International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

An Advanced Information Retrieval System of Relational Keyword Search Scheme

Gholap Dhananjay A.[1], Gumaste S. V.[2]

ME Student, Department of Computer Engineering, Sharadchandrapawar College of Engineering, Otur, Pune, India
Professor, Department of Computer Engineering, Sharadchandrapawar College of Engineering, Otur, Pune, India

Abstract: ***In the last few years, a keyword search technique to relational database has been an interesting area of research system within the relational database and information retrieval (IR) system. A huge number of attempts have been invented and executed, but due some problem , there remains lack of standard system. This lack of standard system it resulted in inaccurate results from different attempts. In this paper present a thought an advanced information retrieval system of relational keyword search scheme. Results shows that large number of existing search schemes do not provide better work for information retrieval tasks. In some schemes, memory consumption prevent many search methods from altering small datasets with 10's of thousands of vertices. Explain connection between implementation time and factors changed in earlier attempts; our analysis shows that these factors have relatively small impact on performance of retrieval scheme.***

Keywords: Information Retrieval, Keyword Search, Datasets, Query Workloads, Schema-based Systems, Graph-based Systems

1. Introduction

The universal search text box has transformed the way people interact with large information. Almost of all Internet users use a search engine daily [10], performing in excess of 3 billion searches [11]. The success of keyword search systems from what it does not need—namely, a special query language or knowledge of the structure of the data. Large number of internet user uses keyword search system.

Interaction for getting information, and it is important to extend this scheme to relational data information. This extension has been an important area of research throughout the past 10 years. We are not conscious of any research projects that have transit from proof-of-concept operations to deploy system. We imagine that the existing ad-hoc evaluations performed by researchers are not indicative of these system's real-world performance, a claim that has surfaced recently in the literature [1], [5], [22].

The large number of research papers being published in this track, existing empirical evaluations reject or only partially address many important points related to search working performance. Baid et al. [1] explain that existing systems have unpredictable performance which undermines their usefulness for real-world retrieval work. This point has little support in the existing literature, but the failure for these systems to gain a foothold implies that robust, independent evaluation is needed. In part, existing performance problems may be obscured by experimental design evaluations such as the selection of datasets or the creation of query workloads.

Therefore consequently, conduct an interesting , independent , advanced evaluation of existing keyword search schemes using a publicly useable benchmark to increase real-world performance for realistic query workload.

A. Overview of Keyword Search

Keyword searching on semi-structured information (e.g., XML) and relational information different from existing IR. A difference presents between the data's physical storage and a logical view of the information. Relational databases are normalized to remove redundancy, and foreign keys find out related information. Search queries regularly cross these relationships (i.e., a subset of search terms is describe in one tuple and the other remaining terms are found in related tuples), which enables relational keyword search systems to recover a logical view of the information. The intrinsic assumption of keyword search that is, the search terms are related , makes complex search process because classically there are many feasible relationships between two search terms. It is almost always possible to insert another occurrence of a search term by including tuples to an existing result. This implementation leads to tension between the conciseness and average search results.

Country		
Code	Name	Capital
A	Austria	Vienna
CH	Switzerland	Bem
D	Germany	Berlin
F	France	Paris
FL	Liechtenstan	Vaduz
I	Italy	Rome

Borders		
C1	C2	Length
A	D	784
A	I	430
CH	A	164
CH	D	334
CH	F	573
CH	I	740
F	D	451
FL	A	37
FL	CH	41

Figure 1: Example relational data from the MONDIAL database (left)

Search in relational data .Consider the query "Switzerland Germany" where the user wants to know how the two countries are related. The borders relation indicates that the two countries are adjacent. However, Switzerland also borders Austria, which borders Germany; Switzerland

borders France, which borders Germany; etc. As shown on the right in the figure, we can continue to construct results by adding intermediary countries, and we are only considering two relations and a handful of tuples from a much larger database.

Creating coherent search results from discrete tuples is the primary reason that searching relational data is significantly more complex than searching unstructured text .Unstructured text allows indexing information at the same granularity as the desired results (e.g., by documents or sections within documents.)

This task is impractical for relational data because an index over logical (or materialized) views is considerably larger than the original data [1], [21]. In this paper, focus on keyword search techniques for relational data, and we do not discuss approaches designed for XML.

