AI Magazine Volume 26 Number 1 (2005) (© AAAI)

Semantic-Integration Research in the Database Community

A Brief Survey

AnHai Doan and Alon Y. Halevy

■ Semantic integration has been a long-standing challenge for the database community. It has received steady attention over the past two decades, and has now become a prominent area of database research. In this article, we first review database applications that require semantic integration and discuss the difficulties underlying the integration process. We then describe recent progress and identify open research issues. We focus in particular on schema matching, a topic that has received much attention in the database community, but also discuss data matching (for example, tuple deduplication) and open issues beyond the match discovery context (for example, reasoning with matches, match verification and repair, and reconciling inconsistent data values). For previous surveys of database research on semantic integration, see Rahm and Bernstein (2001); Ouksel and Seth (1999); and Batini, Lenzerini, and Navathe (1986).

The key commonalities underlying database applications that require semantic integration are that they use structured representations (for example, relational schemas and extensible markup language [XML] document type definitions [DTDs]) to encode the data, and that they employ more than one representation. As such, the applications must resolve heterogeneities with respect to the schemas and their data, either to enable their manipulation (for example, merging the schemas or computing the differences [Batini, Lenzerini, and Navathe 1986; Bernstein 2003]) or to enable the translation of data and queries across the schemas. Many such applications have arisen over time and have been studied actively by the database community.

One of the earliest such applications is *schema integration:* merging a set of given schemas into a single global schema (Batini, Lenzerini, and Navathe 1986; Elmagarmid and Pu 1990; Seth and Larson 1990; Parent and Spaccapietra 1998; Pottinger and Bernstein 2003). This problem has been studied since the early 1980s. It arises in building a database system that comprises several distinct databases and in designing the schema of a database from the local schemas supplied by several user groups. The integration process requires establishing semantic correspondences—matches—between the component schemas and then using the matches to merge schema elements (Pottinger and Bernstein 2003; Batini, Lenzerini, and Navathe 1986).

As databases become widely used, there is a growing need to translate data between multiple databases. This problem arises when organizations consolidate their databases and hence must transfer data from old databases to the new ones. It forms a critical step in data warehousing and data mining, two important research and commercial areas since the early 1990s. In these applications, data coming from multiple sources must be transformed to data conforming to a single target schema to enable further data analysis (Miller, Haas, and Hernandez 2000; Rahm and Bernstein 2001).

In recent years, the explosive growth of information online has given rise to even more application classes that require semantic integration. One application class builds data-integration systems (for example, Garcia-Molina et al. 1997; Levy, Rajaraman, and Ordille 1996; Ives et al. 1999; Lambrecht, Kambhampati, and Gnanaprakasam 1999; Friedman and Weld 1997; Knoblock et al. 1998). Such a system provides users with a uniform query interface (called *mediated schema*) to a multitude of data

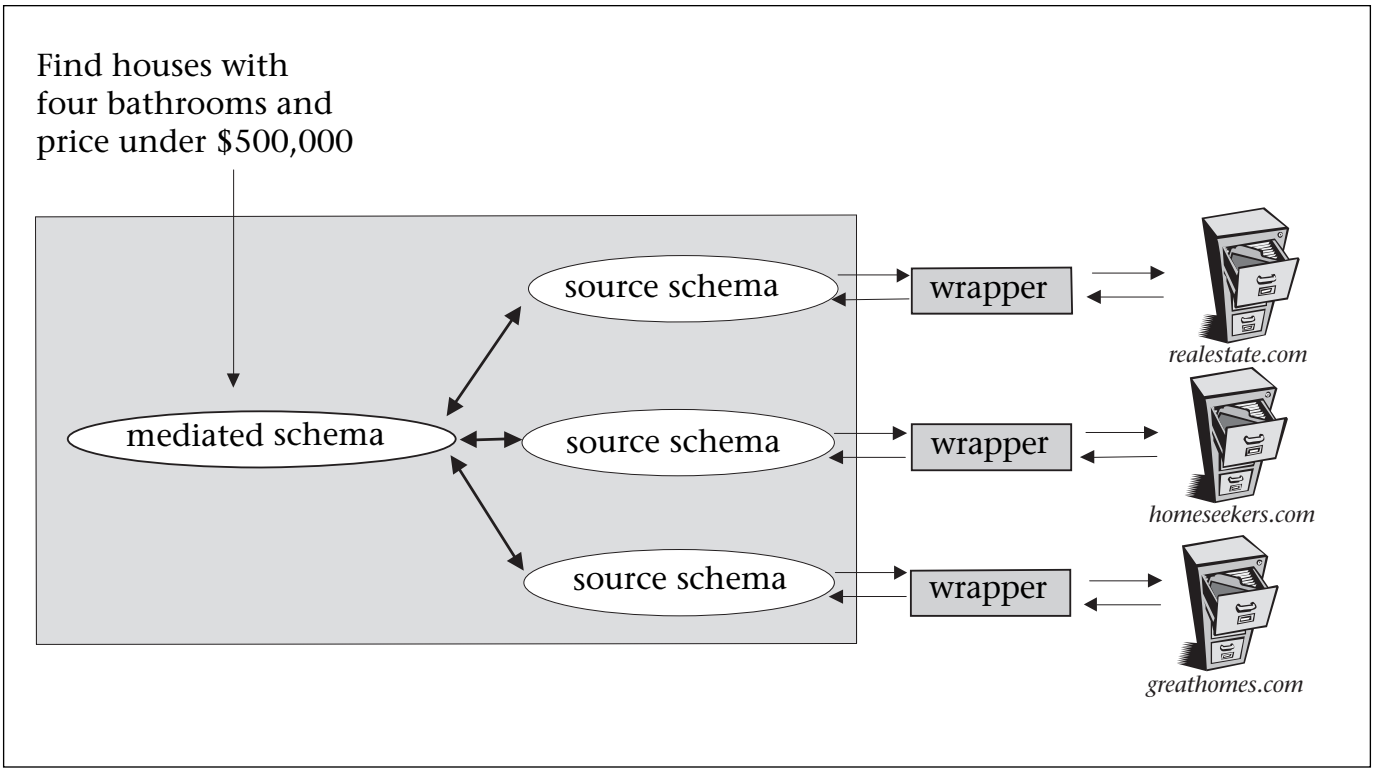

Figure 1. A Data-integration System in the Real Estate Domain.

Such a system uses the semantic correspondences between the mediated schema and the source schemas (denoted with double-head arrows in the figure) to reformulate user queries.

sources, thus freeing them from manually querying each individual source.

