A Two-phase Ranking Method for the Relational Databases

Yingqi Wang, Lianke Zhou, Hongbin Wang
College of Computer Science and Technology
Harbin Engineering University
China
hgcwyq@hrbeu.edu.cn, zhoulianke@hrbeu.edu.cn, wanghongbin@hrbeu.edu.cn

ABSTRACT: *With the extensive application of keyword query in relational database, the ranking algorithm has become an important research topic. Most existing ranking methods have adopted IR-style ranking models without considering the effects of semantic relevancy and diversity on the ranking results. In this paper, we propose a novel ranking method on the basis of semantic relevancy and diversity. First, we define the concepts of basic semantics and complex semantics and then add them to the ranking function. Next, aiming at the generation of redundancy, we design the diversity strategy to re-rank query results. Experimental results demonstrate that our method has better performance in terms of precision and recall.*

Keywords: Relational Database, Keyword Query, Ranking, Semantic Relevancy, Diversity

Received: 1 September 2018, Revised 3 November 2018, Accepted 28 November 2018

DOI: 10.6025/jcl/2019/10/1/1-12

1. Introduction

In the information retrieval field, keyword query has become a prevalent and mature query technology [1-3]. Users are only required to submit a list of keywords to the query system and then they can get the desired information easily. Due to the simplicity and usability of keyword query, more and more researchers apply it to the relational database and conduct in-depth research on it [4, 5]. As a vast amount of information resides in relational databases, there are potentially many results to a keyword query, and users are often only interested in the top-k query results [6-8]. So how to design a ranking function to improve the query quality is of paramount importance [9-11].

Existing ranking methods can be classified into two categories: result size-based ranking function and IR-style ranking function [12-14]. The method based on the result size is simple and easy-to-use, but it does not discuss the rich contents contained in the database which are useful for ranking. To address the above problems, the IR-style method ranks the results using TF*IDF that has been used in information retrieval, then the precision and recall of the methods have been improved. But none of the above methods consider the effects of semantic relevancy and diversity on ranking results.

Example 1. Figure 1 presents two subgraphs G_1 and G_2 for a keyword query $Q\{k_1, k_2, k_3\}$. Two graphs have exactly the same number of keywords, nodes and edges. The structures of G_1 and G_2 are fairly different, and the keywords distribution is also different, but the above two ranking methods do not distinguish this difference and assume that G_1 and G_2 have the same ranking score. Thus the ranking precision has been affected.

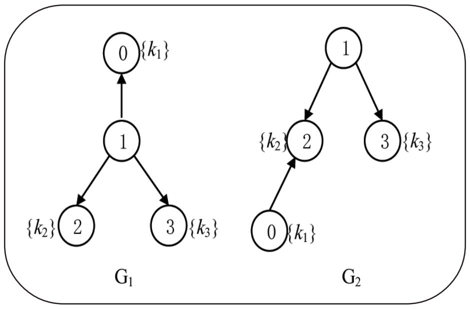


Figure 1. Query result sub-graph

Obviously, this is because the existing two ranking methods assume that the edges between nodes have the same weights and do not analyze the weight of intermediate nodes (such as node 1) which have a significant impact on the quality of ranking. In addition, the above methods only consider the effect of the basic semantic on the ranking results, but do not consider the effect of complex semantic. In order to solve this problem, Section 3 in the paper proposes a detailed solution to further refine the ranking process.

Example 2. Table 1 gives an example of the query results for the keyword query "keyword query databases", once ranked only by relevance, and once re-ranked by diversification. Both rankings provide several possible results, so that users can choose the intended one. However, ranking by estimated relevance bears the danger of redundant results. For example, the result "Spark: Top-*k* keyword query in relational databases" which ranks second overlap with the result "Spark2: Top-*k* keyword query in relational databases" ranked third.

Query: keyword query databases	
Relevance Top-3 results ranking	Relevance & Diversification Top-3 results ranking
Keyword query in relational databases	Keyword query in relational databases
Spark: Top-k keyword query in relational databases	Keyword query cleaning in relational databases
Spark2: Top-k keyword query in relational databases	Enhancing keyword query results over databases

Table 1. Results for a keyword query

In this paper, we aim to develop a two-phase ranking method SD-Rank (Ranking method based on semantic relevancy and diversity). First, we give a formal definition of basic semantics and complex semantics and present a semantic ranking function based on it. Second, we design a diversity strategy to re-rank the results by using the dynamic threshold and greedy algorithm. To the best of our knowledge, SD-Rank is the first work that applies semantic relevancy and diversity into the ranking function, which can return the query results with higher relevancy and diversity to users. Thus the effectiveness of the method is improved.

Our main contributions can be summarized as follows.

1. We present a two-phase ranking method SD-Rank based on semantic relevancy and diversity.

2. We define the concepts of the basic semantics and complex semantics then propose a semantic ranking function based on the concepts.

3. We design a diversity strategy to re-rank the results based on the dynamic threshold and greedy algorithm.

4. We perform extensive experiments using datasets DBLP for confirming the effectiveness of the ranking method in this paper.

The rest of the paper is organized as follows. Section 1 introduces the background and meaning of the study; Section 2 gives a brief review of the related work and analyzes problems in existing methods; Section 3 describes the semantic ranking method; Section 4 provides details of diversity strategy; Section 5 presents the experimental results; Section 6 concludes the paper and discusses possible directions for future work.

