A Platform for Personal Information Management and Integration

Xin (Luna) Dong Alon Halevy

{lunadong, alon}@cs.washington.edu
University of Washington, Seattle

Abstract

The explosion of the amount of information available in digital form has made search a hot research topic for the Information Management Community. While most of the research on search is focussed on the WWW, individual computer users have developed their own vast collections of data on their desktops, and these collections are in critical need for good search tools.

We describe the SEMEX System that offers users a flexible platform for personal information management. SEMEX has two main goals. The first goal is to enable browsing personal information *by semantically meaningful associations*. The challenge it to *automatically* create such associations between data items on one's desktop, and to create enough of them so SEMEX becomes an indispensable tool. Our second goal is to leverage the personal information space we created to increase users' productivity. As our first target, SEMEX leverages the personal information to enable lightweight information integration tasks that are discouragingly difficult to perform with today's tools.

1 Introduction

The explosion of the amount of information available in digital form has made search a hot research topic for the Information Management Community. The *Google Generation* is obsessed (and rightly so!) with finding better ways of searching the vast collection of data available on the WWW. At the same time, thanks to the affordability of large amounts of storage, individual computer users have developed their own vast collections of data on their desktops. However, with the exception of recent keyword-based desktop search tools, most information on our desktop is found by repeatedly traversing directory hierarchies.

As early as 1946, Vannevar Bush [2] described the vision of a Personal Memex, which was motivated by the observation that our mind does not think by way of directory hierarchies, but rather by following *associations* between related objects. For example, we may think of a person, emails sent to us by the person, then jump to thinking of their papers, papers they cited, etc. Today's desktop does not begin to support such associative traversal. Instead, we need to examine directory structures or open specific applications.

We are building the SEMEX System (short for SEMantic EXplorer), that offers users a flexible platform for personal information management. SEMEX has two main goals. Our first goal is to enable browsing personal information by association. The challenge it to *automatically* create associations between data items on one's desktop. While it will never be possible to create all possible associations, our goal is to create enough of them so SEMEX becomes an indispensable tool (in analogy to WWW search engines). Our second goal is to leverage the associations we created to increase users' productivity. In addition to increased productivity offered by a better search tool, we also show how SEMEX leverages the association database to enable lightweight information integration tasks that are discouragingly difficult to perform with today's tools. We consider each of these goals below.

1.1 Browsing By Association

The key impediment to browsing personal information by association is that data on the desktop is stored *by application* and in directory hierarchies, whereas browsing by association requires a *logical view* of the objects on the desktop and the relations between them. As a simple example, information about people is scattered across our email, address book, and text and presentation files. Even answering a simple query, such as finding all of one's co-authors, requires significant work. SEMEX provides a logical view of one's personal information, based on *meaningful* objects and associations. We refer to the instantiation of the logical view as the *association database* or *personal information space*. For example, users of SEMEX can browse their personal information by objects such as Person, Publication and Message and associations such as AuthoredBy, Cites and AttachedTo. Importantly, since users are typically not willing to tolerate any overhead associated with creating additional structure in their personal data, SEMEX attempts to create the association database automatically.

It is impossible to anticipate in advance all the sources of associations between objects in one's personal information. Hence, it is important that SEMEX be extensible in the ways in which associations can be added. SEMEX obtains objects and associations from multiple types of sources. First, some associations are obtained by programs that are specific to particular file types. In the simple case, SEMEX extracts objects and associations from Contacts and email clients (e.g., senders and recipients, phone numbers and email addresses). In more complex cases, SEMEX extracts some associations, e.g., AuthorOf, by analyzing Latex files and Powerpoint presentations. Second, associations can be obtained from external lists or databases (e.g., a list of one's graduate students or departmental colleagues). Finally, complex associations are derived from simple ones (e.g., one's co-authors).

Extracting associations from multiple sources raises one of the important technical challenges for PIM. An association relates two objects in the world, and the objects are represented by references. In order to combine multiple sources of associations and to support effective browsing and querying, SEMEX needs to *reconcile references*, i.e., to decide when two references represent the same object in the world. For example, in the personal data of one author of this paper, there were more than 120 different references to the author. Unlike previous work, the reconciliation problem is exacerbated in our context because each of the references typically contains only little information.

1.2 On-the-fly Data Integration

The second goal of SEMEX is to leverage the association database to increase users' productivity. On-the-fly data integration refers to lightweight data management tasks that require combining information from multiple online sources to achieve a task. As a simple example, suppose the user is considering candidates for a program committee, and wants candidates who have published at the conference, but did not recently serve on its PC. There are disparate sources of data that can help the user in this task, such as a spreadsheet of recent PC members and DBLP, but integrating them with today's tools is tedious at best. With SEMEX, the user would import each one of them into the personal information space, and then be able to query across them and other personal data.

The field of data integration has made substantial progress in recent years, fueled by data sharing opportunities on the WWW, within enterprises and in large scientific projects. Today, data integration projects proceed by identifying needs in an organization, typically focusing on *frequently recurring* queries throughout the organization. As a result, many smaller-scale and more transient data integration tasks that we face on a daily basis are not supported. In particular, integration that involves personal data sources on one's desktop, or in one's laboratory is not supported. The goal of on-the-fly data integration is to fundamentally change the cost-benefit equation associated with integrating data sources. We want to aid non-technical users to easily integrate diverse sources, possibly for transient tasks. The intuition behind our approach is that the association database of SEMEX provides an *anchor* into which we can easily integrate external sources.

SEMEX enables users to easily incorporate new data sources into their association database in several steps. First, SEMEX helps the user *prepare* the data so it can be processed. For example, this may require scraping the data from a web page, file or spreadsheet. Second, SEMEX helps the user establish the semantic relationships between the external data source and the SEMEX domain model. In some cases, this step may involve *extending* the user's personal information model. In the third step, SEMEX imports the data into the personal information space, reconciles references to guarantee high quality import at the instance level. Finally, SEMEX analyzes the imported data to find patterns that may be of interest to the user.

The key insight that enables SEMEX to support these steps easily in comparison to other contexts is that SEMEX leverages the knowledge it has about the domain model and previous activities that the user has performed. In particular, SEMEX leverages previous preparation and mapping activities and the collection of objects and associations it already has in its database. Previously, we have shown how to leverage past schema matching tasks [6, 18]. SEMEX broadens this approach considerably by leveraging other aspects of the information integration process.