Query: "Switzerland Germany"
Results:
1 Switzerland ← [borders] → Germany
2 Switzerland ← [borders] → Austria ← [borders] → Germany
2 Switzerland ← [borders] → France ← [borders] → Germany
4 Switzerland ← [borders] → Italy ← [borders] → Austria ← [borders] →Germany
4 Switzerland ← [borders] → Italy ← [borders] → France ← [borders] →Germany
4 Switzerland ← [borders] → Liechtenstein ← [borders] → Austria ←[borders] → Germany
7 Switzerland ← [borders] → Austria ← [borders] → Italy ← [borders] →France ← [borders] → Germany
Search Results(Right) are ranked by size(number of tuples) which account for the ties in the list.

b. Contributions and Outline

As we discuss later in this paper, many relational keyword search systems approximate solutions to intractable problems. Researchers consequently rely on empirical evaluation to validate their heuristics. Continue this tradition by
evaluating these systems using a benchmark designed for relational keyword search. Our holistic view of the retrieval process exposes the real-world tradeoffs made in the design of many of these systems. For example, some systems use alternative semantics to improve performance while others incorporate more sophisticated scoring functions to improve search effectiveness. These tradeoffs have not been the focus of prior evaluations. The major contributions of this paper are as follows:

- Conduct an independent, An Advanced Information Retrieval System of Relational Keyword Search Scheme , which doubles the number of comparisons as previous work.
- End result will not authenticate previous claims regarding the scalability and presentation of relational keyword search schemes. Present search systems perform weakly for datasets higher than tens of thousands of vertices.
- Describe that the parameters diverted in existing evaluation are at best insecurely related to performance, which is likely due to experiment not using presentive datasets or query workloads.
- The task is the first to merge performance and search usefulness in the assessment of such a large number of systems. Considering these two issues in combination provides better understanding of these two crucial transactions among challenging system designs.

2. Motivation for Independent Evaluation

Most evaluations in the literature disagree about the performance of various search techniques, but significant experimental design differences may account for these discrepancies. Discuss three such differences in this section.

A. Datasets

Table1 summarizes the datasets and the number of queries used in previous evaluations. Even though this table suggests some uniformity in evaluation datasets, their content varies dramatically. Consider the evaluations of BANKS-II , BLINKS [12], and STAR . Only BANKS-II's evaluation includes the entire Digital Bibliography & Library Project (DBLP)3 and the Internet Movie Database (IMDb) dataset. Both BLINKS and STAR use smaller subsets to facilitate comparison with systems that assume the data graph fits entirely within main memory. The literature does not address the representativeness of database subsets, which is a serious threat because the choice of a subset has a profound effect on the experimental results.

Table 1: Statistics Item Previous Evaluations

SYSTEM	DATASET	|V|	|E|	|Q|
BANK[2]	bibliographi	100K	300K	7
DISCOVER [15]	TPC-H			200
DISCOVER-II [14]	DBLP			100
BANKS-II [17]	DBLP	2M	9M	200
	IMDb	2M	9M	
Liu et al. [21]	lyrics	196K	192K	50
DPBF [8]	DBLP	7.9M		500
	MovieLen	1M	1M	600
BLINKS [13]	DBLP	409K	591K	60
	IMDb	68K	248K	40
SPARK [22]	DBLP	882K	1.2M	18
	IMDb	9.8M	14.8M	22
	MONDIAL	10K		35
EASE [20]	DBLife	10K		5
	DBLP	12M		5
	MovieLen	1M		5
	previous 3			5
BANKS-III [6]	DBLP	1.8M	8.5M	8
	IMDb	1.7M	1.9M	4
STAR [18]	DBLP	15K	150K	180
	IMDb	30K	80K	180
	YAGO	1.7M	14M	120

|V| number of nodes (tuples) |E| number of edges in data graph
|Q| number of queries in workload

B. Query Workloads

The query workload is another critical factor in the evaluation of these systems. The trend is for researchers either to create their own queries or to create queries from terms selected randomly from the corpus. The latter strategy

is particularly poor because queries created from randomly-selected terms are unlikely to resemble real user queries . The number of queries used to evaluate these systems is also insufficient. The traditional minimum for evaluating retrieval systems is 50 queries and significantly more may be required to achieve statistical significance [23]. Only two evaluations that use realistic query workloads meet this minimum number of information needs.

C. Experimental Discrepancies

Discrepancies among existing evaluations are prevalent. Table II lists the mean execution times of systems from three evaluations that use DBLP and IMDb databases. The table rows are search techniques; the columns are different evaluations of these techniques. Empty cells indicate that the system was not included in that evaluation.According to its authors,BANKS-II "significantly outperforms" BANKS, which is supported by BANKS-II's evaluation, but the most recent evaluation contradicts this claim especially on DBLP. Likewise, BLINKS claims to outperform BANKS-II[17]"by at least an order of magnitude in most cases" [12], but when evaluated by other researchers, this statement does not hold.