2. Related Work

Ranking keyword query results in relational databases has been widely studied recently. The typical work fall into two main categories: result size-based ranking strategies and IR-style ranking strategies. BANKS [15], DBXplorer [16], and Discover [17] use the size of results to measure the relevance between results and query keywords. The ranking score is inversely proportional to the size of the query results and the content information is not considered in these methods. Thus the methods can't achieve a high precision. On the basis of the method described above, EFFICIENT [18], EFFECTIVE [4], and SPARK [5] calculate the correlation with TF*IDF, where TF represents the term-frequency for a keyword in a tuple, IDF represents the inverse of DF. These methods improve the precision and recall in a certain degree. However, the above methods don't take into account the factors of semantic relevancy and diversity in the ranking process. [19] puts forward a semantic ranking function, taking into account the semantic relationship between tuples, but the weights of edges and result redundancy are not considered. Thus, these methods still can't meet users' query requirements well.

Therefore, the paper proposes a two-phase ranking method SD-Rank based on semantic relevancy and diversity. SD-Rank divides the whole ranking process into two phases. In the first phase, the paper formally defines two concepts of basic semantics and complex semantics, and then a semantic ranking function is put forward. In the second phase, the paper designs the diversity ranking algorithm based on the greedy algorithm and dynamic threshold. Top-*k* query results with higher semantic relevancy and diversity can be returned through the above ranking processes. The method further improves the precision and recall of the ranking in relational databases.

3. Ranking based on Semantic Relevancy

Given a keyword query *Q*, the semantic ranking is intended to define a ranking function which not only considers the tuples containing keywords actually, but also the tuples containing keywords semantically. Thus the precision of ranking results is improved effectively. This paper takes both basic semantics and complex semantics into consideration to put forward a semantic ranking function. Section 3.1 discusses the effect of basic semantics on ranking results from the perspectives of the containment relationship and connection analysis. Section 3.2 formally defines the maximal rooted directed tree and further measures the semantic relevancy of the results. Through in-depth discussions on the semantic relevancy in Section 3.1 and Section 3.2, the paper comes up with the semantic ranking function.

3.1 Basic Semantics

3.1.1 Containment Relationship

- **Direct Containment**

The direct containment ratio is used to measure the degree of direct coverage of subgraph RSG for query keywords. Given the keyword query $Q\{k_1, k_2, ..., k_n\}$, t is any tuple in result subgraph; $A(t)\{t[a_1], t[a_2], t[a_m]\}$ is m attributes of tuple t. Then the direct containment ratio between tuple t and keyword k is shown in (1).

$$s_D(t, k) = \sum_{t[a_i] \in A(t)} \frac{1 + 1n(1 + 1n\, tf)}{len(t[a_i]) / \overline{len}} \cdot ln \frac{N+1}{df} \quad (1)$$

Where tf is the frequency of the keyword k in the attribute $t[a_i]$; $len(t[a_i])$ is the length of $t[a_i]$; $\overline{len}$ is the average length of the attribute value; N is the number of tuples in table where a_i is located; df is the number of tuples containing k in table where a_i is located.

Accordingly, the direct containment ratio between tuple *t* and query *Q* is shown in (2):

$$s_D(t, Q) = \sum_{k \in Q} s_D(t, k) \tag{2}$$

• Indirect Containment

The indirect containment is used to measure the indirect coverage of *RSG* for query keywords. Assuming that tuple *t* doesn't contain the keyword *k* but has primary-foreign key relationship with the tuple t_i which contains *k*, then there is an indirect containment relationship between *t* and *k* . The indirect containment ratio is shown in (3):

$$s_I(t, k) = \sum_{t_i \in S(t)} s_D(t_i, k)^2 \Big/ \sum_{t_i \in S(t)} s_D(t_i, k) \tag{3}$$

Where *S* (*t*) represents the set of tuples which have primary-foreign key relationship with the tuple $t.S_D(t_i, k)$ represents the direct containment ratio between the tuple t_i and the keyword *k*.

Accordingly, the direct containment ratio between the tuple *t* and the query *Q* is shown in (4).

$$s_I(t, Q) = \sum_{k \in Q} s_I(t, k) \tag{4}$$

The direct containment ratio and indirect containment ratio between the tuple t and the query *Q* are calculated with (2) and (4) respectively. We combine the above two containment ratios with (5) to get the containment ratio between the tuple *t* and the query *Q* .

$$s_I(t, Q) = \alpha s_D(t, Q) + (1 - \alpha) s_I(t, Q) \tag{5}$$

Where α is the balance factor to adjust the proportion of the direct containment and indirect containment. According to Section .2, the precision of ranking achieves optimum when α equals to 0.6, which indicates that if a tuple directly contains keywords it will have a stronger correction.

3.1.2 Connection Analysis

In this section we first introduce the aggregation method to calculate the weight of edges. Then, we propose the simple semantic relevancy score function based on the weight of edges and containment relationship. Assuming, $e_{u,v}$ is the edge from the node *u* to the node *v* in the result subgraph, the weight of $e_{u,v}$ is effected by nodes *u*, *v* and query *Q*. The process of calculation is shown in (6).

$$s(e_{u,v}, Q) = \frac{s(u, Q) + s(v, Q)}{2|Q|} \cdot \frac{|Q_{u,v}|}{|Q|} \tag{6}$$

Where *s*(*u*, *Q*) (*s*(*v*, *Q*)) is the containment ratio between the node *u* (*v*) and query *Q*. |*Q*| is the number of keywords in query. $|Q_{u,v}|$ denotes the number of keywords involved with the edge, $e_{u,v}$, which is the number of keywords in nodes *u* and *v* . $|Q_{u,v}|/|Q|$ ensures that edges involving more keywords are assigned to higher weight.