Figure 1 illustrates a data-integration system that helps users find houses on the real estate market. Given a user query over the mediated schema, the system uses a set of semantic matches between the mediated schema and the local schemas of the data sources to translate it into queries over the source schemas. Next, it executes the queries using wrapper programs attached to the sources (for example, Kushmerick, Weld, and Doorenbos [1997]), then combines and returns the results to the user. A critical problem in building a data-integration system, therefore, is to supply the semantic matches. Since in practice data sources often contain duplicate items (for example, the same house listing) (Hernandez and Stolfo 1995; Bilenko and Mooney 2003; Tejada, Knoblock, and Minton 2002), another important problem is to detect and eliminate duplicate data tuples from the answers returned by the sources before presenting the final answers to the user query.

Another important application class is peer data management, which is a natural extension of data integration (Aberer 2003). A peer data management system does away with the notion of mediated schema and allows peers (that is, participating data sources) to query and retrieve data directly from each other. Such querying and data retrieval require the creation of semantic correspondences among the peers.

Recently there has also been considerable attention on model management, which creates tools for easily manipulating models of data (for example, data representations, website structures, and entity relationship [ER] diagrams). Here semantic integration plays a central role, as matching and merging models form core operations in model management algebras (Bernstein 2003; Rahm and Bernstein 2001).

The data-sharing applications described above arise in numerous current real-world domains. They also play an important role in emerging domains such as e-commerce, bioinformatics, and ubiquitous computing. Some recent developments should dramatically increase the need for and the deployment of applications that require semantic integration. The Internet has brought together millions of data sources and makes possible data sharing among them. The widespread adoption of XML as a standard syntax to share data has further streamlined and eased the data-sharing process. The growth of the semantic web will further fuel data-sharing applications and underscore the key role that semantic integration plays in their deployment.

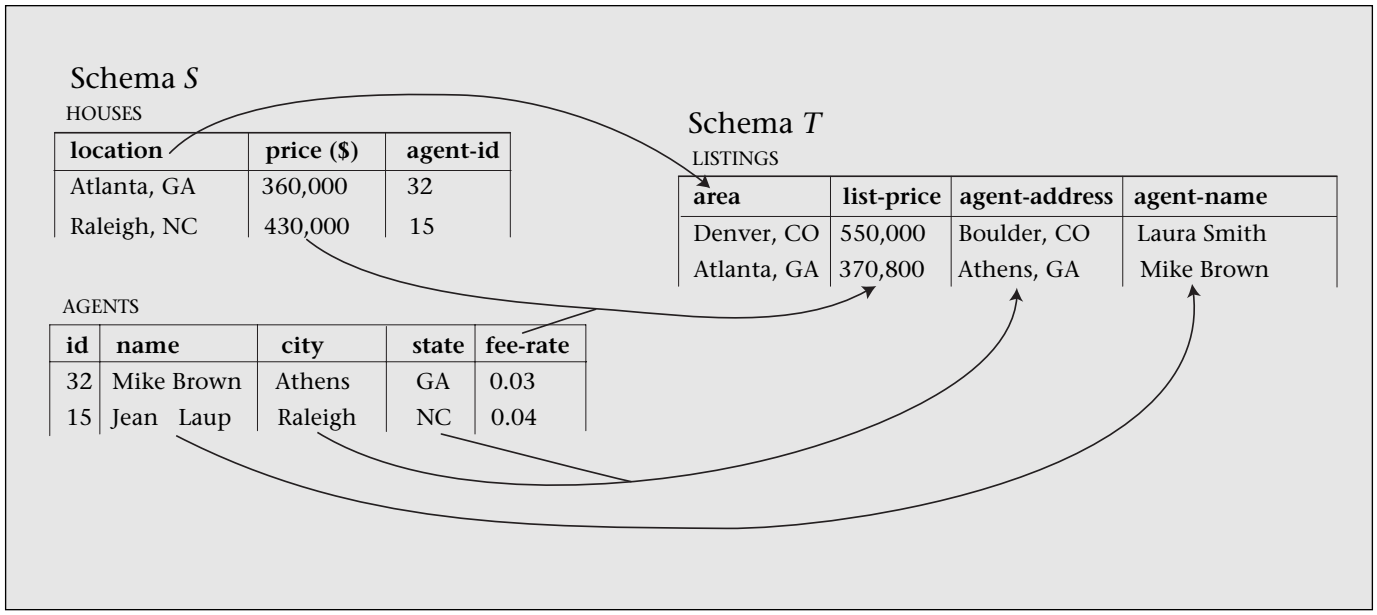

Figure 2. The Schemas of Two Relational Databases S *and* T *on House Listing, and the Semantic Correspondences between Them.*

Database *S* consists of two tables: HOUSES and AGENTS; database *T* consists of the single table LISTINGS.

Challenges of Semantic Integration

Despite its pervasiveness and importance, semantic integration remains an extremely difficult problem. Consider, for example, the challenges that arise during a schema-matching process, which finds semantic correspondences (called *matches*) between database schemas. For example, given the two relational databases on the house listing in figure 2, the process finds matches such as "*location* in schema *S* matches *area* in schema *T*" and "*name* matches *agent-name*."

At the core, matching two database schemas *S* and *T* requires deciding whether any two elements *s* of *S* and *t* of *T* match, that is, whether they refer to the same real-world concept. This problem is challenging for several fundamental reasons.

First, the semantics of the involved elements can be inferred from only a few information sources, typically the creators of data, documentation, and associated schema and data. Extracting semantics information from data creators and documentation is often extremely cumbersome. Frequently, the data creators have long moved, retired, or forgotten about the data. Documentation tends to be sketchy, incorrect, and outdated. In many settings, such as when building data-integration systems over remote web sources, data creators and documentation are simply not accessible. Hence schema elements are typically matched based on clues in the schema and data. Examples of such clues include element names, types, data values, schema structures, and integrity constraints. However, these clues are often unreliable. For example, two elements that share the same name (for example, area) can refer to different real-world entities (the location and square-feet area of the house). The reverse problem also often holds: two elements with different names (for example, *area* and *location*) can refer to the same real-world entity (the location of the house).

Second, schema and data clues are also often incomplete. For example, the name contact-agent suggests only that the element is related to the agent. It does not provide sufficient information to determine the exact nature of the relationship (for example, whether the element is about the agent's phone number or her name).

Third, to decide that element *s* of schema *S* matches element *t* of schema *T,* one must typically examine all other elements of *T* to make sure that there is no other element that matches *s* better than *t.* This global nature of matching adds substantial cost to the matching process.

Finally, to make matters worse, matching is often subjective, depending on the application. One application may decide that *house-style* matches *house-description,* another application may decide that it does not. Hence, the user must often be involved in the matching process. Sometimes, the input of a single user may be considered too subjective, and then a whole committee must be assembled to decide what the correct mapping is (Clifton, Housman, and Rosenthal 1997).

