Data Integration, Entity Reconciliation and Search in Personal Information Networks

Srivatsa Iyengar
IIT Bombay

Apurva Jadhav
Yahoo! Bangalore

Soumen Chakrabarti*
IIT Bombay

Vikas Kedia
IIT Bombay

ABSTRACT

We present SPIN, a system for Searching Personal Information Networks. SPIN seeks to integrate entities and relations from a wide variety of data sources such as address books, emails, documents on file systems, blogs, messenger sessions, social networks and the Web into the PINDB: a universal graph format that has typed nodes, typed edges, and text associated with the nodes. Using the latest techniques for information extraction and deduplication, SPIN adds "soft edges" to the PINDB that correspond to entity aliases and mentions. SPIN's search algorithms then use this representation to respond to a hierarchy of useful semi-structured query classes with proximity, aggregation and ranking semantics that improves substantially beyond simple desktop search tools. SPIN is a self-contained Java application and has a modular architecture where other researchers can plug in new approaches to data integration and search.

1. INTRODUCTION

Storage systems have become dramatically affordable over the last few years, enabling large-scale archival of emails, contacts and documents on personal computers. Large personal storage can be used effectively only if efficient data integration and search tools are available. Recent operating systems can index hard disks and enable keyword search over many file types. Web-based email services include keyword search support. Many Web search companies offer desktop applications that can index and search the file system, emails and browser caches. Many research prototypes exist as well [11, 9, 1, 2].

Most products and prototypes have stayed close to traditional IR: they lack the capability to discover and represent entities and relations from the indexed data as first-class objects or reconcile entities from diverse and heterogeneous sources. Meanwhile, there is a need for extensive evaluation of the cutting-edge research literature on data integration [10, 8], named-entity tagging [16, 12], relation extraction [3], record linkage, deduplication [15, 9] and graph search [5, 4] in large-scale applications.

Our proposed system SPIN (acronym for Searching Personal Information Networks) is a platform where such mining and searching ideas can be tested on real personal data, and is at the same time a powerful personal information management and search application. Our goals overlap with those of the RADAR [2], CALO [1] and IRIS [6] projects, but SPIN is also concerned with efficient data representation, indexing and update issues, as well as integrating advanced semistructured search with the information and graph mining components.

*Contact author soumen@cse.iitb.ac.in

2. SPIN ARCHITECTURE

2.1 PINDB and PINSchema

A **PINDB** consists of typed entities (nodes) and relations (edges) between these entities [5, 4]. Entities can be persons, organizations, places, events, projects, trips, software, subscriptions and other artifacts. These are extracted from mentions in textual and semistructured sources, such as address books, documents and email. PIN edges represent relations. Some are "hard" edges explicitly found in the data, e.g., person *sent* email or email *is-reply-to* email. Others are "soft" or probabilistic edges induced through information extraction, e.g., person *wrote* paper or email *mentions* person. Yet other soft edges are created by reconciling aliases. The **PINSchema** is a concise and accurate structural summary of the PIN graph. It is a graph among node-types connected through edge-types. All data in the PINDB and PINSchema is indexed using Lucene and stored in a Berkeley DB database.

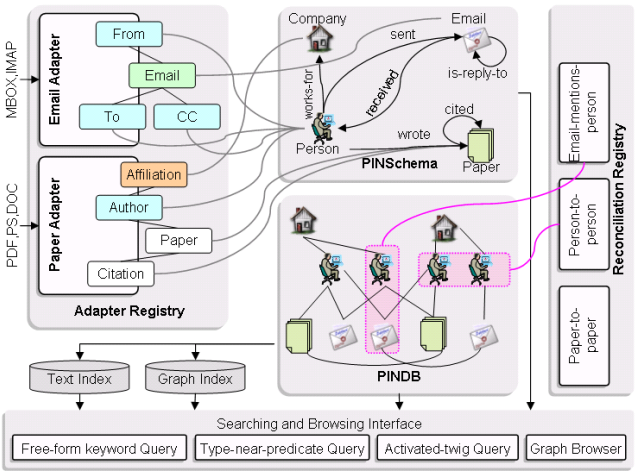

Figure 1: SPIN architecture.

