Automatically Utilizing Secondary Sources to Align Information Across Sources

Martin Michalowski, Snehal Thakkar, and Craig A. Knoblock

■ XML, web services, and the semantic web have opened the door for new and exciting information-integration applications. Information sources on the web are controlled by different organizations or people, utilize different text formats, and have varying inconsistencies. Therefore, any system that integrates information from different data sources must identify common entities from these sources. Data from many data sources on the web does not contain enough information to link the records accurately using state-of-the-art record-linkage systems. However, it is possible to exploit secondary data sources on the web to improve the record-linkage process.

We present an approach to accurately and automatically match entities from various data sources by utilizing a state-of-the-art record-linkage system in conjunction with a data-integration system. The data-integration system is able to automatically determine which secondary sources need to be queried when linking records from various data sources. In turn, the record-linkage system is then able to utilize this additional information to improve the accuracy of the linkage between datasets.

In the recent past, researchers have developed various machine-learning techniques such as SoftMealy (Hsu and Dung 1998) and Stalker (Muslea, Minton, and Knoblock 2001) to easily extract structured data from various web sources. Using these techniques, users can build wrappers that allow them to easily query web sources much like databases. Web-based information-integration systems such as Information Manifold (Levy, Rajaraman, and Ordille 1996b), InfoMaster (Genesereth, Keller, and Duschka 1997), and Ariadne (Knoblock et al. 2001) can provide a uniform query interface for users to query information from various data sources. Furthermore, schema-matching techniques (Dhamankar et al. 2004; Madhavan and Halevy 2003; Miller, Haas, and Hernandez 2000) allow users to align different schemas from various data sources. While schema-matching techniques are useful for aligning attributes from different schemas, they cannot be utilized to determine if two records obtained from different sources refer to the same entity. Therefore, building an application that integrates data from various web sources requires a record-linkage component in addition to wrapper-generation, information-integration, and schema-matching components. For example, two restaurant web sites may refer to the same address using different textual information. Therefore, accurate record linkage is an essential component of any information-integration system used to integrate data accurately from various data sources.

There has been some work done on consolidating data objects from various web sites using textual similarities and transformations [Bilenko and Mooney 2003; Chaudhuri et al. 2003; Dhamankar et al. 2004; Doan et al. 2003; Tejada, Knoblock, and Minton 2002). These approaches provide better consolidation results compared to the exact text-matching techniques in different application domains. How-

Dinesite	Zagat’s	Matched
Broadway Bar and Grill, 1460 3rdstreet Promenade, Santa Monica, CA 90401-2322, 310.393.4211	Broadway Bar&Grill, 1460 Third St. Promenade, Santa Monica, CA, 90401-2322, (310) 393-4211	✓
Beverly Hills Café, 14 N. LaCienaga Boulvard, Beverly Hills, CA 90211-2205, 310.652.1529	Beverly Hills Samurai, 270 S. La Cienaga, Beverly Hills, CA 90211, (310) 360-9688	INCORRECT

Figure 1. Textual Differences in Restaurant Records.

ever, in some application domains it may be extremely difficult to consolidate records. For example, when matching names of people, it would be hard for the aforementioned techniques to determine if “Robert Smith” and “Bob Smith” refer to the same individual. This problem can often be solved by utilizing information from secondary sources on the web. For example, a web site that lists the common acronyms used for the first name may provide information that “Bob” and “Robert” are interchangeable as first names. There are many other application areas where information from secondary data sources can improve the performance of a record-linkage system. Additional examples include utilizing a geocoder to determine if two addresses are the same, utilizing historical area code changes to determine if two telephone numbers are the same, and utilizing the location and officers’ information for different companies to determine if two companies are the same.

In this article, we describe our approach to exploiting secondary sources automatically for record linkage. The goal of the research is to link records more accurately across data sources. We have built a record-linkage system termed Apollo that can utilize information from secondary sources (Michalowski, Thakkar, and Knoblock 2003). In presenting our approach, we describe how Apollo can be combined with an information mediator to automatically identify and utilize information from secondary sources. We provide a motivating example, followed by some background work on record linkage. Subsequently, we present our approach to utilizing the secondary sources during the record-linkage process and an evaluation of our approach using real-world restaurant and company datasets. Finally, we discuss related work and put forward our conclusions and planned future work.

Motivating Example

To clarify the concepts presented in this article, we define the following terms: (1) record linkage, (2) primary data sources, and (3) secondary data sources. *Record linkage* is the process of determining if two records should be linked across some common property. In this article, this common property will be whether the two records refer to the same real-world entity. A *primary data source* is one of the two initial data sources used for record linkage. A *secondary data source* is any source, other then a primary data source that can provide additional information about entities in the primary data sources.

Consider the following primary data sources: (1) Zagat and Dinesite data sources that provide information about various restaurants; (2) Travelocity and Orbitz data sources that provide information about various hotels; and (3) Yahoo and Moviefone data sources that provide information about various theaters.

When the user sends a request to obtain information pertaining to restaurants within a given city, the record-linkage system needs to link records that refer to the same restaurant from the

Zagat and Dinesite data sources. However, due to the textual inconsistencies present in both data sources, determining which records refer to a common entity is a nontrivial task. A similar situation arises when attempting to combine information about hotels from Travelocity and Orbitz or about movies from Yahoo and Moviefone. Figure 1 shows the varying textual inconsistencies found in the restaurant data sources.

Background Work

In this section, we present the Active Atlas (Tejada, Knoblock, and Minton 2001; Tejada, Knoblock, and Minton 2002) record-linkage system. Its robust and extendable framework makes it an ideal candidate for a base system upon which to build a record-linkage system that utilizes secondary sources. For this reason, Active Atlas is used as a base system upon which Apollo is built.

System Overview

Active Atlas's architecture consists of two separate components: a candidate generator and a mapping learner. The overall architecture of the system can be seen in figure 2. Its goal is to find common entities among two record sets from the same domain. The candidate generator proposes a set of potential matches based on the transformations available to the system. The transformation may be one of a number of string comparison types such as EQUALITY, SUBSTRING, PREFIX, SUFFIX, STEMMING, or others and are weighted equally when computing similarity scores for potential matches. Once the candidate generator has finished proposing potential matches, Active Atlas moves on to the second stage and uses the potential matches as the basis for learning mapping rules and transformation weights.

