Discovering Structure in a Corpus of Schemas

Alon Y. Halevy
University of Washington
alon@cs.washington.edu

Jayant Madhavan
University of Washington
jayant@cs.washington.edu

Philip A. Bernstein
Microsoft Research
philbe@microsoft.com

Abstract

This paper describes a research program that exploits a large corpus of database schemas, possibly with associated data and meta-data, to build tools that facilitate the creation, querying and sharing of structured data. The key insight is that given a large corpus, we can discover patterns concerning how designers create structures for representing domains. Given these patterns, we can more easily map between disparate structures or propose structures that are appropriate for a given domain. We describe the first application of our approach to the problem of semi-automatic schema matching.

1 Introduction

Database and knowledge base systems offer their users powerful mechanisms for querying their data. However, such power comes with a significant upfront cost, namely, getting the data into structured form. A key component of the cost is the effort involved in designing a database schema, and for the users, the effort of learning the schema in order to be able to query it. It is instructive to compare the conceptual effort involved in dealing with structured data versus the effort in dealing with text documents [HED$^+$03]. In the latter case, authoring data is a matter of writing coherent text and keyword querying is simple, though providing much less expressive power and accuracy. The differences between the management of structured and unstructured data are exacerbated when people try to *share* data (e.g., on the web, within enterprises and in large scientific projects). Mediating between disparate databases requires understanding the semantic relationships between the data sets which usually involves a large cooperative effort.

The goal of the Revere Project [HED$^+$03] at the University of Washington is to facilitate the activities of authoring, querying and sharing of structured data, so they become palatable to non-expert users. The main components of the project are (1) the Mangrove System [MEH$^+$03], whose goal is to entice people to structure their data, (2) the Piazza System [HITM03, HIST03, HIM$^+$04, MH03] that allows large-scale sharing of structured data without any central mediation, and (3) a set of tools that facilitate the authoring of schemas and the creation of semantic mappings between disparate schemas for data sharing.

This paper focuses on the latter component of Revere, which takes a new approach to developing tools for aiding the management of structured data. The techniques are inspired by one of the main ideas in Information Retrieval (IR), namely the statistical analysis of large corpora of text. Here we examine what we can do with a large corpus of database schemas, possibly with associated data instances and meta-data.

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

We begin in Section 2 by describing the contents of such a corpus and what we aim to achieve with it. In Section 3 we describe its first application to the problem of matching between disparate schemas. In Section 4 we discuss other uses of corpus-based tools, and in Section 5 we describe the main challenges we face in pursuing this approach.

2 A Corpus of Schemas

In Information Retrieval (IR), large corpora of text are analyzed for specific patterns: co-occurrences of terms often indicate that the terms are related or synonymous; high frequency of a term in a particular document (compared to the rest of the corpus) indicates the term's importance to the document. More generally, IR tries to glean clues from patterns in people's use of natural language.

In our context, our goal is to use a corpus of schemas to find patterns in *how people build structures for data*. In particular, creating a database schema involves at least the following steps. In each of these we can search for common patterns:

- Selecting entities and relationships between them: typically done using a conceptual entity/relationship model.
- Naming: choose particular names for the entities, their properties and the relationships, and
- Structuring: take the conceptual model and translate it into a particular data model (e.g., relational tables, object or XML hierarchies). While this step is aided by tools, there is still significant manual work. Furthermore, in this step additional naming decisions are made (e.g., for table names, disambiguating property names).

As in IR, patterns in the above processes cannot be written, but can only be discovered from a large collection of database structures. Once we discover the patterns, we can build tools that assist designers in authoring new schemas, querying schemas, and mapping between disparate schemas. We now briefly discuss the contents of a corpus and the kinds of patterns we can mine from it.

2.1 Contents of a corpus

A corpus can include any kind of information related to structured data, and may give us cues as to the meaning of the data. In particular, it includes the following.

1. Various forms of structures: on the less expressive end, the corpus can include relational and object-oriented database schema or entity/relationship diagrams, XML DTDs or schemas, possibly with associated integrity constraints (e.g., functional dependencies). On the more expressive end, the corpus can include terminologies in a knowledge representation language (e.g., in OWL [DCvH$^+$02]).

2. Instance data: it can include the actual rows of tables (or representative rows) or XML documents. In fact it can include data sets that do not have a schema (e.g., certain file formats). Very often, elements in the schema of one model are instance data in another model. Hence, the distinction between schema and instance data is not clear cut.

3. Validated mappings: previously constructed mappings between different database structures provide significant clues as to how entities are represented differently in disparate models. Whenever we have such mappings, they should be included in the corpus.