1.3 Overarching PIM themes

Before we proceed with the description of SEMEX, we mention several higher-level themes gleaned from our experience with the system, and which we believe will pervade many of the challenges in PIM. First, many of the challenges arise because PIM manages *long-lived* and *evolving* data. In contrast, most data management is used to model database states that capture snapshots of the world. The evolution occurs at the instance level as well as the schema level. So far, the evolution has manifested itself in challenges to querying, reference reconciliation and schema mapping. The second theme is finding the right *granularity* for modeling personal data. It is often possible to model the data at a very fine level. However, since PIM tools are geared toward users who are not necessarily technically savvy, it is important to keep the models as simple as possible. As we continue to investigate this tradeoff, we may find an interesting middle point between the models traditionally used for structured data and

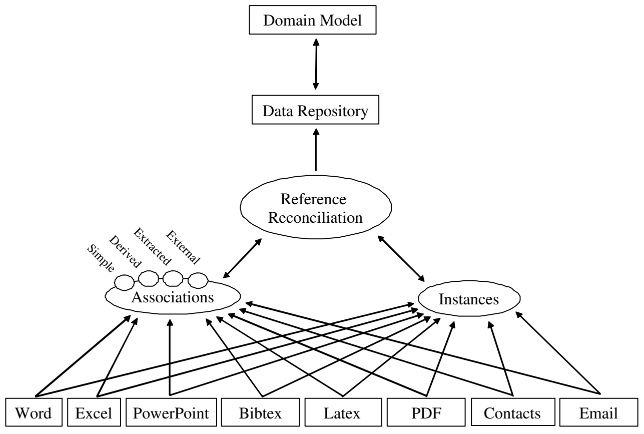


Figure 1: The architecture of SEMEX. SEMEX begins by extracting data from multiple sources. Such extractions create instances of classes in the domain model. SEMEX employs multiple modules for extracting associations, as well as allowing associations to be given by external sources or to be defined as views over other sets of associations. To combine all these associations seamlessly, SEMEX automatically reconciles multiple references to the same real-world object. The user browses and queries all this information through the domain model.

those for unstructured data. Third, when designing PIM systems it is important to think from the perspective of the user and her interactions with data in her daily routine, rather than from the perspective of the database. We need to build systems to support users in their *own* habitat, rather than trying to fit their activities into traditional data management. Finally, there has been a lot of interest in systems that combine structured and unstructured data in a seamless fashion. We believe that PIM is an excellent application to drive the development of such systems, raising challenges concerning storing, modeling and querying hybrid data.

This paper is organized as follows. Section 2 provides an overview of SEMEX and its functionality. Section 3 describes the reference reconciliation algorithm. Section 4 describes how SEMEX helps users import external data sets and perform data integration tasks. Section 5 discusses related work and Section 6 concludes.

2 The Semex System

Personal information management is a topic of growing interest for the data management community [23, 15, 9]. In addition to the challenge of searching our personal information, many challenges arise due to the proliferation of personal electronic devices at our disposal. As a basic goal, we would like our disparate digital devices, be they desktops, laptops, PDAs, cellphones, or GPS systems, to be coordinated with each other. Our data should be consistent across them, updates should propagate seamlessly, and in case one of them fails, it should recover gracefully by communicating with others. Furthermore, when we obtain a new device, it should be able to self-configure and obtain its relevant data. Certainly, we want the devices to be time, location and security aware. In some cases, the vision of PIM extends to ubiquitous computing – we have a multitude of devices in our home, and they should all be able to communicate with each other [9, 13, 15]. From a user's perspective, the system should evolve with its user over time, learn and relearn our preferences. The system should know enough about the user that it can gracefully adapt to new situations. In summary, our goal is to achieve lifelong personal data management.

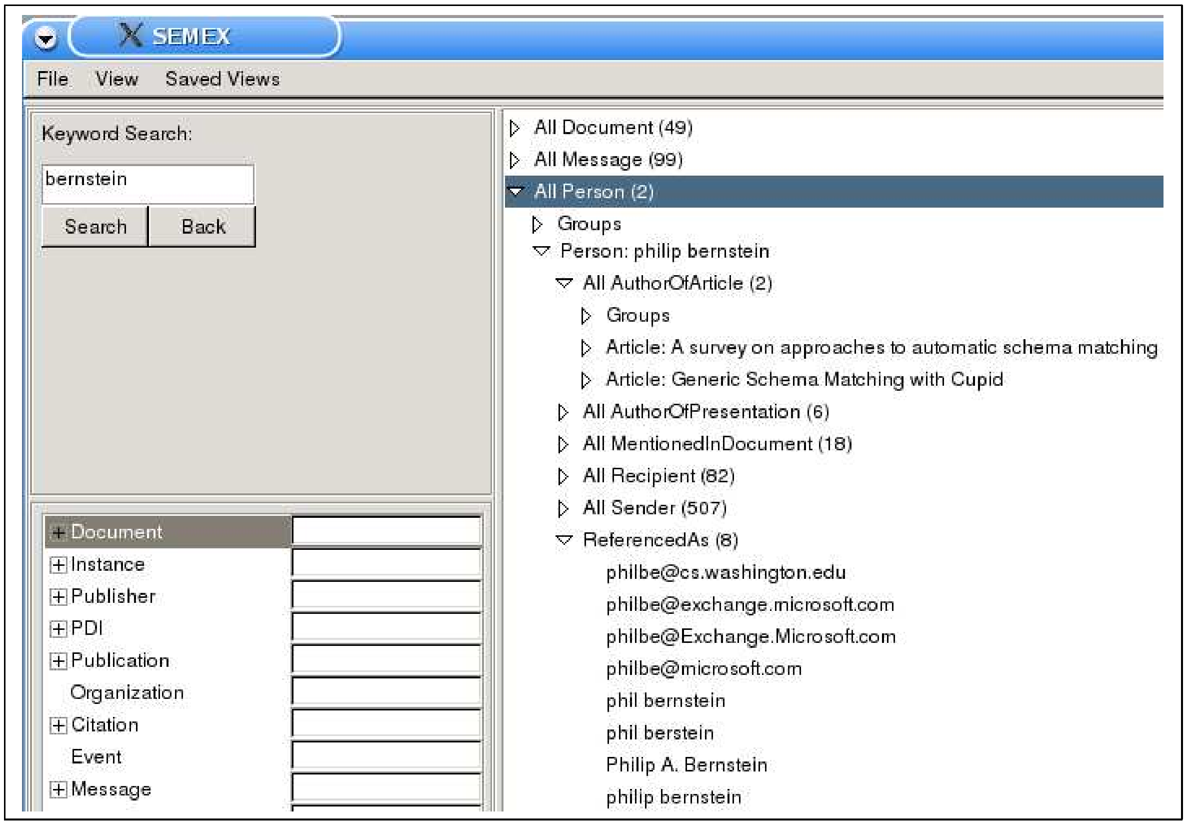
The goal of SEMEX is to raise the level of abstraction involved with personal information management, so we can start considering all the above functionalities. Our initial focus is to enable browsing personal information by meaningful semantic associations. For example, we should be able to browse from a person to their papers, then to the papers cited by those publications, to the authors of the referenced papers, their email addresses, etc. The objects and associations in SEMEX offer a logical view of the data, the association database, which can be used for browsing and for supporting a set of clean interfaces on top of which PIM services can be built.

