International Journal of Advanced Technology in Engineering and Science
Vol. No.4, Issue No. 06, June 2016
www.ijates.com
ijates
ISSN 2348 - 7550

# IMPLEMENTATION OF EFFICIENT KEYWORD ROUTING IN LINKED DATA

**M. Azharuddin[1], Ayeesha Hakeem[2]**

[1]*Assistant professor, Dept. of Computer Science Engineering, S.I.E.T College of Engineering and Technology, Vijayapur, Karnataka, (India)*

[2]*Student, Dept. of Computer Science Engineering, S.I.E.T College of Engineering and Technology, Vijayapur, Karnataka, (India)*

## ABSTRACT

*Keyword search is an intuitive paradigm for searching linked data sources on the web. We propose to route keywords only to relevant sources to reduce the high cost of processing keyword search queries over all sources. We propose a novel method for computing top-k routing plans based on their potentials to contain results for a given keyword query. We employ a keyword-element relationship summary that compactly represents relationships between keywords and the data elements mentioning them. A multilevel scoring mechanism is proposed for computing the relevance of routing plans based on scores at the level of keywords, data elements, element sets, and sub-graphs that connect these elements.*

**Keywords:** ***Database, Keyword query Search, Keyword-element Relationship, Linked Data, Routing***

## I. INTRODUCTION

In recent years the Web has evolved from a global information space of linked documents to one where both documents and data are linked. Underpinning this evolution is a set of best practices for publishing and connecting structured data on the Web known as Linked Data. The adoption of the Linked Data best practices has lead to the extension of the Web with a global data space connecting data from diverse domains such as people, companies, books, scientific publications, films, music, television and radio programs, genes, proteins, drugs and clinical trials, online communities, statistical and scientific data, and reviews. This Web of Data enables new types of applications. There are generic Linked Data browsers which allow users to start browsing in one data source and then navigate along links into related data sources. There are Linked Data search engines that crawl the Web of Data by following links between data sources and provide expressive query capabilities over aggregated data, similar to how a local database is queried today. The Web of Data also opens up new possibilities for domain-specific applications. Unlike Web 2.0 mash ups which work against a fixed set of data sources, Linked Data applications operate on top of an unbound, global data space. This enables them to deliver more complete answers as new data sources appear on the Web.

Investigation of the problem of keyword query routing for keyword search over a large number of structured and Linked Data sources. Implementation of Routing keywords only to relevant sources. This can reduce the high cost of searching for structured results that span multiple sources. To the best of our knowledge, the work presented in this paper represents the first attempt to address this problem.

A graph-based data model is used to characterize individual data sources. In that model, we distinguish between an element-level data graph representing relationships between individual data elements, and a set-level data graph, which captures information about group of elements.

## II. LITERATURE SURVEY

H. He, H. Wang, J. Yang, and P.S. Yu proposed BLINKS [1]: Ranked Keyword Searches on Graphs Query processing over graph-structured data is enjoying a growing number of applications. A top-*k* keyword search query on a graph finds the top *k* answers according to some ranking criteria, where each answer is a substructure of the graph containing all query keywords. Current techniques for supporting such queries on general graphs suffer from several drawbacks, e.g., poor worst-case performance, not taking full advantage of indexes, and high memory requirements. To address these problems, BLINKS was proposed, a bi-level indexing and query processing scheme for top-*k* keyword search on graphs. BLINKS follow a search strategy with provable performance bounds, while additionally exploiting a bi-level index for pruning and accelerating the search. To reduce the index space, BLINKS partitions a data graph into blocks: The bi-level index stores summary information at the block level to initiate and guide search among blocks, and more detailed information for each block to accelerate search within blocks. The experiments show that BLINKS offers orders-of-magnitude performance improvement over existing approaches.

R. Goldman and J. Widom proposed DataGuides[2]: Enabling Query Formulation and Optimization in Semi structured Databases In semi structured databases there is no schema fixed in advance. To provide the benefits of a schema in such environments, DataGuides was introduced, concise and accurate structural summaries of semi structured databases. DataGuides serve as dynamic schemas, generated from the database; they are useful for browsing database structure, formulating queries, storing information such as statistics and sample values, and enabling query optimization. It presents the theoretical foundations of DataGuides along with algorithms for their creation and incremental maintenance.