4. Queries: queries (posed by users or applications) provide important information about how certain data is used. For example, when a database query performs a join over attributes of two different tables, that indicates that the columns are modeling the same domain (and this often is not evident from the schema, which only specifies the data type).

5. Other meta-data: there are many forms of meta-data that accompany database structures. They range from text descriptions of the meaning of different fields to statistics about table cardinalities or histograms on the set of possible values within a column.

It is important to emphasize that a corpus is *not* a logically coherent universal database. Rather, it is a collection of disparate uncoordinated structures. We expect that the structures of the corpus will be stored and accessed using tools for model management [Ber03], which provide a set of operators for manipulating models of data (as opposed to the data itself).

2.1.1 Statistics on the corpus

There is a plethora of possible analyses that can be performed on such a corpus in order to extract interesting patterns. Finding the most effective ones is a long term research challenge. Below we describe certain kinds of statistics that can be computed over the corpus. We classify them according to whether they apply to individual words or terms, partial structures, or elements of particular schemas.

Word and term statistics: these statistics are associated with individual words (in any language) and with noun or verb phrases. These statistics indicate how a word is used in different roles in structured data. For each of these statistics, we can maintain different versions, depending on whether we take into consideration word stemming, synonym tables, inter-language dictionaries, or any combination of these three. The statistics include:

1. Term usage: How frequently the term is used as a relation name, as an attribute name, or in data (as a percentage of all of its uses or as a percentage of structures in the corpus).

2. Co-occurring schema elements: For each of the different uses of a term, which relation names and attributes tend to appear with it? What tend to be the names of related tables and their attribute names? What tend to be the join predicates on pairs of tables? Are there clusters of attribute names that appear together? Are there mutually exclusive uses of attribute names? We can also learn from co-occurrence of terms in *composite* names of schema elements.

3. Similar names: For each of the uses of a term, which other words tend to be used with similar statistical characteristics?

Composite statistics: the same statistics can be applied to *partial structures*. Examples of partial structures are sets of data instances, relations with associated attribute names, a relation with some data (possibly with missing values). In fact, the works in [HC03, KN03] are attempts to learn from partial structures.

Clearly, we need to significantly limit the number of partial structures for which we keep statistics (e.g., use techniques for discovering partial structures that occur frequently e.g., [PG02]). Given statistics for certain partial structures, we can estimate the statistics for other related structures.

Statistics for schema elements: the same word, used in different structures, can have different meanings. Hence, we may want to characterize the specific usages of terms in structures, and relate them to usage of terms in other structures. For example, in [MBC+03] we learn a classifier for every relation and attribute name in the corpus. Following [DDH01], we use meta-strategy learning. The training data used for learning is gleaned from the schema to which the element belongs and the training data of elements that have been mapped to it by a validated mapping in the corpus. Intuitively, the classifier is meant to recognize the particular usage of the term, even if it appears differently in another structure.

3 Using a Corpus for Schema Matching

We now describe the first application of our corpus-based approach, schema matching. Sharing data among multiple data sources and applications is a problem that arises time and again in large enterprises, B2B settings, coordination between government agencies, large-scale science projects, and on the World-Wide Web. While there are many architectures for sharing data (data warehousing, data integration systems [Len02, DHW01], message passing systems (e.g., [MQS03]), web services, and peer-data management systems [HIST03, KNO+02, BGK+02, NWQ+02, AK03, ACMH03, NOTZ03]), a key problem in all of them is the semantic heterogeneity between the structures (e.g., schemas) of data sources that were originally designed independently. Thus, to obtain meaningful interoperation, one needs a *semantic mapping* between the schemas. A semantic mapping is a set of expressions that specify how the data in one database corresponds to the data in another database [MBDH02]. While languages for specifying semantic mappings have been developed and are well understood (see [Len02, Hal01] for surveys), the creation of semantic mappings has become a key bottleneck as it is labor intensive and error-prone.

The goal of schema matching is to assist a human to relate two domain models. Complete automation of the process is unlikely, so the goal is to significantly increase the productivity of human experts. The matching problem is difficult because it requires understanding the underlying semantics of the schemas being matched. While a schema (with its instance data) provides many clues to its intended semantics, it does not suffice in order to relate it to a different schema.