2.1 System Architecture

The components of SEMEX are shown in Figure 1. SEMEX provides access to data stored in multiple applications and sources, such as emails and address book contacts, pages in the user's web cache, documents (e.g., Latex and Bibtex, PDF, Word, and Powerpoint) in the user's personal or shared file directories, and data in more structured sources (e.g., spreadsheets and databases). SEMEX creates a database of objects and associations between them using a collection of association-extraction tools, and provides access to this information through an interface that supports browsing and querying by association. We now briefly explain each of these components in more detail.

Domain model: SEMEX users and applications interact with the system through a domain model (or, ontology) of personal information. The domain model includes a set of classes such as Person, Publication and Message, and associations such as AuthorOf, Cites, Sender, MentionedIn. At the moment SEMEX uses a simple data model of classes and associations, but there is a clear need for supporting subclasses and sub-associations (e.g., AuthorOf is a sub-association of MentionedIn). In a sense, the domain model of SEMEX can be viewed as a *mediated schema* over the set of personal information sources.

Clearly, one of the important features of a PIM system is that users be able to personalize their domain model. SEMEX comes with a generic domain model



Document
Instance
Publisher
PDI
Publication
Organization
Citation
Event
Message

All Document (49)
All Message (99)
All Person (2)
Groups
Person: philip bernstein
All AuthorOfArticle (2)
Groups
Article: A survey on approaches to automatic schema matching
Article: Generic Schema Matching with Cupid
All AuthorOfPresentation (6)
All MentionedInDocument (18)
All Recipient (82)
All Sender (507)
ReferencedAs (8)
philbe@cs.washington.edu
philbe@exchange.microsoft.com
philbe@Exchange.Microsoft.com
philbe@microsoft.com
phil bernstein
phil berstein
Philip A. Bernstein
philip bernstein

Figure 2: A sample screenshot of the SEMEX interface. The user can formulate either a keyword query (top left) or a more specific selection query (bottom left). SEMEX displays all the information about a particular individual and enables browsing the information by association. As seen in the bottom of the right pane, SEMEX needs to reconcile multiple references to the same real-world object.

and offers several ways of manually personalizing it. For example, SEMEX users can extend their domain model by example: the user begins by recording a specific pattern of browsing through instances in SEMEX. The browsing pattern can, in itself, already define a new class or association in the domain model. Alternatively, the user may refine, modify or generalize the pattern or combine it with other patterns to create the desired class. Ultimately, we would like the system to identify interesting clusters of information, and propose extensions based on them.

Associations and instances: The key architectural premise in SEMEX is that it should support a variety of mechanisms for obtaining object and association instances. We describe the main ones as below.

1. *Simple:* In many cases, objects and associations are already stored conveniently in the data sources and they only need to be extracted into the domain model. For example, a contact list already contains several important attributes of persons.
2. *Extracted:* A rich set of objects and associations can be extracted by analyzing specific file formats. For example, authors can be extracted from Latex files and Powerpoint presentations, and citations can be computed from the combination of Latex and Bibtex files.
3. *External:* External sources can explicitly define many associations. For example, if CiteSeer were to publish a web interface, one could extract citation associations directly from there. Alternatively, a professor may wish to create a class MyGradStudents and populate the class with data in a department database.
4. *Defined:* In the same way as views define interesting relations in a database, we can define objects and associations from simpler ones. As simple examples, we can define the association coAuthor, or the concept emailFromFamily.

In each of these cases, associations are obtained by some piece of code. We expect that in practice, the domain model of SEMEX will be extended depending on the availability of code for obtaining associations (or defining complex associations from base ones) in a plug-and-play fashion. Users should be able to incorporate association extractors as they see fit.

Reference reconciliation: Since the data we manage in PIM is very heterogeneous and we need to support multiple sources of associations, it is crucial that the data instances mesh together seamlessly. This leads to one of the key technical challenges in PIM:

reference reconciliation. To truly follow chains of associations and find all the information about a particular individual (or publication, conference, etc.), SEMEX needs to be able to reconcile the many references to the same real-world object. Section 3 describes the reference reconciliation in SEMEX.

The result of the reconciliation algorithm is a high-quality reference list of a set of objects (e.g., people, publications). Except for enabling seamless querying and browsing, SEMEX also leverages this list to extract additional associations. Specifically, we search for occurrences of the names in the reference list in email bodies, spreadsheets, Word and PDF files to create associations such as MentionedIn or isAbout.

Data repository: SEMEX stores the association database in a separate database. While we have not implemented any sophisticated incremental update mechanisms yet, we envision a module that periodically updates the database, so it is transparent to the user.

2.2 Browsing and Querying in Semex

The first and most obvious benefit of the domain model is the ability to *find* information in one's personal data. SEMEX offers its users an interface that combines intuitive browsing and a range of query facilities (see Figure 2).

The user can search her personal data in three ways. First, she can perform keyword search where SEMEX will return all objects that are somehow mentioning the keyword. Note that the answer to such a query can be heterogeneous—it may return sets of objects from different classes, but, importantly, SEMEX already classifies these objects into their classes (Person, Publication, etc). Second, the user may already choose a class in the domain model (e.g., Organization) and enter specific values for some of its attributes, thereby specifying a selection query (see bottom left of Figure 2). Third, SEMEX supports a more sophisticated query interface, through which the user can create *association queries*. An association query is a conjunctive query over triplets, each triplet describing an association between a pair of objects (this is not shown in Figure 2 to avoid clutter).

Given the returned set of objects, the user can browse the data by following the association links, much like web browsing. When a particular object is selected, the user can see *all* of its associated objects, grouped by associations. As examples of possible navigations via associations, consider the following, keeping in mind how difficult it would be to answer these queries with current information organization.