2.2 Adapters

Adapters are responsible for populating the PINDB from various data sources conforming to the PINSchema. SPIN currently has adapters for emails, address books, research papers, and directory services. These adapters readily plug into an adapter registry that tells each of them how to embed graph fragments extracted from its source into the PINDB.

SPIN wraps APIs published by Web search companies with query generators that use PINDB context intelligently. E.g., although there are many Tom Mitchells mentioned on the Web, a PINDB context comprising emails and research papers can enable the search adapter to pad "Tom Mitchell" with other words ("machine learning" and "brain imaging")

to disambiguate the base entity. In future, SPIN will also incorporate trainable Web extractors which can then be used to quickly augment the PINDB with contact information, additional papers, etc.

We are also adding adapters for popular social networks like LinkedIn (`http://www.linkedin.com`) and Orkut (`http://www.orkut.com`), which will let us further augment the PIN graph. As the Semantic Web vision matures, the Web itself will start looking more like a gigantic PINDB. Many of the search paradigms that we will discuss next will extend naturally to the external Web.

2.3 Reconcilers

A single real-world entity (e.g. person) may be represented by multiple nodes in the PINDB because they were extracted from different sources (e.g., email, paper, Web page) using different adapters. At this stage, **reconcilers** take over: using machine learning techniques [15, 9] for alias resolution, the reconcilers introduce *soft edges* into the PINDB. One kind of soft edge connects aliases—nodes that are likely to refer to the same real-world entity. Another kind of soft edge connects mentions of entities to entities, e.g., an edge from an email to a person if the email body is likely to have mentioned that person.

Reconcilers aggregate information about an entity that would otherwise remain scattered in the PINDB, and it improves search accuracy. In activation search (discussed later), scores distributed across alias nodes reinforce each other after reconciliation, and activation from an attribute of one alias can increase the score of another alias.

We use a graphical model for reconciliation [15]. A node in the model represents either a pair of records or a pair of attributes. Edges connect the record pair node to the corresponding attribute pair nodes. E.g., to reconcile persons, we use person name and affiliation as attributes. We iteratively train the model using a voted perceptron [7]. In each iteration we actively select record pairs similar to ones wrongly labeled in the previous iteration and add them to the training set. While inferencing, we first use a canopy [13] to prune out highly dissimilar record pairs. Then we use the graphical model we learnt to infer duplicate pairs using a mincut algorithm [15].

On a collection of research papers collected from a user's file system, SPIN's reconciler obtained an F1 score of about 0.77 on the task of reconciling person nodes. Unlike research prototypes, SPIN is designed to accept continual corrective input from the user and integrate these into its reconciliation system. Some examples of the benefits of reconciliation can be found in Section 4.

3. SEARCHING AND BROWSING

3.1 The 3-tier query paradigm

SPIN includes a proximity-assisted, type-sensitive query system. We provide a hierarchy of query paradigms: **free-form** keyword queries which require the same low cognitive load as Web queries, **type-near** queries that activate entities of a specified type with match predicates, and **twig** queries for users who are ready to exploit the PINSchema. Depending on the information need, even seasoned users find all the query paradigms handy.

The **type-near** paradigm is particularly powerful given its simplicity. E.g., the user can look for a student who graduated in 2001 and went to work at IBM using the query `type=person NEAR org=IBM year=2001`. However, seasoned users can exploit more sophisticated predicates after `NEAR`, involving shallow PINSchema knowledge [14]. In SPIN, every entity is a set of `(field,value)` pairs. E.g., an email has fields `From`, `To`, `Subject` and `Body`. Assisted by the PINSchema, the user can visually edit a query whose string representation might look like `type=org NEAR combine((From OR Subject)= "John Matthews", Email=ANY)`, which will seek organizations near emails that contain "John Matthews" in the `From` or `Subject` fields. In most cases, SPIN autoprompts type and field names.