Because of the above challenges, the manual creation of semantic matches has long been known to be extremely laborious and error prone. For example, a recent project at the GTE telecommunications company sought to integrate 40 databases that have a total of 27,000 elements (that is, attributes of relational tables) (Li and Clifton 2000). The project planners estimated that, without the original developers of the databases, just finding and documenting the matches among the elements would take more than 12 person years.

The problem of matching data tuples also faces similar challenges. In general, the high cost of manually matching schemas and data has spurred numerous solutions that seek to automate the matching process. Because the users must often be in the loop, most of these solutions have been semiautomatic. Research on these solutions dates back to the early 1980s and has picked up significant steam in the past decade due to the need to manage the astronomical volume of distributed and heterogeneous data at enterprises and on the web. In the next two sections we briefly review this research on schema and data matching.

Schema Matching

We discuss the accumulated progress in schema matching with respect to matching techniques, architectures of matching solutions, and types of semantic matches.

Matching Techniques

A wealth of techniques has been developed to find semantic matches semiautomatically. The techniques fall roughly into two groups: rule-based and learning-based solutions (though several techniques that leverage ideas from the fields of information retrieval and information theory have also been developed [Clifton, Housman, and Rosenthal 1997; Kang and Naughton 2003]).

Rule-Based Solutions. Many of the early as well as current matching solutions employ handcrafted rules to match schemas (Milo and Zohar 1998; Palopoli, Sacca, and Ursino 1998; Castano and Antonellis 1999; Mitra, Wiederhold, and Jannink 1999; Madhavan, Bernstein, and Rahm 2001; Melnik, Molina-Garcia, and Rahm 2002).

In general, handcrafted rules exploit schema information such as element names, data types, structures, number of subelements, and integrity constraints. A broad variety of rules have been considered. For example, the TranScm system (Milo and Zohar 1998) employs rules such as "two elements match if they have the same name (allowing synonyms) and the same number of subelements." The DIKE system (Palopoli, Sacca, and Ursino 1998; Palopoli et al. 1999; Palopoli, Terracina, and Ursino 2000) computes the similarity between two schema elements based on the similarity of the characteristics of the elements and the similarity of related elements. The ARTEMIS and the related MOMIS (Castano and Antonellis 1999; Bergamaschi et al. 2001) systems compute the similarity of schema elements as a weighted sum of the similarities of name, data type, and substructure. The CUPID system (Madhavan, Bernstein, and Rahm 2001) employs rules that categorize elements based on names, data types, and domains. Rules therefore tend to be domain-independent, but can be tailored to fit a certain domain. Domain-specific rules can also be crafted.

Rule-based techniques provide several benefits. First, they are relatively inexpensive and do not require training as in learning-based techniques. Second, they typically operate only on schemas (not on data instances) and hence are fairly fast. Third, they can work very well in certain types of applications and for domain representations that are amenable to rules (Noy and Musen 2000).

Finally, rules can provide a quick and concise method to capture valuable user knowledge about the domain. For example, the user can write regular expressions that encode times or phone numbers, or quickly compile a collection of county names or zip codes that help recognize those types of entities. As another example, in the domain of academic course listing, the user can write the following rule: "use regular expressions to recognize elements about times, then match the first time element with start-time and the second element with end-time." Learning techniques, as we discuss shortly, would have difficulties being applied to these scenarios. They either cannot learn the above rules or can do so only with abundant training data or with the right representations for training examples.

The main drawback of rule-based techniques is that they cannot exploit data instances effectively, even though the instances can encode a wealth of information (for example, value format, distribution, frequently occurring words in the attribute values, and so on) that would greatly aid the matching process. In many cases effective matching rules are simply too difficult to handcraft. For example, it is not clear how to handcraft rules that distinguish between "movie description" and "user comments on the movies," both being long textual paragraphs. In contrast, learning methods such as

Naive Bayes can easily construct "probabilistic rules" that distinguish the two with high accuracy, based on the frequency of words in the paragraphs.

Another drawback is that rule-based methods cannot exploit previous matching efforts to assist in the current ones. Thus, in a sense, systems that rely solely on rule-based techniques have difficulties learning from the past, to improve over time. The above reasons have motivated the development of learning-based matching solutions.

Learning-Based Solutions. Many such solutions have been proposed in the past decade, such as those described by Li, Clifton, and Liu (2000); Clifton, Housman, and Rosenthal (1997); Berlin and Motro (2001, 2002); Doan, Domingos, and Halevy (2001); Dhamankar et al. (2004); Embley, Jackman, and Xu (2001); and Neumann et al. (2002). The solutions have considered a variety of learning techniques and exploited both schema and data information. For example, the SemInt system (Li, Clifton, and Liu 2000) uses a neural-network learning approach. It matches schema elements based on attribute specifications (such as data types, scale, the existence of constraints) and statistics of data content (such as maximum, minimum, average, and variance). The LSD system (Doan, Domingos, and Halevy 2001) employs Naive Bayes over data instances and develops a novel learning solution to exploit the hierarchical nature of XML data. The iMAP system (Dhamankar et al. 2004) and also the ILA and HICAL systems developed in the AI community (Perkowitz and Etzioni 1995; Ryutaro, Hideaki, and Shinichi 2001) match the schemas of two sources by analyzing the description of objects that are found in both sources. The Autoplex and Automatch systems (Berlin and Motro 2001, 2002) use a Naive Bayes learning approach that exploits data instances to match elements.

In the past five years, there has also been a growing realization that schema- and data-related evidence in two schemas being matched often is inadequate for the matching process. Hence, several works have advocated learning from the external evidence beyond the two current schemas. Several types of external evidence have been considered. Some recent works advocate exploiting past matches (Doan, Domingos, and Halevy 2001; Do and Rahm 2002; Berlin and Motro 2002; Rahm and Bernstein 2001; Embley, Jackman, and Xu 2001; Bernstein et al. 2004). The key idea is that a matching tool must be able to learn from the past matches, to predict successfully matches for subsequent, unseen matching scenarios.

The work by Madhavan et al. (2005) goes further and describes how to exploit a corpus of schemas and matches in the domain. This scenario arises, for example, when we try to exploit the schemas of numerous real estate sources on the web to help in matching two specific real estate source schemas. In a related direction, the papers by He and Chang (2003) and Wu et al. (2004) describe settings in which one must match multiple schemas all at once. Here the knowledge gleaned from each matching pair can help match other pairs; as a result we can obtain better accuracy than just matching a pair in isolation. McCann et al. (2003) discusses how to learn from a corpus of users to assist schema matching in data-integration contexts. The basic idea is to ask the users of a data-integration system to "pay" for using it by answering relatively simple questions and then use those answers to further build the system, including matching the schemas of the data sources in the system. This way, an enormous burden of schema matching is lifted from the system builder and spread "thinly" over a mass of users.

Architecture of Matching Solutions

The complementary nature of rule- and learner-based techniques suggests that an effective matching solution should employ both—each on the types of information that it can effectively exploit. To this end, several recent works (Bernstein et al. 2004; Do and Rahm 2002; Doan, Domingos, and Halevy 2001; Embley, Jackman, and Xu 2001; Rahm, Do, and Massmann 2004; Dhamankar et al. 2004) have described a system architecture that employs multiple modules called matchers, each of which exploits well a certain type of information to predict matches. The system then combines the predictions of the matchers to arrive at a final prediction for matches. Each matcher can employ one or a set of matching techniques as described earlier (for example, handcrafted rules, learning methods, information retrieval (IR)–based ones). Combining the predictions of matchers can be manually specified (Do and Rahm 2002; Bernstein et al. 2004) or automated to some extent using learning techniques (Doan, Domingos, and Halevy 2001).

Besides being able to exploit multiple types of information, the multimatcher architecture has the advantage of being highly modular and can be easily customized to a new application domain. It is also extensible in that new, more efficient matchers could be easily added when they become available. A recent work (Dhamankar et al. 2004) also shows that the above solution architecture can be extended successfully to handle complex matches.

An important current research direction is to evaluate the above multimatcher architecture in real-world settings. Bernstein et al. (2004) and Rahm, Do, and Massmann 2004 make some initial steps in this direction. A related direction appears to be a shift away from developing complex, isolated, and monolithic matching systems toward creating robust and widely useful matcher operators and developing techniques to quickly and efficiently combine the operators for a particular matching task.

Incorporating Domain Constraints. It was recognized early on that domain integrity constraints and heuristics provide valuable information for matching purposes. Hence, almost all matching solutions exploit some forms of this type of knowledge.

Most works exploit integrity constraints in matching schema elements locally. For example, many works match two elements if they participate in similar constraints. The main problem with this scheme is that it cannot exploit "global" constraints and heuristics that relate the matching of multiple elements (for example, "at most one element matches *house-address*"). To address this problem, several recent works (Melnik, Molina-Garcia, and Rahm 2002; Madhavan, Bernstein, and Rahm 2001; Doan, Domingos, and Halevy 2001; Doan et al. 2003b) have advocated moving the handling of constraints to after the matchers. This way, the constraint-handling framework can exploit "global" constraints and is highly extensible to new types of constraints.

While integrity constraints constitute domain-specific information (for example, *house-id* is a key for house listings), heuristic knowledge makes general statements about how the matching of elements relate to each other. A well-known example of a heuristic is "two nodes match if their neighbors also match," variations of which have been exploited in many systems (for example, Milo and Zohar [1998]; Madhavan, Bernstein, and Rahm [2001]; Melnik, Molina-Garcia, and Rahm [2002]; Noy and Musen [2001]). The common scheme is to iteratively change the matching of a node based on those of its neighbors. The iteration is carried out one or twice, or until some convergence criterion is reached.

Related Work in Knowledge-Intensive Domains

Schema matching requires making multiple interrelated inferences by combining a broad variety of relatively shallow knowledge types. In recent years, several other problems that fit the above description have also been studied in the AI community. Notable problems are information extraction (for example, Freitag [1998]), solving crossword puzzles (Keim et al. 1999), and identifying phrase structure in natural language processing (NLP) (Punyakanok and Roth 2000). What is remarkable about these studies is that they tend to develop similar solution architectures that combine the prediction of multiple independent modules and optionally handle domain constraints on top of the modules. These solution architectures have been shown empirically to work well. It will be interesting to see if such studies converge in a definitive blueprint architecture for making multiple inferences in knowledge-intensive domains.

Types of Semantic Matches

Most schema-matching solutions have focused on finding 1–1 matches such as *"location = address."* However, relationships between real-world schemas involve many complex matches, such as *"name* = concat(*first-name,last-name*)*"* and *"listed-price = price* * (1 + *tax-rate*)." Hence, the development of techniques to construct complex matches semiautomatically is crucial to any practical mapping effort.

Creating complex matches is fundamentally harder than 1–1 matches for the following reason. While the number of candidate 1–1 matches between a pair of schemas is bounded (by the product of the sizes of the two schemas), the number of candidate complex matches is not. There are an unbounded number of functions for combining attributes in a schema, and each one of these could be a candidate match. Hence, in addition to the inherent difficulties in generating a match to start with, the problem is exacerbated by having to examine an unbounded number of match candidates.

There have been only a few works on complex matching. Milo and Zohar (1998) hard-code complex matches into rules. The rules are systematically tried on the given schema pair, and when such a rule fires, the system returns the complex match encoded in the rule. Several recent works have developed more general techniques to find complex matches. They rely on a domain ontology (Xu and Embley 2003), use a combination of search and learning techniques (Dhamankar et al. 2004; Doan et al. 2003b), or employ mining techniques (He, Chang, and Han 2004). Xu and Embley (2003), for example, consider finding complex matches between two schemas by first mapping them into a domain ontology and then constructing the matches based on the relationships inherent in that ontology. The iMAP system reformulates schema matching as a search in an often very large or infinite match space. To search

effectively, it employs a set of searchers, each discovering specific types of complex matches.

Perhaps the key observation gleaned so far from the above few works is that we really need domain knowledge (and lots of it!) to perform accurate complex matching. Such knowledge is crucial in guiding the process of searching for likely complex match candidates (in a vast or often infinite candidate space), in pruning incorrect candidates early (to maintain an acceptable run time), and in evaluating candidates.

Another important observation is that the correct complex match is often not the top-ranked match but somewhere in the top few matches predicted. Since finding a complex match requires gluing together so many different components (for example, the elements involved, the operations, and so on), perhaps this is inevitable and inherent to any complex matching solution. This underscores the importance of generating explanations for the matches, and building effective match design environments, so that humans can effectively examine the top-ranked matches to select the correct ones.

Data Matching

Besides schema matching, the problem of data matching (for example, deciding whether two different relational tuples from two sources refer to the same real-world entity) is also becoming increasingly crucial. Popular examples of data matching include matching citations of research papers, authors, and institutions. As another example, consider again the databases in figure 2. Suppose we have created the mappings and have used them to transfer the house listings from database *S* and another database *U* (not shown in the figure) to those of database *T*. Databases *S* and *U* may contain many duplicate house listings. Hence in the next step we would like to detect and merge such duplicates, to store and reason with the data at database *T*.

The aforementioned tuple-matching problem has received much attention in the database, AI, knowledge discovery and data mining (KDD), and World Wide Web communities, under the names merge/purge, tuple deduplication, entity matching (or consolidation), and object matching.

Research on tuple matching has roughly paralleled that of schema matching, but slightly lagged behind in certain aspects. Just as in schema matching, a variety of techniques for tuple matching have been developed, including both rule-based and learning-based approaches. Early solutions employ manually specified rules (Hernandez and Stolfo 1995), while many subsequent ones learn matching rules from training data (Tejada, Knoblock, and Minton 2002; Bilenko and Mooney 2003; Sarawagi and Bhamidipaty 2002). Several solutions focus on efficient techniques to match strings (Monge and Elkan 1996; Gravano et al. 2003). Others also address techniques to scale up to very large number of tuples (McCallum, Nigam, and Ungar 2000; Cohen and Richman 2002). Several recent methods have also heavily used information-retrieval (Cohen 1998; Ananthakrishna, Chaudhuri, and Ganti 2002) and information-theoretic (Andritsos, Miller, and Tsaparas 2004) techniques.

Recently, there have also been some efforts to exploit external information to aid tuple matching. The external information can come from past matching efforts and domain data (for example, see the article by Martin Michalowski, Snehal Thakkar, and Craig Knoblock in this issue of *AI Magazine)*. In addition, many works have considered settings in which there are many tuples to be matched and examined how information can be moved across different matching pairs to improve matching accuracy (Parag and Domingos 2004; Bhattacharya and Getoor 2004).

At the moment, a definitive solution architecture for tuple matching has not yet emerged, although the work by Doan et al. (2003a) proposes a multimodule architecture reminiscent to the multimatcher architecture of schema matching. Indeed, given that tuple matching and schema matching both try to infer semantic relationships on the basis of limited data, the two problems appear quite related, and techniques developed in one area could be transferred to the other. This implication is significant because so far these two active research areas have been developed quite independently of each other.

Finally, we note that some recent works in the database community have gone beyond the problem of matching tuples into matching data fragments in text and semistructured data (Dong et al. 2004; Fang et al. 2004), a topic that has also been receiving increasing attention in the AI community (for example, see the article by Xin Li, Paul Morie, and Dan Roth in this magazine).

Open Research Directions

Matching schemas and data usually constitute only the first step in the semantic-integration process. We now discuss open issues related to this first step as well as to some subsequent important steps that have received little attention.

User Interaction. In many cases, matching tools must interact with the user to arrive at final correct matches. We consider efficient user interaction one of the most important open problems for schema matching. Any practical matching tool must handle this problem, and anecdotal evidence abounds that deployed matching tools are quickly being abandoned because they irritate users with too many questions. Several recent works have only touched on this problem (for example, Yan et al. [2001]). An important challenge here is to minimize user interaction by asking for absolutely necessary feedback, but maximizing the impact of feedback. Another challenge is to generate effective explanations of matches (Dhamankar et al. 2004).

Formal Foundations. In parallel with efforts to build practical matching systems, several recent papers have developed formal semantics of matching and attempted to explain formally what matching tools are doing (for example, Larson, Navathe, and Elmasri [1989]; Biskup and Convent [1986]; Madhavan et al. [2002]; Sheth and Kashyap [1992]; and Kashyap and Sheth [1996]). Formalizing the notion of semantic similarity has also received some attention (Ryutaro, Hideaki, and Shinichi 2001; Lin 1998; Manning and Schütze 1999). Nevertheless, this topic remains underdeveloped. It should deserve more attention, because such formalizations are important for the purposes of evaluating, comparing, and further developing matching solutions.

Industrial-Strength Schema Matching. Can current matching techniques be truly useful in real-world settings? Are we solving the right schema-matching problems? Partly to answer these questions, several recent works seek to evaluate the applicability of schema-matching techniques in the real world. The work by Bernstein et al. (2004) attempts to build an industrial-strength schema-matching environment, while the work by Rahm, Do, and Massmann (2004) focuses on scaling up matching techniques, specifically on matching large XML schemas, which are common in practice. The works by Seligman et al. (2002) and Rosenthal, Seligman, and Renner (2004) examine the difficulties of real-world schema matching, and suggest changes in data management practice that can facilitate the matching process. These efforts should help us understand better the applicability of current research and suggest future directions.

Mapping Maintenance. In dynamic environments, sources often undergo changes in their schemas and data. Hence, it is important to evolve the discovered semantic mappings. A related problem is to detect changes at autonomous data sources (for example, those on the Internet), verify whether the mappings are still correct, and repair them if necessary. Despite the importance of this problem, it has received relatively little attention (Kushmerick 2000; Lerman, Minton, and Knoblock 2003; Velegrakis, Miller, and Popa 2003).

Reasoning with Imprecise Matches on a Large Scale. A large-scale data-integration or peer-to-peer system inevitably involves thousands or hundreds of thousands of semantic mappings. At this scale, it is impossible for humans to verify and maintain all of them to ensure the correctness of the system. How can we use systems in which parts of the mappings always remain unverified and potentially incorrect? In a related problem, it is unrealistic to expect that some day our matching tools will generate only perfect mappings. If we can generate only reasonably good mappings, on a large scale, are they good for any purpose? Note that these problems will be crucial at any large-scale data integration and sharing scenario, such as the semantic web.

Schema Integration. In schema integration, once matches among a set of schemas have been identified, the next step uses the matches to merge the schemas into a global schema (Batini, Lenzerini, and Navathe 1986). A closely related research topic is model management (Bernstein 2003; Rahm and Bernstein 2001). As described earlier, model management creates tools for easily manipulating models of data (for example, data representations, website structures, and ER diagrams). Here matches are used in higher-level operations, such as merging schemas and computing difference of schemas. Several recent works have discussed how to carry out such operations (Pottinger and Bernstein 2003), but they remain very difficult tasks.

Data Translation. In these applications we often must elaborate matches into mappings to enable the translation of queries and data across schemas. (Note that here we follow the terminologies of Rahm and Bernstein [2001] and distinguish between *match* and *mapping,* as described previously.) In figure 2, for example, suppose the two databases that conform to schemas *S* and *T* both store house listings and are managed by two different real estate companies.

Now suppose that the companies have decided to merge. To cut costs, they eliminate database *S* by transferring all house listings from *S* to database *T.* Such data transfer is not possible without knowing the exact semantic mappings between the relational schemas of

the databases, which specify how to create data for *T* from data in *S*. An example mapping, shown in structured query language (SQL) notation, is shown in figure 3.

In general, a variety of approaches have been used to specify semantic mappings (for example, SQL, XQuery, GAV, LAV, and GLAV [Lenzerini 2002]).

Elaborating a semantic match, such as "list-price = price * (1 + fee-rate)," that has been discovered by a matching tool into the above mapping is a difficult problem and has been studied by Yan et al. (2001), who developed the Clio system. How to combine mapping discovery systems such as Clio with schema-matching systems to build a unified and effective solution for finding semantic mappings is an open research problem.

Peer-to-Peer Data Management. An emerging important application class is *peer data management,* which is a natural extension of data integration (Aberer 2003). A peer data management system does away with the notion of mediated schema and allows peers (that is, participating data sources) to query and retrieve data directly from each other. Such querying and data retrieval require the creation of semantic mappings among the peers. Peer data management also raises novel semantic-integration problems such as composing mappings among peers to enable the transfer of data and queries between two peers with no direct mappings and dealing with loss of semantics during the composition process (Etzioni et al. 2003).

```
list-price  =  SELECT price * (1 + fee-rate)
               FROM   HOUSES, AGENTS
               WHERE  agent-id = id
```

Figure 3. An Example Mapping in SQL Notation.

Concluding Remarks

We have briefly surveyed the broad range of semantic-integration research in the database community. This article, and this special issue of *AI Magazine* in general, demonstrate that this research effort is quite related to that in the AI community. It is also becoming clear that semantic integration lies at the heart of many database and AI problems and that addressing it will require solutions that blend database and AI techniques. Developing such solutions can be greatly facilitated with even more effective collaboration between the various communities in the future.

Acknowledgment

We thank Natasha Noy for invaluable comments on the earlier drafts of this article.

References

Aberer, K. 2003. Special Issue on Peer to Peer Data Management. *SIGMOD Record* 32(3).

Ananthakrishna, R.; Chaudhuri, S.; and Ganti, V. 2002. Eliminating Fuzzy Duplicates in Data Warehouses. In *Proceedings of the Twenty-Eighth International Conference on Very Large Databases (VLDB)*. San Francisco: Morgan Kaufmann Publishers.

Andritsos, P.; Miller, R. J.; and Tsaparas, P. 2004. Information-Theoretic Tools for Mining Database Structure from Large Data Sets. In Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Batini, C.; Lenzerini, M.; and Navathe, S. 1986. A Comparative Analysis of Methodologies for Database Schema Integration. *ACM Computing Survey* 18(4):323–364.

Bergamaschi, S.; Castano, S.; Vincini, M.; and Beneven-Tano, D. 2001. Semantic Integration of Heterogeneous Information Sources. *Data and Knowledge Engineering* 36(3):215–249.

Berlin, J., and Motro, A. 2001. Autoplex: Automated Discovery of Content for Virtual Databases. Paper presented at the Sixth International Conference on Cooperative Information Systems (CoopIS '01), Trento, Italy, September 5–7.

Berlin, J., and Motro, A. 2002. Database Schema Matching Using Machine Learning with Feature Selection. In *Proceedings of the Conference on Advanced Information Systems Engineering (Caise)*. Lecture Notes in Computer Science, volume 2348. Berlin: Springer-Verlag.

Bernstein, P. 2003. Applying Model Management to Classical Meta Data Problems. Paper presented at the Conference on Innovative Database Research (CIDR), Asilomar, CA, January 5.

Bernstein, P. A.; Melnik, S.; Petropoulos, M.; and Quix, C. 2004. Industrial-Strength Schema Matching. *SIGMOD Record* 33(4).

Bhattacharya, I., and Getoor, L. 2004. Iterative Record Linkage for Cleaning and Integration. In Proceedings of the Ninth ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery. New York: Association for Computing Machinery.

Bilenko, M., and Mooney, R. 2003. Adaptive Duplicate Detection Using Learnable String Similarity Measures. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: Association for Computing Machinery.

Biskup, J., and Convent, B. 1986. A Formal View Integration Method. In Proceedings of the 1986 ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Castano, S., and Antonellis, V. D. 1999. A Schema

Analysis and Reconciliation Tool Environment. Paper presented at the International Database Engineering and Applications Symposium (Ideas), Montreal, Quebec, August 1–3.

Clifton, C.; Housman, E.; and Rosenthal, A. 1997. Experience with a Combined Approach to Attribute-Matching Across Heterogeneous Databases. In *Proceedings of the Seventh IFIP Working Conference on Data Semantics (DS-7)*. Amsterdam: Elsevier North-Holland.

Cohen, W. 1998. Integration of Heterogeneous Databases Without Common Domains Using Queries Based on Textual Similarity. In Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Cohen, W., and Richman, J. 2002. Learning to Match and Cluster Entity Names. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: Association for Computing Machinery.

Dhamankar, R.; Lee, Y.; Doan, A.; Halevy, A.; and Domin-Gos, P. 2004. Imap: Discovering Complex Matches Between Database Schemas. In Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Do, H., and Rahm, E. 2002. Coma: A System for Flexible Combination of Schema Matching Approaches. In *Proceedings of the Twenty-Eighth International Conference on Very Large Databases (VLDB)*. San Francisco: Morgan Kaufmann Publishers.

Doan, A.; Domingos, P.; and Halevy, A. 2001. Reconciling Schemas of Disparate Data Sources: A Machine Learning Approach. In Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Doan, A.; Lu, Y.; Lee, Y.; and Han, J. 2003a. Object Matching for Data Integration: A Profile-Based Approach. *IEEE Intelligent Systems* 18(5): 54–59.

Doan, A.; Madhavan, J.; Dhamankar, R.; Domingos, P.; and Halevy, A. 2003b. Learning to Match Ontologies on the Semantic Web. *VLDB Journal* 12(4): 303–319.

Dong, X.; Halevy, A.; Nemes, E.; Sigurdsson, S.; and Domingos, P. 2004. Semex: Toward On-The-Fly Personal Information Integration. Paper presented at the Workshop in Information Integration on the Web, Toronto, Ontario, August 30 (http://cips.eas.asu.edu/iiweb-proceedings.html).

Elmagarmid, A., and Pu, C. 1990. Guest Editors' Introduction to the Special Issue on Heterogeneous Databases. *ACM Computing Survey* 22(3): 175–178.

Embley, D.; Jackman, D.; and Xu, L. 2001. Multi-Faceted Exploitation of Metadata for Attribute Match Discovery in Information Integration. Paper presented at the International Workshop on Information Integration on the Web, Rio de Janeiro, Brazil, April 9–11.

Etzioni, O.; Halevy, A.; Doan, A.; Ives, Z.; Madhavan, J.; McDowell, L.; and Tatarinov, I. 2003. Crossing the Structure Chasm. Paper presented at the Conference for Innovative Database Research, Asilomar, CA, January 6.

Fang, H.; Sinha, R.; Wu, W.; Doan, A.; and Zhai, C. 2004. Entity Retrieval Over Structured Data. Technical Report UIUC-CS-2414, Dept. of Computer Science, Univ. of Illinois, Urbana-Champaign.

Freitag, D. 1998. Machine Learning for Information Extraction in Informal Domains. Ph.D. diss. Dept. of Computer Science, Carnegie Mellon University, Pittsburgh, PA.

Friedman, M., and Weld, D. 1997. Efficiently Executing Information-Gathering Plans. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*. San Francisco: Morgan Kaufmann Publishers.

Garcia-Molina, H.; Papakonstantinou, Y.; Quass, D.; Rajaraman, A.; Sagiv, Y.; Ullman, J.; and Widom, J. 1997. The Tsimmis Project: Integration of Heterogeneous Information Sources. *Journal of Intelligent Information Systems* 8(2): 117–132.

Gravano, L.; Ipeirotis, P.; Koudas, N.; and Srivastava, D. 2003. Text Join for Data Cleansing and Integration in an RDBMS. In Proceedings of Nineteenth International IEEE Conference on Data Engineering. Piscataway, NJ: Institute of Electrical and Electronics Engineers.

He, B., and Chang, K. 2003. Statistical Schema Matching Across Web Query Interfaces. In Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

He, B.; Chang, K. C. C.; and Han, J. 2004. Discovering Complex Matchings Across Web Query Interfaces: A Correlation Mining Approach. In Proceedings of the Tenth ACM SIGKDD Conference on Knowledge Discovery and Data Mining. New York: Association for Computing Machinery.

Hernandez, M., and Stolfo, S. 1995. the Merge/Purge Problem for Large Databases. In Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, 127–138. New York: Association for Computing Machinery.

Ives, Z.; Florescu, D.; Friedman, M.; Levy, A.; and Weld, D. 1999. An Adaptive Query Execution System for Data Integration. In Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Kang, J., and Naughton, J. 2003. On Schema Matching with Opaque Column Names and Data Values. In Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Kashyap, V., and Sheth, A. 1996. Semantic and Schematic Similarities Between Database Objects: A Context-Based Approach. *VLDB Journal* 5(4): 276–304.

Keim, G.; Shazeer, N.; Littman, M.; Agarwal, S.; Cheves, C.; Fitzgerald, J.; Grosland, J.; Jiang, F.; Pollard, S.; and Weinmeister, K. 1999. Proverb: The Probabilistic Cruciverbalist. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, 710–717. Menlo Park, CA: AAAI Press

Knoblock, C.; Minton, S.; Ambite, J.; Ashish, N.; Modi, P.; Muslea, I.; Philpot, A.; and Tejada, S. 1998. Modeling Web Sources for Information Integration. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*. Menlo Park, CA: AAAI Press.

Kushmerick, N. 2000. Wrapper Verification. *World Wide Web Journal* 3(2):79–94.

Kushmerick, N.; Weld, D.; and Doorenbos, R. 1997. Wrapper Induction for Information Extraction. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*. San Francisco: Morgan Kaufmann Publishers.

Lambrecht, E.; Kambhampati, S.; and Gnanaprakasam, S. 1999. Optimizing Recursive Information Gathering Plans. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*. San Francisco: Morgan Kaufmann Publishers.

Larson, J. A.; Navathe, S. B.; and Elmasri, R. 1989. A Theory of Attribute Equivalence in Database with Application to Schema Integration. *IEEE Transaction on Software Engineering* 15(4):449–463.

Lenzerini, M. 2002. Data Integration: A Theoretical Perspective. In Proceedings of the Twenty-First ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. New York: Association for Computing Machinery.

Lerman, K.; Minton, S.; and Knoblock, C. A. 2003. Wrapper Maintenance: A Machine Learning Approach. *Journal of Artificial Intelligence Research* 18: 149–181.

Levy, A. Y.; Rajaraman, A.; and Ordille, J. 1996. Querying Heterogeneous Information Sources Using Source Descriptions. In *Proceedings of the Twenty-Second International Conference on Very Large Databases (VLDB)*. San Francisco: Morgan Kaufmann Publishers.

Li, W., and Clifton, C. 2000. Semint: A Tool for Identifying Attribute Correspondence In Heterogeneous Databases Using Neural Networks. *Data and Knowledge Engineering* 33(1): 49–84.

Li, W.; Clifton, C.; and Liu, S. 2000. Database Integration Using Neural Network: Implementation and Experience. *Knowledge and Information Systems* 2(1):73–96.

Lin, D. 1998. An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*. San Francisco: Morgan Kaufmann Publishers.

Madhavan, J.; Bernstein, P.; Doan, A.; and Halevy, A. 2005. Corpus-Based Schema Matching. In Proceedings of the Eighteenth International Conference on Data Engineering (ICDE). Los Alamitos, CA: IEEE Computer Society.

Madhavan, J.; Bernstein, P.; and Rahm, E. 2001. Generic Schema Matching With Cupid. In *Proceedings of the Twenty-Seventh International Conference on Very Large Databases (VLDB)*. San Francisco: Morgan Kaufmann Publishers.

Madhavan, J.; Halevy, A.; Domingos, P.; and Bernstein, P. 2002. Representing and Reasoning About Mappings Between Domain Models. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*. Menlo Park, CA: AAAI Press.

Manning, C., and Schütze, H. 1999. *Foundations of Statistical Natural Language Processing*. Cambridge, MA: MIT Press.

McCallum, A.; Nigam, K.; and Ungar, L. 2000. Efficient Clustering of High-Dimensional Data Sets with Application to Reference Matching. In Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: Association for Computing Machinery

McCann, R.; Doan, A.; Kramnik, A.; and Varadarajan, V. 2003. Building Data Integration Systems Via Mass Collaboration. Paper presented at the Sixth International Workshop on the Web and Databases (WebDB-03), San Diego, June 12–13.

Melnik, S.; Molina-Garcia, H.; and Rahm, E. 2002. Similarity Flooding: A Versatile Graph Matching Algorithm. In Proceedings of the Eighteenth International Conference on Data Engineering (ICDE). Los Alamitos, CA: IEEE Computer Society.

Miller, R.; Haas, L.; and Hernandez, M. 2000. Schema Mapping As Query Discovery. In *Proceedings of the Twenty-Sixth International Conference on Very Large Databases (VLDB)*. San Francisco: Morgan Kaufmann Publishers.

Milo, T., and Zohar, S. 1998. Using Schema Matching to Simplify Heterogeneous Data Translation. In *Proceedings of the Twenty-Fourth International Conference on Very Large Databases (VLDB)*. San Francisco: Morgan Kaufmann Publishers.