B. Ding, J.X. Yu, S. Wang, L. Qin, X. Zhang, and X. Lin proposed Finding Top-k Min-Cost Connected Trees[3] in Databases It is widely realized that the integration of database and information retrieval techniques will provide users with a wide range of high quality services. In this paper, it shows processing an l-keyword query, p1; p2; _ _ _; pl, against a relational database which can be modeled as a weighted graph, G(V;E). Here V is a set of nodes (tuples) and E is a set of edges representing foreign key references between tuples. Let Vi _ V be a set of nodes that contain the keyword pi. It shows finding top-k minimum cost connected trees that contain at least one node in every subset Vi, and denote our problem as GST-k. When k = 1, it is known as a minimum cost group Steiner tree problem which is NP Complete. It is observed that the number of keywords, l, is small, and propose a novel parameterized solution, with l as a parameter, to find the optimal GST-1, in time complexity O (3ln + 2l((l + log n)n +m)), where n and m are the numbers of nodes and edges in graph G. given

solution can handle graphs with a large number of nodes. OurGST-1 solution can be easily extended to support GST-k, which outperforms the existing GST-k solutions over both weighted undirected/directed graphs.

V. Kacholia, S. Pandit, S. Chakrabarti, S. Sudarshan, R. Desai, and H. Karambelkar proposed Bidirectional Expansion for Keyword Search on Graph Databases [4] .Relational, XML and HTML data can be represented as graphs with entities as nodes and relationships as edges. Text is associated with nodes and possibly edges. Keyword search on such graphs has received much attention lately. A central problem in this scenario is to efficiently extract from the data graph a small number of the "best" answer trees. A Backward Expanding search, starting at nodes matching keywords and working up toward confluent roots, is commonly used for predominantly text-driven queries. But it can perform poorly if some keywords match many nodes, or some node has very large degree. In this paper a new search algorithm is proposed, Bidirectional Search, which improves on Backward Expanding search by allowing forward search from potential roots towards leaves. To exploit this flexibility, a novel search frontier prioritization technique based on spreading activation is devised.

L. Qin, J.X. Yu, and L. Chang proposed Keyword Search in Databases[5]: The Power of RDBMS. Keyword search in relational databases (RDBs) has been extensively studied recently. A keyword search (or a keyword query) in RDBs is specified by a set of keywords to explore the interconnected tuple structures in an RDB that cannot be easily identified using SQL on RDBMSs. In brief, it finds how the tuples containing the given keywords are connected via sequences of connections (foreign key references) among tuples in an RDB. Such interconnected tuple structures can be found as connected trees up to a certain size, sets of tuples that are reachable from a root tuple within a radius, or even multi-center sub-graphs within a radius. In the literature, there are two main approaches. One is to generate a set of relational algebra expressions and evaluate every such expression using SQL on an RDBMS directly or in a middleware on top of an RDBMS indirectly. Due to a large number of relational algebra expressions needed to process, most of the existing works take a middleware approach without fully utilizing RDBMSs. The other is to materialize an RDB as a graph and find the interconnected tuple structures using graph-based algorithms in memory. In this paper it is focused on using SQL to compute all the interconnected tuple structures for a given keyword query. Three types of inter-connected tuple structures are used to achieve that and we control the size of the structures. It is shown that the current commercial RDBMSs are powerful enough to support such keyword queries in RDBs efficiently without any additional new indexing to be built and maintained. The main idea behind this approach is tuple reduction. In our approach, in the first reduction step, prune tuples that do not participate in any results using SQL, and in the second join step, process the relational algebra expressions using SQL over the reduced relations.

V. Hristidis and Y. Papakonstantinou proposed DISCOVER[6] operates on relational databases and facilitates information discovery on them by allowing its user to issue keyword queries without any knowledge of the database schema or SQL. DISCOVER proceeds qualified joining of networks of tuples, that is, sets of tuples that are related because they join on their primary and foreign keys and collectively contain all the keywords of the query. Also proves that the selection of the optimal execution plan (way to reuse common sub expressions) is NP-complete.

G. Li, B.C. Ooi, J. Feng, J. Wang, and L. Zhou proposed that Conventional keyword search engines[7] are restricted to a given data model and cannot easily adapt to unstructured, semi structured or structured data. This paper proposes an efficient and adaptive keyword search method, called EASE, this is for indexing and querying

large collections of heterogeneous data. In order to achieve high efficiency in processing keyword queries, first form unstructured, semi-structured and structured data as graphs, and then sum up the graphs and construct graph indices instead of using traditional inverted indices. Put forward an extended inverted index to facilitate keyword-based search, and present a novel ranking mechanism for enhancing search effectiveness.

