data," *Proc. ACM SIGMOD Int'l Conf. Management of Data*, 2008.

[14] T. Tran, H. Wang, S. Rudolph, and P. Cimiano, "Top-k Exploration of Query Candidates for Efficient Keyword Search on Graph-Shaped (RDF) Data," *Proc. IEEE Int'l Conf. Data Eng. (ICDE)*, 2009.

[15] S. Agrawal, S. Chaudhuri, and G. Das, "DBXplorer: Enabling Keyword Search over Relational Databases," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD)*, 2002.

[16] Y. Chen, W. Wang, and Z. Liu, "Keyword-Based Search and Exploration on Databases," *Proc. 27th Int'l Conf. Data Eng. (ICDE)*, 2011.

[17] V. Hristidis, L. Gravano, and Y. Papakonstantinou, "Efficient IRStyle Keyword Search Over Relational Databases," *Proc. 29th Int'l Conf. Very Large Data Bases (VLDB),* 2003.

[18] A. Polleres, "From SPARQL to Rules (and Back)," *Proc. 16th Int'l Conf. World Wide Web (WWW),* 2007.