Finally, the basic semantic relevancy integrates the containment ratio of all nodes and weights of all edges. Compared with the existing ranking methods, the method proposed in this paper assigns different weights to different edges and further details the ranking process to improve the precision of ranking.

The calculation of the basic semantic correlation is shown in (7).

$$S_{simp}(RSG, Q) = \sum_{t,e \in RSG} s(t, Q) + s(e, Q) \tag{7}$$

3.2 Complex Semantics

This section analyzes the complex semantic relations between tuples in the query result subgraph RSG . First, the concept of maximal rooted directed tree is defined formally. There are strong semantic relevancy between tuples in the rooted directed tree. Second, the basic semantic score function in Section 3.1 is extended by using the maximal rooted directed tree.

Definition 1 Rooted directed tree. A nontrivial directed tree *DT* is defined as a rooted directed tree if the in-degrees of all nodes in *DT* are 1 except the root node.

Definition 2 Maximal rooted directed tree *max*(*RDT*). The subtree *ST* in result subgraph is defined as a maximal rooted directed tree if *ST* satisfies the following conditions:

1. *ST* contains only one rooted directed tree;

2. There is not t_i , $t_i \notin ST$ and $ST \cup \{t_i\}$ contains only one rooted directed tree.

Assuming that all the tuples $RSG(t_1, t_2, ...)$ in the result subgraph *RSG* are contained in a same maximal rooted directed tree, there is a tuple which can uniquely determine all other tuples. The semantic relevancy between tuples in result subgraph *RSG* are stronger. Conversely, if the tuples $RSG(t_1, t_2, ...)$ in result graph *RSG* are not contained in a same maximal rooted directed tree, the tuples can't be uniquely determined by one original tuple, and the semantic relevancy between tuples are weaker. Thus the semantic relevancy between tuples in *RSG* is inversely proportional to the number of *max* (*RDT*) in *RSG*, as shown in (8).

$$s_{comp}(RSG, Q) = 1/|\,maxRDT(RSG)| \tag{8}$$

Where $|maxRDT(RSG)|$ is the number of *max* (*RST*) in *RSG* obtained by marking and counting the number of node whose in-degrees 0.

For example, there are two result subgraphs RSG_1 and RSG_2 for query *Q* as shown in Figure 2. In result subgraph RSG_1 there is *max*(*RDT*) which represents the complex correlation between tuples, that is $|maxRDT(RSG_1)\} = 1$. In RSG_2 there is not a *max*(*RDT*) containing all the tuples, but there are two maximal rooted directed trees $t_1 \to t_2 \to t_3$ and $t_3 \to t_4 \to t_5$ which represent the complex correlation between tuples. So RSG_2 contains two maximal rooted directed trees, that is $|maxRDT(RSG_2)| = 2$.

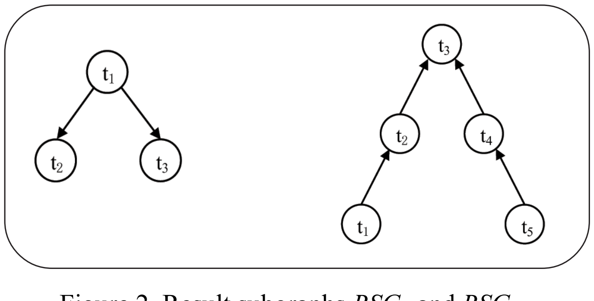


Figure 2. Result subgraphs RSG_1 and RSG_2

Through the analysis of basic semantics and complex semantics in Section 3.1 and 3.2 we can get the following semantic ranking function.

$$s_{semantics}(RSG, Q) = s_{simp}(SG, Q) + 1/|maxRDT(RSG)| \tag{9}$$

4. Ranking based on Diversity Strategy

4.1 Ranking Function

If we only use the above semantic ranking function to rank the keyword query results, we will get overlapping results. Ranking based on diversity balances the relevance and diversity to solve the above problems well, which avoids the risk of users' dissatisfaction. For example, for the same keyword query "Apple", some users may be interested in apples, while some users want to get information about the Apple company. Therefore, it is necessary to propose a ranking method to diversify the query results and satisfy different users. Whereas diversification of query results on unstructured documents has been well studied and is widely understood, diversification of query results in relational database attracted much less attention.

The diversification of query results is intended to make the results have both higher relevance and weaker similarity. Assume the subgraphs RSG_i and RSG_j are the results for keyword query Q. This paper measures the similarity between RSG_i and RSG_j with Jaccard similarity coefficient, as shown in (10).

$$Sim(RSG_i, RSG_j) = \frac{RSG_i \cap RSG_j}{RSG_i \cup RSG_j} \tag{10}$$

The similarity of query results takes range value from 0 to 1, where 1 represents two results have the highest similarity. After the semantic ranking in Section 3, we can obtain r results RSG_1, RSG_2,...., RSG_r related to the keyword query Q. The diversity score of subgraph RSG is the average of the similarities between RSG and all the other subgraphs, as shown in (11).

$$s_{diversity}(RSG) = \sum_{SG_i \in SGS} Sim(RSG, RSG_i) / |\mathrm{RSGS}| \tag{11}$$

We integrate the above semantic ranking score and diversity score to get the final ranking formula (12) as follows:

$$Score(RSG) = \beta \,.\, s_{semantic}(RSG, Q) - (1-\beta).\ s_{diversity}(RSG) \tag{12}$$

Where, β is the balance factor to adjust the semantic relevancy and diversity, when β equals 0 the ranking method only considers diversity.

