Improving the Performance of Keyword Search over Relational Database

P. Sathishkumar[1], Dr. M. Gunasekaran[2]
[1]Assistant Professor/ CSE, Bannari Amman Institute of Technology, Sathyamangalam, Erode, Tamilnadu
[2]Associate Professor/ IT, Bannari Amman Institute of Technology, Sathyamangalam, Erode, Tamilnadu

Abstract
The necessity of relational databases grows very larger .Relational database is structured to recognize relations between stored items of information. Extending the keyword search to relational data has been an active area of research. Many techniques have been proposed but all those techniques suffer from lack of standardization. Lack of standardization results in contradictory results. Keyword queries on databases provide easy access to data, but often suffer from low ranking quality, i.e., low precision and/or recall, as shown in recent benchmarks. It would be useful to identify queries that are likely to have low ranking quality to improve the user satisfaction. For instance, the system may suggest to the user alternative queries for such hard queries. In this paper, we analyze the characteristics of hard queries and propose a novel framework to measure the degree of difficulty for a keyword query over a database, In summary, our work confirms previous claims regarding the unacceptable performance of these search techniques and underscores the need for standardization in evaluations

Keywords — *Keyword search, relational, database,Graph.*

I. INTRODUCTION

The amount of information in the world is increasing exponentially. Keyword search has proven to be an effective method to discover and retrieve information online as evidenced by the success of Internet search engines. Unfortunately, many common information management systems do not support the familiar keyword search interface that people now expect. Web sites, corporations, and governments all use relational databases to manage information, but keyword search in relational databases is difficult due to data transformations that eliminate redundancy and ensure consistency. Relational keyword search enables users to retrieve information and to explore the relationships among that information all via a familiar interface.

Unlike many evaluations that appear in the literature, our benchmark uses realistic data sets and realistic queries to investigate the numerous tradeoffs made in the design of these search techniques. Our benchmark is the only one to date in the literature that satisfies the *minimum* criteria established by the IR community for the evaluation of retrieval systems.

The major contributions of this paper are as follows:

- We conduct an independent, empirical evaluation of the runtime performance of seven relational keyword search techniques. Our evaluation is the most extensive and thorough one to appear to date in the literature.
- Our results do not substantiate previous claims regarding the scalability and performance of relational keyword search techniques. Existing search techniques perform poorly on databases exceeding tens of thousands of tuples or require an inordinate amount of memory.
- We show that many parameters varied in existing evaluations are at best loosely
- correlated with runtime performance. The lack of a meaningful relationship gives merit to previous claims of unpredictable performance [6] for existing search techniques.

- Our work is the first to combine performance and search effectiveness in the evaluation of such a large number of search techniques. Considering these two issues in conjunction provides better understanding of these two critical trade offs among competing approaches.

The remainder of this paper is organized as follows: Section 2 formally defines the problem of keyword search in relational data graphs and describes the search techniques included in our evaluation. Section 3 describes our experimental setup, including our evaluation benchmark and metrics. In Section 4, we present our experimental results. We review related work in Section 4 and provide our conclusions in Section 5. Online appendices provide greater detail about our evaluation benchmark and summarize implementation details of the search techniques.

II. KEYWORD SEARCH IN RELATIONAL DATABASES

Keyword search on semi-structured data (e.g., XML) and relational data differs considerably from traditional IR. For instance, the granularity of search results must be reconsidered for both these data sources. An XML document might contain a single element that is pertinent to a given query along with many unrelated elements. The XML dump of the Digital Bibliography & Library Project (DBLP)4 currently contains more than 1.3 million publications; searching this repository for a particular paper should return only the information about that paper and not the complete bibliography.

Identifying relevant results is further complicated due to the fact that a physical view of the data often does not match a logical view of the information. Relational databases normalize data to eliminate redundancy. Logically related information often appears in different relations, and foreign keys provide the only link between the data. Whenever search queries cross a relationship modeled in the database (i.e., a subset of search terms is present in one tuple and the remaining terms are found in related tuples), the data must be mapped back to a logical view to obtain meaningful search results.

