Ranking Issues for Information Integration[*]

Aditya Telang, Roochi Mishra and Sharma Chakravarthy
IT Laboratory and Department of Computer Science & Engineering
The University of Texas at Arlington, Arlington, TX 76019.
{telang, mishra, sharma}@cse.uta.edu

Abstract

Ranking of query/search answers, although introduced by early information retrieval systems, has become mandatory for internet searches. When the answers of a query or search are varying in quality and are large in numbers, it is necessary to rank/order these answers based on some criteria. From a users' viewpoint, ranking is extremely useful especially when associated with the retrieval of a few (top-k) answers. Currently, the notion of ranking is being applied to database query answers (ordering based on user criteria) in order to retrieve top-k answers. One of the challenges is to push ranking computation into the query processing stage to make it efficient and eliminate post processing operations.

In this paper, we argue that top-k answers and ranking will become mandatory as well for searches that involve integration of information from heterogeneous domains. However, it is not clear how ranking can be supported as sources are autonomous and support different characteristics and capabilities. We analyze possible alternatives for supporting ranking while answering queries in this context and propose potential approaches. The discussion is in the context of InfoMosaic*, a framework proposed by the authors for information integration.*

1 Introduction

The volume of information accessible via the web is staggeringly large and growing rapidly. The coverage of information available from web sources is difficult to match by any other means. Hence, over the last decade or so, search engines [1] have become extremely popular and have facilitated users to quickly and effortlessly retrieve information from individual sources. Conceptually, search engines perform the equivalent of a simple *lookup* operation on one or more keywords, followed by a sophisticated `ranking` of the large volume of generated results [2]. These ranking mechanisms may be *user-specified* (e.g., airfare ranking based on prices, flights, class) or *system-specified* (e.g., Google page-rank).

On the other hand, despite the robustness and efficient query-handling capabilities of database management systems, the lack of effective means to order large number of results affects the usability of these systems on the web. Even in cases where the database holds correct and consistent data, there is a need to retrieve top-k results based on *user-specified criteria*. As a result, the issue of *top-k answers* and in turn, the need for ranking in database systems [3] has become extremely important.

The simplicity associated in using search engines and/or querying web-databases makes it difficult to specify queries that require extraction of data from multiple repositories across diverse domains. For example, consider the query: *"Retrieve all castles within 2 hours by train from London"*. Although all the information for answering this query is available on the web, it is currently **not possible** to frame it as a query and get all or some of the answers. The current process for identifying answers to the above query essentially involves the following steps: i) get a list of cities with castles using a search engine (e.g., Google search on "castles in UK"), ii) choose some cities (from the list of castles) and look up the distances between that city and London, iii) check train connectivity with a specific city by finding an appropriate source that provides train schedules in UK (another Google search on "Train schedules in UK" followed by a search on a web-database exposed by a query interface), iv) *manually* retrieve the time for the travel, and v) *manually* **rank** the final results based on some user criteria such as antiquity of the castle, how much time can be spent at the castle etc. Thus, the answer to the above (as well as similar) query can be put together *manually with some/considerable effort* and the **rank of the results computed manually** by retrieving and combining different pieces of information from different individual web sites in an *intelligent manner*.

Although the gist of information integration has not changed and this topic has been investigated over two

[*]The work was supported, in part, by NSF grants IIS - 0534611, IIS - 0326505, and EIA - 0216500.

decades, the problem at hand is quite different and far more complex than the one attempted earlier. As the number of repositories/sources will increase steadily, there is no other option but to find a solution for integrating information from different autonomous sources as needed for a search/query whose answers have to be retrieved, integrated and efficiently ranked from multiple domains and sources. Towards this end, we are currently developing *InfoMosaic*, a framework with well-defined components, domain and source knowledge, and a set of new techniques/algorithms. Instead of limiting integration of information from multiple sources that belong to a single domain, this framework considers multiple autonomous domains that include spatial, temporal, as well other data types existing in various formats (structured, semi-structured and unstructured). In this project, we are addressing several issues of such multi-domain information integration – from query/search specification to query elaboration, alternate plan generation and optimization, maintenance of domain knowledge and source semantics, data extraction from individual repositories (web, database or others), integration of information, and finally generating answers based on user as well as system-specified *ranking* metrics. We will elaborate on the last component in this short paper.