## III. PROBLEM DEFINITION

The web is no longer only a collection of textual documents but also a web of interlinked data sources. One prominent project that largely contributes to this development is Linking Open Data. It is difficult for the typical web users to exploit this web data by means of structured queries using languages like SQL or SPARQL. To this end, keyword search has proven to be intuitive. Keyword Search is the Information Retrieval Methodology. In which users are not aware of a query language as well as the structure of data. The problem defines of the keyword search over linked database. Investigation of the problem of keyword query routing for keyword search over a large number of structured and Linked Data sources. Routing keywords only to relevant sources can reduce the high cost of searching for structured results that span multiple sources. Also reducing the number of potential results that increase exponentially with the number of links and sources between them and eliminating the search results from irrelevant sources and concentrating more results from relevant sources.

## IV. ARCHITECTURE

Routing keyword to the relevant sources reduces higher processing cost of query on all sources. We propose a novel method to generate top k-routing plans that contain the requested query keyword. Unlike the existing system that employs the binary relationship between keywords where there will be a lot of false sub-trees, we employ element level relationship by developing a graph between the keywords at elementary levels. Fig.1 gives a high-level overview of the interrelationships between elements at different level and search space in general. Keywords mentioned in any entity description at the element level are linked with set-level element with a relation like type. The set-level elements are present in the sources. Thus there will be an advantage if both the queried keyword elements are connected via a path. A correct routing plan is then selected based on the graphs generated based on relationships between keywords present in the query. This relationship is considered at different levels like element level, set level or keyword level. The final goal is to generate a plan that can search for the keyword from multiple sources and produce the most relevant results.

In record recovery, numerous question extension strategies are taking into account data contained in the top-positioned recovered records in reaction to the unique client inquiry. Essentially, our methodology is in light of performing a starting recovery of assets agreeing to the unique keyword query. Thereafter, further assets will be inferred by utilizing the first recovered ones.

The inter-relationships between elements at different levels are illustrated in Fig.1. A keyword is mentioned in some entity descriptions at the element level. Entities at the element level are associated with a set-level element via type. A set-level element is contained in a source. There is an edge between two keywords if two elements at the element level mentioning these keywords are connected via a path.

