Middle-East Journal of Scientific Research 24 (12): 3711-3715, 2016
ISSN 1990-9233
© IDOSI Publications, 2016
DOI: 10.5829/idosi.mejsr.2016.3711.3715

Effective Dynamic Query Processing in Keyword Search

[1]M. Indra Devi and [2]G. Balakrishnan

[1]Department of CSE Kamaraj College of Engineering and Technology, Virudhunagar, India
[2]Department of CSE, K.L.N. College of Information Technology, Pottapalayam, Sivaganga District, India

Abstract: Now a days, retrieving data in multiple databases is more complex one. Connecting and implementing database queries in large amount of database is the problematic one. To solve this problem, keyword search is the effective solution in database research. This paper proposes an effective method to search keywords in different databases using dynamic query processing and implementing steiner tree. This method consists of two parts. One is searching keywords in different databases using key-level relationship and the second one is implementing keyword search to analyze effective results. Experimental results shows optimal performance in keyword search and reduce the search time in search.

Key words: Keyword search · Steiner tree · Key-level relationship

INTRODUCTION

Keyword search is an interesting approach in database research. Keyword search focuses an effective keyword query that retrieves relevant results in different databases. Connecting multiple databases without affecting their structure and information is the interesting approach in keyword search. It is difficult to implement SQL query concept directly in this approach but without SQL, we cannot achieve anything in structured databases. This concept is the base of web linked data. Web is a collection of textual documents and that are linked another one. These web linked data have different structured data sources containing more number of RDF triples. RDF triples contains lot of web links and these are links and its structure is more complex.

Keyword search applies both databases and web linked data. This involves to analyze the database structure. Database consists of SQL queries to create two dimensional tables and tables are linked two one another by using integrity constraints. This creates large number data containing the data source that are structured and linked to one another.

In this, we investigate the problem of retrieving data in large number linked databases and analyze the database using its structure. Many SQL queries have implemented to analyse this problem but SQL queries focuses only in connecting of tables. If the table's data increases then query executing time increases. There are lot of query optimization policies are implemented to create an effective queries in query analysis but all are focusing in a separate databases. Linked data analysis in databases is having no effective performance using these optimization techniques. In this paper, we provide the following contributions.

- Analyze database integrity constraints and identify the effective relationships between the fields. Using the relationship, connect all tables by using effective route.
- Implement steiner tree to reduce the repeated path in the database table routes. In this, we propose a method to create a complete link between all tables in the database.
- Generate dynamic queries using these links to get keywords in the database. Apply keywords search implement IR style ranking method to get the effective results.

We implement the approach in 20 normalized data sets. Normalized data sets are implemented integrity constraints and analyzed the multilevel relevance datasets, such as considered as keywords or entities.

Related Work

Database Anaysis: Modern database have large number of entities. These entities are sometimes structured. Entities are linked to web information and retrieving data

Corresponding Author: M. Indra Devi, Department of CSE Kamaraj College of Engineering and Technology, Virudhunagar, India.

from normal queries in these entities become more complex. Customized query processing mechanisms focuses these types of problems. Query forms are analyzing theses issues [1]. Database tables are considered as two types, one is relation-based records and another one is tuple based records. Relation-based records utilize the schema in the database but tuple-based records implements the integrity constraints in the databases. In this approach, tuples and its foreign key references are stored and retrieve keywords in using the links in graph approach [4].

Previous works define a database keyword relationship that connects relational database tables and using relevant structures. These works focuses keyword search over structured database queries [7]. Effective databases have fully normalized form, in this integrity constraint identification is the important work and analyzing the foreign-key relationship between tables implements the links in the databases. This work avoids the repeated relationship in the databases and identifies the normalized link in the database.

Approaches: Existing approaches implements keyword query search in relational databases using schema based and without schema based. Database relevant keywords are processed using schema and applying queries to take keywords in the database. This approach is implemented only in schema based [2]. There is no mid-work in this approach; keywords are taken directly by implementing this approach.

Instead of schema-based approach, database structure analysis is the important work. In this, database structure is completely analyzed using integrity constraints and that structure will be converted in a pictorial form by analyzing repeated links. Steiner trees and graphs [2] are implemented to compute this process. This process implements dynamic query programming.

Now days, multisource schema matching techniques are implemented in various approaches [4], it is the schema analyzing and relevant links in different schemas are employed in this approach. Data base structure is the main key of this method.

In this paper, we analyze the structure approach in database and implements streiner tree method to analyze the integrity constraint in database. IR style ranking method is implemented in the method to discover effective keyword search over databases.