We use Table II to motivate two concerns that we have regarding existing evaluations. First, the difference in the relative performance of each system is startling. We do not expect the most recent evaluation to downgrade the orders of magnitude performance improvements to performance degradations, which is the certainly the case on the DBLP dataset. Second, the absolute execution times for the search techniques vary widely across different evaluations.

Table 2: Example of contradictory results in the literature

	execution time (s)					
	DBLP			IMDb		
System	[17]	[13]	[18]	[17]	[13]	[18]
BANKS [2]	14.8		5.9	5.0		10.6
BANKS-II [17]	0.7	44.7	7.9	0.6	5.9	6.6
BLINKS [13]		1.2	19.1		0.2	2.8
STAR [18]			1.2			1.6

3. Relational Keyword Search Systems

Given our focus on empirical evaluation, we adopt a general model of keyword search over data graphs. This section presents the search technique included in our evaluation. Problem definition: We model a relational database as a graph G=(V,E). Each vertex v ϵ V corresponds to a tuple in the relational database. An edge (u,v)ϵ E represents each relationship (i.e., foreign key) in the relational database. Each vertex is decorated with the set of terms it contains. A query Q comprises a list of terms. A result for Q is a tree T that is reduced with respect to Q' Subset Q; that is, T contains all the terms of Q' but no proper subtree that also contains all of them. Results are ranked in decreasing order of their estimated relevance to the information need expressed by Q.

A. Schema-Based Systems

Schema-based approaches support keyword search over relational databases via direct execution of SQL commands. These techniques model the relational schema as a graph where edges denote relationships between tables. The database's full text indices identify all tuples that contain search terms, and a join expression is created for each possible relationship between these tuples.

DISCOVER [14] creates a set of tuples for each subset of search terms in the database relations. A candidate network is a tree of tuple sets where edges correspond to relationships in the database schema. DISCOVER enumerates candidate networks using a breadth-first algorithm but limits the maximum size to ensure efficient enumeration. A smaller size improves performance but risks missing results. DISCOVER creates a join expression for each candidate network, executes the join expression against the underlying database to identify results, and ranks these results by the number of joins.

Hristidis et al. [13] refined DISCOVER by adopting pivoted normalization scoring to rank results:

$$\sum_{t \in Q} \frac{1+\ln(1+\ln tf)}{1-s+s.\frac{dl}{avgdl}} . qtf . \ln(\frac{N+1}{df}) \quad \text{--- (1)}$$

where t is a query term, (q)tf is the frequency of the (query) term, s is a constant (usually 0.2), dl is the document length, avgdl is the mean document length, N is the number of documents, and df is the number of documents that contain t. The score of each attribute (i.e., a document) in the tree of tuples is summed to obtain the total score. To improve scalability, DISCOVER-II creates only a single tuple set for each database relation and supports top-k query processing because users typically view only the highest ranked search results.

B. Graph-based Systems

The objective of proximity search is to minimize the weight of result trees. This task is a formulation of the group Steiner tree problem [9], which is known to be NP-complete [29]. Graph-based search techniques are more general than schema based approaches, for relational databases, XML, and the Internet can all be modeled as graphs .

BANKS[2] enumerates results by searching the graph backwards from vertices that contain query keywords. The backward search heuristic concurrently executes copies of Dijkstra's shortest path algorithm [7], one from each vertex that contains a search term. When a vertex has been labelled with its distance to each search term, that vertex is the root of a directed tree that is a result to the query. BANKS-II augments the backward search heuristic [2] by searching the graph forwards from potential root nodes. This strategy has an advantage when the query contains a common term or when a copy of Dijkstra's shortest path algorithm reaches a vertex with a large number of incoming edges. Spreading activation prioritizes the search but may cause the bidirectional search heuristic to identify shorter paths after creating partial results. When a shorter path is found, the existing results must be updated recursively, which potentially increases the total execution time.

Although finding the optimal group Steiner tree is NP-complete ,there are efficient algorithms to find the optimal tree for a fixed number of terminals (i.e., search terms). DPBF [8] is a dynamic programming algorithm for the optimal solution but remains exponential in the number of search terms. The algorithm enumerates additional results in approximate order.
He et al. [12] propose a bi-level index to improve the performance of bidirectional search . BLINKS partitions the graph into blocks and constructs a block index and intra block index. These two indices provide a lower bound on the shortest distance to keywords, which dramatically prunes the search space.