Twig queries are similar to their XML counterpart in that they (visually) specify a small schema skeleton, such as `email` ←*sent*– `person` –*works-for*→ `org`, but the twig is activated by other predicates as in type-near queries. Twig queries are useful while searching for events, e.g. "find an email about XML sent by someone working in Germany."

3.2 Browsing search results

The response to a SPIN query is an ordered list of nodes or twigs. In Web search, text snippets associated with each URL are very valuable for user relevance judgment. In SPIN, a "snippet" is really a graph fragment that explains the large score of the response nodes or twigs. A great deal of care is needed to maintain a display of a screen-sized portion of graph contexts as "snippets." As the user clicks around in the response fragments, SPIN continually computes the worth of displaying nodes on (precious) screen area, and evicts unnecessary nodes unobtrusively. Screenshots of SPIN can be found at `http://www.cse.iitb.ac.in/~soumen/tmp/kdd2006spin`

4. INITIAL EXPERIENCE

We compared SPIN with commercial desktop search systems by running similar queries on all systems, with similar data. E.g., we ran the query `type=paper NEAR ("HITS")`, hoping to retrieve Kleinberg's famous HITS paper. The results obtained are summarized in the following table. Ranking produced by Google Desktop does not seem to exploit the linkage structure of citations. SPIN nails the best document right at the top of the list.

SPIN
Authoritative sources in a hyperlinked environment
Improved algorithms for topic distillation in a hyperlinked...
Focused crawling a new approach to topic specific web...
Mining sequential patterns
Learning to probabilistically identify authoritative documents
Google Desktop
Not all hits are created equal cooperative proxy caching...
A unified framework for web link analysis
Link fusion, a unified link analysis framework for multitype...
Entropy based link analysis for mining web informative...
Learning to create customized authority lists

Different papers may list the same author slightly differently, as with *Jon Kleinberg* and *J. M. Kleinberg* in the sample graph in Figure 2. These start out as different nodes in SPIN because they were imported from different papers, or perhaps even by different adapters (email and papers, say). These nodes are connected to papers authored by Jon Kleinberg. In SPIN, all links are bidirectional, so high prestige of a paper is conducted to its author.

Before reconciliation, the endorsement from high-prestige papers would be divided among the multiple alias nodes for Kleinberg. Reconciliation involves connecting aliases by an edge or hyperedge (shown as a "supernode" around the two alias nodes). After reconciliation, papers transfer

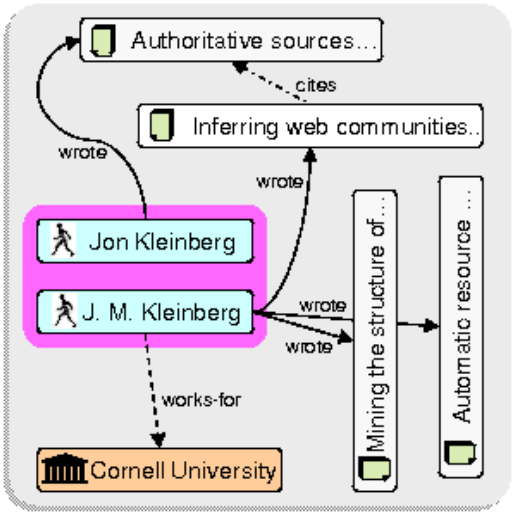

Figure 2: Effect of reconciliation on ranking.

some of their prestige to the single supernode for the reconciled authors, thereby improving its rank. Thus Kleinberg is more likely to top the list of responses to the query `type=person NEAR ("web graph" "link topology")` after reconciliation.

Reconciliation also helps to combine information captured from various adapters. In the above example a course newsgroup adapter reads an email containing a mention of Kleinberg in the body and creates a graph fragment as shown in Figure 3. (Like hyperedges, the "mentioned-in" edge is a soft edge.)