The process of generating a semantic mapping has traditionally been divided into two phases. The first phase finds a *match* between the two schemas. The match result is a set of *correspondences* between elements in the two schemas, stating that these elements are *somehow* related. For example, a correspondence may state that buyer in one schema model corresponds to customer in another. The second phase builds on the correspondences by creating the mapping expressions. The mapping expressions, often expressed as queries or rules, enable translating data from one data source to another, or reformulating a query over one data source into a query on the other. A plethora of techniques have been proposed for schema matching: see [RB01] for a survey, and [NM02, DMDH02, DR02] for some work since then. Collectively, these techniques mirror the heuristics that a human designer may follow. For example, techniques have considered exploiting relationships between names of elements in the schemas, structural similarities between them, similarities in data values, and even correlations between values in different attributes. Several recent works on schema matching are based on *combining* multiple techniques in a principled fashion [MBR01, DDH01, DR02].

Corpus-based matching

Schema matching is often facilitated by a detailed knowledge about the domain in which the matching is being performed. However, creating a knowledge base is often hard, and furthermore, the result may be brittle in the sense it only helps on its domain of coverage, and only provides a *single* perspective on the domain. Our approach, corpus-based matching, complements a knowledge base by gleaning relevant knowledge from a large corpus of database schemas and previously validated mappings. There are two types of knowledge that we can glean from such a corpus. First, we can learn the different ways in which words (or terms) are used in database structures (i.e., as relation names, attribute names and data values). Second, the validated mappings show how variations in term usages correspond to each other in disparate structures.

Although such a corpus is not easy to construct, it is a very different kind of activity than building a detailed and comprehensive knowledge base. It does not require the careful ontological design as a knowledge base does, nor the careful control of its contents, thereby removing key bottlenecks present in the design of knowledge bases. The corpus offers *multiple perspectives* on modeling a particular domain, including different levels of coverage of the domain. Thereby, it is more likely to provide knowledge that is useful for matching two disparate schemas.

The LSD System [DDH01] investigated the benefit of learning from previously validated mappings. That worked considered the case where multiple data sources are mapped to a single *mediated schema*, on which users pose queries. LSD was provided with the mediated schema and a set of *training matches* for some data sources. LSD used these matches to learn models of the elements of the mediated schema. Since no single learning algorithm captures all the cues from the domain, LSD used a multi-strategy approach that combined the predictions of several learners. LSD was then asked to predict matches between the mediated schema and a set of test schemas. The experiments in [DDH01] showed that (1) it is possible to achieve high accuracy with multi-strategy learning, and (2) additional accuracy is obtained by considering domain constraints (i.e., a simple form of domain knowledge). Overall, LSD achieved matching accuracy of 75-90% on small to medium sized schemas of data sources on the Web. LSD was then extended to consider simple taxonomies of concepts in [DMDH02].

In recent work [MBC$^+$03], we investigate the benefit of a corpus of schemas and matches, and the ability to use such a corpus to predict mappings between a *pair* of schemas that have not been previously seen. Like in LSD, we learn models for elements in the corpus, using both the information available in the schema and validated matches that are provided in the corpus. Given two schemas, S_1 and S_2, we calculate for each element in them a *similarity vector* w.r.t. the corpus, i.e., how similar each element in S_i is to each element in the corpus. Very roughly speaking, if the similarity vectors of two elements $a_1 \in S_1$ and $a_2 \in S_2$ are similar to each other, then we predict that a_1 matches a_2. The results of our experiments show that (1) even with a modest corpus of 10 schemas we are able to achieve good accuracy, and (2) the correct matches found by using the corpus and those found by other previously known techniques overlap, but have significant differences. Hence, the use of the corpus is finding matches that would not have been predicted by other techniques.

4 Other Uses of the Corpus

While schema matching was the first application that motivated our corpus-based approach, we believe the corpus is a general tool that is applicable elsewhere [1] We now briefly describe some of these applications.

4.1 Creating and Querying Structured Data

As argued at the outset, one of the greatest impediments to using database and knowledge base technology is the conceptual difficulty of dealing with structured data. Hence, we are faced with the challenge of creating tools that facilitate the creation and querying of structured data.

One such tool would be a *schema design advisor*, which assists in the authoring of structured data, much in the spirit of an auto-complete feature. A user of such a tool creates a schema fragment and some data in a particular domain, and the tool then proposes extensions to the schema using the corpus. The user may choose a schema from the list and modify it further to fit the local context. Note that besides time savings, such a tool has other advantages, such as possibly resulting in better designs and helping users conform to certain standards, when these are applicable.

On the querying side, the corpus-based approach can facilitate the querying of unfamiliar data. Specifically, consider a tool that enables you to pose a query using *your own terminology* to any database. The tool would then use the corpus to propose reformulations of your query that are well formed w.r.t. the schemas of the database at hand. The tool may propose a few such queries (possibly with example answers), and let you choose among them or refine them.