STAR[16] is a pseudo polynomial-time algorithm for the Steiner tree problem. It computes an initial solution quickly and then improves this result iteratively. Although STAR approximates the optimal solution, its approximation ratio is significantly better than previous heuristics.

4. Related Work

Existing evaluations of relational keyword search systems are ad hoc with little standardization. Webber [22] summarizes existing evaluations with regards to search effectiveness. Although Coffman and Weaver [5] developed the benchmark that we use in this evaluation, their work does not include any performance evaluation. Baid et al. [1] assert that many existing keyword search techniques have unpredictable performance due to unacceptable response times or fail to produce results even after exhausting memory. Our results—particularly the large memory footprint of the systems—confirm this claim.A number of relational keyword search systems have been published beyond those included in our evaluation. Chen et al. [4] and Chaudhuri and Das [3] both presented tutorials on keyword search in databases. Yu et al. [35] provides an excellent overview of relational keyword search techniques.

Liu et al. and SPARK [22] both propose modified scoring functions for schema-based keyword search. SPARK also introduces a skyline sweep algorithm to minimize the total number of database probes during a search. Qin et al. [20] further this efficient query processing by exploring semi-joins. Baid et al. [1] suggest terminating the search after a predetermined period of time and allowing the user to guide further exploration of the search space.In the area of graph-based search techniques, EASE indexes all r-radius Steiner graphs that might form results for a keyword query. Golenberg et al. [12] provide an algorithm that enumerates results in approximate order by height with polynomial delay. Dalvi et al. [6] consider keyword search on graphs that cannot fit within main memory. CSTree provides alternative semantics—the compact Steiner tree—to answer search queries more efficiently.

In general, the evaluations of these systems do not investigate important issues related to performance (e.g., handling data graphs that do not fit within main memory). Many evaluations are also contradictory, for the reported performance of each system varies greatly between different valuations. Our experimental results question the validity of many previous evaluations, and we believe our benchmark is more robust and realistic with regards to the retrieval tasks than the workloads used in other evaluations. Furthermore, because our evaluation benchmark is available for other researchers to use, we expect our results to be repeatable.

5. Proposed System

In this proposed system, we are going to make An Advanced Information Retrieval System of Relational Keyword Search Scheme. Existing system in which many existing search techniques do not provide satisfactory performance for realistic retrieval tasks. In particular systems, memory utilization consist of many search techniques. We are going to explain relationship between execution time and factors different in previously evaluations; our investigation indicates that these factors have moderately little conflict on performance. In summary, our work will confirm the previous claim which is regarding with the improper performance of these systems and underscores the need for the consistency as represent by the IR area when we are going to examine these retrieval systems.

6. Algorithms

1. It results the files on basis of the file usage by *Breadth-First algorithm.*
2. Chart represents the ranking of the keyword searched by the user using *Djikstra's shortest path algorithm.*
3. Keyword search is essential for computing the results quickly by using *Steriner Tree Problem* and improves time-taken for the search by using *PseudoPolynomial Time alogirthm.*
4. Discovers the files by its keyword and executes it in a fraction of second for the user by using *Sparse algorithm.*

7. Conclusion and Future Work

Unlike many of the evaluations reported in the literature, ours is designed to investigate not the underlying algorithms but the overall, end-to-end performance of these retrieval systems. Hence, we favor a realistic query workload instead of a larger workload with queries that are unlikely to be representative (e.g., queries created by randomly selecting terms from the dataset).Overall, the performance of existing relational keyword search systems is somewhat disappointing, particularly with regard to the number of queries will be completed in proposed query workload. Given previously published results, we were especially surprised by the number of timeout and memory exceptions that we witnessed. Because our larger execution times might only reflect our choice to use larger datasets, we focus on two concerns that we have related to memory utilization.

References

[1] A. Baid, A. Doan, I. Rae, J. Li, and J. Naughton, “Towards Scalable Keyword Search over Relational Data,” VLDB Endowment, vol. 3, no. 1, pp. 140–149, 2010.
[2] G. Bhalotia, S. Chakrabarti , A. Hulgeri, C. Nakhe, and S. Sudarshan , “ Keyword Searching and Browsing in Databases using BANKS,” in Proceedings of the 18th

International Conference on Data Engg., ICDE 02, February 2002, pp. 431–440.

[3] G. Das ,and Chaudhuri S, "Keyword Querying and Ranking in Database,"VLDB Endowment, vol. 2, pp. 1658–1659, August 2009. .

[4] W. Wang, Z. Liu, and X. Lin, Y. Chen, "Keywords Searching on Structured and Semi-Structured Data," in Proceedings of the 35th SIGMOD International Conference on Mgt of Data, ser. SIGMOD 09, June 2009, pp. 1005–1010.