4.2 Diversity Strategy

In order to obtain top-k query results with high relevance and diversity, this paper puts forward a diversity strategy based on greedy algorithm and dynamic threshold algorithm to re-rank the results. Given query Q and set L of top-r results obtained by the semantic ranking, first the most relevant result is returned. Then at each iteration, the result with highest score by (12) is returned. The realization process is shown in Algorithm 1.

Algorithm 1 Diversity Algorithm

Input: List L of top-r query results ranked by semantics;

Output: list L' ;

Method:

```
1. $L'[0] = L[0]$; $i = 1$
2. while($i < k$){
3.     $j = i$; threshold = 0;
4.     while ($L[j]$ = null){
5.     if (threshold > $\beta.s_{semantic}(L[j])$) break;
6.     if((Score (L[j]) > threshold){
7.             threshold = Score(L[j]);
8.             $t = j$;
9.                     }j++
10.               }
11.    $L'[i] = L[t]$;
12.     Swap $L$ [$i$ … t - 1]) and $L[t]$;
13.     i++;
14.             }
15. Return L′;
```

At each iteration, we can identify the subgraph with the highest score and without checking all the result subgraphs.

The time complexity of algorithm 1 is $O(r * k)$, where is number of subgraphs in list L obtained by the semantic ranking; k is the number of subgraphs in re-ranking list L'.

5. Experimental Results

The experiments are conducted on DBLP dataset [20]. Section 5.1 describes briefly the dataset and experimental environment. Section 5.2 evaluates the quality of SD-Rank by comparing it with DISCOVER and SPARK, since DISCOVER and SPARK are known as the most effective one in the result size based methods and IR-style methods respectively. In order to test the effectiveness of the ranking method SD-Rank, we used precision, recall and F-score to do the evaluation. The precision is a ratio of the relevant results over the returned results. The recall is a ratio of the number of relevant results searched over the overall number of relevant results in the database. The formulas are shown as follows:

$$precision = \frac{|\{relevant\ results\} \cap \{retrieved\ results\}|}{|\{retrieved\ results\}|}$$

$$recall = \frac{|\{relevant\ results\} \cap \{retrieved\ results\}|}{|\{relevant\ results\}|}$$

$$Fscore = \frac{2 \times precision \times recall}{precision + recall}$$

For each keyword query, the paper uses the results obtained by executing the corresponding SQL queries as the relevant results.

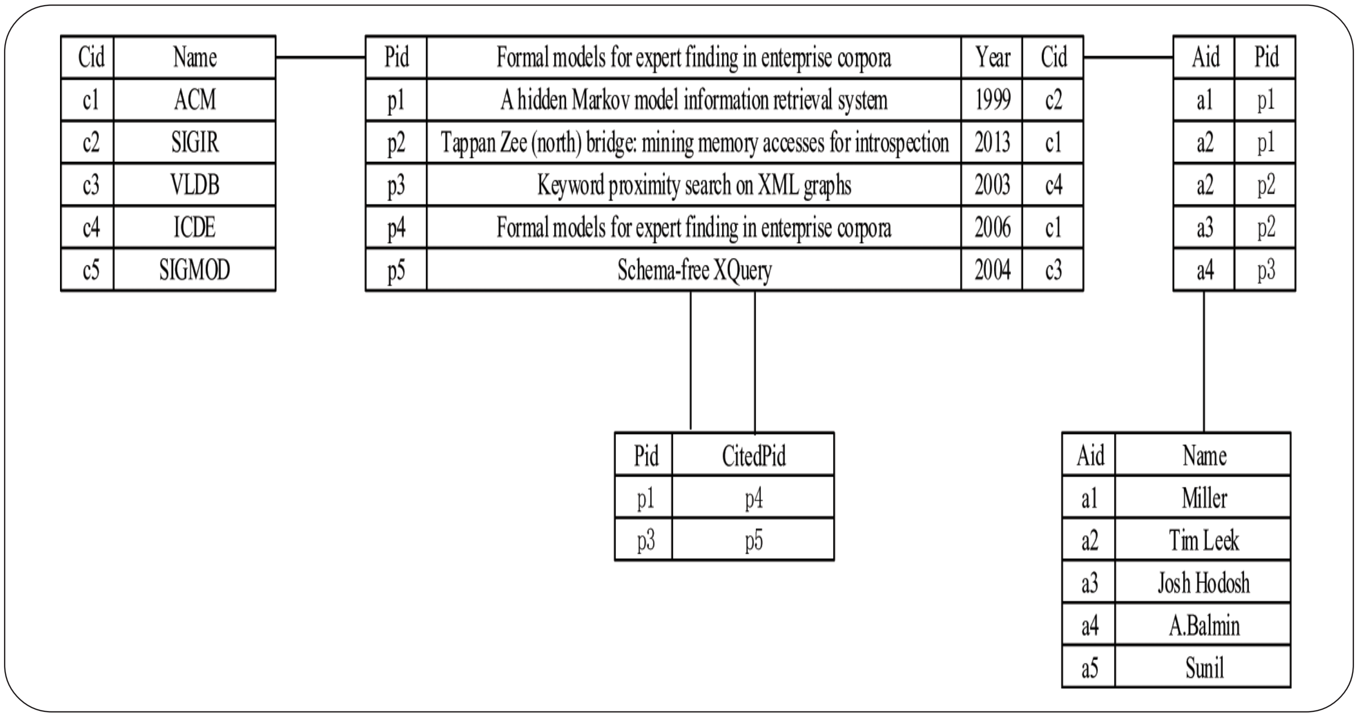
5.1 Experimental Setup
Dataset
The original form of DBLP dataset is XML and the download address is http://dblp.uni-trier.de/xml. We first transform the dataset to a relational database using SAX parsing technology and then obtain the data graph. The schema of DBLP database is shown as Figure 3 which contains five tables: Author, Paper, Conference, Cite and Write. Where, tables Author, Paper and Conference contain information about scholars, papers and conference respectively. Tables Cite and Write are relationship tables. The former specifies the reference relationships and the latter contains the writing relationships between scholars and papers.

