CSE 705: Seminar Report on
Learning to Create Data-Integration Queries

Noel Gunasekar (nvg2@buffalo.edu)

Overview:

This paper which was presented in VLDB 2008 conference, discusses about a system that enables non-expert users to pose queries across multiple data sources. In the field of Life Sciences there are hundreds of large inter-linked data resources on fields like proteomics, genomics, disease studies and pharmacology. A typical information need would look like - "What are the proteins and genes associated with the disease Narcolepsy?" This involves querying of three different fields with each field containing multiple databases. Example information on proteins alone could be in databases like UniProt, PIR, PRF, and Protein Database [1]. At present, a life sciences researcher, who is not an expert in querying, relies on web forms and associated query templates to meet the information need. Unfortunately, due to varied information needs of the user, one standard web-form doesn't suffice the purpose.

In this paper the authors have presented a Q-System that allows non-expert user to author new query templates and Web-forms, to be reused by anyone with related information needs. Q-System works in two phases - in the first phase the author poses keyword queries that are matched against tables and attributes and uses the associations like foreign keys to create multiple ranked queries. These set of queries are then attached to a web-form. This web-form is then used for querying by the author and associated users. Further, the query answers that are obtained from the web-form are rated, providing feedback to the Q-system to alter the queries that are being attached to that web-form. In the second phase - after creating such a reusable web-form, which caters particular information need, user pose future queries through it.

Q-systems architecture, which is discussed in-detail in the paper, basically consists of four components – Initial Schema Loader, Query Template Creation, Query Execution and Learning through Feedback. Before the user actually poses keyword queries, the Q-system is loaded with data sources, their schema and associations. The Schema Loader encodes all these information as a schema graph, nodes representing relations and edges representing the associations. Edges may have a cost that captures its likely utility to the user: this may be based on reliability, trustworthiness, etc., and the system will attempt to learn that cost based on user feedback.

The Query Template creation phase constitutes of creating top k queries matching user given keyword and attaching it to a Web-Form. The user defining a query poses keyword query like "protein, plasma membrane, gene, disease" which is matched against the schema graph by the Steiner Tree Generator. Steiner tree is a tree of minimal cost in a graph which includes all the required nodes(here the query keywords).The Tree Generator gives top k Steiner Trees ranked by the cost, which is computed by adding the cost of all edges of the tree. Tree Generation is accomplished by formulating the problem as a multi-commodity flow problem and solving using Mixed Integer Programming. This works fine for smaller schema graphs but for larger graphs before applying the algorithm the graph is pruned by Shortest Paths Complete Subgraph Heuristics.

Each of these Steiner trees is then converted to equivalent query by using node information and the edge associations by the Query Formulator. Finally, with user refinement a web-form is attached to these queries. Then author and the associated users query by providing parameters in the web-form to get the query result. Here to execute a given query using the provided parameters we require a query execution engine that is capable of querying remote sources and also record data provenance. ORCHESTRA is used for this purpose.

Once the query is executed the query answers are verified and users mark the tuple either as relevant or irrelevant, thereby providing feedback to the system. With the data provenance these feedback are traced to the source query and their respective edges cost are updated to reflect the user's preference. Q-System uses an online learning algorithm, i.e., an algorithm that updates its weights after receiving each training example. The algorithm used here is based on Margin Infused Ranking Algorithm (MIRA).

The paper concludes with experiments mainly evaluating the accuracy of the learning algorithm, the response time for the feedback and query response, and comparison of performance of different Steiner Tree generation algorithms. Overall the learning algorithm was able to work fine after 40-60% of training in a sample set of 25 queries. It takes less than two seconds to generate top 5 queries, which makes the system highly interactive. Also it was found that it is possible to generate Steiner Trees for larger graphs with no loss using the approximation algorithm.

Detail Comments:

Strengths: The key strength of the system is the ability to eliminate the need to understand the full complexity of the underlying schemas, source relationships, or query languages, while posing queries. Also the system is built from the input provided by the user thereby tailored to the user's context and the information need. The system's accuracy in terms of learning is impressive.

Weakness: System works only at the meta-data level i.e., the web-form is formed based on input keywords that are matched against relation names and attributes. There might be requirement to match at the data-level, but in that case the schema graph might be really large and the scalability and performance of the algorithm is a big question. The experiments were conducted in a more controlled scenario, there should have been some experimental results showing the user-satisfaction in a more realistic scenario were the users feedback would be more varying.

Technical Depth: Paper is really technically sound as it covers in-depth the algorithms and methodologies that were used to address various problems that were faced. The major problems that were discussed are the generation of top-K queries, incorporating the user feedback and the data provenance.
Paper discusses Steiner Trees generation, how it is formulated as a multi-commodity flow problem and solved using mixed integer programming. Edge cost updating methodology and algorithm used for online learning is discussed in detail. Also details on data provenance are provided.

Comparison with Related Work: Though this is similar to information retrieval, here the system generates ranked queries for keywords rather than ranked results. This keyword based querying approach though it is similar to Natural language interface; this work differs because the data sources are so large and diverse. At high level the system is like a keyword based querying system on a database, but the key differentiation is that the system learns based on the user feedback. Schema matching tools are also complementary as the schema information is loaded along with the cost and modified based user feedback.

Discussions: Some of the discussions were related to the justification of using feature and weight, as the features demonstrated were only of binary values. There were also comments made on the ranking model as the system tends to prefer Steiner trees with lesser nodes as the cost is less, which may cause undesired side-effects.

Additional References:
Protein Databases: http://www.bioscience.org/urllists/protdb.htm