[5] A. C. Weaver, J. Coffman, "A Framework for Evaluating Database Keyword Search Strategy," in Proceedings of the 19th ACM International Conference on Information and Knowledge Management, ser. CIKM 10, October 2010, pp. 729–738.

[6] M. Kshirsagar, B. Dalvi and S. Sudarshan, "Keyword Search on External Memory Data Graph," Proceedings of the VLDB Endowment,vol. 1, no. 1, pp. 1189–1204, 2008.

[7] Dijkstra E, "A Note on Two Problems in Connexion with Graph," Numerische Mathematik, vol. 1, no. 1, pp. 269–271, 1959.

[8] B. Ding, J. X. Yu, S. Wang, L. Qin, X. Zhang, and X. Lin, "Finding Topk Min-Cost Connected Trees in Databases," in ICDE '07: Proceedings of the 23rd International Conference on Data Engineering, April 2007,pp. 836–845.

[9] R. A. Wagner and S. E. Dreyfus, "The Steiner Problem in Graph," Networks, vol. 1, no. 3, pp. 195–207, 1971.

[10] D. Fallows, "Search Engine Use," Pew Internet and American Life Project, Tech. Rep., August 2008, http://www.pewinternet.org/Reports/2008/Search-Engine-Use.aspx.

[11] Y. Sagiv, B. Kimelfeld, K. Golenberg ,"Keyword Proximity Search in Complex Data Graph," in Proceedings of the 2008 ACM SIGMOD International Conference on Mgt of Data, ser. SIGMOD 08,June 2008, pp. 927–940.

[12] H. Wang, J. Yang, and H. He, P. S. Yu, "BLINKs: Ranked Keyword Searches on Graph," in Proceedings of the 2007 ACM SIGMOD International Conference on Mgt of Data, ser. SIGMOD 07,June 2007, pp. 305–316.

[13] L. Gravano, V. Hristidis and Y. Papakonstantinou, "Efficient IR-style Keyword Search over Relational Databases," in Proceedings of the 29th International Conference on Very Large Data Base, ser. VLDB '03, September 2003, pp. 850–861.

[14] V. Hristidis and Y. Papakonstantnou, "DISCOVER: Keyword Search in Relational Databases," in Proceedings of the 29th International Conference on Very Large Data Bases, ser. VLDB '02. VLDB Endowment, August 2002, pp. 670–681.

[15] M. Sozio, F. M. Suchanek G. Kasneci, M. Ramanath, , and G. Weikum,"STAR: Steiner-Tree Approximation in Relationship Graphs," in Proceedings of the 25th International Conference on Data Engg, ser.ICDE 09, March 2009, pp. 868–879.

[16] S. Pandit, S. Chakrabarti, S. Sudarshan, R. Desai, V. Kacholia and H. Karambelkar, "Bidirectional Expansion For Keyword Search on Graph Database," in Proceedings of the 31st International Conference on Very Large Data Bases, ser. VLDB 05, August 2005, pp. 505–516.

[17] *X. Lin*, W. Wang, Y. Luo, and X. Zhou, " Top.k Keywords Query in Relational Database SPARK," in Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, ser. SIGMOD '07,June 2007, pp. 115–126.

[18] C Manning, H. Schutze ,and P. Raghavan "Introduction to Information Retrieval". New York, NY: Cambridge University Press, 2008.

[19] Nandi A. and H. Jagdish, "Qunits : queried unit for databases search," in CIDR '09: Proceedings of the 4th Biennial Conference on Innovative Data Systems Research, January 2009.

[20] L. Qin and J. X. Yu and L. Chang, "Keyword Search in Database :The Power of RDBMS," in Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data, ser. SIGMOD ,June 2009, pp. 681–694.

[21] L. Qin, J. Yu and L. Chang, Y. Tao, "Querying Communities in Relational Database," in Proceedings of the 25th International Conference on Data Engg,ser. ICDE , March 2009, pp. 724–735.

[22] Webber W, "Evaluating the Effectiveness of Keyword Search," IEEE Data Engg Bulletin, vol. 33, no. 1, pp. 54–59, 2010.

[23] A. Moffat,, W. Webber and J. Zobel, "Statistical Power in Retrieval Experimentation," in CIKM '08: Proceeding of the 17th ACM International Conference on Information and Knowledge Mgt, 2008,pp. 571–580.

[24] L. Qin, and L. Chang, J. X. Yu ,"Keyword Search in Databases", 1st E.Morgan and Claypool Publisher, 2010.