Experimental Environment
The experiments have been performed on a computer with an Intel Pentium 3.2GHz CPU and 4 GB of RAM on Windows XP platform.

5.2 Parameter Setting
This section studies the effect of two parameters of SD-Rank, α and β. α is used to adjust the contribution ratio of the direct containment and indirect containment to the semantic ranking. β is used to adjust the contribution ration of the relevance and

diversity to the complex ranking function. Keep β equal to 1 and change the value of α constantly, the precision under different α is shown in Figure 4. From Figure 4 we can see that the precision of the ranking function is highest when α equals to 0.6.

Keep α equal to 0.6 and change the value of β constantly, the Figure 5 shows the precision under different values of β. When equals to 0.55, the precision of the ranking function reaches a maximum. When β gets larger, the ranking performance begins to deteriorate. In the following experiment, the values of α and β are set to 0.6 and 0.55 respectively, then the performance of SD-Rank ranking method is optimum.



Cid	Name
c1	ACM
c2	SIGIR
c3	VLDB
c4	ICDE
c5	SIGMOD

Pid	Formal models for expert finding in enterprise corpora	Year	Cid
p1	A hidden Markov model information retrieval system	1999	c2
p2	Tappan Zee (north) bridge: mining memory accesses for introspection	2013	c1
p3	Keyword proximity search on XML graphs	2003	c4
p4	Formal models for expert finding in enterprise corpora	2006	c1
p5	Schema-free XQuery	2004	c3

