Proving Search Techniques Do Not Provide Acceptable Performance for Realistic Retrieval Tasks in Data Mining

S . I . S . Jaffarvalli
M.TECH student
AITS
Computer Science and Engineering
Rajampet, India.

M . Gunasekhar
M.TECH student
AITS
Computer Science and Engineering
Rajampet, India

***Abstract*— To search some information on the web still today we use paradigm to relational data has been an active area of research within the database .however this is not a best approach to retrieve or to mine data on the web. The ubiquitous search text box has transformed the way people interact with information. The ubiquitous search text box has transformed the way people interact with information. The following results indicate that many existing search techniques do not provide acceptable performance for realistic retrieval tasks. In particular, memory consumption precludes many search techniques from scaling beyond small datasets with tens of thousands of vertices.**

Keywords—component; formatting; style; styling; insert (key words)

I. INTRODUCTION

- The ubiquitous search text box has transformed the way people interact with information. Nearly half of all Internet users use a search engine daily [8], performing in excess of 4 billion searches [1]. The success of keyword search items from what it does not require—namely, a specialized query language or knowledge of the underlying structure of the data. Internet users increasingly demand keyword search interfaces for accessing information, and it is natural to extend this paradigm to relational data. This extension has been an active area of research throughout the past decade. However, we are not aware of any research projects that have transitioned from proof-of-concept implementations to deployed systems.

Keyword search on semi-structured data (e.g., XML) and relational data differs considerably from traditional IR.1 A discrepancy exists between the data's physical storage and a logical view of the information. Relational databases are normalized to eliminate redundancy, and foreign keys identify related information. Search queries frequently cross these relationships (i.e., a subset of search terms is present in one tuple and the remaining terms are found in related tuples), which forces relational keyword search systems to recover a logical view of the information. The implicit assumption of keyword search that is, the search terms are related complicates the search process because typically there are many possible relationships between two search terms It is almost always possible to include another occurrence of a search term by adding tuples to an existing result. This realization leads to tension between the compactness and coverage of search results.

II. MOTIVATION FOR INDEPENDENT EVALUATION

Consider the evaluations of BANKS-II [7], BLINKS [3] and STAR [1]. Only BANKS-II's evaluation includes the entire Digital Bibliography & Library Project (DBLP) 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. For example, a subset containing 1% of the original data is two orders of magnitude easier to search than the original database due to fewer tuples containing search terms bias our results toward the schema based approaches. The schema-based systems offload much of their work to the underlying database, which swaps temporary data (e.g., the results of a join) to disk as needed. Hence, DISCOVER and DISCOVER-II might also require a significant amount of memory, and a more fair evaluation would allow the graph-based techniques to page data to disk. To investigate this possibility, we ran all the systems9 with 3 GB of physical memory and 5 GB of virtual memory.10 Note that once a system consumes the available physical memory, the operating system's virtual memory manager is responsible for paging data to and from disk. Table X contains the results of this experiment. The overall trends are relatively unchanged from Table V although BLINKS does complete all the MONDIAL queries with the help of the additional memory. The precipitous drop in execution time suggests that Java's garbage collector was responsible for the majority of Blinks' execution time, and this overhead was responsible for Blinks' poor performance. The other graph-based systems do not significantly improve from the

additional virtual memory. In most cases, we observed severe thrashing, which merely transformed memory exceptions into timeout exceptions.

TABLE 1 STATISTICS FROM PREVIOUS EVOLUTIONS

System	Dataset	$\|V\|$	$\|E\|$	$\|Q\|$
BANKS [2]	bibliographic	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
	MovieLens	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
	MovieLens	1M		5
	previous 3			5
Golenberg *et al.* [12]	MONDIAL			36
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

.

III. RELATIONALKEYWORD SEARCH SYSTEMS

Given our focus on empirical evaluation, we adopt a general model of keyword search over data graphs. This section presents the search techniques included in our evaluation; other relational keyword search techniques are mentioned in Section VI. Problem definition: We model a relational database as a graph G = (V;E). Each vertex v 2 V corresponds to a tuple in the relational database. An edge (u; v) 2 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. 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 [15] 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.

TABLE 2 DATASET RESULTS

Dataset	$\|V\|$	$\|E\|$	$\|T\|$
MONDIAL	17	56	12
IMDb	1673	6075	1748
Wikipedia	206	785	750

IV. DESCRIPTION OF EXISTING SEARCH TECHNIQUES

A. *DISCOVER*

DISCOVER [HP02] creates a set of tuples for each subset of search terms that appear in the database relations. Determining the optimal execution strategy that DISCOVER [HP02] creates a set of tuples for each subset of search terms that appear in the database relations. Determining the optimal execution strategy that has the lowest total evaluation cost for a set of candidate networks is an NP-complete problem so a greedy algorithm is proposed to enumerate results. The greedy algorithm prioritizes candidate networks that generate the fewest intermediate results and also candidate networks that share common sub expressions with other candidate networks as the lowest total evaluation cost for a set of candidate networks is an NP-complete problem so a greedy algorithm is proposed to enumerate results. The greedy algorithm prioritizes candidate networks that generate the fewest intermediate results and also candidate networks that share common sub expressions with other candidate networks

B. *DISCOVER2*

Hristidis et al. [HGP03] reined DISCOVER by adopting an IR-style scoring function to Rank results.

$$score(T,Q) = \frac{1}{|T|}\sum_{a} score(a,Q)$$

Pivoted normalization weighting [SCH+99] is a state-of-the-art scoring function [Sin01]: where t is a query term, (q) is the frequency of the (query) term, s is a constant (usually 0.2), dl is the document length, average dl is the mean document length, N is the number of documents in the collection, and f the number of documents that contain t. The values of each database attribute form a distinct document collection. The total score for a result T is the sum of all the individual attribute scores normalized by the size of the result.