Example 2.1 *Suppose the user wants to search for all the publications authored by Bernstein in the user's personal data. She first types the keyword* `Bernstein`. SEMEX *returns a set of persons, messages, and documents related to Bernstein. When the user selects a particular person object,* SEMEX *presents all objects associated with Bernstein, categorized into papers authored by him, messages sent to him, etc. The user can simply browse the category* AuthorOfArticle *to find Bernstein's publications.*

As an example of a more complex search, suppose the user is trying to find a specific reference to insert in a paper. She does not remember the title or authors of the paper, but does remember that she used this reference in a previous paper that also cited a paper by Bernstein. Having found the papers by Bernstein, she can find which ones were cited in her papers by following the CitedBy *association. Then, she can determine which was the previous paper of hers that cited both, and follow the* Cites *association to find the reference in question. Finally, a more sophisticated user may choose not to follow this browsing chain and formulate an association query directly.* □

The navigation discussed thus far allows the user to inspect only one object at a time and see the objects related to it; however, searches in personal information (and elsewhere) often require examining a *set* of objects. For example, the user may want to find all of her co-authors. She can select all of her publications, but then she needs to follow the AuthorOf association for each publication in isolation. SEMEX offers a novel feature that tightens the interconnection of browsing and querying. When a particular object is selected, SEMEX creates links that represent chains of length two from the object. For example, starting from the object corresponding to herself, and navigating via the association AllAuthorOfArticle, the user will see a set of Author associations, corresponding to all authors of her papers, i.e., her co-authors.

3 Reference Reconciliation in Semex

This section describes our preliminary work on reconciling references in SEMEX. Reference reconciliation is a crucial element in order for all the objects and associations in SEMEX to mesh seamlessly together and for importing and integrating external data sources. As an example, the following references were obtained from contacts, email and Bibtex data. In order to fuse all the information about Mike Carey, SEMEX must realize they all refer to the same person.

name, phone	:	Mike Carey, (123)456 − 7890
email	:	carey@almaden.ibm.com
name	:	M. Carey

Reference reconciliation has been the subject of several recent research efforts (see [1] for a recent survey); however, previous work falls short in our context. Broadly speaking, earlier approaches focus on reconciling tuple references from a single database table. Hence, they typically make the following assumptions:

(1) all the references share the same set of attributes, (2) each attribute has a single value, and (3) references have a reasonable number of attributes (not necessarily many, but certainly not one or two).

In our context, these assumptions do not hold. First, the data sources in SEMEX are heterogeneous, thereby containing different sets of attributes. In the above example, the attributes of the first and the second references even do not overlap. Second, attributes may have multiple values; e.g., it is common for persons to have multiple email addresses. Third, each reference may contain very limited information; i.e., have only one or two attributes.

In addition to the aforementioned difficulties, we cannot apply the supervised learning methods in previous work, since we cannot assume the existence of training data suitable for users from all different organizations and areas. Nor can we rely on statistical analysis of large collections of references [4], given the relatively small size of personal data.

Section 3.1 explains how SEMEX addresses these challenges in the context of reconciling references to persons, and shows its results on an initial experiment. Reconciling person references is the hardest reconciliation problem we face, and impacts the user's experience most. We then briefly consider two additional challenges to reference reconciliation that arise in the PIM context. Section 3.2 describes how SEMEX needs to reconcile references to objects in *multiple* classes (e.g., in addition to Person, also Publication, Conferences, and Institutions). Finally, Section 3.3 discusses how to reconcile references to objects that *evolve* over time, which is common in the PIM context.

We use the following terminology in our discussion. A *reference* is one of several representations of some *object* in the domain; it consists of a set of *attributes*. The attributes in a reference are a subset of the attributes associated with the class in the domain model. The specific set of attributes in a reference varies depending on the data source from which we extract the attributes. Some attributes contain a set of values (e.g., author of a Publication), and some may actually be references to other objects. For example, the author of Publication is a set of references to Person instances. These embedded references will be important in reconciling references of multiple classes, as we describe in Section 3.2.

Classes in our domain model may be associated with one or more keys. Each key is a set of attributes that uniquely define an object in that class (though recall that an object may have multiple values for a key). For example, for the class Person, email address is a key and so is the combination of first and last name.[1]

[1]While the second key is not perfect, it is typically not violated in personal information data sets. Furthermore, we can often detect false positives for this case based on other information, but we do not discuss the details here.

3.1 Reconciling Person References

Traditionally, the reference reconciliation problem was solved by considering only independent *pairwise* decisions. In the case of persons, and in personal information management in general, each single reference is rather weak (i.e., contains few attributes). Hence, the key idea underlying our algorithm is to gradually *enrich* the references when they are matched with others. The enriched references contain more information about the domain object, thereby enabling more sophisticated matching decisions. Specifically, an enriched reference to an object includes a *set* of values for each of its attributes, rather than a single value. For example, we may have both "AT&T Labs" and "AT&T Shannon Laboratories" as possible values for the name attribute of an Institute, or may have multiple spellings of someone's last name in an enriched Person reference. The values of mutli-valued attributes are grouped: e.g., for a Publication, we may have multiple references to each of its authors, but the references will be grouped as well as possible so each group refers to a single author.

The reference reconciliation algorithm takes a set of references as input. Steps 1 and 2 of the algorithm merge the input references into richer ones, and step 3 exploits the information in the richer references to perform further reconciliations.

Step 1: Reconciling based on shared keys: The first step merges two references r_1 and r_2 if they share a value on a key. Specifically, there should be a key k such that for each attribute a of k, the values that r_1 has for a intersect with the values that r_2 has for a.

Step 2: Reconciling based on string similarity: The second step merges two references r_1 and r_2 based on string similarity. Specifically, when on each of their common attributes a, one of a's values in r_1 has high string similarity with one of a's values in r_2, we merge r_1 and r_2. As in [1], we employ edit distance to measure string similarity. In addition, we employ domain dependent heuristics to compare certain strings. For example, we compare email addresses by exploiting knowledge of the different components of the address and recognizing certain mail software idiosyncrasies. In the case of phone numbers, we allow for missing area codes or extension numbers.

Step 3: Applying global knowledge: Now that we have grouped multiple references into richer ones, we can make additional merging decisions based on analysis of the entire references. We give two important examples of such global knowledge, but one can imagine other matching heuristics as well. We note that in all stages of the algorithm, we opt for conservative decisions, as we consider avoiding false positives more important to guarantee quality browsing of personal information.

- *Time-series comparison:* The time-series analyzer selects pairs of references that were judged similar

	Count	%
Messages	18037	—
Contacts	240	—
Files	7085	100%
Latex	582	9.4%
Bibtex	25	0.4%
PDF	97	1.6%
PostScript	668	10.8%
Plain text	51	0.8%
Rich text	31	0.5%
HTML/XML	666	10.8%
Word	400	6.5%
PowerPoint	777	12.6%
Excel	55	0.9%
Multimedia	539	8.7%
Archives	475	7.7%
Other	1809	29.3%

Table 1: The characteristics of our experimental data set. The personal dataset is made up of messages, contacts, and different types of file documents. The first column shows for each type the number of items, and the second column, when applies, shows its fraction of the number of files.

