SPIN: Searching in Personal Information Networks

Prof. Soumen Chakrabarti
soumen@cse.iitb.ac.in
IIT Bombay

Harsh Jain
harsh@cse.iitb.ac.in
IIT Bombay

Srivatsa Iyengar
vatsa@cse.iitb.ac.in
IIT Bombay

Abstract

SPIN integrates entities and relations from a wide variety of data sources (such as address books, emails, documents on file systems, blog sessions, messenger sessions, social networks and the Web at large) into a lowest common denominator format of a directed graph with typed edges, typed nodes, and text associated with nodes. SPIN supports a novel class of proximity search algorithms on top of this graph structure.

1 Introduction

Large scale storage has become dramatically affordable over the last decade, enabling large-scale archival of email, contacts, documents, images, and music. Large personal storage becomes all the more useful with effective search tools. Recent operating systems can index PC hard disks and enable keyword search over many file types. Most Web-based email services include keyword search. Most search companies have released desktop search tools that enable search on file systems and browser caches. Many research prototypes exist as well [3, 5].

With some exceptions, prototypes and products have stayed close to traditional IR: they largely lack the capability to discover and reconcile entities and relations that pervade the user's life from diverse and heterogeneous sources, to represent these in a graphical model, and to enable powerful but user-friendly queries on such graphs. These are the goals of our proposed system, SPIN.

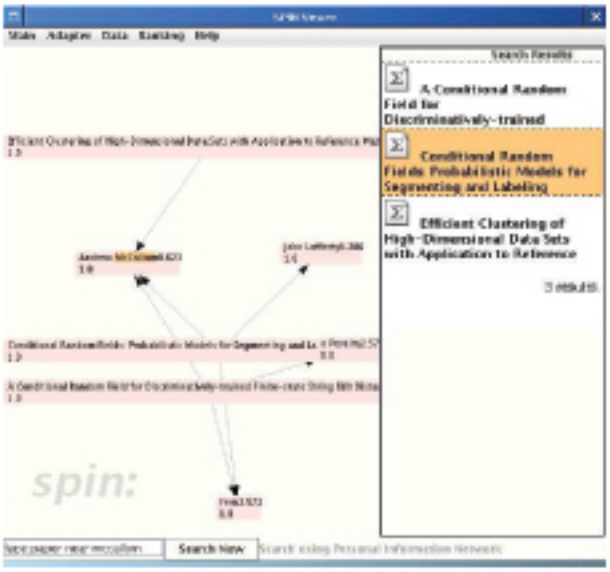

Figure 1: SPIN:Searching in personal information network

2 Personal Information Networks

A Personal Information Network(PIN) consists of several entities(nodes) and relations(edges) between these entities. PIN 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 wrote 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.

International Conference on Management of Data
COMAD 2005b, Hyderabad, India, December 20-22, 2005

2.1 PIN Schema

PIN-Schema is concise and accurate structural summary of the PIN graph. It is a graph among node-types connected through edge-types. PIN-Schema is maintained and dynamically modified by adapter registry, which controls the type of nodes and edges inserted into the PIN by an adapter. PIN-Schema is useful for browsing PIN structure, formulating queries and storing information such as statistics and sample values and is useful for query optimization. In SPIN context, PIN-Schema is most useful in reconciliation. Reconciliation algorithms take advantage of the strict schema the PIN has to adhere to, while making reconciliation decisions.