The mapping learner establishes which of the potential matches are correct by adapting the mapping rules and transformation weights to the specific domain. Due to the fact that the initial similarity scores are very inaccurate, the system uses an active learning approach to refine and improve the transformation weights and mapping rules. This approach uses a decision tree (Quinlan 1996) committee model with three committee members. The key idea behind the committee learning approach is to divide the training data set into three parts and have each committee member learn a decision tree based on that member's respective part. During active learning, the mapping learner then selects the most informative potential example. The most informative potential example is defined as a potential match with the

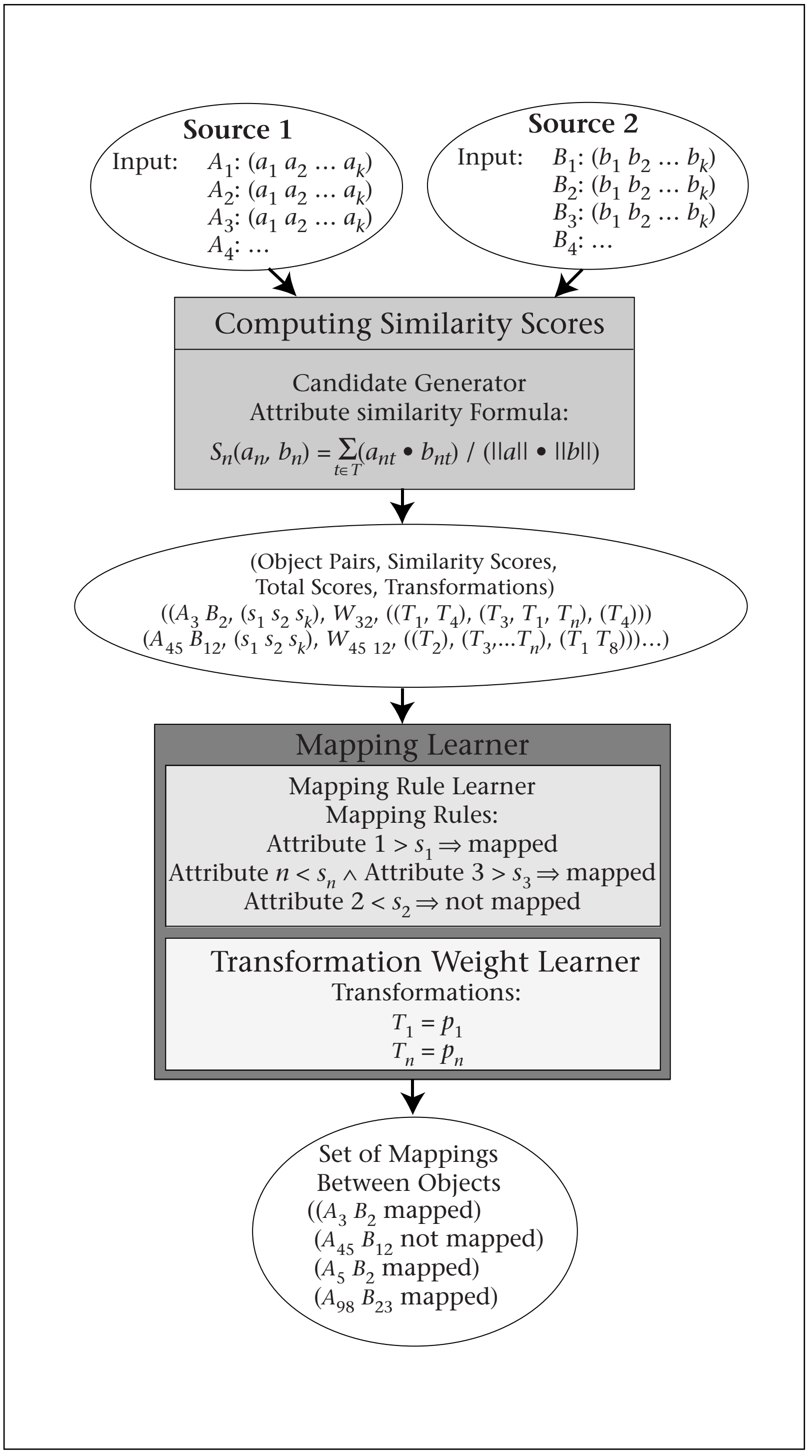

Source 1
Input: A_1: (a_1 a_2 … a_k)
A_2: (a_1 a_2 … a_k)
A_3: (a_1 a_2 … a_k)
A_4: …
Source 2
Input: B_1: (b_1 b_2 … b_k)
B_2: (b_1 b_2 … b_k)
B_3: (b_1 b_2 … b_k)
B_4: …
Computing Similarity Scores
Candidate Generator
Attribute similarity Formula:
$S_n(a_n, b_n) = \sum_{t \in T}(a_{nt} \bullet b_{nt}) / (||a|| \bullet ||b||)$
(Object Pairs, Similarity Scores,
Total Scores, Transformations)
((A_3 B_2, (s_1 s_2 s_k), W_{32}, ((T_1, T_4), (T_3, T_1, T_n), (T_4)))
(A_{45} B_{12}, (s_1 s_2 s_k), $W_{45\ 12}$, ((T_2), (T_3,…T_n), (T_1 T_8)))…)
Mapping Learner
Mapping Rule Learner
Mapping Rules:
Attribute 1 > s_1 ⇒ mapped
Attribute n < s_n ∧ Attribute 3 > s_3 ⇒ mapped
Attribute 2 < s_2 ⇒ not mapped
Transformation Weight Learner
Transformations:
$T_1 = p_1$
$T_n = p_n$
Set of Mappings
Between Objects
((A_3 B_2 mapped)
(A_{45} B_{12} not mapped)
(A_5 B_2 mapped)
(A_{98} B_{23} mapped)

Figure 2. Active Atlas System Architecture.

highest disagreement among the members. It then asks the user to label this example as either a match or nonmatch. The user's response is used to refine and recalculate the transformation weights, learn new mapping rules, and reclassify record pairs. This process continues until (1) the committee learners converge and agree on one decision tree or (2) the user has been asked a predefined number of questions. Once the mapping rules and transformation weights have been learned, Active Atlas uses them to classify all the potential matches in the system as matched or not matched. The results are then made available to the user.