V. INITIAL MEMORY CONSUMPTION

To better understand the memory utilization of the systems particularly the overhead of an in-memory data graph, we measured each system's memory footprint immediately prior to executing a query. The results of the graph representation of the database; the right column of values gives the total size of all data structures used by the search techniques (e.g., additional index structures). As evidenced by the table, the schema-based systems consume very little memory, most of which is used to store the database schema. In contrast, the graph-based search techniques require considerably more memory to store their data graph. When compared to the total

amount of virtual memory Available, the size of the MONDIAL data graphs are quite small, roughly two orders of magnitude smaller than the size of the heap. Hence, the data graph itself cannot account for the high Memory utilization of the systems; instead, the amount of state maintained by the algorithms (not shown by the table) must account for the excessive memory consumption. For example Bank's worst-case memory consumption is O (jVj2) where j is the number of vertices in the data graph. It is easy to show that in the worst case BANKS will require in excess of 1 GB of state during a search of the MONDIAL database even if we ignore the overhead of the requisite data structures (e.g., linked lists).

TABLE 3 INITIAL MEMORY CONSUMED

System	✓			exec. (s)	speedup (%)
BANKS	30	20	—	1817.3	3.7
DISCOVER	**50**	—	—	6.1	-10.9
DISCOVER-II	**50**	—	—	6.3	-12.5
BANKS-II	**50**	—	—	238.7	15.4
BLINKS	**50**	—	—	20.3	91.5
STAR	**50**	—	—	0.3	33

(a) MONDIAL

System	✓			exec. (s)	speedup (%)
BANKS	3	40	—	3448.8	—
DISCOVER	**50**	—	—	221.6	-0.5
DISCOVER-II	**50**	—	—	195.0	-0.6
BANKS-II	—	18	—	3607.0	—
BLINKS	—	—	50	—	—
STAR	—	—	50	—	—

(b) IMDb

System	✓			exec. (s)	speedup (%)
BANKS	4	46	—	3324.5	-4.7
DISCOVER	**50**	—	—	34.0	-3.3
DISCOVER-II	**50**	—	—	33.1	-4.1
BANKS-II	11	36	—	2909.4	9.2
BLINKS	—	—	50	—	—
STAR	—	—	50	—	—

(c) Wikipedia

VI. CONCLUSION

Overall, the performance of existing relational keyword search systems is somewhat disappointing, particularly with regard to the number of queries completed successfully in our query workload (see Table VI). Given previously published results (Table II), 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. First, no system admits to having a large memory requirement. In fact, memory consumption during a search has not been the focus of any previous evaluation. Making the original source code (or a binary distribute on that accepts a database URL and query as input) available to other researchers would be ideal and greatly reduce the likelihood that observed differences are implementation Artifacts.

REFERENCES

[1] A. Baid, I. Rae, J. Li, A. Doan, and J. Naughton, "Toward Scalable Keyword Search over Relational Data," Proceedings of the VLDB Endowment, vol. 3, no. 1, pp. 140–149, 2010.
[2] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan," Keyword Searching and Browsing in Databases using BANKS," in Proceedings of the 18th International Conference on Data Engineering's. ICDE '02, February 2002, pp. 431–440.
[3] S. Chaudhuri and G. Das, "Keyword Querying and Ranking in Databases,"Proceedings of the VLDB Endowment, vol. 2, pp. 1658–1659, August2009. [Online]. Available:
[4] Y. Chen, W. Wang, Z. Liu, and X. Lin, "Keyword Search on Structured and Semi-Structured Data," in Proceedings of the 35th SIGMOD International Conference on Management of Dataset. SIGMOD '09, June 2009, pp. 1005–1010.
[5] J. Coffman and A. C. Weaver, "A Framework for Evaluating Database Keyword Search Strategies," in Proceedings of the 19th ACM International Conference on Information and Knowledge Management, ser. CIKM '10, October 2010, pp. 729–738. [Online]. Available:
[6] B. B. Dalvi, M. Kshirsagar, and S. Sudarshan, "Keyword Search on External Memory Data Graphs," Proceedings of the VLDB Endowment, vol. 1, no. 1, pp. 1189–1204, 2008.
[7] E. W. Dijkstra, "A Note on Two Problems in Connection with Graphs," Numeric Mathematics, 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] S. E. Dreyfus and R. A. Wagner, "The Steiner Problem in Graphs," Networks, vol. 1, no. 3, pp. 195–207, 1971. [Online]. Available: