

Data Integration: A Status Report

Alon Y. Halevy

University of Washington, Box 352350, Seattle WA 98195-2350, USA,
alon@cs.washington.edu

1 Introduction

Integration of data from multiple sources is one of the longest standing problems facing the database research community. In addition to being a problem in most large enterprises, research on this topic has been fueled by the promise of integrating data on the WWW. In the past few years, the community has made very significant progress on data integration, from the conceptual and algorithmic aspects, to the systems and commercial aspects. Below I briefly explain which areas we've made progress on, and what challenges lie ahead. This short paper is not meant to be a comprehensive survey of the state of the art in data integration, and represents only my personal perspective. I apologize in advance for the skewed list of references.

2 Recent Progress

Since the mid-90's, the database and AI communities have made significant progress in several aspects of data integration.

Schema mediation languages: a data integration system usually employs a *mediated schema*, which is purely a logical schema for the purpose of posing queries. In order for several autonomous data sources to interoperate, we need semantic mappings between the schemas of the different sources. The semantic mappings provide translations between the schemas of the sources and the mediated schema. Research on data integration has provided a set of rich and well understood schema mediation languages. The two commonly used formalisms are the *global-as-view* (GAV) approach used by [18, 22, 2], in which the mediated schema is defined as a set of views over the data sources; and the *local-as-view* (LAV) approach of [32, 12, 34], in which the contents of data sources are described as views over the mediated schema. The GLAV formalism [16] combines LAV and GAV. The semantics of the formalisms are defined in terms of *certain answers* to a query [1]. Surveys of these languages and their properties are given in [31, 24].

Query answering algorithms: associated with the mediation languages, significant progress was made on developing algorithms for answering queries in a data integration system. The specific problem of interest is to reformulate a

query posed over the mediated schema into a query that refers to the schemas of the data sources. In the case of the GAV formalism, answering queries reduces to view unfolding. In the LAV approach, answering queries translates to the problem of *answering queries using views* [24], for which several very efficient algorithms have been developed [38, 19].

Query optimization: the autonomy and heterogeneity of data sources present many new challenges to query optimization. In addition to the usual issues that arise in distributed query processing [37], in the data integration context data sources vary in their capabilities: some are simple web pages (or web-form interfaces to databases) and others are full-fledged relational query engines [22, 43, 15]. Handling *binding-patterns limitations*, i.e., the need to provide certain inputs to a data source in order to obtain answers has also received significant attention [40, 32, 14, 17, 11].

Query execution: the main line of work in the area of query execution for data integration systems was based on the observation that without statistics about the data sources, standard query optimization methods (i.e., cost-based optimization) do not apply. In answer to this challenge, several works considered *adaptive query processing*, where the system starts with some plan and adapts it as the execution proceeds [26, 28, 27, 3, 41, 6, 29, 42, 20]. The innovations in that work included the use of query operators that adapt as the data streams from the network, and different architectures for modifying query plans during execution.

Industry development: even three years ago, the term *data integration* in the commercial world referred explicitly to data warehousing. Practitioners and analysts were unwilling to believe any alternative architecture to data integration, mostly for two reasons: (1) lack of scalability and (2) inability to respect the autonomy of data sources. At the time, data warehouse products were already very well developed and accepted, and with the promise of *real-time* data warehousing, the need for other data integration architectures seemed diminished.

Today there is a data integration market. There is plenty of analyst coverage of different data integration products and architectures (often referred to as EII - Enterprise Information Integration). Companies such as IBM and BEA have announced initiatives leading to products in this realm, and a handful of startup companies such as Nimble Technology [10, 35] and Enosys [13] have delivered data integration products to customers, based on virtual integration architectures. One of the main issues facing these companies is the interaction between EII products and EAI (Enterprise Application Integration) products. It is reasonable to expect that in the near future we will start seeing integrated products from these two categories.

In summary, through years of research, development of prototypes and experimentation, we now know how to build the basic blocks of a data integration system. This is not to claim that the problems mentioned above are all solved, but a reasonable body of knowledge already exists, and now commercial efforts are starting to benefit from this body of knowledge.

3 Current Challenges

Despite the immense progress, several very significant challenges remain. I discuss a few of these below.

3.1 Generating Semantic Mappings

One of the major bottlenecks in building data integration applications (or any other application that requires multiple data sources to interoperate) is generating the semantic mappings between the data sources and the mediated schema. Currently, semantic mappings are written manually in a labor intensive and error-prone process. Tools that aid this process are crucial to the scalability of data sharing systems.

Unfortunately, complete automation of the generation of semantic mappings is unlikely to be possible. The crux of the problem is that writing a correct mapping requires an understanding of the underlying semantics of the schemas being matched. While the schema and data instances provide clues on their intended semantics, they do not fully capture the full semantics.