Open Research Problem

A difficult problem encountered when performing record linkage is the degree of certainty with which matches are proposed and rejected. A record-linkage system is only as good as the labeled data it has received and is therefore limited in accuracy with respect to its classification of matches. A record-linkage system is able to classify obvious matches and nonmatches. In our research of record linkage, we have found that there exists a "gray" area in the classification of potential matches. Classifying the matches in this "gray" with a high degree of confidence often requires additional knowledge or information. This is due to the fact that primary sources often lack sufficient information to resolve all ambiguous matches.

These ambiguous matches cannot be classified with full confidence as a match, yet they are similar enough to be considered as potentially matched. This presents the need for a secondary source to help resolve this ambiguity. A secondary source would provide the system with additional information that it could use to help in the classification of the match. The following example helps illustrate the need for secondary sources.

Consider the restaurant domain. Record linkage is performed on two different data sources, each source composed of records referring to a particular restaurant. The system returns all matches; however, there exists one returned match that is similar enough to be classified as matched by the system but that also contains enough inconsistencies across attributes to raise doubt that it is a true match. Manually looking closer at the record, it is determined that the telephone number field is the major source of the inconsistency. With the availability of a secondary source that contains a telephone area code's history, it could be determined that the telephone numbers in question are in fact the same, since one area code is the successor of the other. This information could then be used by a system to confidently classify the record as a true match.

Exploiting Secondary Sources for Record Linkage

In this section, we describe our approach to performing more accurate record linkage by utilizing information from secondary sources. The intuition in this article is to utilize the domain knowledge from a data-integration system to obtain information automatically about available secondary sources and utilize the available secondary sources to improve the record-linkage process. When Apollo needs to link records from two data sources, it first obtains information about available secondary sources. Next, it queries the secondary sources and utilizes the additional information in the record-linkage process.

Which Secondary Sources to Query?

As we described earlier, primary data sources often do not contain enough information to distinguish between two entities purely based on textual similarities. Therefore, it may be valuable to obtain information from secondary data sources. Apollo utilizes the Prometheus mediator (Thakkar, Ambite, and Knoblock 2003) to obtain information pertaining to which secondary data sources should be queried. In this section, we provide a brief overview of the Prometheus mediator and describe how it is utilized in Apollo.

Overview of Data-integration Systems. Various data-integration systems such as TSIMMIS (Garcia-Molina et al. 1995), Information Manifold (Levy, Rajaraman, and Ordille 1996a), InfoMaster (Genesereth, Keller, and Duschka 1997), InfoSleuth (Bayardo et al. 1997), and Ariadne (Knoblock et al. 2001) have been used to provide an integrated view of information from heterogeneous data sources. Traditionally, these systems model data sources in the form of relations. These systems also contain a set of virtual domain relations that the user utilizes to specify the queries to the mediator system. Every data-integration system must specify a set of domain rules to relate the source relations to the domain relations. The user then sends queries to the data-integration system using these domain relations. The data-integration system then reformulates the user query into a set of source queries, executes the source queries, and provides the answer to the user's query.

Utilization in Apollo. Apollo has access to a data-integration system (the Prometheus mediator)