III. EVALUATION FRAMEWORK

In this section, we present our evaluation framework. We start with our benchmark and then describe our metrics and experimental setup. We refer the reader to the benchmark's original description [3] for additional details that space precludes us from repeating here.

A. Benchmark Overview

Our evaluation benchmark includes the three data sets shown in Table 3: MONDIAL [21], IMDb, and Wikipedia. Two data sets (IMDb and Wikipedia) are extracted from popular websites. As shown in Table 3, the size of the data sets varies widely: MONDIAL is more than two orders of magnitude smaller than the IMDb data set, and Wikipedia lies in between. In addition, the schemas and content also differ considerably.

B. Metrics

Schema-based search techniques support keyword search over relational databases via direct execution of SQL commands. These techniques model the relational schema as a graph where vertices are relational tables and edges denote foreign keys between tables. Query processing follows three phases. First, database tuples that contain search terms are identified. Second, candidate networks (SQL expressions) that could relate these tuples are systematically enumerated. Third, these SQL expressions are executed against the database to identify results, which are returned to the user. precision value at the last result. We also use MAP to measure retrieval effectiveness at greater retrieval depths.

C. Graph-based Approaches

Graph-based approaches assume the database is modeled as a weighted graph where the weights of edges indicate the importance of relationships. Proximity search strategies attempt to minimize the weight of result trees. This task is a formulation of the group Steiner tree problem [DW71], which is known to be

NP-complete [RW90]. In addition to ranking results by their total edge weight, many search techniques also include a prestige (i.e., node weight) factor to prefer results that contain more highly-referenced database tuples. Graph-based search techniques are more general than schema-based approaches, for relational databases, XML, and the Internet can all be modeled as graphs.

Data Sets

Two of the most common datasets (DBLP and IMDb) lack a canonical relational schema, which leads to several different schemas appearing in the literature. The information contained within each dataset also varies For example, BANKS-II, BLINKS, and STAR all use a DBLP and IMDb dataset, but only BANKS-II's evaluation includes the entire database. Both BLINKS and STAR use smaller subsets to facilitate comparison with search techniques that assume the data graph.

IV. IMPLEMENTATION

Our implementation of BANKS adheres to its original description although it queries the database dynamically to identify nodes (tuples) that contain query keywords. Our implementation of DISCOVER borrows its successor's query processing techniques. Both DISCOVER and DIS- COVER-II are executed with the sparse algorithm, which provides the best performance for queries with AND semantics [17]. BLINKS's block index was created using breadth-first partitioning and contains 50 nodes per block. STAR uses the edge weighting scheme proposed by Ding et al. [12] for undirected graphs.

Experimental Setup

Our experimental setup is comparable to those reported in previous evaluations. We execute each query on a Linux machine running Ubuntu 10.04 with dual 1.6-GHz AMD Opteron 242 processors and 3 GB of RAM. We compiled each implementation using `javac` version 1.6 and ran the implementations with the Java HotSpot 64-bit server VM. PostgreSQL was our database management system.

Ranking Schemes

Relatively little work focuses on just ranking relational keyword search results. Most research papers propose schemes to improve runtime performance and search effectiveness rather than focusing on either enumeration or ranking. The evaluation presented indicates that structured cover density ranking has much more reliable performance than existing ranking schemes, and SVM rank outperforms all previous schemes that have been described in the literature.