The problem of semi-automatically generating semantic mappings is an age-long problem in the database and AI communities (see [39] for a survey), and has recently received renewed attention [33, 8, 36, 9, 7]. Two principles are guiding the current work on this topic. First, a matching system needs to employ a collection of tools, each of which uses certain clues from the input to propose mappings. Second, a matching system should improve over time – every successful matching task provides experience to the system, and with the appropriate use of Machine Learning techniques, can be used to propose subsequent matches [8].

3.2 Peer-Data Management

Data integration systems are a significant step forward in data sharing systems, but still fall short of the grand vision of data sharing. In particular, consider the success of the World-Wide Web and that of P2P file sharing system. We still do not know how to create the analog of these two types of systems where the data being shared has rich semantics. The vision for sharing structured data has also recently been known as the *Semantic Web* [4]. Hence, an important challenge is to build data sharing architectures that enable such wide-scale authoring and querying.

One effort in this direction is known as *Peer-data management systems* (PDMS). PDMS offer a flexible architecture for sharing data, where instead of requiring the participants to create a central mediated schema, they can interoperate by specifying local semantic mappings, and queries can be answered by chaining mappings [21, 25, 30, 5]. As a result, sharing data is made easier, because peers can map their schema to the most convenient peer in the system rather than the mediated schema. In addition, they can query the system using their own schema, rather than using a mediated schema that may be foreign to them.

3.3 Crossing the Structure Chasm

The difficulties of creating large-scale data sharing systems are rooted in the profound differences between two types of data management: the unstructured world of text versus the structured world of data and knowledge bases [23]. In the structured world, authoring and querying data involve a conceptual effort of building and/or understanding a schema, while in the unstructured world, they merely involve writing text or entering keywords. The difficulties of sharing data in the structured world have been mentioned above, where as sharing is straightforward in the unstructured world. Clearly, one cannot expect management of data to be equally easy in both world – you need to *pay to play*: – you must put some effort into creating your data if you expect the benefits of complex querying offered by the tools in the structured world. However, there is a long way to go in making the interactions with data in the structured world much easier. In [23] this challenge is named *crossing the structure chasm*. This is a multi-decade project that will require substantial effort from the community.

4 Conclusion

Data management has traditionally focused on systems in which data is logically and physically centralized. The World-Wide Web, Ubiquitous Computing, Personal Data Management present challenges to data management where data needs to be created everywhere and by anyone, and accessed anytime and through various media. Data integration represents one point in the exciting journey from central data management to the vision of ubiquitous data. We have made significant progress over the last few years, and we must continue to focus towards the ultimate goal.

References

1. S. Abiteboul and O. Duschka. Complexity of answering queries using materialized views. In *Proc. of PODS*, pages 254–263, Seattle, WA, 1998.
2. S. Adali, K. Candan, Y. Papakonstantinou, and V. Subrahmanian. Query caching and optimization in distributed mediator systems. In *Proc. of SIGMOD*, pages 137–148, Montreal, Canada, 1996.
3. R. Avnur and J. M. Hellerstein. Eddies: Continuously adaptive query processing. In *Proc. of SIGMOD*, 2000.
4. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, May 2001.
5. P. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, and I. Zaihrayeu. Data management for peer-to-peer computing : A vision. In *ACM SIGMOD WebDB Workshop 2002*, 2002.
6. R. L. Cole. A decision theoretic cost model for dynamic plans. *IEEE Data Engineering Bulletin*, 23(2):34–41, 2000.
7. H.-H. Do and E. Rahm. COMA - a system for flexible combination of schema matching approaches. In *Proc. of VLDB*, 2002.
8. A. Doan, P. Domingos, and A. Halevy. Reconciling schemas of disparate data sources: a machine learning approach. In *Proc. of SIGMOD*, 2001.

9. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *Proc. of the Int. WWW Conf.*, 2002.
10. D. Draper, A. Y. Halevy, and D. S. Weld. The nimble integration system. In *Proc. of SIGMOD*, 2001.
11. O. Duschka, M. Genesereth, and A. Levy. Recursive query plans for data integration. *Journal of Logic Programming, special issue on Logic Based Heterogeneous Information Systems*, 43(1):49–73, 2000.
12. O. M. Duschka and M. R. Genesereth. Answering recursive queries using views. In *Proc. of PODS*, pages 109–116, Tucson, Arizona., 1997.
13. <http://www.enosysmarkets.com>, 2003.
14. D. Florescu, A. Levy, I. Manolesu, and D. Suciu. Query optimization in the presence of limited access patterns. In *Proc. of SIGMOD*, 1999.
15. D. Florescu, A. Levy, and A. Mendelzon. Database techniques for the world-wide web: A survey. *SIGMOD Record*, 27(3):59–74, September 1998.
16. M. Friedman, A. Levy, and T. Millstein. Navigational plans for data integration. In *Proceedings of the National Conference on Artificial Intelligence*, 1999.
17. M. Friedman and D. Weld. Efficient execution of information gathering plans. In *Proceedings of the International Joint Conference on Artificial Intelligence, Nagoya, Japan*, pages 785–791, 1997.
18. H. Garcia-Molina, Y. Papakonstantinou, D. Quass, A. Rajaraman, Y. Sagiv, J. Ullman, and J. Widom. The TSIMMIS project: Integration of heterogeneous information sources. *Journal of Intelligent Information Systems*, 8(2):117–132, March 1997.
19. J. Goldstein and P.-A. Larson. Optimizing queries using materialized views: a practical, scalable solution. In *Proc. of SIGMOD*, pages 331–342, 2001.
20. G. Graefe and R. Cole. Optimization of dynamic query evaluation plans. In *Proc. of SIGMOD*, Minneapolis, Minnesota, 1994.
21. S. Gribble, A. Halevy, Z. Ives, M. Rodrig, and D. Suciu. What can databases do for peer-to-peer? In *ACM SIGMOD WebDB Workshop 2001*, 2001.
22. L. Haas, D. Kossmann, E. Wimmers, and J. Yang. Optimizing queries across diverse data sources. In *Proc. of VLDB*, Athens, Greece, 1997.
23. A. Halevy, O. Etzioni, A. Doan, Z. Ives, J. Madhavan, L. McDowell, and I. Tatarinov. Crossing the structure chasm. In *To appear in the Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR)*, 2003.
24. A. Y. Halevy. Answering queries using views: A survey. *VLDB Journal*, 10(4), 2001.
25. A. Y. Halevy, Z. G. Ives, D. Suciu, and I. Tatarinov. Schema mediation in peer data management systems. In *Proc. of ICDE*, 2003.
26. Z. Ives, D. Florescu, M. Friedman, A. Levy, and D. Weld. An adaptive query execution engine for data integration. In *Proc. of SIGMOD*, 1999.
27. Z. Ives, A. Halevy, and D. Weld. An xml query engine for network-bound data. *VLDB Journal, Special Issue on XML Query Processing*, 2003.
28. Z. Ives, A. Levy, D. Weld, D. Florescu, and M. Friedman. Adaptive query processing for internet applications. *Data Engineering Bulletin* 23(3), 2000.
29. N. Kabra and D. J. DeWitt. Efficient mid-query re-optimization of sub-optimal query execution plans. In *Proc. of SIGMOD*, pages 106–117, Seattle, WA, 1998.
30. P. Kalnis, W. Ng, B. Ooi, D. Papadias, and K. Tan. An adaptive peer-to-peer network for distributed caching of olap results. In *Proc. of SIGMOD*, 2002.
31. M. Lenzerini. Data integration: A theoretical perspective. In *Proc. of PODS*, pages 233–246, 2002.
32. A. Y. Levy, A. Rajaraman, and J. J. Ordille. Querying heterogeneous information sources using source descriptions. In *Proc. of VLDB*, pages 251–262, Bombay, India, 1996.

33. J. Madhavan, P. Bernstein, and E. Rahm. Generic schema matching with cupid. In *Proceedings of the International Conference on Very Large Databases (VLDB)*, 2001.
34. I. Manolescu, D. Florescu, and D. Kossmann. Answering xml queries on heterogeneous data sources. In *Proc. of VLDB*, pages 241–250, 2001.
35. <http://www.nimble.com>, 2003.
36. N. F. Noy and M. A. Musen. PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2000.
37. M. T. Ozsu and P. Valduriez. *Principles of Distributed Database Systems*. Prentice Hall, Upper Saddle River, New Jersey, 1999.
38. R. Pottinger and A. Halevy. Minicon: A scalable algorithm for answering queries using views. *VLDB Journal*, 2001.
39. E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, 2001.
40. A. Rajaraman, Y. Sagiv, and J. D. Ullman. Answering queries using templates with binding patterns. In *Proc. of PODS*, pages 105–112, San Jose, CA, 1995.
41. T. Urhan and M. J. Franklin. Xjoin: A reactively-scheduled pipelined join operator. *IEEE Data Engineering Bulletin*, 23(2):27–33, 2000.
42. T. Urhan, M. J. Franklin, and L. Amsaleg. Cost based query scrambling for initial delays. In *Proc. of SIGMOD*, pages 130–141, Seattle, WA, 1998.
43. V. Vassalos and Y. Papakonstantinou. Describing and using the query capabilities of heterogeneous sources. In *Proc. of VLDB*, Athens, Greece, 1997.