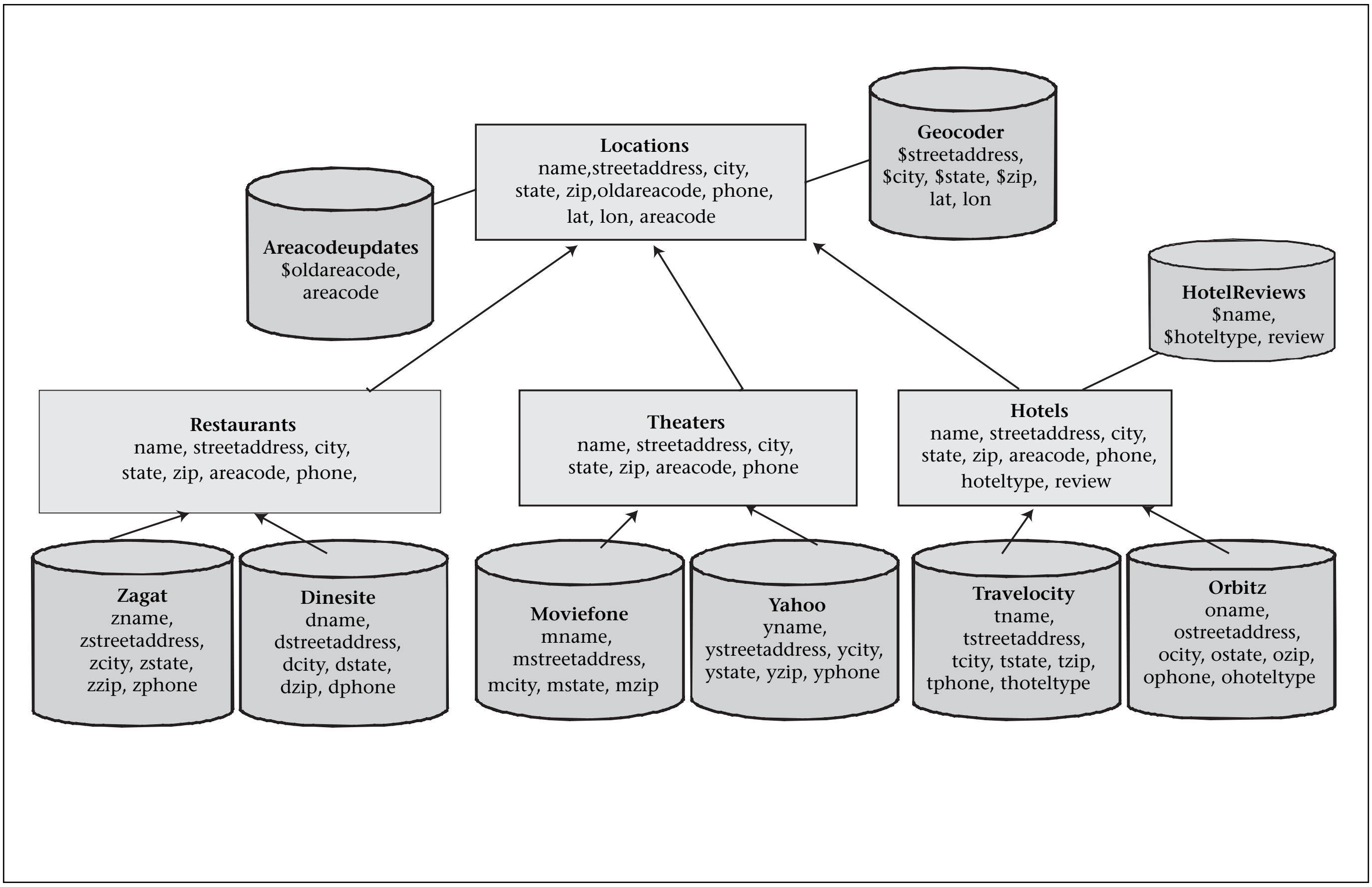


Figure 3. Example Domain Model.

which models various data sources. These data sources include all primary data sources as well as all available secondary data sources. When Apollo receives a request to link records from two data sources, it sends a request to the Prometheus mediator to obtain related data sources.

To further illustrate how additional information is obtained, consider the domain model shown in figure 3. Our example domain model contains various data sources to obtain hotel, restaurant, and theater information. The *Zagat* and *Dinesite* data sources provide information about restaurants, the *Travelocity* and *Orbitz* data sources provide information about hotels, and the *Yahoo* and *Moviefone* data sources provide information about theaters. The data-integration system also has access to a Geocoder data source that provides geographic coordinates for a given address, an area code updates data source that provides information about area code changes, and a hotel review data source that accepts the name of a hotel and a hotel type and provides the review for the hotel.

The data-integration system models the available data sources as source relations. In addition to these source relations, it has a set of domain relations. These domain relations are defined as views over the source relations. In our example, *Locations, Theaters, Hotels,* and *Restaurants* represent domain relations.

Consider the situation where Apollo is attempting to link hotels obtained from the *Orbitz* and *Travelocity* data sources. Apollo sends a request to the Prometheus mediator to obtain all possible secondary sources that are related to the *Orbitz* or *Travelocity* source relations. Upon receiving the request, Prometheus first finds all domain relations that are defined as views over the given sources. In our example, it finds that the domain relation *Hotels* is defined as a join between the *HotelReviews* source relation and the union of the *Orbitz* and *Travelocity* source relations. In the second step, the mediator analyzes the view definition of the domain relations found in the first step to obtain all source relations that participate in the view definition. In the given example, Prometheus finds the *HotelReviews, Orbitz,* and *Travelocity* source relations. Next, it picks the source relations other than the primary source relations as available secondary data sources. From the three source relations, Prometheus picks *HotelReviews* as the available secondary source, as both *Orbitz* and *Travelocity* are primary sources.

Next, the mediator repeats the above-mentioned steps with the domain relation found in the first step. In the given example, Prometheus first finds that the view definition for the *Locations* domain relation includes the *Hotels* domain relation. Moreover, Prometheus finds that the *Locations* domain relation is defined as a join of *Hotels, AreaCodeUpdates,* and *Geocoder* source relations. Therefore, it adds the *AreaCodeUpdates* and *Geocoder* sources to the set of available secondary sources. Prometheus repeats the process with *Locations* as input and finds no qualifying domain relations. Therefore, for the given example, Prometheus returns the *HotelReviews, Areacodeupdates,* and *Geocoder* source relations as available secondary sources. It is important to note that the goal of the mediator is to find all available secondary sources, even the ones that may not be useful. In the next section, we describe how Apollo utilizes the information found in these secondary sources and determines which secondary sources are useful.

How to Utilize Information from Secondary Sources

Current record-linkage systems do a good job of learning how to weigh attributes of different records across data sources. Using machine-learning techniques such as decision trees (Quinlan 1996) and bagging and boosting (Abe and Mamitsuka 1998; Breiman 1996), they are able to determine which attributes are most relevant to consider when trying to match records across different data sources. Apollo takes advantage of this process by augmenting data sources with additional attributes. The machine-learning component of the record-linkage system is then able to use these attributes in learning the correct mapping rules used in the linkage process. Using the labeled examples provided, the system is able to learn whether the newly added attributes are informative enough to incorporate into the mapping rules.

With the flexible nature of the record-linkage framework used in Apollo, incorporating the additional information from secondary sources is an easy and efficient process. A key point needs to be kept in mind when utilizing the information from the secondary sources: When augmenting primary data sources with additional information, it is important not to overload and confuse the record-linkage system. We address this point using a systematic approach to adding additional information discussed below.

Secondary data sources fall into one of two possible categories: (1) one-to-one mapping source, (2) one-to-*N* mapping source. A one-to-one mapping source is a secondary data source that takes an input and produces only one possible answer for each unique data record passed in. The geocoder data source, which takes as input a street address and produces a corresponding latitude and longitude for that address, is a one-to-one mapping source. As we know, each address will have only one latitude (lat) / longitude (lon) combination associated with it. For example, passing the address "4676 Admiralty Way, Marina del Rey, CA 90292" to the source yields the following output:

```
<lat>33.980304877551021</lat>,
<lon>-118.44027018504094 </lon>.
```

A one-to-*N* mapping source is a secondary data source that takes an input and produces multiple possible answers for each unique data record passed in. The area code updates secondary source is an example of such a source because one unique area code may yield multiple past area codes (area codes could have been merged to produce the area code in question). For example, inputting 619 into this source yields the following past area codes: 760, 858, and 935.