	Count before reconciliation	%
Instances	23318	100%
Person	5014	22%
Message	17322	74%
Document	805	3%
Publication	177	1%
Associations	38318	100%
senderOf	17316	45%
recipientOf	20530	54%
authorOf	472	1%

Table 2: The number of instances extracted from the raw data for classes in the domain model. For example, after scanning all the sources, we have 5014 person references that need to be reconciled.

in the previous passes, but not combined. It then collects for each reference a set of time stamps of the email messages associated with the reference. If the time series have little or no overlap, the references are merged. This heuristic works well for detecting people who move from one institution to another.

- *Search-engine analysis:* Our search-engine analyzer feeds texts from a pair of references into the Google search engine, and compares the top hits. Two references to the same person object tend to obtain similar top hits.

3.1.1 Preliminary experiments

We describe the results of experiments applied to a personal data set of one author of this paper[2]. The data set spans six years of activities and consists of the usual variety of personal data (though probably more Latex files than typical computer users). Table 1 details the characteristics of the raw data, and Table 2 shows the number of instances extracted from the raw data for several of the classes in the domain model.

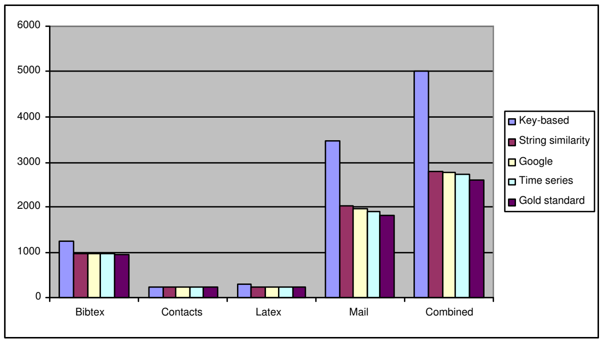


Figure 3: This figure shows the progress of the reference reconciliation algorithm w.r.t. its different steps. The right-most set of bars concerns the entire data set, while the other sets consider individual components of the data set. In a set, each of the first four bars shows the number of references after a pass of reconciliation, and the rightmost bar indicates the actual number of distinct objects.

We limit the following discussion to person instances. Figure 3 shows the results of the reference reconciliation algorithm *within* each of the component data sets in isolation (i.e., for Bibtex, contacts, email, latex), and then the results for all these components combined. The rightmost column (*gold standard*) in each group indicates the *actual* number of distinct objects in the domain. The other columns report the numbers of references after each reconciliation step.

We observe that the first two steps of the algorithm find 91% of the reconciliations in the raw data (i.e., the difference between the resulting number of references and the gold standard is reduced by 91%). The time-series and search-engine analyzers successively remove an additional 1.7% of the beginning total of extra references each, but more importantly, these correspond to 18% and 29% of the references that still need to be reconciled. We also observed that changing the order of the time-series and search-engine analyzers does not change the results substantially.

In summary, while we clearly need to perform more comprehensive experiments and tune our algorithm further, our experiments suggest that the technique of growing rich references holds quite a bit of promise. The results of the reconciliation already yield a reasonable browsing experience on this particular data set.

3.2 Reconciling Objects of Different Classes

The second challenge that arises in the context of PIM is that we must reconcile objects of multiple classes. Conceivably, we could apply the algorithm described

[2]To further complicate matters, this author changed his name from Levy to Halevy a few years ago.

in the previous section, with appropriate class-specific heuristics, to other classes in our domain model. However, applying the algorithm to each class in isolation would miss the many interactions between objects in different classes. For example, having merged several different references to the same Institution will help reconcile references to Person, and in turn, help reconcile references to Publication.

A complete description of our multi-class reconciliation algorithm is beyond the scope of this paper. We illustrate the algorithm with the following example, where we reconcile references to persons, publications, institutions and publication venues.

paper *pa*1=(*"Conflict Detection Tradeoffs for Replicated Data", "703-746",* {pe1,pe2},c1)
*pa*2=(*"Conflict detection tradeoffs for replicated data", "703-746",* {pe3,pe4},c2)
person *pe*1=(*"M.J.Carey",* null, i1)
*pe*2=(*"M.Livny",* null, i1)
*pe*3=(*"Mike Carey", "carey@cs.wisc.edu",* i2)
*pe*4=(*"Miron Livny", "miron@cs.wisc.edu",* i2)
inst *i*1=(*"Univ. of Wisconsin Madison",* null)
*i*2=(*"WISC", "1210 W. Dayton St., Madison, WI 53706-1685"*)
conf *c*1=(*"ACM Transactions on Database Systems", "1991"*)
*c*2=(*"TODS", "1991"*)

The first phase of the algorithm applies reconciliation to each class in isolation. In our example, suppose that this phase decides to merge the two journal references, c_1 and c_2. The first phase also identifies a set of *candidate pairs*: pairs of references that *might* still be reconciled if the algorithm gets more evidence.

In the second phase, Semex creates a *dependency graph*. There is a node in the graph for every merged pair and every candidate pair of references for each class in the domain model. Each node is associated with a similarity score (between 0 and 1), computed in Phase 1 (see Figure 4). An edge from node m to node n in the graph implies that when we recompute the similarity for m, we should reconsider the similarity for n.

Given the dependency graph, Semex may proceed as follows. The fact that $c1$ and $c2$ are merged triggers the merging of the publication references $pa1$ and $pa2$. Given this decision, Semex may decide to merge the person references in the author lists of $pa1$ and $pa2$, i.e., to merge $pe1$ with $pe3$ and $pe2$ with $pe4$. Finally, based on the merging of person references, Semex may decide to merge the institution references $i1$ and $i2$.