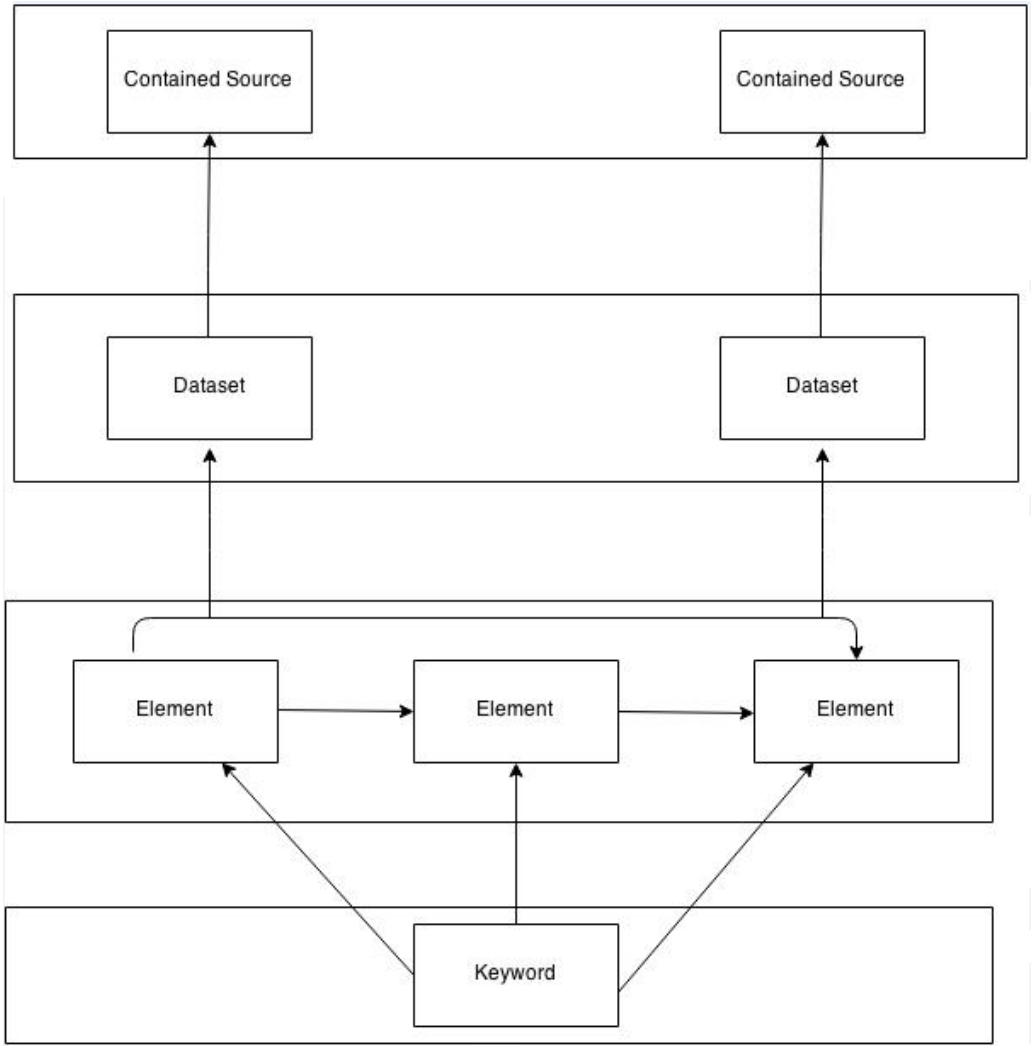


**Figure 1: Multi-Level Inter-Relation Graph**

The system is proposed to have the following modules along with functional requirements.

1. Keyword Search
2. Keyword Query Routing
3. Multilevel Inter-Relationship
4. Set - Level

### 4.1 Keyword Search

There are schema-based approaches implemented on top of off-the-shelf databases .A keyword query is processed by mapping keywords to elements of the database (called keyword elements). Then, using the schema, valid join sequences are derived, which are then employed to join ("connect") the computed keyword elements to form so-called candidate networks representing possible results to the keyword query. Schema-agnostic approaches operate directly on the data. Structured results are computed by exploring the underlying data graph. The goal is to find structures in the data called Steiner trees (Steiner graphs in general), which connect keyword elements.

### 4.2 Keyword Query Routing

We propose to investigate the problem of keyword query routing for keyword search over large number of structured and Linked Data sources. Routing keywords only to relevant sources can reduce the high cost of searching for structured results that span multiple sources. To the best of our knowledge, the work presented in this paper represents the first attempt to address this problem. A solution to keyword query routing can address these problems by pruning unpromising sources and enabling users to select combinations that more likely contain relevant results. For the routing problem, we do not need to compute results capturing specific elements at the data level, but can focus on the more coarse-grained level of sources.

### 4.3 Multilevel Inter-Relationship

The search space of keyword query routing using a multilevel inter-relationship graph. The inter-relationships between elements at different levels are shown in Fig 2.1. A keyword is mentioned in some entity descriptions at the element level. Entities at the element level are associated with a set-level element via type. A set-level element is contained in a source. There is an edge between two keywords if two elements at the element level mentioning these keywords are connected via a path. We propose a ranking scheme that deals with relevance at many levels.

### 4.4 Set Level

We extract keywords and relationships from the data. Then, based on the elements and sets of elements in which they occur, we create keyword-element relationships. Pre-computing relationships (i.e., paths) between data elements are typically performed for keyword search to improve online performance. These relationships are stored in specialized indexes and retrieved at the time of keyword query processing to accelerate the search for Steiner graphs. For database selection, relationships between keywords are also pre-computed. This work neither considers relationships between keywords nor relationships between data elements but between keyword-elements that collectively represent the keywords and the data elements in which they occur.

## V. RESULTS AND DISCUSSIONS

Routing has a large positive effect on the performance of keyword search. Keyword search without routing is especially problematic when the number of keywords is large. Hence, routing can be seen as a promising alternative paradigm especially for cases, where the information need is well described and available as a large amount of texts.

The merits of this project are as follows

- It is a web based application
- User can get their information with appropriate multimedia that will help them to grasp information easily.
- User can also send request about their query to any other users in the community.