It is important to differentiate between the two when deciding how to use the data from the secondary source. Take as an example a one-to-*N* mapping source that returns five unique values for a single input. If the primary sources contain 100 records respectively and are augmented with data from this secondary source, the total number of unique records in each primary source would increase to 500. This increase clouds the real record-linkage problem at hand and would in fact make it harder to solve because of the increased complexity and the lack of a guarantee that the additional information is beneficial. In this work, we address this issue by only using one-to-*N* mapping sources when they are used to augment only one of the primary sources (our approach to choosing which primary source is discussed later). This approach avoids large increases in complexity and the problem introduced by extraneous records in each primary data source. We plan to conduct research on solving the problem of needing to augment both primary data sources with a one-to-*N* mapping source, and this work is discussed in the "Conclusion and Future Work" section. This problem does not exist if there are multiple one-to-one data sources available for a given primary data source, as each one-to-one secondary data source would add one or more attributes to each record from the primary data sources. Therefore, the number of records stays the same.

Another issue that needs to be dealt with is the question of which primary source or sources should be augmented with additional information. Semantic types and attribute bindings are used to make this determination. The Apollo record-linkage system looks at the attribute types found in each primary source and compares them to the output types of the secondary source. If any of the primary data sources already contain an attribute of the same type as the output type of the secondary source, the primary source does not need to be augmented with information from the secondary source. We do not want to add preexisting attributes, because this addition may cause inconsistencies in data for each record in the primary source. These inconsistencies could lead to inaccurate classification of matches between the primary data sources, defeating the purpose of using secondary sources for improving record linkage.

Once the attribute type data information has been added to the primary sources, a binding is made between the added information attribute types in each dataset. For example, if one primary data source contains the latitude and longitude attributes and the second contains a street address but no latitude and longitude attributes, the Apollo system would query the geocoder secondary source using the street address from the second primary source and augment this primary source with the returned data. Once the primary source has been augmented with this data, a binding is made between the latitude and longitude attributes in the two primary data sources and the record-linkage process is run. This step is necessary because the record-linkage system needs an explicit declaration of the mappings of attributes from one primary data source to the other.

As discussed previously, the data-integration component of the Apollo system handles the querying of secondary data sources. Once the data from the secondary source or sources is queried, the incorporation of additional data into one or both of the primary data sources is done. The Apollo system currently does this by appending the additional information to the record as an additional attribute. As mentioned earlier in this section, once an attribute is appended to a primary data source, a binding is created between this attribute and the corresponding attribute in the other primary data source.

Experimental Evaluation

We tested the Apollo system in two real-world domains, restaurants and companies. In the restaurant domain, we used wrapper technology discussed by Muslea, Minton, and Knoblock (2001) to extract restaurant records from the Zagat[1] and Dinesite[2] web sources. Each web source provided a restaurant's name, address, city, state, telephone number, and cuisine type. Because of the inconsistencies between the two sources, a record-linkage system is required to find common restaurants. Furthermore, we used the geocoder[3] web service as a secondary source. This source takes in an address as input and outputs the corresponding latitude and longitude coordinates. The Zagat data source contained 897 records, while the Dinesite data source contained 1,257 records. There were 136 matching records in the two datasets.

As a first step we ran the candidate generator from the Active Atlas system to obtain about 9000 candidate matches. From the candidate matches, we randomly selected 100, 150, 200, and 250 record pairs, labeled them as matches or nonmatches, and used this as input into the Apollo system. The goal of the experiments was to show that by utilizing the geocoder secondary source, the system was able to link records more accurately across the two primary data sources using fewer labeled examples. The restaurant domain experimental results are shown in figures 4 and 5.

As shown in figures 4 and 5, the addition of a secondary source led to a significant improvement in precision and recall. With the use of a secondary source, Apollo reached 100 percent precision and 76 percent recall with only 150 total labeled examples. Out of the 150 labeled examples, there were, on average, only 6 positive examples. In case of 50 labeled examples and 100 labeled examples, there were only 2 and 3 positive examples respectively. Therefore, there was not enough information (from positive-labeled examples) for the decision tree learner to utilize information from secondary sources. Without the secondary source, precision and recall levels were lower, even with 250 total labeled examples, than the levels seen in Apollo with just 150 total labeled examples. The improvement brought about by the secondary source is due to the secondary source's ability to handle inconsistencies for the given attribute better than string transformations.

With 250 training examples, a decrease in recall is caused by the fact that the decision tree learner begins to overfit the training data. In general, decision trees do not have the problem of overfitting when provided with more training examples. However, in our case, the ratio of pos-

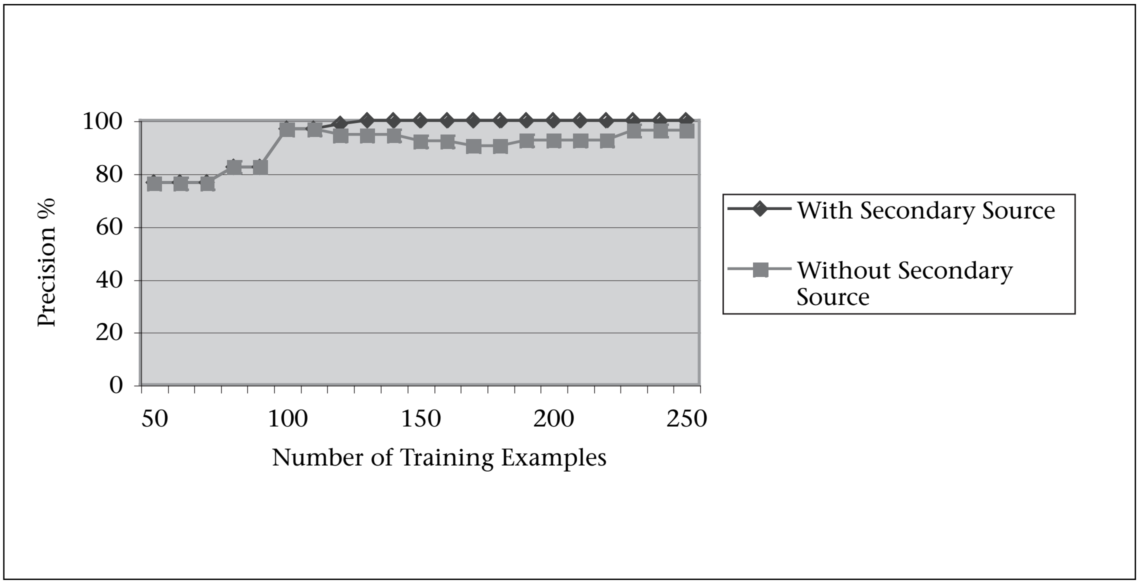


Figure 4. Restaurant Domain Precision Results.