Proposed Work: The proposed work analysis database structure and identify the key-relationships then implementing the Steiner tree approach. Steiner tree elements are stored to generate dynamic queries for retrieving effective keywords in the database. Initially keywords are captured in set level and then extracted to element level. Set level analysis implemented from dynamic query form results. This approach combines both database structure and keywords. Element level approach implements the results from set level approach and identify the keyword based relationship in the approach.

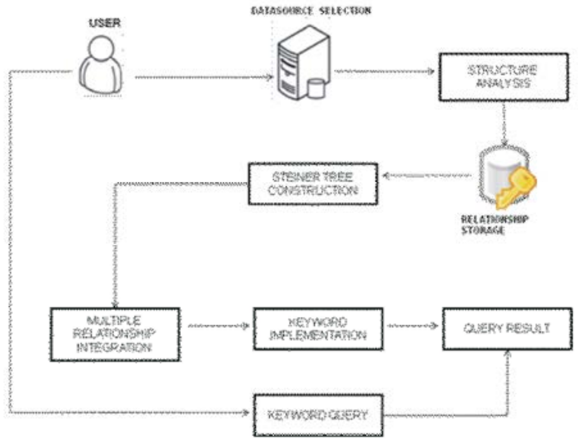

Fig. 3.1: System Architecture

Initially set level key constraint analysis is implemented. In this Given a dataset D, candidate key C_k and the foreign key F_k then, a Set S_i

$$S_i = \{ C_{k1}, C_{k2}\ldots\ldots C_{kn}\} \rightarrow \{F_{k1},F_{k2}\ldots F_{kn}\} \quad (1)$$

In the (1) equation each candidate key in the set is linked with the foreign keys, then, this system finds the one or more foreign keys linked with each candidate key, for example a candidate key C_{ki} may be referenced with one or more foreign keys,

$$\{F_{ki}, F_{kj}\} \rightarrow C_{ki} \quad (2)$$

$$\{F_{ki\,j}\} \rightarrow C_{ki} \quad (3)$$

We implement this to create set-level key element matrix and extract that matrix elements to identify the internal relationship between candidate keys and foreign keys.

$M(s_i) = \{ * F_{k1}\ F_{k2} \ldots\ldots F_{kn} \}$
C_{k1} 0 1 ……. 1
C_{k2} 1 0 ……..0
.
.
C_{kn} 1 0 ……. 1 (4)

In this matrix we identify the key relationship between candidate keys and foreign keys as distance and count distance if one candidate key refers multiple foreign keys then Matrix of the set can be

$D(M(s_i))= \{* \; F_{k1} \; F_{k2} \; ..... \; F_{kn}\}$
C_{k1} 0 1 ……. 1
C_{k2} 1 0 ……..2
.
.
C_{kn} 4 1 ……. 3 (5)

Distance level set Matrix is extracted from maximum distance by using the following rule

$$C_{ki} \rightarrow d\,(F_{k1\ldots n}) \quad (6)$$

In every key C_k relationship $C_{ki} \rightarrow F_{ki}$ consists the sub relationship in $S_p = C_{ki} \rightarrow F_{ki}$ covers the additional distance d in the relationship. We extract the matrix distance for each relationship and move the distance value to element level.

Element level key relationship identifies internal relationship distances and avoids it to reduce the distance in the key relationship, for example a candidate key relationship link,

$$L_i = C_{k1} \rightarrow F_{k1} \rightarrow C_{k2} \rightarrow F_{k2} \rightarrow C_{k3} \rightarrow F_{k3} \quad (7)$$

Then

$$L_j = C_{k2} \rightarrow F_{k2} \rightarrow C_{k3} \rightarrow F_{k3} \quad (8)$$

Link L_i distance is 5 and Link L_j distance is 3 but link L_j is the sub-link of Link L_i and the this distance is the internal distance so it is avoidable distance, this can be identified in element level key relationships.

$$S_i = L_i \leftarrow L_j \quad (9)$$

Element level key relationship also covers some additional links with previous links and that link is added to previous link, for example a candidate key relationship link,

$$L_i = C_{k1} \rightarrow F_{k1} \rightarrow C_{k2} \rightarrow F_{k2} \rightarrow C_{k3} \rightarrow F_{k3} \quad (10)$$

Then

$$L_j = C_{k2} \rightarrow F_{k2} \rightarrow C_{k3} \rightarrow F_{k3} \rightarrow C_{k4} \rightarrow F_{k4} \quad (11)$$

Some links L_j is the sub link of L_i then L_j is avoided and the additional link in L_j is added to the L_i then L_j distance is avoided and L_i distance increases. Then

$$L_i = C_{k1} \rightarrow F_{k1} \rightarrow C_{k2} \rightarrow F_{k2} \rightarrow C_{k3} \rightarrow F_{k3} \rightarrow C_{k4} \rightarrow F_{k4} \quad (12)$$

The following part implements dynamic query formation using set-level and element level key relationships and analyzes multisource key relationship implementation.