The idea of exploiting influences between multiple reconciliation decisions has been considered in a series of works in the AI community [24, 19, 22]. Unlike our work that considers objects of different classes, these works consider reconciling tuples in a single table; however, some of them delegate the matching of attributes to a separate matching decision (e.g., when the tuples

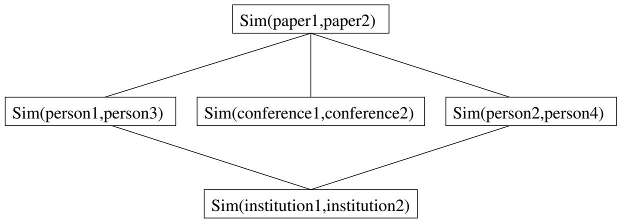


Figure 4: Dependency graph for Example 3.1. The graph indicates the dependency of the similarity between the papers and those between its authors and between its conferences, and also the dependency of the similarity between the persons and that between the institutes they belong to.

being reconciled correspond to publications, they separate the matching of author names as separate decisions). The main difference between our algorithm and theirs is that they learn from training data a detailed probabilistic model of the domain, and base the *entire* reconciliation process on the probabilistic model. In contrast, our approach employs for each class in the domain model the reconciliation algorithm tuned to that particular class. Our algorithm exploits dependencies between reconciliation decisions only to decide when to trigger similarity recomputation. Hence, we can apply any domain specific heuristics that we deem effective, which is extremely important in practice. In addition, we cannot assume the existence of training data, and thus cannot create a detailed probabilistic model of the domain.

3.3 Reconciling Evolving Objects

Unlike traditional reference reconciliation that considers a snapshot of a particular database, PIM involves managing rich objects that persist and evolve over a long period of time. For example, a publication goes through many versions in its lifecycle: its title and authors may change over time, not to speak of its contents. The same happens with projects, software, presentations, and other objects managed in PIM. Taken to the extreme, in some cases it is not even clear when we should model a set of data as a single object or multiple ones. In principle, we can often refine our model to circumvent this problem, e.g., we can explicitly model the different versions of a publication. However, we would like to avoid overly complex models in PIM for the sake of usability.

In Semex we have started addressing this problem as follows. We distinguish between fine-grained reconciliation, which is performed by the algorithms described thus far, and coarse-grained reconciliation. In the coarse-grained reconciliation we consider candidate pairs resulting from the fine-grained phase, but look for clues that imply the involution of one object from the other. This analysis is based on examining the global knowledge we have of the reference. As an illustration, consider reconciling publications. After applying the fine-grained reconciliation algorithm, Semex finds a set of Publication objects with similar titles and authors, but possibly distinct values for other attributes (e.g., associated with different conferences). Semex then analyzes the time stamps associated with the objects to see whether there is some obvious temporal progression. The results of coarse-grained reconciliation are presented differently to the user: first, Semex informs the user that several objects are being shown as one; second, Semex may show the user only the reference to the most recent object, and let the user explore previous ones on demand.

4 On-the-fly Integration with Semex

So far we described how Semex builds a logical view of all the information on the user's desktop which, in effect, entailed *integrating* data from multiple sources. One of the main objectives of Semex is to leverage this association database to enable other tasks more easily. In particular, Semex uses the association database as an *anchor* to help integrating external data sources, and from which to generate data in other formats. Such an anchor offers a basis on which to facilitate a wide range of data integration tasks. We refer to them as *on-the-fly integration*, because they are light-weight tasks performed by individuals for relatively transient goals. We begin with an example that illustrates on-the-fly integration with Semex.

Example 4.1 *Suppose the user is forming a program committee for a conference, and she needs to select people who recently published at the conference, but did not serve on its program committee in the last two years. There are disparate sources of data that can help her in this task, such as a spreadsheet of recent PC members and DBLP. Even with today's tools, combining the information from these sources is tedious, and therefore typically done manually, if at all.*

With Semex*, the user would proceed as follows. She begins by importing each of these data sets into her personal information space. For example, she would import the relevant subset of the DBLP data as a set of instances and associations into the* Semex *database. She can choose to either import this data into her association database or mark it as a temporary data set that can be deleted later. Similarly, she imports the PC member spreadsheet. At that point, she can pose queries, such as finding all the people who have published at the conference but did not serve on the PC in the last two years. Moreover, she can intersect the result with her association database to see which of these people are her co-authors or have interacted with her on other program committees. Finally, once she created the list of people for the PC, she can export the data into a spreadsheet that includes their email addresses and institutions for future use.* □

The key building block for on-the-fly integration is Semex's support for importing external data sources. Semex provides tools to guide the user in the import process with the following steps:

1. **Data preparation and wrapping**: Transform the data into a structured form that Semex can manipulate further. For example, this may involve extracting data from a web site into XML, or reading the cells of a spreadsheet.
2. **Schema mapping**: Establish the semantic relationships between the data source and the Semex domain model. The result is a set of query expressions that enable importing the external data into Semex.
3. **Data import**: Given the schema mapping, import data from the external data sources. In this step Semex reconciles references so all the associations mesh seamlessly together.
4. **Data analysis and summarization:** In this optional step Semex analyzes the imported data and tries to find patterns that may be of interest to the user. For example, when importing a spreadsheet of people, Semex may find that a large fraction of them are the user's co-authors or overlap with other sources the user has integrated in the past.

Each of the above steps has received significant attention in the literature, and is known to be rather challenging in itself. However, we argue that because we are integrating sources into the personal information space, Semex offers several benefits that make the process easier. First, the user is already familiar with the domain model of Semex. This considerably reduces the user's effort during an import process. While in some cases the user may need to extend the domain model of Semex to accommodate new concepts from the data source, these extensions will typically be minor. Second and more importantly, the system is familiar with Semex's domain model and already has extensive experience with it, which can be leveraged to facilitate the import process. Specifically, Semex can leverage several kinds of previous experience and background knowledge:

1. **Previous mappings:** Previous work on schema mapping [6, 5, 18] has shown how to leverage past mapping activities. In fact, it is easier to leverage past mappings when one of the schemas (in our case, the domain model of Semex) is fixed in all the mapping activities.
2. **Previous wrapping tasks:** Many of the external data sources the user may integrate have been used previously by herself or by others in the same department or community. Hence, Semex can leverage previous wrapping tasks [16].
3. **Data-level background knowledge:** The Semex database contains a large number of instances and associations. When examining an external information source, Semex can find overlap

between the data in the source and in its database. Such overlap can be used to suggest the appropriate way to import the data [25, 17]. For example, if we encounter a known title in an external XML file, we can identify that it is a publication and recognize that the `conf` tag corresponds to the inConference association.

4. **Related domain models:** Domain models of individual users are likely to be related, as they may evolve from common roots. Consequently, past activities of one user can benefit colleagues. Moreover, when a user needs to extend her domain model to accommodate a new source, SEMEX can leverage other domain models to propose appropriate extensions.

4.1 Mapping external data sources

In what follows we focus on one component of the import process, namely SEMEX's schema mapping algorithm. To appreciate the novel challenges involved, we first explain why the rich previous body of work in this area does not directly apply to SEMEX.