2 Related Work

There has been a great deal of work on document ranking based on *vector-space* models [4], *probabilistic* models [5], *fuzzy-sets* [6], etc. Most of these used the content of the documents – *term frequency* (TF) and *inverse document frequency* (IDF) – to determine the relevance and rank of a document. Existing search engines have indexed a significant amount of information available on the internet. Consequently, the number of pages providing information associated with any search topic can range in the order of thousands (or more depending on the search keyword). In order to filter the relevant data from these vast reservoirs of information, search engines adopt sophisticated ranking mechanisms. Some domain-specific web-portals (e.g., http://www.expedia.com, http://www.amazon.com, etc.) provide users a fixed, pre-defined number of choices and filter the results based on these (for example, ranking airfare based on prices, ranking books based on author-popularity, etc.). However, *all* search engines have an inbuilt ranking criteria (e.g., *google sauce* in Google) that facilitates generating results based on pre-determined metrics. There is a significant difference between ranking in earlier information retrieval systems and web search engines. Unlike information retrieval systems, the number of documents (or web pages) are very large, uncontrolled, and the kind and quality of information is unknown. As a result, earlier techniques could not be directly used for this purpose. As a result, a number of new approaches and algorithms have been developed. Google's PageRank algorithm [7] is an example which used the equivalent of scholarly citation criteria for ranking web pages. Currently, there continues to be significant amount of research in document and page ranking for search engines.

As databases become available on the web in the form of hidden-web [8], it becomes necessary for the query interfaces exported by these database to support IR-like queries. Additionally, as the results generated by the database are extremely large (and needs to be retrieved from remote sources), it becomes mandatory to rank and present a few results at a time instead of all the possible answers. In many applications, *ranking* the query results based on some criteria is an integral part of the query semantics. Hence, the increasing importance and applicability of ranked data retrieval warrants an efficient support of ranking in practical database systems. To this effect, several techniques such as probabilistic ranking [9], automated ranking [10] and other SQL-based ranking algorithms [3, 11] have been proposed.

In the information integration environment, several frameworks such as Havasu [12], MetaQuerier [13], Ariadne [14], TSIMMIS [15], InfoMaster [16], Information Manifold [17], and others have addressed and tackled a number of challenges in a delimited context. However, to the best of our knowledge, *ranking* has not been addressed explicitly in any of the major projects on information integration.

3 The InfoMosaic Framework

In this section, we briefly describe the architecture of InfoMosaic [18] along with the functional modules (Figure 1). The user query is accepted in an intuitive manner using domain names and keywords of interest, and elaborated/refined using a *domain knowledge* in the form of taxonomies and dictionaries (synonyms etc.). Requests are refined and presented to the user for feedback (Query Refinement module). Once the query is finalized, it is represented in a canonical form (e.g., query graphs) and transformed into a query plan using a two-phase process: 1) generation of logical plans using domain characteristics, and 2) generation of physical plans using *source semantics*. The plan is further optimized by applying several metrics (Query planner and Optimizer module). The Query Execution and Data Extraction module generates the actual source queries that is used by the extractor to retrieve the requisite data. It also determines whether a previously retrieved answer can be reused. There is an XML Repository that stores extracted results from each source in a system-determined format. A separate PostGIS repository is used for storing spatial data extracted from sources. The Data Integrator formulates XQueries (with external functions for handling spatial component) on these repositories to compute the final answers and format them for the user. A number of knowledge-bases

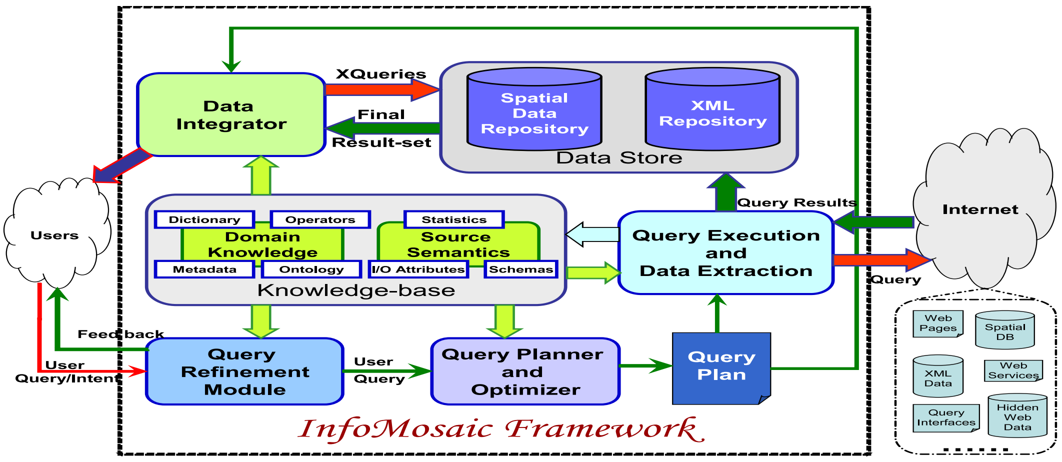

Figure 1. InfoMosaic Architecture

(e.g., *domain knowledge* and *source semantics*) blend all the pieces together in terms of the information used by various modules. The adaptive capability of the system is based on the ability of the InfoMosaic components to update these knowledge-bases at runtime.