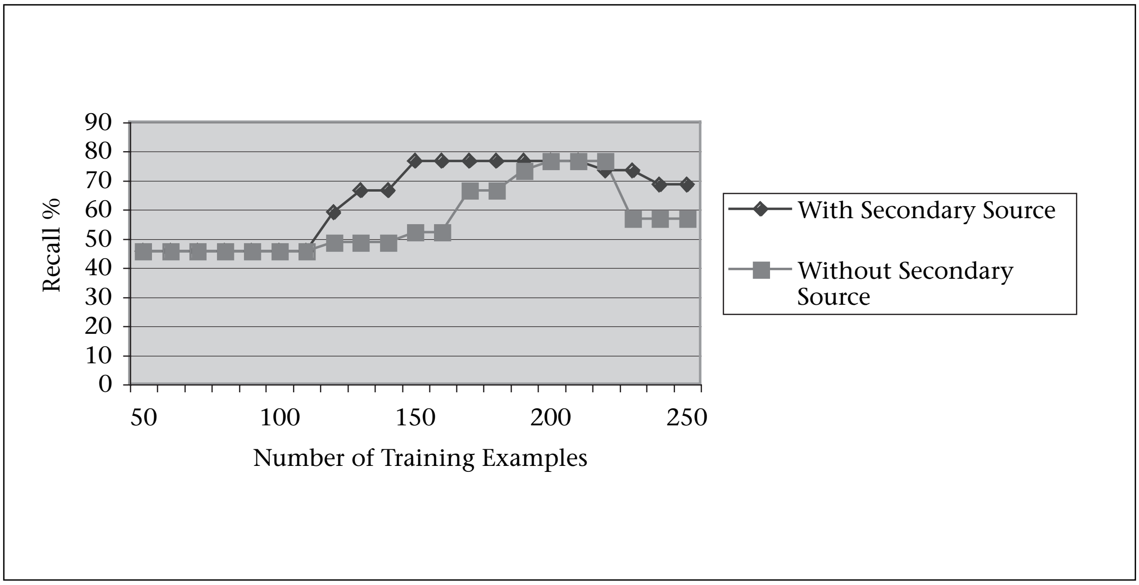


Figure 5. Restaurant Domain Recall Results.

itive to negative examples is very small. Therefore, when we increase the number of training examples, the decision trees learn rules specific to the few positive-labeled examples, causing the decrease in recall. However, we can see that the precision (100 percent) and recall values are still better when using a secondary source.

In the company domain, we extracted company records from two primary data sources: (1) A company database containing company name and ticker symbol for 21,281 companies, and (2) using wrapper technology discussed by Muslea, Minton, and Knoblock (2001) to extract company information, containing company name, person name, and position from various news articles for 8,800 companies. The key challenge with the company domain data was that company name was the only common attribute and companies were referred to using different names in different articles. We used Yahoo Finance as a secondary source, and it provided the company name, ticker symbol, and three top officials for a given company.

As shown in figures 6 and 7, record linkage

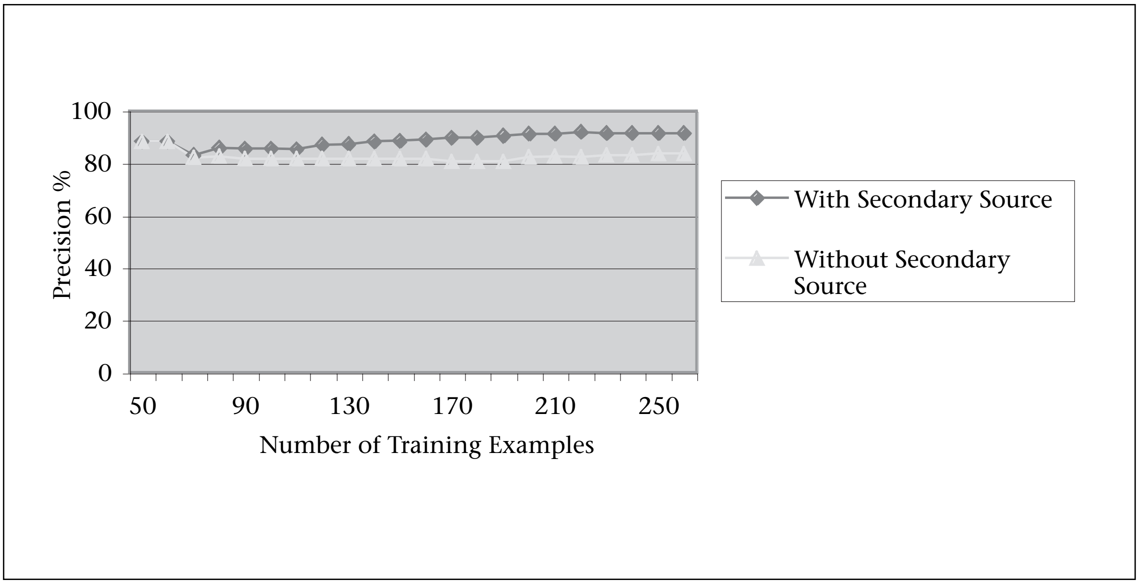


Figure 6. Business Domain Precision Results.

