DB&IR Integration: Report on the Dagstuhl Seminar "Ranked XML Querying" [1]

Sihem Amer-Yahia, Djoerd Hiemstra, Thomas Roelleke, Divesh Srivastava, Gerhard Weikum [2]

Contact Author: Gerhard Weikum, Max Planck Institute for Informatics,
D-66123 Saarbruecken, Germany, email: weikum@mpi-inf.mpg.de

This paper is based on a five-day workshop on „Ranked XML Querying" that took place in Schloss Dagstuhl in Germany in March 2008 and was attended by 27 people from three different research communities: ***database systems (DB), information retrieval (IR),*** and ***Web***. The seminar title was interpreted in an IR-style „andish" sense (it covered also subsets of {Ranking, XML, Querying}, with larger sets being favored) rather than the DB-style strictly conjunctive manner. So in essence, the seminar really addressed the ***integration of DB and IR*** technologies with Web 2.0 being an important target area.

1 Why DB&IR with Integration?

DB and IR have evolved as separate communities for historical reasons. They were spawned in the sixties with focus on very different application areas: accounting and reservation systems on the DB side, and library and patent information on the IR side. Consequently, they have emphasized different methodological paradigms: precise querying over schematized data, based on logic and algebra (DB), vs. keyword search and ranking over text and uncertain data, based on statistics and probability theory (IR). However, there are now many applications that require managing both structured and unstructured data and thus mandate serious consideration on how to integrate the DB and IR worlds at both foundational and software-system levels. These applications include Web and Web 2.0 use cases as well as more corporate-oriented scenarios such as customer support and health care. All three communities that participated in the seminar (DB, IR, Web) agreed on the importance of the general direction and came up with ten tenets, from different viewpoints, on why DB&IR integration is desirable.

DB1: Preference search over travel portals or product catalogs often poses a ***too-many-answers*** problem. Narrowing the query conditions can easily overshoot by producing too few or even no results; in general, interactive reformulation and browsing is time-consuming and may irritate customers. Large result sets inevitably require ranking, based on data and/or workload statistics as well as user profiles.

DB2: Adding ***text-matching*** functionality to DB systems often entails approximate matching (e.g., because of misspellings or spelling variants) and, when text fields refer to named entities, leads into ***record linkage*** for matching entities (e.g., to reconcile William J. Clinton and Bill Clinton or M-31 and the Andromeda nebula). Naturally, approximate matching by similarity measures requires ranking.

DB3: It has become the norm that applications access multiple databases, often with a run-time choice of the data sources. Even if each of these sources contains structured, exact data records and comes with an explicit schema, there is no unified global schema unless some magic could perform perfect on-the-fly data integration. So the application program has to cope with the ***heterogeneity*** of the underlying schema names, XML tags, or RDF properties, and queries need to be schema-agnostic or tolerant to ***schema relaxation***. In addition to this fact of life, many application builders (e.g., for e-science portals) do not want to start with a lengthy schema design process and rather want to be immediately productive by first entering data and only later adding and evolving metadata in a ***pay-as-you-go*** manner.

DB4: Textual information contains named entities and relationships between them in natural-language sentences. These can be made explicit by ***information-extraction***

[1] http://www.dagstuhl.de/en/program/calendar/semhp/?semnr=08111

[2] The seminar was organized by Sihem Amer-Yahia, Divesh Srivastava, and Gerhard Weikum; and this report was written by the three organizers together with Djoerd Hiemstra and Thomas Roelleke.
Additional participants were Peter Apers, Holger Bast, Mariano Consens, Emiran Curtmola, Debora Donato, Ingo Frommholz, Irini Fundulaki, Ihab Ilyas, Panos Ipeirotis, Benny Kimelfeld, Stefan Klinger, Amelie Marian, Maarten Marx, Yosi Mass, Sebastian Michel, Ralf Schenkel, Harald Schoening, Pierre Senellart, Kostas Stefanidis, Martin Theobald, David Toman, and Arjen de Vries.