4.1.1 Previous work

Previous work divided schema mapping into two separate steps. In the first, often called *schema matching*, we find correspondences between the attributes of the two schemas. In the second, referred to as *query discovery* [21], we build on the correspondences and create mapping expressions that specify how to translate data from one schema to another. The vast majority of previous work on schema matching [28, 5, 20, 18, 12, 14, 17] has only considered the case where both schemas have a *single* table (or XML DTD). When translated to relational terms, the classes and associations of SEMEX's domain model correspond to multiple tables. Hence, the matching phase cannot assume a single table. In practice, this means that the matching phase will yield a much larger number of candidate correspondences. For example, many classes have an attribute similar to *name,* each of which can be a match to a column of a spreadsheet with names in it. Furthermore, this means that we need to blur the distinction between the two phases of schema mapping. The second phase involves both creating the mapping expressions and pruning many of the candidate matches produced in the first phase. In contrast to previous work, the query discovery phase cannot assume a given and correct set of correspondences.

4.1.2 Schema mapping algorithm

We illustrate SEMEX's schema mapping algorithm with an example of importing data from a spreadsheet listing the PC members of major database conferences into the domain model. The columns of the spreadsheet are name, conf, year and comment. The different steps of the algorithm are shown in Figure 5. In our description, we distinguish two kinds of binary relationships in the domain model (in the spirit of E/R diagrams): attributes and associations. Attributes relate objects with simple values, and associations relate between pairs of objects.

Step 1: Attribute matching: SEMEX first finds correspondences between attributes in the source schema and attributes in the domain model. In addition to the usual techniques employed for finding correspondences, which include leveraging previous matching tasks, SEMEX also exploits additional clues. First, SEMEX considers the names of the classes attached to the attributes. For example, the spreadsheet has a column named conf, but the domain model does not have an attribute with a similar name. However, the class Conference in the domain model hints at the correspondence between conf and some attribute in the Conference class. Second, SEMEX considers knowledge of overlapping data. In the example, SEMEX observes that several instances of the conf column occur as instances of attribute name of class Conference. Hence, SEMEX proposes to map conf to the attribute name of Conference. Similarly, the comment column in the spreadsheet is matched to the track attribute of the Committee class, because they have overlapping values (e.g., "research", "industry"). The result of this step is a set of triples of the form (a, c, p), denoting that the attribute a in spreadsheet maps to the attribute p of the class c in the domain model. Figure 5(a) shows the correspondences proposed by SEMEX.

Step 2: Class discovery: In this step, the algorithm identifies candidate combinations of classes in the domain model (with their associated attributes) that will be populated by a row of data in the external source. For example, Figure 5(b) shows that a row of the spreadsheet will populate instances of Person, Conference and Committee, whereas Figure 5(c) shows an alternative combination. Note that the associations between these instances will only be determined in the third step. The output of Step 2 is a set of *class combinations*, each of which consists of a bag of classes to populate, each associated with a set of triples output by Step 1.

The algorithm begins by considering all subsets of triples proposed in Step 1, and prunes some of these subsets based on the background knowledge provided by SEMEX and previous experience. In our example, the attribute year in the spreadsheet can be matched to birthyear of Person, year of Conference, or both. The name attribute in the spreadsheet can map to person name, conference name or institution name. Since it is unlikely that we have a person's birth date without his name, we can rule out certain subsets. More generally, we prune subsets that contain at least one class whose non-nullable associations do not match any of the attributes in the external source. The classes in-

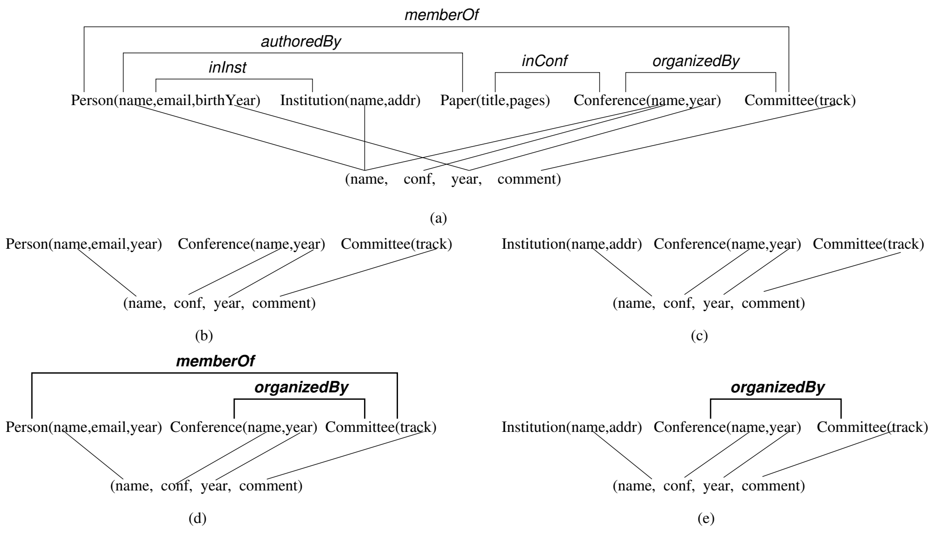


Figure 5: An example schema matching scenario. Step 1: (a) The domain model of SEMEX and candidate correspondences with the columns of the spreadsheet. Step 2: (b and c) show the possible class combinations: the set of objects that will be populated by a row in the spreadsheet. Step 3: (d and e) show candidate mappings proposed by the algorithm. In this step the algorithm discovers the possible associations between objects in the class combinations, and thereby further prunes the number of candidate mappings.

volved in a subset of triples form the class combination associated with that subset.

Step 3: Association discovery: To produce a candidate schema mapping, the third step proposes the associations between class instances in the combinations proposed in Step 2. For example, in this step the algorithm will propose that the instance of Committee is related to that of Person by the association memberOf, and related to that of Conference by the association organizedBy. Two alternative mappings, corresponding to the class combinations in Figure 5(b) and (c) respectively, are shown in Figure 5(d) and (e).

The algorithm considers all the possible associations that relate classes in a given class combination, and prunes proposed mappings that violate integrity constraints. Finally, it uses a set of common heuristics to order the candidate mappings. For example, the mapping in Figure 5(e) is deemed less likely based on the heuristic that a class is usually *not* isolated from all the other classes in the combination. The result of this step is a set of proposed mappings that the user can refine, provide feedback on, or accept.

Finally, we note that in some cases the algorithm may not find a good mapping because the domain model is missing some required classes or associations. In this case, SEMEX will guide the user in extending the domain model. We do not discuss the details here.

5 Related Work

A number of PIM projects studied ways to effectively organize and search information. They all attempt to go beyond the traditional hierarchical directory model of storing data and present a unified user interface for personal data.