Mitra, P.; Wiederhold, G.; and Jannink, J. 1999. Semi-Automatic Integration of Knowledge Sources. In Proceedings of the Second International Conference on Information Fusion (Fusion'99). Los Alamitos, CA: IEEE Computer Society.

Monge, A., and Elkan, C. 1996. The Field Matching Problem: Algorithms and Applications. In *Proceedings of the Second International Conference Knowledge Discovery and Data Mining*. Menlo Park, CA: AAAI Press.

Neumann, F.; Ho, C.; Tian, X.; Haas, L.; and Meggido, N. 2002. Attribute Classification Using Feature Analysis. In Proceedings of the Eighteenth International Conference on Data Engineering (ICDE). Los Alamitos, CA: IEEE Computer Society.

Noy, N., and Musen, M. 2001. Anchor-Prompt: Using Non-Local Context for Semantic Matching. Paper presented at the IJCAI Workshop on Ontologies and Information Sharing, Seattle, August 4–5.

Noy, N., and Musen, M. 2000. Prompt: Algorithm and Tool for Automated Ontology Merging and Alignment. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*. Menlo Park, CA: AAAI Press.

Ouksel, A., and Seth, A. P. 1999. Semantic Interoperability in Global Information Systems. *SIGMOD Record* 28(1): 5–12.

Palopoli, L.; Sacca, D.; Terracina, G.; and Ursino, D. 1999. A Unified Graph-Based Framework for Deriving Nominal Interscheme Properties, Type Conflicts, and Object Cluster Similarities. Paper presented at the Fourth International Conference on Cooperative Information Systems (CoopIS '99), Edinburgh, Scotland, September 2–4.

Palopoli, L.; Sacca, D.; and Ursino, D. 1998. Semi-Au-

tomatic, Semantic Discovery of Properties from Database Schemes. In Proceedings of the International Database Engineering and Applications Symposium (IDEAS-98), 244–253. Los Alamitos, CA: IEEE Computer Society.

Palopoli, L.; Terracina, G.; and Ursino, D. 2000. The System Dike: Towards the Semi-Automatic Synthesis of Cooperative Information Systems and Data Warehouses. In Proceedings of the Symposium on Advances in Databases and Information Systems, Enlarged Fourth East-European Conference on Advances in Databases and Information Systems. New York: Association for Computing Machinery.

Parag, and Domingos, P. 2004. Multi-Relational Record Linkage. Paper presented at the Third SIGKDD Workshop on Multi-Relational Data Mining, Seattle, August 22.

Parent, C., and Spaccapietra, S. 1998. Issues and Approaches of Database Integration. *Communications of the ACM* 41(5):166–178.

Perkowitz, M., and Etzioni, O. 1995. Category Translation: Learning to Understand Information on the Internet. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*. San Francisco: Morgan Kaufmann Publishers.

Pottinger, R. A., and Bernstein, P. A. 2003. Merging Models Based on Given Correspondences. In *Proceedings of the Twenty-Ninth International Conference on Very Large Databases (VLDB)*. San Francisco: Morgan Kaufmann Publishers.

Punyakanok, V., and Roth, D. 2000. The Use of Classifiers in Sequential Inference. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS-00)*. Cambridge, MA: MIT Press.

Rahm, E., and Bernstein, P. 2001. A Survey of Approaches to Automatic Schema Matching. *VLDB Journal* 10(4): 334–350.

Rahm, E.; Do, H.; and Massmann, S. 2004. Matching Large XML Schemas. Special Issue on Semantic Integration, *SIGMOD Record* 33(3).

Rosenthal, A.; Seligman, L.; and Renner, S. 2004. From Semantic Integration to Semantics Management: Case Studies and A Way Forward. Special Issue on Semantic Integration, *SIGMOD Record* 33(3).

Ryutaro, I.; Hideaki, T.; and Shinichi, H. 2001. Rule Induction for Concept Hierarchy Alignment. Paper presented at the Second Workshop on Ontology Learning, Seattle, August 4.

Sarawagi, S., and Bhamidipaty, A. 2002. Interactive Deduplication Using Active Learning. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: Association for Computing Machinery.

Seligman, L.; Rosenthal, A.; Lehner, P.; and Smith, A. 2002. Data Integration: Where Does the Time Go? *IEEE Data Engineering Bulletin* 25(3): 3–10.

Seth, A., and Larson, J. 1990. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Survey* 22(3):183–236.

Sheth, A. P., and Kashyap, V. 1992. So Far (Schematically) Yet So Near (Semantically). In *Proceedings of the IFIP WG 2.6 Database Semantics Conference on Interoperable Database Systems (DS-5)*. Amsterdam: Elsevier North-Holland.

Tejada, S.; Knoblock, C.; and Minton, S. 2002. Learning Domain-Independent String Transformation Weights for High Accuracy Object Identification. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: Association for Computing Machinery.

Velegrakis, Y.; Miller, R. J.; and Popa, L. 2003. Mapping Adaptation Under Evolving Schemas. In *Proceedings of the Twenty-Ninth International Conference on Very Large Databases (VLDB)*. San Francisco: Morgan Kaufmann Publishers.

Wu, W.; Yu, C.; Doan, A.; and Meng, W. 2004. An Interactive Clustering-Based Approach to Integrating Source Query Interfaces on the Deep Web. In Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Xu, L., and Embley, D. 2003. Using Domain Ontologies to Discover Direct and Indirect Matches for Schema Elements. Paper presented at the Semantic Integration Workshop, Second International Semantic Web Conference (ISWC 2003), Sanibel Island, FL, October 20 (http://Smi.Stanford.Edu/Si2003).

Yan, L.; Miller, R.; Haas, L.; and Fagin, R. 2001. Data Driven Understanding and Refinement of Schema Mappings. *SIGMOD Record* 30(2): 485–496

AnHai Doan is an assistant professor in computer science at the University of Illinois, Urbana-Champaign. His interests cover databases and AI, with a current focus on data integration, schema and ontology matching, and machine learning. Selected recent honors include the William Chan Memorial Dissertation Award from the University of Washington, the ACM Dissertation Award in 2003, and the CAREER Award in 2004. Selected recent professional activities include coeditoring a special issue on semantic integration for *SIGMOD Record*. He can be e-mailed at anhai@cs.uiuc.edu.

Alon Halevy is an associate professor of computer science at the University of Washington at Seattle. His research interests span the AI and database communities and address various aspects of information integration and semantic heterogeneity. In particular, he is interested in applying machine learning and knowledge representation to schema mapping and constructing semantic web applications based on peer data management systems. He received his Ph.D. in computer science from Stanford University in 1993 and his Bs.C. in computer science from Hebrew University in 1988. His e-mail address is alon@cs.washington.edu.