Dagstuhl Seminar Proceedings 08111
Ranked XML Querying
http://drops.dagstuhl.de/opus/volltexte/2008/1535

techniques (pattern matching, statistical learning, NLP). This can potentially lead to large knowledge bases whose facts, however, exhibit some uncertainty. Querying the extracted facts thus entails the need for ranking. Moreover, it is often desirable that such information can be conveniently queried using keywords rather than sophisticated expressions in SQL or XQuery. With the extracted data organized in graph structures, this entails new research problems like determining when keyword occurrences are interconnected in a meaningful way and efficiently computing answers in ranked order,

IR5: ***Digital information*** really comprises both record-oriented and document-oriented data. The DB and IR fields have ***common roots*** even before the two areas became historically and somewhat artificially separated. In the fifties, Hans-Peter Luhn foresaw computer-based business intelligence and invented automatic indexing; this line of research led to text IR, but included what would now be seen as DB issues. It may be noteworthy that Luhn started his career with a punchcard-based algorithm for searching files of chemical compounds. Another anecdotal evidence for DB&IR commonalities is that both HTML/XML and thus the prevalent Web formats and the relational DB technology can be traced back to IBM Almaden, namely, to the seminal works of Charles Goldfarb and Ted Codd.

IR6: Information in digital libraries, enterprise intranets, e-science portals, and business-oriented Web sites is increasingly demanding ***structured IR*** that goes beyond keyword search by understanding attributes, XML tags, and metadata. The most successful approach along this line is the ***faceted IR*** paradigm that underlies most Internet merchant sites for product search, result refinement, and interactive exploration.

IR7: Search-result ***personalization***, adapting to the information-oriented tasks of the user, and proactive support for the user's information needs, are key directions towards better search precision/recall and user satisfaction. To this end, user preferences and profiling based on the user's long-term history of queries, clicks, and data usage, can be exploited, but also short-term behavior in the context of the current task needs to be considered. Such approaches are already pursued for Web, news, and blog search, and have enormous potential especially for individualizing and thus enhancing ***desktop search***.

IR8: Recognizing and tagging entities in text sources allows ***entity-search*** queries about electronics products, travel destinations, movie stars, etc., thus boosting the search capabilities on intranets, portals, news sites, and the business- and entertainment-oriented parts of the Web. Likewise, extracting binary relations between entities and also place and time attributes can pave the way towards ***semantic IR*** on digital libraries (e.g., PubMed), news, and blogs. Such capabilities are also a key asset for opinion mining and natural-language question answering.

Web9: As the surface Web is more and more dominated by portals, dynamic content loading (using Ajax and CMS's), data feeds, and mashups, understanding and querying the so-called ***Deep Web*** (aka. Hidden Web) of structured databases underneath the surface becomes an increasingly pressing issue.

Web10: Modern ***Web 2.0*** platforms for user-generated content and social networks have a mix of structured and unstructured data such as photos or videos with rich metadata, and an additional wealth of user-behavior and community information like tagging, rating, recommendations, friendships and other social relations, and so on.

Notwithstanding the general sense of agreement, the three communities also expressed major cultural and technical differences. For example, DB3, IR6, and Web10 all address the need for structure, whereas DB3 emphasizes relaxation of structure, IR6 emphasizes adding structure to information, and Web10 takes a mix of structured and unstructured data for granted. As another example, DB2, DB4, and IR8 address the need for named entities resulting from NLP techniques, whereas DB2 and DB4 emphasize approximate matching and ranking, and IR8 emphasizes adding relationships between entities. Generally, what this paper refers to as DB&IR integration would be naturally called IR&DB integration for the IR community, and the Web folks would not resist occasional remarks that the Web has come with its own software technology and has been very successful by ignoring both standard DB systems and traditional IR engines. These cultural and technical differences are partly reflected in the topics discussed below.

2 Hot Issues and Emerging Themes

2.1 XQuery Full-Text Scoring and Ranking

Both DB and IR participants agreed that XQuery Full-Text, XQFT for short, is troublesome (the Web people did not seem to care about it). XQFT is the designated W3C standard, currently in draft mode, for incorporating text-matching, scoring, and ranking functionality into the XQuery language. It offers great flexibility for applications to customize their own tokenization (e.g., word/phrase/name/sentence boundaries, stemming, etc.), thesauri, and scoring functions. However, this highly flexible programming comes with ***semantic pitfalls***, and there is hardly any guidance for application developers on making appropriate use of the various operators and score-aggregation options.

For example, what are the semantic differences between

searching for *„Billy AND the AND Kid“*, *„Billy OR the OR Kid“*, or *„(Billy AND the) OR (the AND Kid)“*, or the phrase (sequence) *„Billy the Kid“*, in the same XML element or spread across arbitrary elements? Is the phrase search guaranteed to return a subset of the conjunctive search? Is the ranked result list of the former a prefix of the latter's result? What if the conjunction is expressed at the XQuery level rather than in the text condition? For example, are the three conditions 1) *$a ftcontains "Billy" ftor "Kid"*, 2) *$a ftcontains ("Billy", "Kid")*, 3) *$a ftcontains "Billy" or $a ftcontains "Kid"* equivalent formulations, and if so, are they guaranteed to produce the same rankings and therefore the same top-10 results for any IR model instantiation? How does the tokenization plug-in affect the outcome? How do scores for primitive conditions propagate into scores for composite subqueries?