[1] In [HM03] we argue that the corpus can form a basis for a new class of Knowledge Representation systems.

4.2 Web Search and Query Answering

Another class of applications concerns various information finding tasks on the Web (or intranet). The first application is *query answering* (e.g., [KEW01, RFQ$^+$02]): a natural language query is posed to a web search interface, and rather than finding relevant pages, the search engine tries to find the answers to the query. A second, related application is *focused crawling* (e.g., [SBG$^+$03]), where the search engine is given a particular topic and tries to find pages relevant to it by crawling from an initial set of pages. In both of these applications, the presence of additional domain knowledge has been shown or argued to be useful. However, the cost of constructing knowledge bases with such wide domain coverage is prohibitive. In contrast, a corpus-based approach can yield a more robust solution.

5 Challenges

Our initial work has revealed significant challenges to building corpus-based tools. We briefly describe some of these challenges that provide many exciting research opportunities.

Creating the corpus: The first challenge is, of course, creating a corpus of interest. Naturally, organizations will not be quick to freely share their schemas. Fortunately, there are many publicly available schemas that are already useful. Furthermore, our initial experiments have shown that significant advantages can be obtained even by learning from a relatively small number of schemas.

A more subtle issue is the *focus* of the corpus – how closely do the domains of the schemas in the corpus need to be related to the domain for which the corpus is being used. For example, if we are concerned with mapping between disparate schemas of purchase orders, should our corpus include only schemas in this domain? Will schemas in other domains help or detract from the effectiveness of the corpus-based methods? Ideally, we would like to be able to collect a large corpus of schemas without carefully controlling their domains, and devise methods that exploit only the relevant information from the corpus.

Granularity in the corpus: The corpus will include many disparate schemas. At one extreme, we can view *every* term in *every* schema as a separate 'concept' and try to learn patterns about its usage. However, such an approach will quickly get out of hand. Instead, we need to devise techniques that cluster elements in the corpus into larger concepts and learn patterns about these clusters.

Tuning the corpus: Whatever automatic methods we use to analyze the corpus, there is no doubt that manual tuning of the corpus can be useful. The tuning can be of several forms: removing useless schemas, adding particularly useful mappings between schemas in the corpus, helping in clustering terms in the corpus, etc. In addition, combining the corpus with a manually constructed domain model raises interesting challenges.

In conclusion, while the challenges for corpus-based tools are enormous, we believe the payoffs could be huge, and the results can profoundly impact how we create and use structured data.

Acknowledgments

Many colleagues have contributed to the ideas that led to corpus-based representation. We would like to thank Anhai Doan, Pedro Domingos, Oren Etzioni, Zack Ives, Pradeep Shenoy, and Igor Tatarinov. Funding was provided by an NSF CAREER/PECASE Grant IIS-9985114, ITR Grant IIS-0205635, a Sloan Fellowship, and gifts from Microsoft Research, Intel, NEC and NTT.

References

[ACMH03] Karl Aberer, Philippe Cudre-Mauroux, and Manfred Hauswirth. The chatty web: Emergent semantics through gossiping. In *Twelfth International World Wide Web Conference*, 2003.

[AK03] Rene J. Miller Anastasios Kementsietsidis, Marcelo Arenas. Mapping data in peer-to-peer systems: Semantics and algorithmic issues. In *SIGMOD '03*, 2003.

[Ber03] Philip A. Bernstein. Applying Model Management to Classical Meta Data Problems. In *Proceedings of the Conference on Innovative Data Systems Research (CIDR)*, 2003.

[BGK+02] P. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, and I. Zaihrayeu. Data management for peer-to-peer computing : A vision. In *Proceedings of the WebDB Workshop*, 2002.

[DCvH+02] M. Dean, D. Connolly, F. van Harmelen, J. Hendler, I. Horrocks, D. McGuinness, P. Patel-Schneider, and L. Stein. OWL web ontology language 1.0 reference, 2002. Manuscript available from `http://www.w3.org/2001/sw/WebOnt/`.

[DDH01] Anhai Doan, Pedro Domingos, and Alon Halevy. Reconciling schemas of disparate data sources: a machine learning approach. In *Proc. of SIGMOD*, 2001.

[DHW01] Denise Draper, Alon Y. Halevy, and Daniel S. Weld. The nimble integration system. In *Proc. of SIGMOD*, 2001.

[DMDH02] Anhai Doan, Jayant Madhavan, Pedro Domingos, and Alon Halevy. Learning to map between ontologies on the semantic web. In *Proc. of the Int. WWW Conf.*, 2002.

[DR02] Hong-Hai Do and Erhard Rahm. COMA - a system for flexible combination of schema matching approaches. In *Proc. of VLDB*, 2002.

