Ariel Fuxman
Statement of Research Interests

When we integrate data from multiple sources, the resulting database may contain errors and inconsistencies. Ideally, all errors should be cleaned automatically. In reality, though, data cleaning necessarily requires human intervention. To facilitate this, I envision a data integration process where it is possible to obtain meaningful and consistent answers from an integrated database even if it is partially dirty.

The challenge of query answering over dirty databases is that we cannot naively reuse existing database technology, which is designed to work on clean databases. For this reason, I designed, implemented and evaluated *ConQuer* [SIGMOD05,VLDB05], a scalable system for query answering over dirty databases.[1] This system helps users take advantage of the query results in order to interactively clean the integrated database.

While it is well known how to answer queries over clean databases, it is necessary to give meaning to the notion of *clean answers* over dirty databases. In my thesis, I explore two different semantics for clean answers. One [ICDT05] is based upon certain answers, a concept which is prevalent in the data integration theory literature; the other [ICDE06], which is drawn from the area of probabilistic databases, assigns each tuple a probability of being clean. The key contribution of my work is to bridge the gap between theory and practice by providing an *efficient and scalable system* to obtain clean query answers from dirty databases.

In order to compute "clean" answers efficiently, *ConQuer* builds upon existing database management technology. In particular, it implements a *query rewriting* approach. That is, given a query q, *ConQuer* rewrites q into another query Q^* such that Q^* always retrieves the "clean" answers for q on every dirty database. The rewritten query is a SQL query that can be executed using any commercial Database Management System (DBMS). *ConQuer* also provides optimization techniques that would never be found by standard query optimizers which are in general unaware of data conflicts. In an extensive set of experiments, I showed that the rewritten queries have little overhead when compared to the original (non-rewritten) ones.

ConQuer provides an interface that enables the user to gradually clean the database. In particular, when a query is submitted, the system shows the clean answers together with a query explanation. The explanation can be extremely valuable, since it often points to underlying errors in the database that require attention from the user.

In my thesis work, I considered integrated databases that may violate a set of primary key constraints. Such databases are present in most organizations. For example, in the domain of Customer Relationship Management (CRM), data sources often contain conflicting information about the same customer. Notably, commercial CRM tools provide limited support for merging tuples corresponding to the same customer into one tuple in the integrated database. Although they typically support some form of conflict resolution rules (e.g., rules that take the average between two conflicting incomes of the same customer), these rules may be difficult to design.

[1] *ConQuer*'s web page can be found at `www.cs.toronto.edu/db/conquer`

In the absence of conflict resolution rules, some CRM tools transfer all conflicting tuples to the integrated database. Thus, even if the sources satisfy the key constraints, the integrated database may not.

I am currently extending *ConQuer* to support dirty databases that may violate foreign key constraints. Together with the support for primary keys, this would cover all integrity constraints commonly used in database systems. I will also consider other constraints, such as constraints arising from business rules (e.g., a rule saying that a car insurance policy cannot be held by people who are younger than 18 years old). Regarding the data model, *ConQuer* currently works on relational databases. In the near future, I would like to extend support to semi-structured data, such as XML documents.

Besides query answering, data integration is concerned with transforming source data to fit the schema of the integrated database. In the last decade, the research community has focused mostly on *schema mappings*, which are declarative specifications of such transformations. I myself have contributed to this effort, through my work on peer data exchange [PODS05]. There is an alternative approach for specifying data transformation, which is more operational than schema mappings: Extract-Transform-Load (ETL) scripts. Although ETL tools clearly dominate the market, they have not yet received much attention in the research community. There are interesting questions as to how ETL can be used in an interactive data integration process. For example, I would like to explore how to automatically build ETL scripts that could be used by the user in response to query explanations such as the ones produced by *ConQuer*.

My thesis work is just a starting point towards a more interactive data integration process. There are many exciting directions that I am looking forward to pursue. One such direction has to do with *exploiting lineage information*. *ConQuer* currently deals with errors and inconsistencies that occur after the source data has been transformed to conform to the schema of the integrated database. However, we sometimes need to trace the origins of data (aka lineage). For instance, in the probabilistic query answering semantics, probabilities may be related to the quality of the sources: the higher the quality of a source, the greater probability of being correct that its tuples should have. As another example, *ConQuer* gives query explanations using tuples from the integrated database. However, users are sometimes more familiar with the source schemas, and would prefer an explanation in terms of tuples from the data sources.

Another promising direction involves *learning models of user's preferences*. *ConQuer* can query a database even if it has little or no knowledge about how to clean it. But as the user interacts with the system (e.g., by reacting to query explanations), it may be possible to *learn* some of the user preferences on how to resolve data conflicts. In turn, these preferences could be made part of the semantic model for clean query answering in order to improve the quality of future query results. For example, suppose that a user tends to prefer the tuples coming from one particular source whenever there is a conflict. Then, in the probabilistic query answering semantics, we could increase the probabilities of the tuples coming from this source. The problem becomes even more interesting when we consider a multi-user environment. In this case, the system could create a profile of each user's preferences, and personalize the query results according to the user who issued the query.

In summary, I plan to contribute to the development of a new generation of tools that make the process of searching and organizing data more flexible, interactive and tailored to the user's preferences. I believe that a research lab is an ideal place to build these tools and ensure that they have a broad impact in the marketplace.

Bibliography

[ICDE06] P. Andritsos, A. Fuxman, R. J. Miller. Clean Answers over Dirty Databases: A Probabilistic Approach. To appear in *International Conference on Data Engineering (ICDE)*, 2006.

[SIGMOD05] A. Fuxman, E. Fazli, R. J. Miller. ConQuer: Efficient Management of Inconsistent Databases. In *ACM SIGMOD Conference*, 155-166, 2005.

[VLDB05] A. Fuxman, D. Fuxman, R. J. Miller. ConQuer: A System for Efficient Querying Over Inconsistent Databases. In *International Conference on Very Large Data Bases (VLDB)*, 1354-1357, 2005.
System demonstration.

[PODS05] A. Fuxman, P. Kolaitis, R. J. Miller, W. Tan. Peer Data Exchange. In *ACM Symposium on Principles of Database Systems (PODS)*, 160-171, 2005.
Invited for submission to a special issue of *Transactions of Database Systems (TODS)* with the best papers from PODS 2005.

[ICDT05] A. Fuxman, R. J. Miller. First-Order Query Rewriting for Inconsistent Databases. In *International Conference of Database Theory (ICDT)*, 337-351, 2005.
Invited for submission to a special issue of *Journal of Computer and Systems Sciences (JCSS)*.