IR people felt that the scoring facilities for XQFT were mere ad-hoc and restricted, since the XQFT approach lacks the theoretical underpinnings of modern IR models like probabilistic IR or statistical language models. Such IR models have a range of desirable properties including sound reasoning about score composability. Also, the XQFT property that all scores – even for subquery results – must be between 0 and 1 seems very limiting and would rule out a straightforward implementation of some of the most successful IR scoring functions such as BM25 or log-likelihood ratios. DB people, on the other hand, felt that a clean algebraic approach would help for reasoning about equivalent query formulations (execution plans). When users formulate different queries that are not really equivalent in the underlying algebra, DB people would blame it on the user (i.e., programmer in the case of XQFT). IR people would be more concerned about users understanding the principles behind the scoring model. For example, how do global statistics about idf values or average document/element lengths affect the scoring? How can such aspects be incorporated into XQFT? Can application builders really cope with the flexibility of XQFT?

2.2 Search with Context

DB applications seem to be getting more and more user-oriented (bringing the field closer to IR where awareness of human-user aspects has a long tradition), as opposed to the classical, now perfectly mastered, business-platform applications. Examples are personalized Web exploration, desktop search and personal information management, and social networks. This trend raises the issue of how to take into account the ***context*** in which a user poses queries and explores information. The context includes environmental parameters like the location, time, device, and situation (e.g., business meeting vs. tourist tour) of the user, but should also consider inherent preferences and long-term behavior of the individual. For the latter, building and maintaining user profiles is a popular approach, based on statistics about prior queries, clicks, and others actions in the user's history. The profile may in turn be encoded in the form of constraints and rules that can drive query rewriting for simple relational queries or sophisticated XQuery programs. Of course, such approaches have a long tradition in IR, but relevance feedback, query expansion, user-specific result ranking, and other related techniques were mostly explored for keyword search; the structure of XML data adds opportunities as well as research challenges.

A particularly intriguing case for context-aware functionality, customized to an individual user, is ***desktop search***. Path labels of email folders and directory paths, along with attributes about dates, authors, and other context, and content keywords together provide powerful ways of searching and ranking. All this can be cast into XML-centric DB&IR methods; particular attention needs to be paid to approximate matchings of paths and other sub-structures as users often do not remember their directories that well. But the potential goes way beyond XML similarity search: unlike in a Web setting, the user's own desktop data (i.e., the file system on her PC or mobile device) can be analyzed in a much deeper way for more expressive and strongly individualized rewriting, expansion, and ranking strategies. Last but not least there are great opportunities for observing the user – on the client side without any privacy breaches – and customizing system actions to the current task of the user. For example, the last few emails read, the last few new items seen on a Web site, the last few MP3 songs listened to, or the last few incoming phone calls provide clues about the user's current information needs and can enable opportunities for task-oriented search and even proactive information supply.

2.3 Ranking over Uncertain Databases

The best years of exact data seem to be over. Most of the interesting applications face uncertain data for various reasons: 1) in sensor networks there are natural and unavoidable sources of measurement errors so that the data often needs to be interpreted in view of the error variance or with confidence intervals; 2) in Web 2.0 forums, the most valuable data is manifested in social recommendations and ratings; but this „wisdom-of-crowds“ data can only be interpreted in statistically aggregated form, with natural uncertainty; 3) information integration and pay-as-you-go data-acquisition applications are bound to end up with missing values, inconsistent values, and multiple alternatives for critical fields or entire records; consequently, querying the resulting data amounts to searching a potentially huge number of „possible worlds“; 4) as text continues to be prevalent in content production in news, blogs, and literature, information-

extraction methods are the way to lift text statements into value-added relational facts; however, this process is inherently error-prone so that confidence-aware search and support for exploring uncertain data are crucial.