## VI. CONCLUSION

This paper proposes the idea of routing keyword query to produce more relevant results by implementing relationship graphs between the keywords at different levels. This idea proposes to reduce the high cost of searching for structured data spanning across multiple resources by routing the keywords only to the relevant sources. A correct routing plan will be selected by using graphs developed based on the relationships between keywords in the query at different level. This project is tested with a database having 856 records in four different datasets. The records in the datasets are created such that each dataset will some kind of data about the keywords. By this we created a web of data similar to the Linked data on the internet where information about a keyword may encompass on different sources.

Queries with more keywords would also generate effective results, but they cannot be handled efficiently. For example, if we give a query with more keywords as a query in the existing system, it would also give effective results, but it might take a higher time which is not desirable in a present day's demand of high responsiveness. Keyword search without routing is problematic when the query has many words. That is the reason for routing of queries having more number of keywords.

## REFERENCES

[1]. H.He, H. Wang, J. Yang, and P.S. Yu, "Blinks: Ranked Keyword Searches on Graphs," Proc. ACM SIGMOD Conf., pp. 305-316, 2007.

[2]. R.Goldman and J. Widom, "DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases," Proc. 23rd Int'l Conf. Very Large Data Bases (VLDB), pp. 436-445, 1997.

[3]. B. Ding, J.X. Yu, S. Wang, L. Qin, X. Zhang, and X. Lin, "Finding Top-K Min-Cost Connected Trees in Databases," Proc. IEEE 23rd Int'l Conf. Data Eng. (ICDE), pp. 836-845, 2007.

[4]. V.Kacholia, S. Pandit, S. Chakrabarti, S. Sudarshan, R. Desai, and H. Karambelkar, "Bidirectional Expansion for Keyword Search on Graph Databases," Proc. 31st Int'l Conf. Very Large Data Bases (VLDB), pp. 505-516, 2005.

[5]. L. Qin, J.X. Yu, and L. Chang, "Keyword Search in Databases: The Power of RDBMS," Proc. ACM SIGMOD Conf., pp. 681-694, 2009.

[6]. V. Hristidis and Y. Papakonstantinou, "Discover: Keyword Search in Relational Databases," Proc. 28th Int'l Conf. Very Large Data Bases (VLDB), pp. 670-681, 2002.

[7]. G. Li, B.C. Ooi, J. Feng, J. Wang, and L. Zhou, "Ease: An Effective 3-in-1 Keyword Search Method for Unstructured, Semi-Structured and Structured Data," Proc. ACM SIGMOD Conf., pp. 903-914, 2008.

[8]. B.Yu, G. Li, K.R. Sollins, and A.K.H. Tung, "Effective Keyword- Based Selection of Relational Databases," Proc. ACM SIGMOD Conf., pp. 139-150, 2007.

[9]. Y. Luo, X. Lin, W. Wang, and X. Zhou, "Spark: Top-K Keyword Query in Relational Databases," Proc. ACM SIGMOD Conf., pp. 115-126, 2007.

[10]. G. Li, S. Ji, C. Li, and J. Feng, "Efficient Type-Ahead Search on Relational Data: A Tastier Approach," Proc. ACM SIGMOD Conf., pp. 695-706, 2009.

[11]. M. Sayyadian, H. LeKhac, A. Doan, and L. Gravano, "Efficient Keyword Search Across Heterogeneous Relational Databases," Proc. IEEE 23rd Int'l Conf. Data Eng. (ICDE), pp. 346-355, 2007.

[12]. V. Hristidis, L. Gravano, and Y. Papakonstantinou, "Efficient IR-Style Keyword Search over Relational Databases," Proc. 29th Int'l Conf. Very Large Data Bases (VLDB), pp. 850 861, 2003.

[13]. Q.H. Vu, B.C. Ooi, D. Papadias, and A.K.H. Tung, "A Graph Method for Keyword-Based Selection of the Top-K Databases," Proc. ACM SIGMOD Conf., pp. 915-926, 2008.

[14]. T. Tran, H. Wang, and P. Haase, "Hermes: Data Web Search on a Pay-as-You-Go Integration Infrastructure," J. Web Semantics, vol. 7, no. 3, pp. 189-203, 2009.

[15]. F. Liu, C.T. Yu, W. Meng, and A. Chowdhury,"Effective Keyword Search in Relational Databases," Proc. ACM SIGMOD Conf., pp. 563-574, 2006.

[16]. G. Ladwig and T. Tran, "Index Structures and Top-K Join Algorithms for Native Keyword Search Databases," Proc. 20th ACM Int'l Conf. Information and Knowledge Management (CIKM),pp. 1505-1514, 2011.