Databases for Semistructured Data: How Useful Are They? (position paper)

Len Seligman, Ken Smith, Inderjeet Mani, Barbara Gates
The MITRE Corporation
McLean, VA 22102
{seligman, kps, imani, blgates}@mitre.org

Abstract

There is much activity in the database research community on managing semistructured data but little experience to date in applying this research to substantial problems. This paper describes an effort to assess the applicability of this technology to organizations with large quantities of semistructured (and structured) information. We consider the issue of appropriate data models for semistructured data and then describe an evaluation framework currently under construction.

1 Introduction

Semistructured data has some structure but is difficult to describe with an explicitly specified, rigid schema. Reasons for this difficulty include irregularity or rapid evolution of the data, or structure that is implicit or not known to the user. Researchers are developing data managers for semistructured data in order: (a) to provide the advantages of database managers (e.g., sophisticated query, support for multiple views, constraint management, etc.) for data that lacks a schema and (b) to facilitate integration of sources containing semistructured data with other sources, including structured ones. Several projects are investigating the use of graph-structured data models (GSDM) that represent all data as labeled directed graphs (e.g., [PGMW95]). Query languages have been developed for GSDMs (e.g., Lorel [AQMW97] and UnQL [BDS95]). Promising prototypes have been developed, but there has been little experience using "semistructured databases" for substantial applications. Assessment using realistic data is a logical next step for semistructured database research.

In an effort to provide such an assessment, MITRE has initiated an applied research effort in Semistructured Data Management. Its goal is to assess the usefulness of semistructured databases for managing and integrating realistic sized heterogeneous, semistructured information sources. We have begun to develop an evaluation framework, which is described briefly below. We are experimenting with Stanford's LORE graph-structured DBMS and a few semistructured intelligence community data sources. In addition, we plan to use LORE to perform an integration effort between two intelligence databases, one semistructured and one relational. We are constructing wrappers to import data with implicit structure into semistructured databases. In addition, we will compare the costs/benefits of wrapping data so that it can be imported into semistructured databases with the costs/benefits of importing it into commercial databases with extensions for handling specific data types (e.g., Oracle and its ConText text management tool).

2 Picking the Right Data Model

An essential part of identifying the right data manager for a particular application is to assess the appropriateness of the underlying data model. A data model includes both a set of operations used to manipulate data (a subset of which is exposed in the API) and a set of rules according to which data can be constructed. There is a great diversity of data models, including not only those implemented by traditional DBMSs, but also the WWW, information retrieval/document management tools, GSDMs, etc.

The choice of a data model should be motivated by these main factors:

1. What *query functionality* is required (i.e., what operations and predicates)?
2. What *other database operations* must be supported beyond query? Examples include update, support for multiple views of the data,

Proceedings of the 5th KRDB Workshop
Seattle, WA, 31-May-1998
(A. Borgida, V. Chaudhri, M. Staudt, eds.)
http://sunsite.informatik.rwth-aachen.de/Pulications/CEUR-WS/Vol-10/

transaction management, and constraint enforcement. Also, what are the specific requirements for supporting those operations? For example, if constraint enforcement is required, what kinds of predicates should be supported and over what data organization?

3. What *representational constructs* are required? Answers to questions (1) and (2) are significant factors in determining which formal representations would provide the greatest benefit.
4. What are the *costs and benefits* of producing (and maintaining) different representations of the data? Often, user communities need to access heterogeneous sources (often with ad hoc representations) over which they may have little control. One must assess the cost of producing desired representations, given the current state of the data and available technologies for (semi) automatically extracting the required structure. Engineering trade-offs must be made, depending upon the costs and benefits.

Examples of required query operations include:

- Relational algebra (e.g., select, project, join, set operations, etc.)
- Text-based information retrieval (IR) queries, including boolean, weighted vectors, operator-free natural language or “document similarity” queries, proximity queries (e.g., “within the same sentence as”)
- Geographic queries (e.g., “in the Potomac River floodplain”)
- Image queries (e.g., “find images most like this one, based on edge shape, texture, color, etc.)
- Graph traversal. For example (using Lorel), “Select LogisticsDB.Equipment.Description where LogisticsDB.Equipment.#.LocatedAt.Name = "Ramstein";” (i.e., “List the description for all Equipment objects in LogisticsDB that, after following zero or more links, one can follow a LocatedAt arc to an object that has a name of ‘Ramstein’.”).
- Arbitrary functions (e.g., “Select * from Emp where TopPerformer(Emp)”, assuming Emp has a boolean method TopPerformer).
- Hybrid queries (e.g., “Get information about Doctors who practice in hospitals within 5 miles of Times Square and who wrote treatment notes last month about emphysema”, which includes relational operations, text and geographic predicates).

In addition to query, one must consider other required operations, such as view support, constraint management, and transaction management. The ability to construct and maintain multiple views of data with a minimum of administrative effort is one of the most significant achievements of database technology. Ad hoc representations (e.g., unrestricted free-text) are ill-suited to view support, while HTML only lends itself to simple selection views, using syntactic tags. Research is just beginning on views for graph-structured databases [ZGM98].