TABLE 1
Summaries of Queries Completed and Exceptions

System	TO	VM	?	exec.	
BANKS	29	21	—	—	1910.9
DISCOVER	50	—	—	—	8.0
DISCOVER-II	50	—	—	—	6.5
BANKS-II	50	—	—	—	190.2
DPBF	50	—	—	—	11.1
BLINKS	50	—	—	—	23.6
STAR	50	—	—	—	0.3

(b) IMDb

BANKS	7	39	—	4	3239.7
DISCOVER	50	—	—	—	227.9
DISCOVER-II	50	—	—	—	201.8
BANKS-II	—	18	—	32	3604.3
DPBF	5	45	—	—	3399.3
BLINKS	—	—	50	—	—
STAR	—	—	50	—	—

V. EXPERIMENTAL RESULTS

The results shown in the table 1 that the number of queries executed successfully by each search technique for our data sets and also the number and types of exceptions we encountered. Of interest is the number of queries that were not completed successfully. Queries fail due to time outs (i.e., the algorithm had not terminated after 1 hour of execution time) or exhausting

virtual memory. In the table, these exceptions are indicated by "TO" and "VM."

A. *Execution Time*

In particular, the range in execution times for a search technique shown in fig 1 is often several orders of magnitude. Most search techniques also have outliers in their execution times; these outliers indicate that the performance of these search heuristics varies considerably. Previous evaluations—most of which report only the mean execution time for queries—have not acknowledged the existence of such outliers.

A number of previous evaluations [8], [12], [16], [17]

report mean execution time for queries that contain different numbers of search terms to show that performance remains acceptable even when queries contain more keywords.Some search techniques fail to complete some queries, which accounts for the omissions in the graph. As evidenced by the graph, queries that contain more search terms require more time to execute on average than queries that contain fewer search terms.

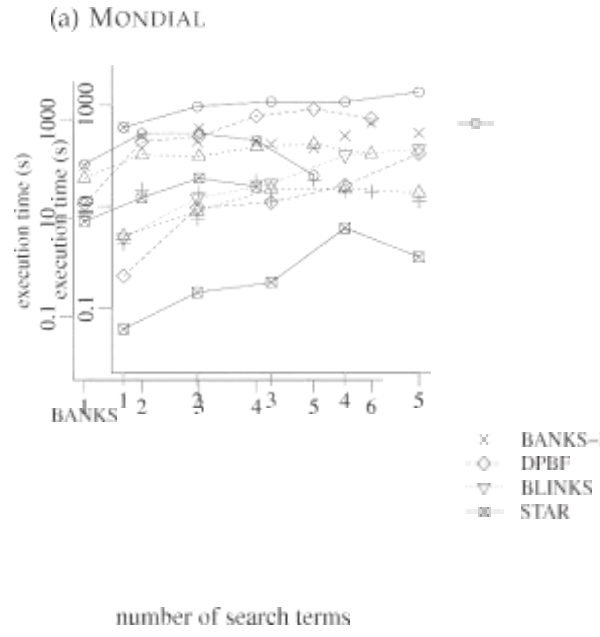

Fig. 1. Execution time versus query length for MONDIAL and Wikipedia queries; lower execution times are better. IMDb is omitted due to the few search techniques that successfully complete its queries. Note that the *y*-axis has a log scale.

VI. CONCLUSION

Unlike many evaluations reported in the literature, ours investigates the overall, end-to-end performance of rela- tional keyword search techniques. 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 data set).

Our experimental results do not reflect well on existing relational keyword search techniques. Runtime performance is unacceptable for most search techniques. Memory consumption is also excessive for many search techniques. Our experimental results question the scalability and improvements claimed by previous evaluations. These conclusions are consistent with previous evaluations that demonstrate the poor runtime performance of existing search techniques as a prelude to a newly-proposed approach.

A. *Future Work*

Further research is unquestionably necessary to investigate the myriad of experimental design decisions that have a significant impact on the evaluation of relational keyword search systems. For example, our results indicate that existing systems would be unable to search the entire IMDb database, which underscores the need for a progression of data sets that will allow researchers to make progress toward this objective.

Our results should serve as a challenge to this community because little previous work has acknowledged these challenges. Moving forward, we must address several issues. First, we must design algorithms, data structures, and implementations that recognize that main memory is limited. Search techniques must manage their memory utilization efficiently, swapping

data to and from disk as necessary.