Key Constraints Analysis: Data source of the database is analyzed and its SQL structured definitions are identified. Selected database table's integrity constraint is checked and also column structure analyzed. The main part of this work is to identify the primary key-foreign key relationships and column type analysis is also the important work. Different data sources have different column structures and different key implementations. All are analyzed and identified.

Key Relationship Implementation: Set level and element level key relationship are implemented. In each level, we can analyze the distance of each link and identifies the maximum links in set-level key relationship implementation. Element level key relationship implementation reduces link distance by eliminating repeated sub links and joining additional links. After this, we can implement Steiner tree implementations to join all links to create a tree form. In this link hierarchy is identified.

Keyword Integration: We can identify keyword by applying keyword integration algorithm to generate dynamic queries for fetching keyword set from the Steiner tree links in the database. We can apply this process repeatedly in different databases to identify the common keyword dataset and common key relationship links. Stop word removing process is also implemented. We can filter repeated keywords in the dataset from different databases and the keyword dataset is stored in a repository with its identifier.

Ranking Process: We implement IR style ranking process in the keyword searching from the stored repository. We search the keyword from the repository and searching process implemented the IR style ranking to retrieve the most relevant data for this keyword and analyze the fetching time in this process. Effective results reduce the fetching time and this can be done by the ranking process. To compute ranking process, we generate runtime queries to repository based on Steiner tree links and calculate the distance from the repository.

Algorithm Implementation: The Algorithm implements to get keyword set from Steiner tree links and analyzes its edges between element level key relationships and captures possibly complex paths can be directly retrieved and employed for multiple links. Standard query join is implemented to calculate the possible link distances. While other standard join implementations can be used, the system employs hash join in the experiment. Because the order of joins does not have an effect on the top-k procedure, the system employs in the end, pairs of keywords are chosen randomly and iteratively until all query keywords are covered.

```
Algorithm
    Input : The query k
    Output : Set of keywords.
1.  Split k into $k_1, k_2, k_3 \ldots\ldots k_n$
2.  t- a table where every tuple captures a join sequence of foreign key relationships and the score of each t and the combined score of the join sequence; it is initially empty.
3.  Generate the dynamic query by using element-level join sequences
4.  On each row of t do
<$t_i$, $t_j$> ← t.pop();
Compute the relevant keyword r for t.
Each keyword perform the matching process to t
    If matched then
     Identify the field name and add it         -to the tree
    Else
     Repeat the process
Compute the score in t in the tree
        Sort the score.
```

For computing element level links from the algorithm there are some factors, those are analyzed. We can implement the complexity analysis; if database table increases and more number of element level links identified then it takes time and space. In this we implement set level joins and reduces more complex links in the database. This results the complexity O($link_{max}$(i,j)), wehere $link_{max}$ refers the maximum link in element level key relationship.

Completeness is the factor in this algorithm. In this algorithm, we cover all possible queries for all element-level key relationships. If the links grows then queries also grows. Therefore we concentrate internal sub links and reduce number sub links then we cover all possible links.

We reduce more number of links by increasing number of joins in the algorithm. These joins are same for different element level key relationship models in different databases. Aggregate links is also an additional work to reduce the number of links. Aggregate operation can be implemented in join operation in the element-level key relationships by grouping the common keys in the database and also identifies different types of common keys groups in different databases.

RESULTS AND DISCUSSION

The modal is implemented on a PC with a Pentium Dual-Core processor running Microsoft Windows XP. All the algorithms are implemented in Java. The topic repository uses the PHP when directory and the data is stored in MYSQL open Server.

To implement this system, we take 5 databases with fully normalized form. Each database consists of minimum 15 tables and each table consists of minimum 1000 tuple. This is reasonable in practice and does not contradict our criticism on hierarchical methods that an efficient assignment of hierarchy requires a priori knowledge on what to be shared. We apply key-relationship integration in these databases and implement Steiner tree link construction.

We have implemented our algorithm for element level key relationships in different databases and the algorithm generated keyword sets. Table 1 shows the statistical information about keyword generation.

In the keyword search process, we have implemented different keywords with different lengths and keyword relevance with effective time is also calculated.