[Hal01] Alon Y. Halevy. Answering queries using views: A survey. *VLDB Journal*, 10(4), 2001.

[HC03] Bin He and Kevin Chen-Chuan Chang. Statistical schema integration across the deep web. In *Proc. of SIGMOD*, 2003.

[HED+03] Alon Halevy, Oren Etzioni, Anhai Doan, Zachary Ives, Jayant Madhavan, Luke McDowell, and Igor Tatarinov. Crossing the structure chasm. In *Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR)*, 2003.

[HIM+04] Alon Y. Halevy, Zachary G. Ives, Jayant Madhavan, Peter Mork, Dan Suciu, and Igor Tatarinov. The piazza peer-data management system. *Transactions on Knowledge and Data Engineering, Special issue on Peer-dta management, to appear*, 2004.

[HIST03] Alon Y. Halevy, Zachary G. Ives, Dan Suciu, and Igor Tatarinov. Schema mediation in peer data management systems. In *Proc. of ICDE*, 2003.

[HITM03] Alon Halevy, Zachary Ives, Igor Tatarinov, and Peter Mork. Piazza: Data management infrastructure for semantic web applications. In *Proc. of the Int. WWW Conf.*, 2003.

[HM03] Alon Y. Halevy and Jayant Madhavan. Corpus-based knowledge representation. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1567–1572, 2003.

[KEW01] Cody Kwok, Oren Etzioni, and Dan Weld. Scaling question answering to the web. In *Proc. of the Int. WWW Conf.*, pages 150–161, 2001.

[KN03] Jaewoo Kang and Jeffrey Naughton. On schema matching with opaque column names and data values. In *Proc. of SIGMOD*, 2003.

[KNO+02] 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.

[Len02] Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proc. of PODS*, 2002.

[MBC+03] Jayant Madhavan, Philip Bernstein, Kuang Chen, Alon Halevy, and Pradeep Shenoy. Matching schemas by learning from others. In *Working notes of the IJCAI-03 workshop on Data Integration on the Web*, 2003.

[MBDH02] Jayant Madhavan, Philip Bernstein, Pedro Domingos, and Alon Halevy. Representing and reasoning about mappings between domain models. In *Proceedings of AAAI*, 2002.

[MBR01] Jayant Madhavan, Phil Bernstein, and Erhard Rahm. Generic schema matching with cupid. In *Proceedings of the International Conference on Very Large Databases (VLDB)*, 2001.

[MEH+03] Luke McDowell, Oren Etzioni, Alon Halevy, Henry Levy, Steven Gribble, William Pentney, Deepak Verma, and Stani Vlasseva. Enticing ordinary people onto the semantic web via instant gratification. In *Proceedings of the Second International Conference on the Semantic Web*, October 2003.

[MH03] Jayant Madhavan and Alon Halevy. Composing mappings among data sources. In *Proc. of VLDB*, 2003.

[MQS03] http://www-3.ibm.com/software/ts/mqseries/, 2003.

[NM02] Natalya Noy and Mark A. Musen. PROMPTDIFF: A fixed-point algorithm for comparing ontology versions. In *Proceedings of AAAI*, 2002.

[NOTZ03] Wee Siong Ng, Beng Chin Ooi, Kian-Lee Tan, and Aoying Zhou. PeerDB: A P2P-based system for distributed data sharing. In *SIGMOD '03*, 2003.

[NWQ+02] Wolfgang Nejdl, Boris Wolf, Changtao Qu, Stefan Decker, Michael Sintek, Ambjorn Naeve, Mikael Nilsson, Matthias Palmer, and Tore Risch. EDUTELLA: A P2P networking infrastructure based on RDF. In *Eleventh International World Wide Web Conference*, pages 604–615, 2002.

[PG02] Neoklis Polyzotis and Minos N. Garofalkis. Statistical synopses for graph-structured XML databases. In *SIGMOD '02*, 2002.

[RB01] Erhard Rahm and Philip A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, 2001.

[RFQ+02] Dragomir R. Radev, Weiguo Fan, Hong Qi, Harris Wu, and Amardeep Grewal. Probabilistic question answering on the web. In *Proc. of the Int. WWW Conf.*, pages 408–419, 2002.

[SBG+03] S. Sizov, M. Biwer, J. Graupmann, S. Siersdorfer, M. Theobald, G. Weikum, and P. Zimmer. The BINGO! system for information portal generation and expert web search. In *Proceedings of the First Biannual Conference on Innovative Data Systems Research (CIDR), Asilomar(CA)*, 2003.