As elaborated in Section 2, the problem of ranking in information integration is different from that of information retrieval and database systems. Moreover, we believe that as search mechanisms shift their focus from basic keyword retrieval to true information integration, the challenge of efficiently ranking large volume of integrated data will become critical. To address these issues, we consider the task of addressing *ranking* techniques in our framework.

4 Ranking Issues in Information Integration

It is clear that unlike the domains of information retrieval or even databases, the computation of ranking in information integration is more complex due to: i) autonomous nature of sources, ii) lack of information about the quality of information from a source, iii) lack of information about the amount of information (equivalent of cardinality) for a query on the source, iv) lack of support for retrieving results in some order or based on some metrics, and v) presence of support but need to be understood and applied properly. Our goal in InfoMosaic is to identify the kinds of information needed to address all aspects of information integration including ranking. For example, in the *castles query*, if we can get train schedules in a time ordered manner starting at a particular time (which is actually possible), it can be effectively used for one of the user-specified ranking metrics (maximum time at the destination). Based on the above, our approach in InfoMosaic will be to strengthen the knowledge base and develop techniques to leverage known information effectively. Of course, there is the concomitant issue of collecting or inferring this information. Our belief is that once we know what information is needed and useful, we can work towards acquiring that information. Some of the extant techniques, such as mining, sampling, and others will provide ways to collet required information.

InfoMosaic has multiple ways of executing user requests. One of them is to extract answers for each subquery (of the entire query) into InfoMosaic storage and combine/join them to generate the result. In this approach, ranking the answers will be relatively easier as each subset of results can be sorted and ranked at the middleware based on the local criterion and then the results integrated to generate top-k answers that meets the overall raking metric. Even in this approach, certain qualitative issues need to be resolved or inferred using domain semantics and inferencing. For example, in the *castles query*, the notion of antiquity needs to be mapped to age of the castle. If total time available at the destination is a ranking criterion, that has to be mapped to the difference in times of the train schedule. Even for this approach, it is possible to extract portions of results from each source and the answers computed using the approach described above. However, this will only provide local ranking of answers instead of global ranking of answers.

A more effective approach will be to make use of characteristics and support provided by the source, and push ranking computation as low as possible in the query plan. In this approach we will consider the combination of query plans and source capabilities to help compute the ranking function. Ranking will be addressed at – the domain level to map ranking metric to attributes of the domain (where possible) and the source level to use the characteristics as explained below. For example, in the *castles query*, it is possible to get all the castles and their cities from a source and use each city from the list to check the train connectivity. If, for example, in addition to city, the age (or the date of construction) of the castle is available, then the castles can be ranked based on the age before retrieving the train connectivity. In this case the source may not have to support ranking explicitly; through the knowledge base, the system can infer relationship between the two and if the *date* information can be retrieved, it can be effectively used for ranking. As an alternative, a source may explicitly provide ranked results (e.g., hotels based on price) which can be used by the system to calculate ranking of the results. It is possible that some sources support ranking and some do not. This may entail reordering of the evaluation of sub-queries in order to minimize the information retrieved and to process the query efficiently. The choice of sources will be influenced by ranking as well.

A third alternative will be to sample frequently used sources to gather information that would be useful at run time for ranking or even use the data extracted for different sub-queries to infer additional ranking information. Since adaptability is central to the success of InfoMosaic, updating the knowledge base is critical as it is used for all aspects of query processing including ranking. The data extraction and data integrator modules of InfoMosaic deal with ranking based on the source level query plan generated and will use additional knowledge associated with a source (domain and source semantics in Figure 1). Furthermore, since a user is an important part of the InfoMosaic framework, we plan to incorporate a number of optional user-specified inputs including ranking criteria for answers. We intend to facilitate query/search specifications that will include the following: i) specification of *soft* (or *imprecise*) queries instead of *hard* queries, ii) request for *top-k* answers (if appropriate knowledge about the sources is available), iii) *ranking* metric (could be a combination that can mao to different sources), iv) and *query relaxation* for retrieving approximate answers in the absence of exact results that satisfy all the user-specified constraints, and v) optional *quality-of-service* (QoS) criteria to be used during retrieval of information. We plan on exploring both system-defined ranking metrics (where appropriate and possible) and user-specified ranking metrics.