Table 1: Keyword Set Statistics

Database	Average tuples	D_{max}	Average keyword sets
1	1K>2K	21	40
2	2K>2.5K	18	28
3	1K>3K	14	32
4	1K>2K	16	38
5	1K>2K	19	23

Table 2: Performance Analysis

Keyword	No. of Results	Tim (in sec)	O(i)
1	10	0.71	0.86
2	8	0.58	0.89
3	15	0.64	0.69
4	12	0.69	0.72
5	9	0.61	0.67

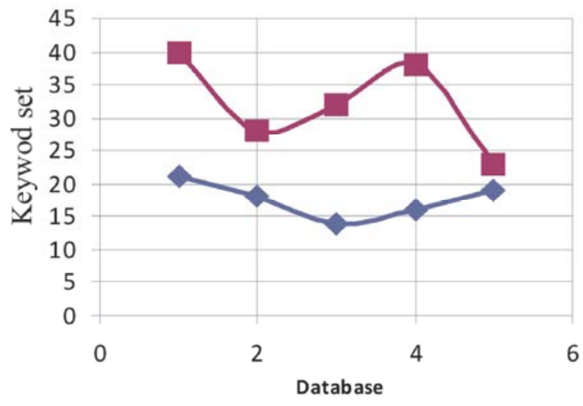

Fig. 5.1: Keyword Set Statistics

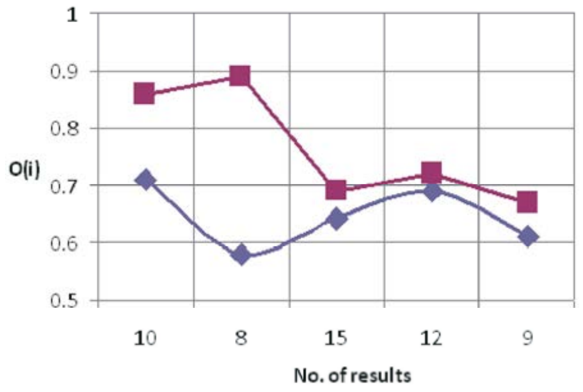

Fig. 5.2: Performance Analysis

In the table O(i) is the relevance accuracy of each keyword in the keyword search and in the table, if number of results increases the O(i) gradually increases. Deviation of time and relevance accuracy is very small and this achieves optimal results in the keyword search.

Number of results in the keyword search also contains ranking process. Most relevant results are obtained from the IR style ranking process and that are retrieved from the keyword search element level key relationship link process.

CONCLUSION

This paper proposes the concept to retrieve data in multiple databases and implementing database queries in large amount of database is the problematic one. Keyword search is the effective solution in database research. This paper implements an effective method to search keywords in different databases using dynamic query processing and implementing Steiner tree. This method consists of searching keywords in different databases using key-level relationship and the implements keyword search to analyze effective results. Experimental results improve performance of keyword search. In future, this method of keyword search is extended to unstructured databases and different types of web links.

REFERENCES

1. Thanh Tran and Lei Zhang, 2014. “keyword Query Routing”, IEEE Transactions on Knowledge and Data Engineering.
2. Ding, J.X.Yu, S. Wang, L. Quin, X. Zhang and X. Lin 2007. “Finding Top-K Min-Cost Connected Trees in Databases”, Proc. IEEE 23rd International Conference on Data Engineering.
3. Sayyadin, M., H.L. Ekhac, A. Doan and L. Gravano, 2007. “Efficient keyword search across hetrogenious relational databases” ,Proceedings in the IEEE 23rd International Conf. on Data Engg.
4. Vu, O.H., B.C. Ooi, D. Papadias and A.K.H. Tung, 2008. “A graph method for keyword based selection of the Top-K databases”, Proceedings in ACM SIGMOD Conference.
5. Yu, B., G. Li, K.R. Sollins and A.K.H. Tung, 2007. “Effective keyword based selection of Relational databases”, Proceedings in ACM SIGMOD Conference.
6. Luo, Y., X. Lin, W. Wang and X. Zhou, 2007. “Spark: Top-K keyword Query in Relational databases”, Proceedings in ACM SIGMOD Conference.
7. Hristidis, V., L. Gravano and Y. Papakonstantinou, 2003. “Efficient IR-style keyword search over relational databases” Proceedings in 29th Internation conf. in VLDB.
8. Qin, L., J.X. Yu and L. Chang, 2009. “Keyword search in databases: The power of RDBMS” Proceedings in ACM SIGMOD Conference.