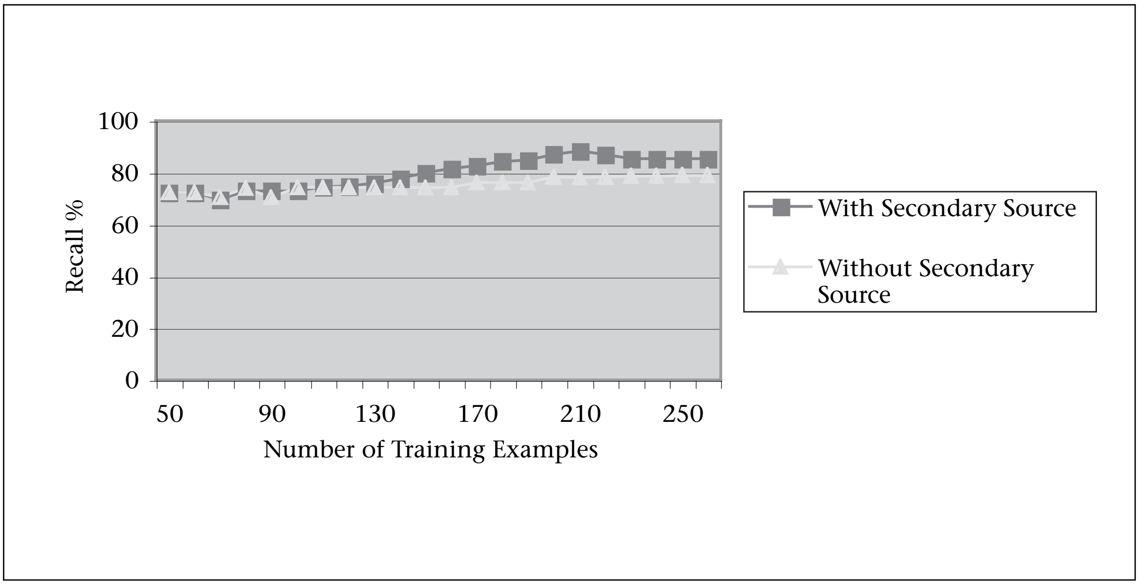


Figure 7. Business Domain Recall Results.

using a secondary source achieves better precision and recall values as it is able to utilize person and company names in the linkage process. It is worth noting that company articles may mention people other than the top three officials, in which case our secondary source is less useful.

Related Work

There has been significant work done on solving the record-linkage problem. We divided the related work into three categories: (1) record linkage (Bilenko and Mooney 2003; Doan et al. 2003; Hernández and Stolfo 1995; Jin, Li, and Mehrotra 2003; Monge and Elkan 1996; Sarawagi and Bhamidipaty 2002; Tejada, Knoblock, and Minton 2002), (2) record linkage for data cleaning (Chaudhuri et al. 2003; Raman and Hellerstein 2001), and (3) schema matching (Dhamankar et al. 2004; Madhavan and Halevy 2003; Miller, Haas, and Hernandez 2000). We describe the most closely related works in each of the aforementioned areas next.

Research in the record-linkage area includes research on entity matching (Doan et al. 2003; Jin, Li, and Mehrotra 2003), object consolidation (Jin, Li, and Mehrotra 2003; Tejada, Knoblock, and Minton 2002), and deduplication (Bilenko and Mooney 2003; Hernández and Stolfo 1995; Monge and Elkan 1996; Sarawagi and Bhamidipaty 2002). The problem of *entity matching* is to determine whether two given records refer to the same real-world entity. *Object consolidation* is defined as the process of merging records from two data sets, such that there is only one record per real-world entity. *Record linkage* is defined as a process of linking two records based on a set of common attributes. All systems utilize some form of textual similarity measures to determine whether two records should be linked. However, none of the systems incorporate the idea of utilizing secondary sources to obtain relevant information and use this information to improve the record-linkage process. Doan et al. (2003) describe a profiler-based approach to improving entity matching. The key idea in the article is to design profilers by mining large amounts of data from different web sources, obtaining input from domain experts, or by examining previously matched entities. The profilers generate rules that determine relationships between various attributes of entities; for example, someone with age 9 is not likely to have a salary of $200,000. This idea is complementary to our approach of utilizing secondary sources to provide additional attributes.

Record linkage has been used for data cleaning by linking records from an unclean dataset to a data source that contains clean records. Chaudhari et al. (2003) describe a fuzzy matching approach for data cleaning in online data catalogs. The goal of their work is to determine the closest records in the reference relation with the given record from a relation containing erroneous records. This work relies on a reference relation to map records from a data source containing inconsistencies or errors to a clean data source. Such an approach is limited by the availability of reference relations. The Potter's wheel system (Raman and Hellerstein 2001) provides a graphical user interface for linking records across two data sources. Potter's wheel generates candidate matches between records from two data sets and presents them to the user. The user can then determine which records are matches or nonmatches. In these systems, Apollo could be utilized to provide additional information about the records, which would assist the user in interactively labeling the records in Potter's wheel or provide additional information when cleaning records in system developed by Chaudhari et al. (2003).

Identifying how one or more attributes of a dataset are related to one or more attributes of another dataset is often referred to as the schema-mapping problem (Dhamankar et al. 2004; Madhavan and Halevy 2003; Miller, Haas, and Hernandez 2000). This is quite different from the record-linkage problem addressed in this article. However, data-integration or data-sharing applications may require modules to solve both schema-mapping and record-linkage problems. The work done in this area could be used in conjunction with the work presented in this article to create a robust and efficient data-integration application.

Conclusion and Future Work

In this article, we presented our approach to utilizing secondary sources to improve the accuracy of record linkage. We showed how our Apollo record-linkage system discovers and utilizes secondary sources. Our experimental evaluation, in two different real-world domains, shows that Apollo reduces the number of labeled examples required as well as improving the precision and recall values for each domain.

In the future we plan to address the issues associated with augmenting both primary data sources with information from one-to-*N* mapping secondary sources. Furthermore, we are researching ways in which we can reduce the number of queries sent to secondary sources, since queries may be expensive or time consuming. Finally, even though the transformations used in Apollo are quite comprehensive, they do not cover all possible sets of transformations. To address this problem, we are working on improving the field (attribute) level matching process. This work applies specific sets of transformations depending on the semantic types of different attributes and leads to more accurate confidence measures for the given attributes.

Acknowledgements

This material is based upon work supported in part by the National Science Foundation under Award No. IIS-0324955, in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under Contract No. NBCHD030010, in part by the Air Force Office of Scientific Research under grant numbers F49620-01-1-0053 and FA9550-04-1-0105, in part by the United States Air Force under contract number F49620-02-C-0103, and in part by a gift from the Microsoft Corporation.

The U.S. government is authorized to reproduce and distribute the authors' original reports for governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of any of the above organizations or any person connected with them.

Notes

1. www.zagats.com

2. www.dinesite.com

3. terraworld.isi.edu/Geocode/Service1.asmx

References

Abe, N.; and Mamitsuka, H. 1998. Query Learning Strategies using Boosting and Bagging. In *Proceedings of the Fifteenth International Conference on Machine Learning*. San Francisco: Morgan Kaufmann Publishers.