In summary, we propose several approaches for addressing ranking in the context of information integration: i) apply ranking metrics before combining results that have been retrieved from individual sources, ii) infer attributes and their values that can be retrieved from a source using which ranking for subquery results can be computed for use in the overall query, iii) exploit explicit ranking schemes provided by a source, iv) ordering of subqueries based on the ranking computation available, and v) prior sampling of sources to gather information about sources that are specific to ranking. The above is derived from our analysis of the problem, understanding of ranking approaches/techniques in information retrieval, search, database exploration. Of course, for each of the above categories, algorithms and techniques needs to be fleshed out and we are currently doing that in the context of InfoMosaic.

5 Conclusion

In this paper, we introduced the problem of ranking in the domain of information integration. We articulated the issues associated with ranking and how it differs from the problem of ranking in other domains. We discussed several alternative approaches in the context of InfoMosaic to handle ranking. Currently, we are working on ranking as well as other aspects of information integration.

References

[1] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine." *Computer Networks*, vol. 30, no. 1-7, pp. 107–117, 1998.

[2] M. Sahami, V. O. Mittal, S. Baluja, and H. A. Rowley, "The Happy Searcher: Challenges in Web Information Retrieval." in *PRICAI*, 2004.

[3] C. Li, M. A. Soliman, K. C.-C. Chang, and I. F. Ilyas, "RankSQL: Supporting Ranking Queries in Relational Database Management Systems." in *VLDB*, 2005, pp. 1342–1345.

[4] D. L. Lee, H. Chuang, and K. E. Seamons, "Document Ranking and the Vector-Space Model." *IEEE Software*, vol. 14, no. 2, pp. 67–75, 1997.

[5] W. R. Greiff and J. M. Ponte, "The Maximum Entropy approach and Probabilistic IR Models." *ACM Trans. Inf. Syst.*, vol. 18, no. 3, pp. 246–287, 2000.

[6] M. He and B. Feng, "Intelligent Information Retrieval Based on the Variable Precision Rough Set Model and Fuzzy Sets." in *RSFDGrC*, 2005, pp. 184–192.

[7] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web," Stanford Digital Library Technologies Project, Tech. Rep., 1998.

[8] K. C.-C. Chang, B. He, and Z. Zhang, "MetaQuerier over the Deep Web: Shallow Integration across Holistic Sources." in *IIWeb - VLDB*, 2004.

[9] S. Chaudhuri, G. Das, V. Hristidis, and G. Weikum, "Probabilistic Ranking of Database Query Results." in *VLDB*, 2004, pp. 888–899.

[10] S. Agrawal, S. Chaudhuri, G. Das, and A. Gionis, "Automated Ranking of Database Query Results." in *CIDR*, 2003.

[11] I. F. Ilyas, R. Shah, W. G. Aref, J. S. Vitter, and A. K. Elmagarmid, "Rank-aware Query Optimization." in *SIGMOD Conference*, 2004, pp. 203–214.

[12] S. Kambhampati, U. Nambiar, Z. Nie, and S. Vaddi, "Havasu: A Multi-Objective, Adaptive Query Processing Framework for Web Data Integration." Arizona State University, Tech. Rep., 2002.

[13] K. C.-C. Chang, B. He, and Z. Zhang, "Toward Large Scale Integration: Building a MetaQuerier over Databases on the Web." in *CIDR*, 2005, pp. 44–55.

[14] C. A. Knoblock, S. Minton, J. L. Ambite, N. Ashish, I. Muslea, A. Philpot, and S. Tejada, "The Ariadne Approach to Web-Based Information Integration." *Int. J. Cooperative Inf. Syst.*, vol. 10, no. 1-2, pp. 145–169, 2001.

[15] J. Hammer, H. Garcia-Molina, K. Ireland, Y. Papakonstantinou, J. D. Ullman, and J. Widom, "Integrating and Accessing Heterogeneous Information Sources in TSIMMIS." in *AAAI*, 1995, pp. 61–64.

[16] O. M. Duschka and M. R. Genesereth, "Infomaster: An Information Integration Tool." in *Intelligent Information Integration*, 1997.

[17] A. Y. Levy, "Information Manifold Approach to Data Integration." *IEEE Intelligent Systems*, pp. 1312–1316, 1998.

[18] A. Telang and S. Chakravarthy, "Information Integration across Heterogeneous Domains: Current Scenario and Challenges." University of Texas, Arlington, Tech. Rep., 2006.