Based on the required operations, one can assess how well different data models support those operations. If D_m is the original form of the data (represented in data model m) and D_n is the same data transformed into data model n, we can characterize the transformation as $D_n = \text{wrap}_n(D_m)$, where wrap_n is a function (which may include manual processing) that constructs a wrapper around data such that the operations of data model n can be used.

3 Preliminary Evaluation Framework

Before deciding what data model to use, one should perform an analysis of the benefits and costs of producing (and maintaining) different representations of the data. The benefit of a transformation from data model m to model n can be characterized as

$$\text{Benefit} = (\text{Value } [t_0 \rightarrow t_1](\text{Operations}_n)) - (\text{Value } [t_0 \rightarrow t_1](\text{Operations}_m))$$

This represents the added value to users of being able to manipulate D_n over the life cycle of the system (i.e., time interval $[t_0 \rightarrow t_1]$) using the operations supported by model n as compared to manipulating D_m using the operations supported model m. The cost can be characterized as

$$\text{Cost} = \text{InitialCost}(\text{Wrap}_n(D_m)) + \text{MaintenanceCost } [t_0 \rightarrow t_1](\text{Wrap}_n(D_m))$$

i.e., the initial cost of building a wrapper plus the cost of maintaining it over the system life-cycle. Costs are determined both by the current state of the data and by the available technologies for extracting the required structure. Both initial and maintenance costs can be substantial, especially when manual effort is required to do the transformation.

Admittedly, it will often be impossible to develop meaningful quantitative measures of the sort described above, particularly for Value. However, we believe qualitative assessments of these trade-offs can and should be done.

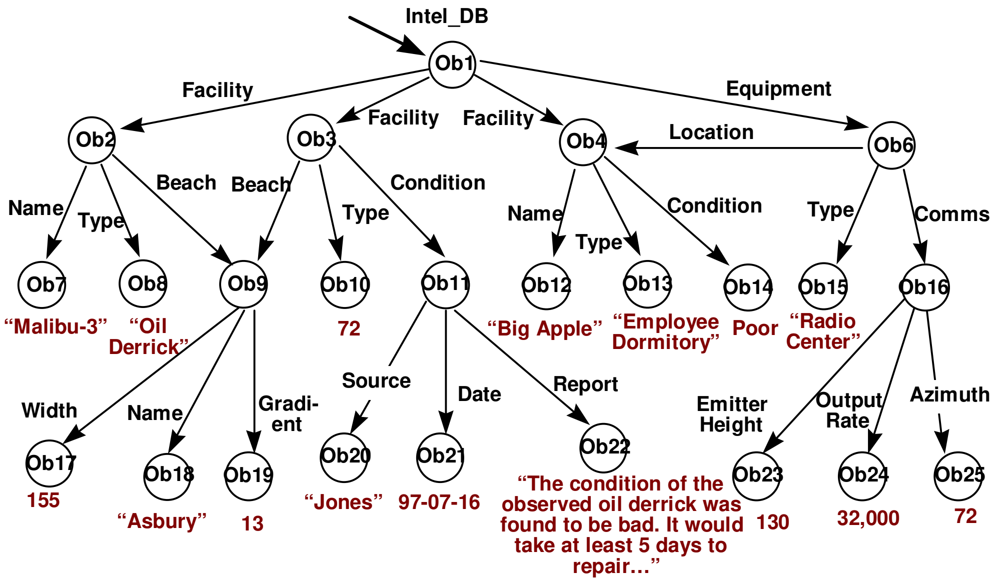

Figure 1. Fictitious Intelligence Database in LORE

As an example, consider the CIA World Factbook (CWF),[1] a web site that provides information on the economy, geography, defense, etc. of all countries. While there are clearly marked "fields" that appear in most countries' pages, the structure of these documents is implicit and therefore supports only simple text search. We have developed a simple wrapper for CWF data which allows importation into LORE so that relational as well as graph traversal queries can be asked over this data. Because the structure (although implicit) is fairly regular, the cost of developing wrappers is small. In addition, because document structure evolves slowly, wrapper maintenance costs are low. The benefit of bringing the data into LORE (e.g., more powerful query processing) should more than outweigh the low wrapper costs. Of course, given the regularity and slow evolution of the CWF, it is reasonable to ask if the benefit/cost ratio might be even better for bringing the information into a relational database. We are still too early in our experimentation to be able to answer this.