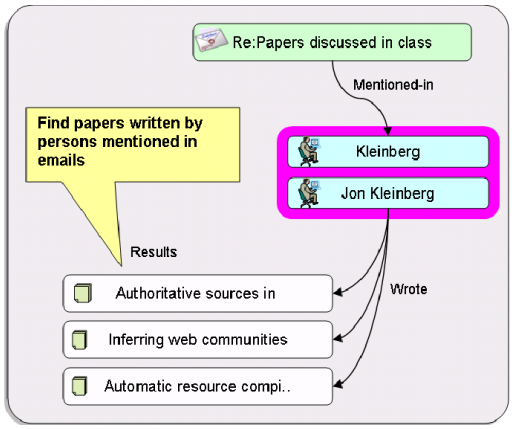

Figure 3: Reconciliation and "mention" edges.

The query "find papers written by persons mentioned in newsgroup postings" (which can be expressed as a SPIN twig query) is unlikely to return the papers written by Kleinberg since the mention node "Kleinberg" remains disconnected from the graph fragments emitted by other adapters. A reconciler for person nodes will likely reconcile this node with other nodes in the PIN graph and the above query will return results including the some of the papers written by Kleinberg.

5. DEMONSTRATION SUMMARY

We will present a live demonstration of many of SPIN's capabilities. We will show the important adapters (email, papers, address books, and a publication-oriented Web adapter) at work. We will also show how the reconciler introduces soft edges between aliases, and between mentions and entities. We will demonstrate some of the dominant query paradigms, and argue, through examples, that the linkage through the PINDB lead to better scoring and ranking than possible with a standard IR-based desktop search solution. We will also show the benefits of the browsing interface, with its intuitive graph contexts explaining why the top-ranking responses are related closely to the query.

6. REFERENCES

[1] CALO - Cognitive Assistant that Learns and Organizes. `http://www.ai.sri.com/project/CALO`.

[2] RADAR - Reflective Agents with Distributed Adaptive Reasoning. `http://www.radar.cs.cmu.edu`.

[3] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *ICDL*, pages 85–94. ACM, 2000.

[4] A. Balmin, V. Hristidis, and Y. Papakonstantinou. Authority-based keyword queries in databases using ObjectRank. In *VLDB*, Toronto, 2004.

[5] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan. Keyword searching and browsing in databases using BANKS. In *ICDE*. IEEE, 2002.

[6] A. Cheyer, J. Park, and R. Giuli. IRIS: Integrate. Relate. Infer. Share. In *ISWC 2005 Workshop on The Semantic Desktop*, 2005.

[7] M. Collins. Ranking algorithms for named entity extraction: Boosting and the voted perceptron. In *ACL Conference*, pages 489–496, 2002.

[8] A. Doan, P. Domingos, and A. Y. Levy. Learning source description for data integration. In *WebDB (Informal Proceedings)*, pages 81–86, 2000.

[9] X. Dong, A. Halevy, and J. Madhavan. Reference reconciliation in complex information spaces. In *ACM SIGMOD Conference*, pages 85–96, 2005.

[10] D. Florescu, D. Koller, and A. Y. Levy. Using probabilistic information in data integration. In *The VLDB Journal*, pages 216–225, 1997.

[11] D. R. Karger and D. Quan. Haystack: A user interface for creating, browsing, and organizing arbitrary semistructured information. In *Human Factors in Computing Systems (ACM CHI)*, pages 777–778, 2004.

[12] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[13] A. McCallum, K. Nigam, and L. H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *SIGKDD Conference*, pages 169–178, 2000.

[14] D. Metzler and W. B. Croft. Combining the language model and inference network approaches to retrieval. *Information Processing and Management*, 40(5):735–750, 2004.

[15] Parag and P. Domingos. Multi relational record linkage. In *SIGKDD Multi-Relational Data Mining Workshop*, 2004.

[16] S. Soderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1-3):233–272, 1999.