Bayardo Jr., R. J.; Bohrer, W.; Brice, R.; Cichocki, A.; Flower, J.; Helal, A.; Kashyap, V.; Ksiezyk, T.; Martin, G.; Nodine, M.; Rashid, M.; Rusinkiewicz, M;. Shea, R.; Unnikrishnan, C.; Unruh, A.; and Woelk, D. 1997. Infosleuth: Agent-based Semantic Integration of Information in Open and Dynamic Environments. In Proceedings of the ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Bilenko, M.; and Mooney, R. J. 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.

Breiman, L. 1996. Bagging Predictors. *Machine Learning* 24(2): 123–140.

Chaudhuri, S.; Ganjam, K.; Ganti, V.; and Motwani, R. 2003. Robust and Efficient Fuzzy Match for Online Data Cleaning. In Proceedings of the ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Dhamankar, R.; Lee, Y.; Doan, A.; Halevy, A.; and Domingos, P. 2004. iMAP: Discovering Complex Semantic Mappings between Database Schemas. In Proceedings of the ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

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

Garcia-Molina, H.; Hammer, J.; Ireland, K.; Papakonstantinou, Y.; Ullman, J.; and Widom, J. 1995. Integrating and Accessing Heterogeneous Information Sources in TSIMMIS. In Information Gathering from Heterogeneous, Distributed Environments: Papers from the AAAI Spring Symposium. Technical Report SS-95-08. Menlo Park, CA: American Association for Artificial Intelligence.

Genesereth, M. R.; Keller, A. M.; and Duschka, O. M. 1997. InfoMaster: An Information Integration System. In Proceedings of the ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Hernández, M. A.; and Stolfo, S. J. 1995. The Merge/Purge Problem for Large Databases. In Proceedings of the ACM SIGMOD International Conference on Management of Data. New York: Association for Computing Machinery.

Hsu, C.-N.; and Dung, M.-T. 1998. Generating Finite-State Transducers for Semistructured Data Extraction from the Web. *Information Systems Journal* 23(8): 521–538.

Jin, L.; Li, C.; and Mehrotra, S. 2003. Efficient Record Linkage in Large Data Sets. In Proceedings of the Eighth International Conference on Database Systems for Advanced Applications (DASFAA 2003). Piscataway, NJ: Institute of Electrical and Electronics Engineers.

Knoblock, C., S.; Minton, J. L.; Ambite, N.; Ashish, N.; Muslea, I.; Philpot, A.; and Tejada, S. 2001. The ARIADNE Approach to Web-Based Information Integration. *International Journal on Intelligent Cooperative Information Systems* 10(1-2): 145–169.

Levy, A. Y.; Rajaraman, A.; and Ordille, J. J. 1996a. Query-Answering Algorithms for Information Agents. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*. Menlo Park, CA: AAAI Press.

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

Madhavan, J.; and Halevy, A. 2003. Composing Mappings among Data Sources. In *Proceedings of the Twenty-Ninth International Conference on Very Large Databases*. San Francisco: Morgan Kaufmann Publishers.

Michalowski, M.; Thakkar, S.; and Knoblock, C. A. 2003. Exploiting Secondary Sources for Automatic Object Consolidation. Paper presented at the First KDD Workshop on Data Cleaning, Record Linkage, and Object Consolidation, Washington, D.C., 27 August.

Miller, R. J.; Haas, L. M.; and Hernández, M. A. 2000. Schema Mapping as Query Discovery. In *Proceedings of the Twenty-Sixth International Conference on Very Large Data Bases*. San Francisco: Morgan Kaufmann Publishers.

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

Muslea, I.; Minton, S.; and Knoblock, C. A. 2001. Hierarchical Wrapper Induction for Semistructured Information Sources. *Autonomous Agents and Multi-Agent Systems* 4(1): 93–114.

Quinlan, J. R. 1996. Improved Use of Continuous Attributes in C4.5. *Journal of Artficial Intelligence Research*. 4: 77–90.

Raman, V.; and Hellerstein, J. M. 2001. Potter's Wheel: An Interactive Data Cleaning System. In *Proceedings of the Twenty-Seventh International Conference on Very Large Data Bases*, 381–390. San Francisco: Morgan Kaufmann Publishers.

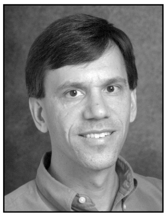
Sarawagi, S.; and Bhamidipaty, A. 2002. Interactive Deduplication Using Active Learning. In Proceedings of the Eighth ACM SIGKDD Conference on Knowledge Discovery and Data Minning. New York: Association for Computing Machinery.

Tejada, S.; Knoblock, C. A.; and Minton, S. 2001. Learning Object Identification Rules for Information Integration. *Information Systems* 26(8): 607–633.

Tejada, S.; Knoblock, C. A.; 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.

Thakkar, S.; Ambite, J. L.; and Knoblock, C. A. 2003. A View Integration Approach to Dynamic Composition of Web Services. Paper presented at the 2003 ICAPS Workshop on Planning for Web Services, Trento, Italy, 10 June.

Martin Michalowski is a doctoral student in the Computer Science Department at the University of Southern California. His research interests include information integration, record linkage, constraint-based integration, heuristic-based planning, and machine learning. He received his B.CompE. in computer engineering from the University of Minnesota in 2001 and M.S. in computer science from the University of Southern California in 2003. He can be reached at martinm@isi.edu; or www.isi.edu/~martinm.



Craig A. Knoblock is a senior project leader at the Information Sciences Institute and a research associate professor in computer science at the University of Southern California. His current research interests include information integration, automated planning, machine learning, and constraint reasoning and the application of these techniques to geospatial and biological data integration. He received his Ph.D. in computer science from Carnegie Mellon University. He is a member of AAAI and ACM. He can be reached at knoblock@isi.edu or www.isi.edu/~knoblock.

Snehal Thakkar is a Ph.D. student in computer science at the University of Southern California. His research interests include information integration, automatic web service composition, and geographical information systems. He received his MS in computer science from the University of Southern California. He can be reached at thakkar@isi.edu or www.isi.edu/~thakkar.