Figure 1 shows a fictitious, LORE intelligence database which better highlights the strengths of semistructured databases. LORE represents all information as a directed labeled graph. From the root (Ob1), there are arcs pointing to Facilities and Equipment. This example shows two kinds of irregularity, neither of which causes problems for LORE. First, there are mismatched types (e.g., Intel_DB.Facility.Condition is in one case a short string (Ob14), while in another it is a complex object (Ob11) that includes a text document (Ob22)). Second, missing attributes are handled gracefully without the use of Null values (e.g., not all facilities are named). An additional feature is the ability to support unconstrained annotation. For example, suppose that no database designer knew a priori of a relationship between Equipment and Facility entities. Despite this, a user can represent the fact that Equipment Ob6 is located at Facility Ob4, simply by creating a new arc and giving it an appropriate label (e.g., Location).[2]

Let us consider this example in the context of the benefit/cost ratios described above. The ability to cope with irregularity can result in lower costs for developing and maintaining wrappers; one can still provide database functionality, without having to develop wrappers to transform all information to a common representation. Added value is provided by the ability to combine the relational algebra with graph traversal and regular expression operations. In addition, support for unconstrained annotation may be of great benefit to certain user communities, especially those (like intelligence analysts) who are in the business of hypothesizing and exploring (often unanticipated) relationships. Unconstrained annotation is also especially useful in the early stages of design (e.g. of a mechanical artifact or chemical process), when structural information is still tentative and somewhat fluid.

As a counter-example, consider a university department's online library of technical reports. Reports do not change, but several different formats

[1] http://www.odci.gov/cia/publications/nsolo/wfb-all.htm

[2] Of course, this flexibility may come at the expense of diminished data quality. Constraint management in semistructured databases is an important topic for future research.

exist, and more formats may be added. An existing web interface supports searches using boolean queries over author, title, and abstract fields as well over the full text. Once documents are located, they can be viewed or downloaded as postscript or PDF.

This library could be migrated to a semistructured database, exposing the internal structure of documents including sections, subsections, paragraphs, sentences, equations, and citations. Many new types of operations could be supported using this structure, for example: "Display all paragraphs containing the word 'semistructured'"; "Display the abstracts of all technical reports cited in TR-101"; or "List the author names of all documents that contain the words 'semistructured' in a section whose title contains 'future' and 'research'." However, these new operations do not appear to give users much additional power in locating relevant reports compared to the existing web interface. In addition, unconstrained annotation is of no value here, since the data is read-only. On the other hand, the initial and maintenance costs for wrappers (e.g., developing parsers to extract the structure despite format heterogeneity) are nontrivial and are likely to outweigh the benefits of the new query capabilities offered by the semistructured database.

4 Conclusions

To revisit our original question: are semistructured databases useful? We believe there is no single answer. This paper has presented a simple cost/benefit model based on several important variables and illustrated its use. As we gain experience, we will refine this model. In addition, we expect to gain a much better understanding of (1) when semistructured databases are appropriate and (2) the utility of different approaches to creating wrappers.

In closing, we note that the cost of wrapper generation often depends upon the difficulty of making implicit structure explicit, so that the data can be represented using some formal representation. At MITRE's Artificial Intelligence Center, we are applying information extraction technology to extract structure from text news sources. Examples include extracting information about entities such as people, organizations, and places, and characterizing texts in terms of key phrases and summaries, using machine learning and language processing techniques [Mani97, Clif97, MMM97]. We have also been applying knowledge discovery methods to merge information from structured and unstructured sources on air traffic accident reports. We have begun to collaborate among research efforts in information extraction and semistructured data management and expect to see interesting interdisciplinary results.

Acknowledgments

The authors thank the anonymous reviewers for their helpful comments. In addition, the authors would like to thank the following individuals at MITRE for their support of this work: Reza Eftekari, Steve Huffman, Paul Lehner, Howard Carpenter, and Dave Lehman. Finally, we thank Jason McHugh, Roy Goldman, and Jennifer Widom of Stanford for their help with LORE.

Bibliography

[AQMW97] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. Wiener. The Lorel Query Language for Semistructured Data. *Int. Journal on Digital Libraries*, 1(1):68-88, April 1997.

[BDS95] P. Buneman, S. Davidson, D. Suciu, "Programming Constructs for Unstructured Data," *Proc. of Int. Workshop on Database Programming Languages*, 1995.

[Clif97] Clifton, C., Rosenthal, A., and Ullman, J.D., "Knowledge Discovery in Text", *First Federal Data Mining Symposium*, AFCEA, Washington, DC, December 16-17, 1997.

[Mani97] Mani, I., and Bloedorn, E., "Summarizing Similarities and Differences among Related Documents", *Proceedings AAAI '97*, pp. 622-228, AAAI Press, Menlo Park, CA, 1997.

[MMM97] Maybury, M., Merlino, A., and Morey, D., "Broadcast News Navigation using Story Segments", *ACM International Multimedia Conference*, Seattle, WA, November 1997.

[PGMW95] Y. Papakonstantinou, H. Garcia-Molina, J. Widom, "Object Exchange Across Heterogeneous Information Sources," *International Conference on Data Engineering*, 1995.

[ZGM98] Y. Zhuge and H. Garcia-Molina, "Graph Structured Views and their Incremental Maintenance," *International Conference on Data Engineering*, 1998.