REFERENCES

[1] D. Fallows, "Search Engine Use," technical report, Pew Internet and Am. Life Project, http://www.pewinternet.org/Reports/ 2008/Search-Engine-Use.aspx. Aug. 2008.

[2] comScore, "Global Search Market Grows 46 Percent in2009," Global_Search Market_Grows_46_%_in_2009, Jan. 2010.

[3] J. Coffman and A.C. Weaver, "A Framework for Evaluating Database Keyword Search Strategies," Proc. 19th ACM Int'l Conf. Information and Knowledge Management (CIKM '10), pp. 729-738, Oct. 2010.

[4] Y. Chen, W. Wang, Z. Liu, and X. Lin, "Keyword Search on Structured and Semi-Structured Data," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '09), pp. 1005-1010, June 2009.

[5] W. Webber, "Evaluating the Effectiveness of Keyword Search,"IEEE Data Eng. Bull., vol. 33, no. 1, pp. 54-59, Mar. 2010.

[6] A. Baid, I. Rae, J. Li, A. Doan, and J. Naughton, "Toward Scalable Keyword Search over Relational Data," Proc. VLDB Endowment, vol. 3, no. 1, pp. 140-149, 2010.

[7] Q. Su and J. Widom, "Indexing Relational Database Content Offline for Efficient Keyword-Based Search," Proc. Ninth Int'l Database Eng. and Application Symp. (IDEAS '05), pp. 297-306, July 2005.

[8] 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 '05), pp. 505-516, Aug. 2005.

[9] H. He, H. Wang, J. Yang, and P.S. Yu, "BLINKS: Ranked Keyword Searches on Graphs," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '07), pp. 305-316, June 2007.

[10] G. Kasneci, M. Ramanath, M. Sozio, F.M. Suchanek, and G. Weikum, "STAR: Steiner-Tree Approximation in Relationship Graphs," Proc. Int'l Conf. Data Eng. (ICDE '09), pp. 868-879, Mar. 2009.

[11] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan, "Keyword Searching and Browsing in Databases Using BANKS," Proc. 18th Int'l Conf. Data Eng. (ICDE '02), pp. 431-440, Feb. 2002.

[12] B. Ding, J.X. Yu, S. Wang, L. Qin, X. Zhang, and X. Lin, "Finding Top-k Min-Cost Connected Trees in Databases," Proc. 23rd Int'l Conf. Data Eng. (ICDE '07), pp. 836-845, Apr. 2007.

[13] 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 Int'l Conf. Management of Data (SIGMOD '08), pp. 903-914, June 2008.

[14] L. Qin, J. Yu, L. Chang, and Y. Tao, "Querying Communities in Relational Databases," Proc. IEEE Int'l Conf. Data Eng. (ICDE '09), pp. 724-735, Mar. 2009.

[15] G. Li, J. Feng, X. Zhou, and J. Wang, "Providing Built-in Keyword Search Capabilities in RDBMS," The VLDB J., vol. 20, pp. 1-19, Feb. 2011.

[16] V. Hristidis and Y. Papakonstantinou, "DISCOVER: Keyword Search in Relational Databases," Proc. 28th Int'l Conf. Very Large Data Base (VLDB '02), pp. 670-681, Aug. 2002.

[17] 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 '03), pp. 850-861, Sept. 2003

[18] A. Singhal, J. Choi, D. Hindle, D. Lewis, and F. Pereira, "AT&T at TREC-7," Proc. Seventh Text REtrieval Conf. (TREC-7), pp. 239-252, Nov. 1999.

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

[20] G. Reich and P. Widmayer, "Beyond Steiner's Problem: A VLSI Oriented Generalization," Proc. 15th Int'l Workshop Graph-Theoretic Concepts in Computer Science, pp. 196-210, 1990.