Aid	Pid
a1	p1
a2	p1
a2	p2
a3	p2
a4	p3

Pid	CitedPid
p1	p4
p3	p5

Aid	Name
a1	Miller
a2	Tim Leek
a3	Josh Hodosh
a4	A.Balmin
a5	Sunil

Figure 3. The schema graph of DBLP

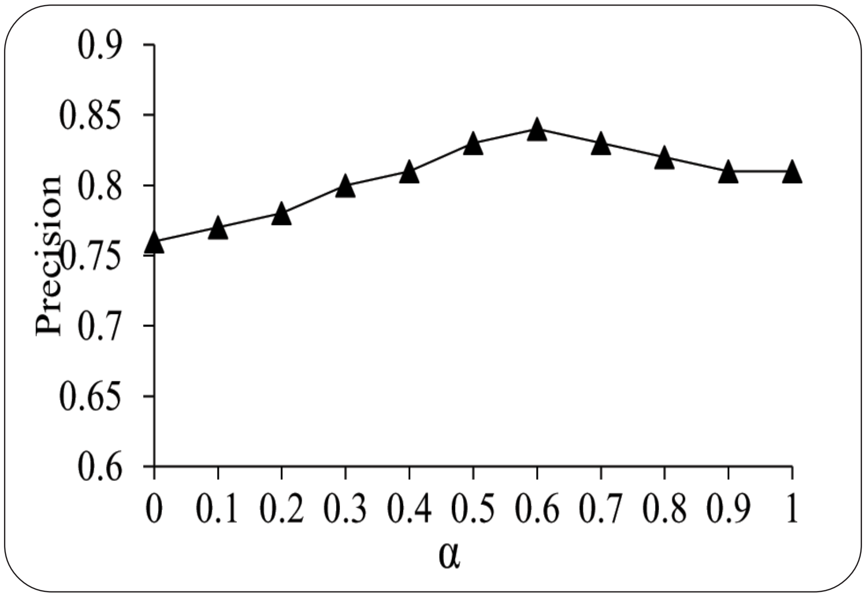


Figure 4. The effect of β on precision

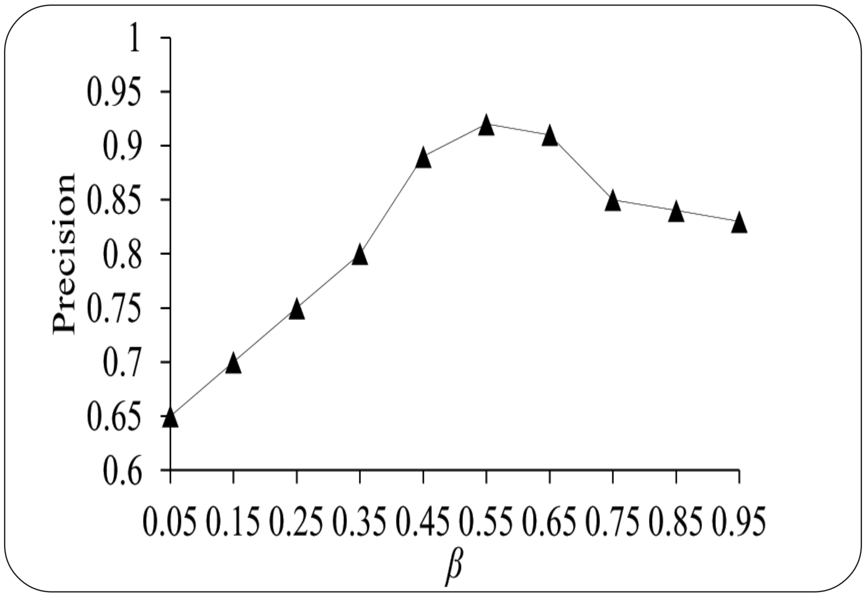


Figure 5. The effect of on precision β

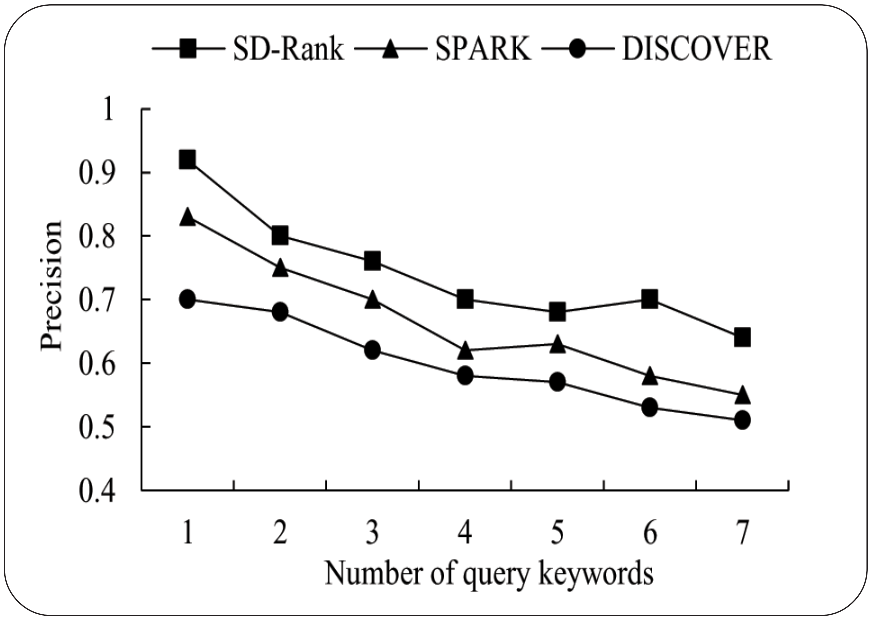


Figure 6. Comparison of precision

5.3 Performance Study

This section evaluates the performance of the ranking methods with three metrics: precision, recall and F-score.

Precision

The experiment can be divided into 7 groups according to the number of query keywords, and each group contains 30 queries.

In Figure 6, the x-axis represents the number of keywords and the y-axis represents the precision of ranking results. Figure 6 shows the precision of DISCOVER, SPARK and SD-Rank. We can see that SD-Rank work much better than DISCOVER and SPARK. Specifically, when the number of keywords is 4, the SD-Rank method increases the precision by 20.7% and 12.9%, compared with DISCOVER and SPARK, respectively. The reason for effective and improved ranking of SD-Rank is as follows. First we investigate the reason that the precision of DISCOVER is much lower than the other two methods and conclude that it is because DISCOVER simply ranks the query results based on the size of results, the influence factors are too simple. This method can't accurately measure the correlation between the query and results, so it is difficult to return accurate results. While SPARK uses the TF*IDF correction measure to accurately measure the containment correlation between query and results, and the precision of results is improved. SD-Rank improves the ranking function on the basis of the above two methods, which not only considers the effect of content but also emphasizes the effect of semantics. SD-Rank first formally defines the concept of semantic correction and comprehensively measures it from two aspects of simple semantics and complex semantics. The simple semantic contains containment relationship and connection analysis, which fully considers the effects of direct containment, indirect containment and the distribution of keywords on the ranking results. Thus the process of ranking can be further refined. And the complex semantic can return the results with higher semantic correlation to effectively improve the ranking precision.

Recall

As Figure 7 shows, the recall values of SD-Rank all achieved more than 0.68, and among which reaches 0.86 when the number of keywords is 6, while the highest recall values of DISCOVER and SPARK are 0.64 and 0.69 respectively. SD-Rank performs much better than DISCOVER and SPARK in terms of recall as the number of keywords varying from 1 to 7. Therefore, SD-Rank can get more relevant results. For example, SD-Rank achieves 0.79 average recall, which leads to about 21.5% and 29.5% over SPARK and DISCOVER. Particularly when the number of keywords is 3, SD-Rank works significantly better than DISCOVER, with more than 30.0% relative improvement. Furthermore, SD-Rank also works better than SPARK. The reason for the results is as follows: as compared to DISCOVER and SPARK, SD-Rank proposes a two-phase ranking method, which not only considers the factors of structures, contents and semantics but also adds the diversity into the integrated ranking function. Thus the redundant results are effectively avoided and more relevant and comprehensive results are returned.