In all these settings, the ***uncertainty*** that arises from different ***„possible worlds“*** strongly suggests ranking of query results. Thus, ***top-k queries*** are a dominant part of the workload, and this calls for efficient algorithms and a smart query optimizer. While top-k queries over „hard“ data, such as multimedia features, frequency values in traffic logs, or precomputed scores in IR-style inverted lists, have been intensively studied, there is little work on the situation when the uncertain data incur an additional dimension of combinatorial choices. Optimizing top-k queries over a „possible-worlds“ dataset or social-network-based ratings, where each user may be seen as a „possible world“, poses great challenges for the DB community. At the same time, the ranking should follow a principled model, for example, based on a generative stochastic process; this aspect is in the main expertise of the IR community. Needless to say that ranking semantics and computational complexity and thus efficient algorithms are intertwined and should, therefore, be best considered together. And to reassure the DB hardcore folks: yes, the ranking (ordering) of search results is an aspect of query semantics, although it may be based on a statistical model.

2.4 Light-weight DB&IR Engines

For several years, there have been strong advocates against the heavy footprint, overly broad functionality, claim of universality, and thus hardly manageable complexity of the traditional brand of commercial database engines. In view of this discussion, various light-weight engines for DB&IR were presented at the workshop: open-source systems for XML IR (TIJAH based on a column store (MonetDB) and TopX 2.0 based on a homegrown file manager) and also the very-light-weight CompleteSearch engine for faceted IR with extensions for DB operations. An interesting discussion emerged from these presentations as to whether DB or IR is the better starting point for such a light-weight DB&IR or IR&DB kernel.

2.5 Miscellaneous

Many other interesting topics were presented and discussed in the seminar. Some were highly creative in pursuing approaches off the beaten paths; some were provocative and controversial. As a small selection, three of them are pointed out here.

For ***opinion mining*** in product reviews (or in blogs), instead of attempting to analyze natural-language statements such as „incredible delivery time“ (most likely to denote slow delivery and thus a negative opinion), one can build a correlation model between text snippets and numerical attributes such as prices paid by the customers. This way, ***econometrics*** aids the otherwise very difficult task of opinion analysis.

A largely unexplored issue that was felt to develop increasing importance is search and mining of the ***history*** of Web, intranet, news, blogs, or social-tagging data. Digital heritage can be a gold mine for journalists, sociologists, market analysts, lawyers dealing with intellectual-property rights, and everybody interested in the evolution of cultural and sub-cultural zeitgeist. Many data sources have their built-in versioning (e.g., when using a Wiki for collaborative input). So the mechanics for indexing and query execution is present, but there are tremendous scalability challenges and a widely open question about ranking the results of ***time-travel*** queries.

Finally, a few participants advocated that text would be a more ***natural*** form of ***data representation*** compared to structured records (the DB hardcore participants took this heresy with serenity). It is easier to enter, easier to interpret by the user, and can go a very long way for advanced search capabilities. One participant, Holger Bast, even cited John 1:1: „In the beginning was the word“, and pointed out that there is no mention of „in the beginning was the table“ anywhere. The audience interpreted this as another pitch for the pay-as-you-go philosophy.

3 Speculative Directions

The workshop participants spent some time in breakout groups on three speculative topics. The following reflects the issues that were identified and discussed, without any claims about paths towards solutions.

3.1 XML vs. RDF

XML is prevalent in business applications, digital libraries, and as a ubiquitous exchange format. But RDF is gaining momentum as the Semantic Web starts taking shape and is fairly popular in the e-science community as an import/export format. Both XML and RDF have been conceived as representation models for semistructured data that does not necessarily have a rigorous schema and combines text with categorical and numerical values. The question that the breakout group addressed was whether there really is a need for both XML and RDF; perhaps one could subsume the other, or they could be reconciled into a single model, or another, possibly newly devised, model could take this unifying role.

The group first tried to characterize the two data models and put them in perspective with other models. XML was seen as ***tree-oriented***, with XLinks being added as an afterthought (with no XPath support for traversal along non-tree connections and across document boundaries). RDF is a ***graph model*** with triples representing

relationships that form the graph's edges. It may come with a schema, but schema-less RDF is popular. Entire triples can in turn participate in relationships; so RDF is essentially an extended entity-relationship model. One thought that came up towards bridging the two models was to cast the RDF graph into a (small) number of (different) spanning trees (possibly with a few additional non-tree edges) that together cover the graph. This is reminiscent of the „colorful XML" proposal in the literature. Of course, connections to object-relational models were obvious, too. But the goal of the group was not to head for the „best" data model per se; rather the discussion focused on the context of search and ranking.