The LifeStreams Project [10] organizes documents in chronological order and allows the user to view the documents from different viewpoints in terms of time. The Stuff I've Seen Project (SIS) [8] emphasizes access through text search which is independent of the application storing the data. It indexes all types of information and provides a unique full-text keyword-search interface. Placeless Documents [7] annotates documents with property/value pairs, and groups documents into overlapping collections according to the property value. It also enables annotation with executable code in the form of active properties. These properties were later leveraged in the Haystack Project [26] to create *dynamic hierarchies*, where users can use the properties in an arbitrary order to narrow down the search space (in contrast to following a fixed order in the traditional directory model).

The Haystack [27] and MyLifeBits [11] projects view personal data as a graph of information. Nodes in the graph represent documents and annotation metadata; edges represent the annotates relationship. MyLifeBits focuses on integrating text and multimedia objects, allowing to annotate a file by linking it to another file, and by manually adding text annotation or audio annotation. Haystack, which emphasizes extensive personalization, supports more refined annotations, such as author or type of a document. Annotations in Haystack come from three sources, from the user, from automatic extraction from certain fields of documents (e.g., *to, from, subject* in email messages), and from observing the user's behaviors, such as browsing trails.

SEMEX's database can also be viewed as a graph, but it is guided by a schema: a set of classes representing real-world entities and associations that represent meaningful relationships between the classes. These relationships are more fine-grained than those supported by Haystack or MyLifeBits, and they play a key role in browsing the personal information space and integrating external information sources. SEMEX puts much more emphasis on extracting associations from multiple sources and on ensuring that they all mesh together seamlessly.

6 Conclusions

We described SEMEX, a system that offers a concrete path for data management researchers to address the challenges involved in PIM. The first key idea in SEMEX is to automatically construct a database of instances and associations from the information on one's machine. The immediate benefit of this database is to enable querying and browsing personal information by association, in the spirit of the Memex vision [2]. In addition, this database can support tasks such as coordination between multiple personal devices and context-aware behaviors. The second key idea in SEMEX is that the personal information space can be used as an anchor for importing external information sources, thereby offering an environment for performing on-the-fly integration tasks. An initial version of SEMEX is currently being integrated into the CALO Project [3], a broader program developing cognitive assistants that learn their users' behavior over time.

Acknowledgements

We thank Jayant Madhavan, Ema Nemes, Pedro Domingos and Stebbi Sigurdsson for many useful discussions concerning SEMEX. The research was funded by NSF CAREER Grant #9985114, NSF ITR Grant #0205635, and DARPA Contract #03-000225.

References

[1] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg. Adaptive name matching in information integration. *IEEE Intelligent Systems Special Issue on Information Integration on the Web*, September 2003.

[2] V. Bush. As we may think. *The Atlantic Monthly*, July 1945.

[3] http://www.ai.sri.com/project/CALO, 2004.

[4] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani. Robust and efficient fuzzy match for online data cleaning. In *Proc. of SIGMOD*, 2003.

[5] H.-H. Do and E. Rahm. COMA - A System for Flexible Combination of Schema Matching Approaches. In *Proc. of VLDB*, 2002.

[6] A. Doan, P. Domingos, and A. Halevy. Reconciling schemas of disparate data sources: a machine learning approach. In *Proc. of SIGMOD*, 2001.

[7] P. Dourish, W. K. Edwards, A. LaMarca, J. Lamping, K. Petersen, M. Salisbury, D. B. Terry, and J. Thornton. Extending document management systems with user-specific active properties. *ACM TOIS*, 18(2), 2000.

[8] S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. C. Robbins. Stuff i've seen: A system for personal information retrieval and re-use. In *SIGIR*, 2003.

[9] M. Franklin, M. Cherniack, and S. Zdonik. Data management for pervasive computing: A tutorial. Tutorial at the 2001 VLDB Conference, 2001.

[10] E. Freeman and D. Gelernter. Lifestreams: a storage model for personal data. *SIGMOD Bulletin*, 1996.

[11] J. Gemmell, G. Bell, R. Lueder, S. Drucker, and C. Wong. Mylifebits: Fulfilling the memex vision. In *ACM Multimedia*, 2002.

[12] B. He and K. C.-C. Chang. Statistical schema integration across the deep web. In *Proc. of SIGMOD*, 2003.

[13] Z. Ives, A. Levy, J. Madhavan, R. Pottinger, S. Saroiu, I. Tatarinov, E. Jaslikowska, and T. Yeung. Self-organizing data sharing communities with SAGRES: System demonstration. In *Proc. of SIGMOD*, page 582, 2000.

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

[15] M. Kersten, G. Weikum, M. Franklin, D. Keim, A. Buchmann, and S. Chaudhuri. Panel: A database striptease, or how to manage your personal databases. In *Proc. of VLDB*, 2003.

[16] N. Kushmerick, R. Doorenbos, and D. Weld. Wrapper induction for information extraction. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, 1997.

[17] Y. Lee, A. Doan, R. Dhamankar, A. Y. Halevy, and P. Domingos. imap: Discovering complex mappings between database schemas. In *Proc. of SIGMOD*, pages 383–394, 2004.

[18] J. Madhavan, P. Bernstein, A. Doan, and A. Halevy. Corpus-basd schema matching. In *Proc. of ICDE*, 2005.

[19] A. McCallum and B. Wellner. Toward conditional models of identity uncertainty with application to proper noun coreference. In *IJCAI*, 2003.

[20] S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity Flooding: A Versatile Graph Matching Algorithm. In *Proc. of ICDE*, 2002.

[21] R. J. Miller, L. M. Haas, and M. A. Hernandez. Schema mapping as query discovery. In *VLDB*, 2000.

[22] Parag and P. Domingos. Multi-relational record linkage. In *Proceedings of the Third Workshop on Multi-Relational Data Mining*, 2004.

[23] T. L. Participants. The lowell database research self assessment. *CoRR cs.DB/0310006*, 2003.

[24] H. Pasula, B. Marthi, B. Milch, S. Russell, and I. Shpitser. Identity uncertainty and citation matching. In *NIPS*, 2002.

[25] M. Perkowitz and O. Etzioni. Category translation: Learning to understand information on the internet. In *Proc. of IJCAI*, 1995.

[26] D. Quan, K. Bakshi, D. Huynh, and D. R. Karger. User interfaces for supporting multiple categorization. In *INTERACT*, 2003.

[27] D. Quan, D. Huynh, and D. R. Karger. Haystack: A platform for authoring end user semantic web applications. In *ISWC*, 2003.

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