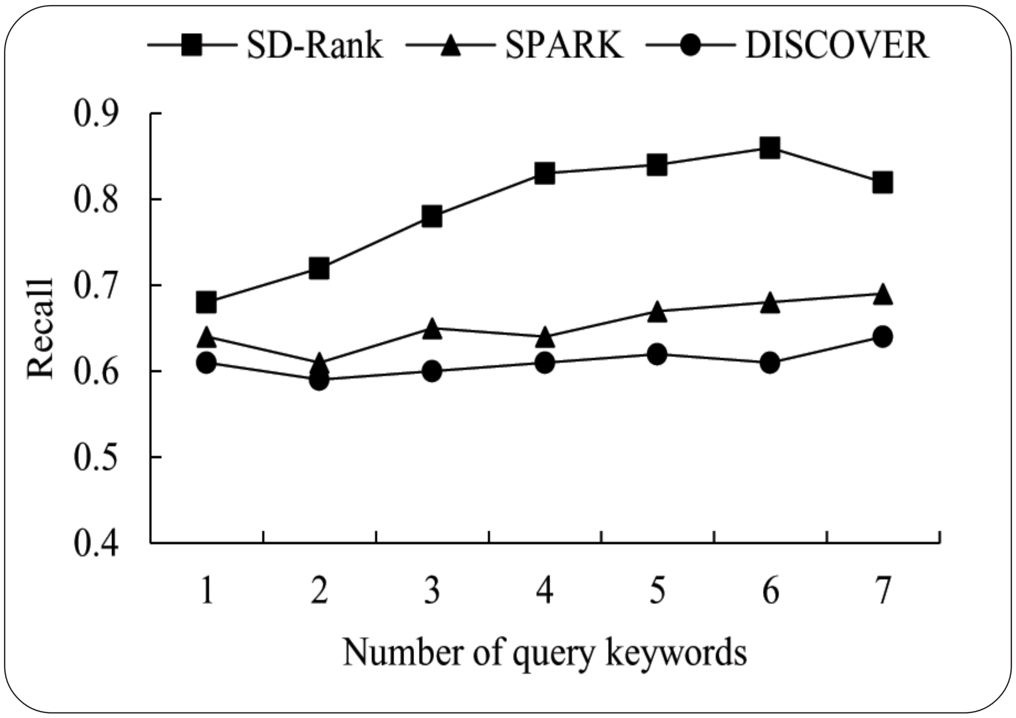


Figure 7. Comparison of recall

F-score

Figure 8 shows the F-score values of DISCOVER, SPARK and SD-Rank. As we can see in the above figure, among three ranking methods DISCOVER has the lowest precision and SD-Rank has the highest one. Overall, the F-score of DISCOVER method is little higher than SPARK. And the F-score of SD-Rank is improved obviously compared with the other two. More precisely, when the number of keywords is 3, the F-score of DISCORE and SPARK is 0.61 and 0.67 respectively, while the one of SD-Rank is 0.77. As expected, the F-score of SD-Rank is approximately 14.9% higher than that of the SPARK method, and it is even higher when compared with the other method DISCOVER. In summary, SD-Rank can really outperform the baseline methods.

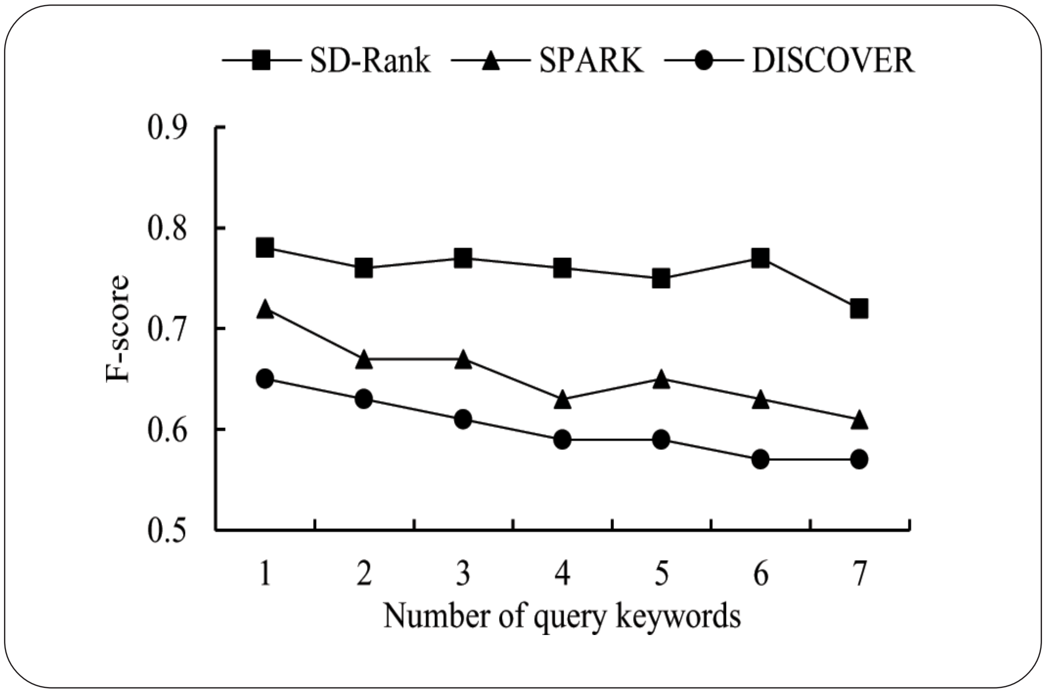


Figure 8. Comparison of F-score

6. Conclusion and Future Work

The paper mainly focuses on the research of the ranking method in relational databases. A simple and effective ranking method SD-Rank is proposed. Compared with the traditional ranking method, SD-Rank optimizes the ranking function by combining the semantic correlation with the diversity of results. First, the concepts of basic semantics and complex semantics are defined and the semantic ranking function is proposed based on the concepts, then the first phase of ranking would have been finished. Second, SD-Rank method uses the dynamic threshold and greedy algorithm to design the diversity strategy for the second phase of ranking. We have performed experiments on the DBLP dataset to show the effectiveness of the two-phase ranking method. In the future work, we will continue to study the effect of users' feedback on the ranking results based on SD-Rank.