The discussion led to the general insight that XML and RDF have different strengths and weaknesses regarding their ***ease of use*** in different application classes and for different tasks. The discussion of applications ranged from collaborative Wikipedia authoring and social-tagging forums, on the less structured side, to e-science data and content management systems for museums, on the more structured side. The tasks under consideration included data entering, query formulation, search, ranking, result presentation, and the long-term evolution of data and applications. It was recognized that trees (XML) allow easier input, easier query formulation, and simpler ways of presenting the results because answer units are always subtrees (although this is not true for XQuery). Graphs (RDF), on the other hand, seem to be more flexible for the long-term evolution of the data; the data-first-schema-later paradigm of dataspaces fits nicely with this representation, whereas XML trees may have to undergo more disruptive restructuring in the lifecycle of a dataspace. As for ranking, generative probabilistic models in the IR tradition are easily applied to either of the two; extensions for both trees and graphs have been proposed in the literature.

3.2 Ranking vs. Scoring

The traditional model for ranking search results is by first associating a numerical score with each result item independently and then ordering the items by descending scores. The scores are usually based on various forms of probabilistic/statistical models, and can be largely precomputed so that run-time scoring boils down to aggregating the score values. The question addressed in this breakout group was whether one really needs both ranking and scoring, and whether numerical scores are the only or most appropriate way of computing rankings. From an end-user perspective, scores are hardly interpretable and only ranking matters, that is, the relative ordering of search results. However, the relative distances between ranked items is of potential interest, since a small distance means closeness in relevance whereas a large distance clearly differentiates retrieved items. When search is a service with an API for application programs, the programs may need access to scores, for example, to analyze the skew of score distributions in the top-1000 results for finding suitable cut-off points, or to cluster and visualize results. So both ranking and scoring seem to be indispensable, for different purposes.

An interesting thought that came up in the discussion was to question the traditional wisdom of score aggregation, and rather aim for a notion of ***holistic ranking***. Considering global statistics (idf values, length distributions, background models for smoothing, etc.) for the final ranking may be viewed as a first step in this direction. Likewise, there is a recent trend of automatically learning (coefficients of) a high-dimensional scoring function by regression, using merely ordinal information as input, namely, pairwise preferences of users about query results. This input can be easily obtained from click logs and other user-behavior data.

But holistic ranking may go much further. It is often desirable that the top-10 ranks of the search result exhibit a natural ***diversity***. This may be needed for minimizing the risk of misinterpreting the user's actual information demand, for example, when the query input is ambiguous (e.g., searching for „Ajax" may give information on Web programming, the Iliad, the soccer club Ajax Amsterdam, etc.) or the best results would depend on the user's prior expertise and knowledge which can only be guessed by the system (e.g., searching for „online community" should give different results to accomplished sociologists vs. laymen or children). Another reason for diversity that the group speculated about is „user happiness": when users see ten excellent web pages in the top-10 results, they may feel uneasy, whereas showing a few excellent ones mixed with a few mediocre ones may be psychologically better (even if many excellent results are available).

Holistic ranking could entail interesting technical questions. For example, is it really an axiom that the items in a top-k result should be a prefix of the items in a top-(k+1) ranking? Maybe, it is not even necessary that the top-k result is a subset of the top-(k+1) items. Diversity and other considerations may well justify such deviation from established practice.

3.3 Social Wisdom

Web 2.0 comes with a wealth of „social wisdom" that can be harnessed for improving search results and the entire search experience. In fact, the standards of user expectations are rising. Search engines should consider context and background information about the user and the user's current task, and they should know about the user's social relations, her friends' preferences and recent cyberspace activities. For example, queries about recent movies should not simply return Web pages, but should

place the user right into the experience of social-network ratings and discussions on the movie. Similarly, a query about a particular digital-camera model should know/predict whether the user has the intention of buying a new camera – and then point right into the best consumer reviews and merchant sites –, or whether the user has recently purchased this model – and then point into troubleshooting or photo competition forums. Beyond user-initiated search, such a richer kind of ***experience engine*** should perform many tasks in a proactive information-supply manner. Obviously, search-related advertisements can be seen as an aspect of this ambitious approach, but the experience engine should go much further and leverage the social community aspects of Web 2.0.

4 Conclusion

All three of the participating communities – DB, IR, and Web – felt that looking across the fence paid off very well, and that the communities should continue learning from each other. Challenges are ahead in areas like Web 2.0, personal information management, and entity-relationship search; these will remain difficult and rewarding areas for a while. Combining the different and quite complementary expertises from DB and IR would be vital towards well-founded and practically viable solutions.