References

[1] Pawar, S. S., Manepatil, A., Kadam, A., Jagtap, P. (2016). Keyword search in information retrieval and relational database system: Two class view. *In:* International Conference on Electrical, Electronics, and Optimization Techniques, ICEEOT 2016. p. 4534-4540. Institute of Electrical and Electronics Engineers Inc. March 3, 2016 - March 5, 2016.

[2] Fernandez, M., Cantador, I., Lopez, V., Vallet, D., Castells, P., Motta. E. (2011). Semantically enhanced Information Retrieval: An ontology-based approach. *Journal of Web Semantics* 9 (4) 434-452.

[3] Wilson, M. L., Kules, B., Schraefel, M. C., Shneiderman, B. (2010). From keyword search to exploration: Designing future search interfaces for the web. *Foundations and Trends in Web Science* 2 (1) 1-97.

[4] Liu, F., Yu, C., Meng, W., Chowdhury. A. (2006). Effective keyword search in relational databases. *In:* 2006 ACM SIGMOD international conference on Management of data. p. 563-574. ACM. June 27, 2006 - June 29, 2006.

[5] Luo, Y., Lin, X., Wang, W., Zhou, X. (2007). Spark: top-k keyword query in relational databases. *In:* SIGMOD 2007: ACM SIGMOD International Conference on Management of Data. p. 115-126. ACM. June 12, 2007 - June 14, 2007.

[6] Pahikkala, T., Waegeman, W., Airola, A., Salakoski, T., De Baets, B. (2010). Conditional ranking on relational data. *In:* European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML PKDD 2010, September 20, 2010 - September 24, 2010. p. 499-514. Springer Verlag.

[7] Kothawade, A., Harak, M., Bagul, J., Patil. B. (2015). Ranking based prediction of keyword over big databases. *In:* 1st International Conference on Green Computing and Internet of Things, ICGCIoT 2015, October 8, 2015 - October 10, 2015. p. 899-903. Institute of Electrical and Electronics Engineers Inc.

[8] Yang, S., Han, F., Wu,Y., Yan. X. (2016). Fast top-k search in knowledge graphs. *In:* 32nd IEEE International Conference on Data Engineering, ICDE 2016. p. 990-1001. Institute of Electrical and Electronics Engineers Inc. May 16, 2016 - May 20, 2016.

[9] Zhou, J., Yu, X., Liu, Y., Yu, Z. (2014). Ranking keyword search results with query logs. *In:* 3rd IEEE International Congress on Big Data, BigData Congress 2014. p. 770-771. Institute of Electrical and Electronics Engineers Inc. June 27, 2014 - July 2, 2014.

[10] Chen, Y., Wang, W., Liu, Z. (2011). Keyword-based search and exploration on databases. *In:* 2011 IEEE 27th International Conference on Data Engineering. p 1380-1383. IEEE Computer Society. April 11, 2011 - April 16, 2011.

[11] Chaudhuri, S., Das, G. (2009). Keyword querying and ranking in databases, *In:* Proceedings of the VLDB Endowment 2 (2) 1658-1659.

[12] Bicer, V., Tran, T., Nedkov, R. (2011). Ranking support for keyword search on structured data using relevance models. *In:* 20th ACM Conference on Information and Knowledge Management. pages 1669-1678. Association for Computing Machinery. October 24, 2011 - October 28, 2011.

[13] Pujari, P., Ade. R. (2016). Enhancing performance of keyword query over structured data. *In:* 2nd International Conference on Computing, Communication, Control and Automation. Institute of Electrical and Electronics Engineers Inc. August 12, 2016 - August 13, 2016.

[14] Kim, I.-J., Whang, K .-Y., Kwon, H.-Y. (2014). SRT-rank: Ranking keyword query results in relational databases using the strongly related tree. *IEICE Transactions on Information and Systems* E97-D (9) 2398-2414.

[15] Bhalotia, G., Hulgeri, A., Nakhe, C., Chakrabarti, S., Sudarshan. S. (2002). Keyword searching and browsing in databases using BANKS. *In:* 18th International Conference on Data Engineering. p. 431-440. IEEE Computer Society. February 26, 2002 - March 1, 2002.

[16] Agrawal, S., Chaudhuri, S., Das, G. (2002). DBXplorer: A system for keyword-based search over relational databases. *In:* 18th International Conference on Data Engineering. p. 5-16. IEEE Computer Society. February 26, 2002 - March 1.

[17] Hristidis, V., Papakonstantinou, Y. (2002). Discover: keyword search in relational databases. *In:* The 28th international conference on Very Large Data Bases. p. 670-681. VLDB Endowment. August 20 - 23, 2002

[18] Hristidis, V., Gravano, L., Papakonstantinou, Y. (2003). Efficient IR-style keyword search over relational databases. *In:* The 29th international conference on Very Large Data Bases. p. 850-861. VLDB Endowment. September 9, 2003 - September 12, 2003.

[19] Wang, B., Yang, X.-C., Wang. G.-R. (2008). Top-K keyword search for supporting semantics in relational databases. *Ruan Jian Xue Bao/Journal of Software* 19 (9) 2362-2375.

[20] *DBLP*. Available: http://dblp.uni-trier.de/