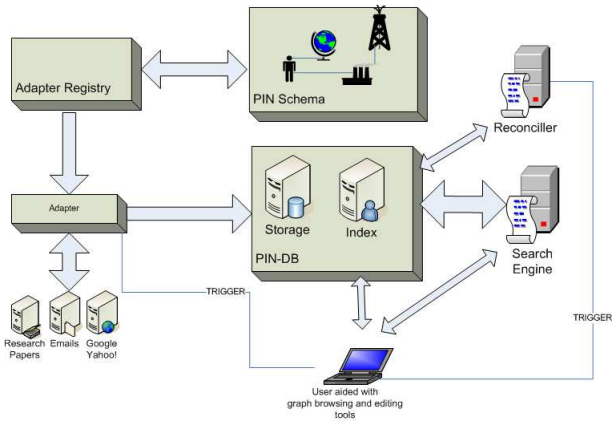

Figure 2: SPIN Architecture

2.2 Adapters

Adapters are primarily responsible for populating the PIN from various data sources. They introduce new entities and new relations in the PIN. While it is mostly trivial to introduce hard edges, as these are explicitly mentioned in most data sources, the real challenge for adapters is to introduce non-explicit relations or "soft" edges. SPIN currently has adapters to extract information from emails, address books, and research papers. We employ several machine learning techniques to create these soft edges, for example extracting names of persons from emails and authors from papers.

3 Reconciliation

Since SPIN deals with multiple information sources, it is possible that several different representations of the same real world entity are created in the system. Reconciliation addresses this problem by identifying duplicate entities and combining them to create a single node for each real world entity in the PIN graph. Existing systems largely use record linkage [2] and probabilistic approaches using graphical models [7] like Conditional Random Fields (CRFs) [6]. SPIN uses a novel approach to reconciliation, using a variant of SimRank [4] to quickly identify candidate pairs for reconciliation.

3.1 Reconciliation System details

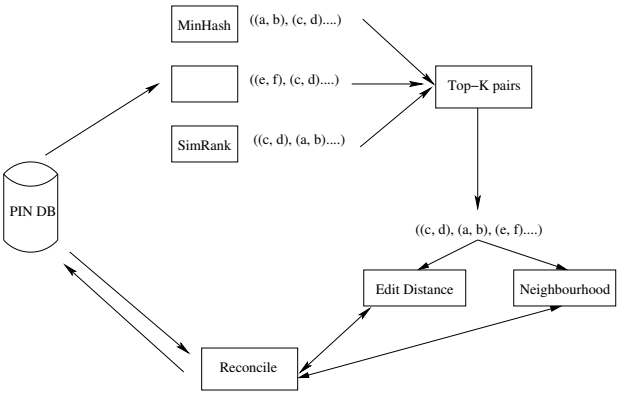

Figure 3: Entity Reconciliation Module in SPIN

Entity reconciliation module uses the services provided by the SPIN database PIN-DB, for accessing PIN graph. Lists of candidate pairs for reconciliation are generated using SimRank and a host of other methods and merged using a top-k strategy. Pairs in the merged list are compared using a stronger string similarity metric like edit distance. Based on the similarity, neighborhood constraints and PIN schema constraints, pairs of nodes which can not be reconciled are eliminated. A transitive closure of the remaining pairs is computed and all the nodes in a single equivalence-class are reconciled into a single node.

3.2 Reconciliation and search

The motivation behind entity reconciliation in SPIN like setting is twofold, reconciliation puts together information about an entity disseminated in the PIN graph and it improves search accuracy. Rank scores distributed across different nodes reinforce after reconciliation and the rank score of the reconciled node will be higher. Reconciliation module in SPIN keeps the old graph structure intact by creating "soft" reconciliation nodes and edges and search algorithms may use either of the PIN graphs to rank the search results thereby making it easy to evaluate the effect of reconciliation on search.

4 Searching PIN

SPIN provides a proximity-based, type-sensitive query model to search the PIN and a query API to programmatically create such queries. One dominant query paradigm is *type=TypeName? NEAR Predicates*. The user can look for a student who graduated around 2001 and went to work at IBM using the query *type=person NEAR org=IBM year=2001*.

To illustrate the power of complex queries supported, SPIN also supports an INDRI style grammar. It allows complex phrase matching, synonyms, weighted expressions, boolean filtering, numeric(and dated) fields, and the extensive use of document structure(fields) among others. In SPIN, every entity is a set of (field, value) pairs. For example, an email has fields such as "From", "To", "Subject" etc. The query API allows user to create queries which restricts search to some particular field or particular type of nodes. For example,*#combine(John Matthews.From,Subject #any.Email)* will look for emails containing "John Matthews" in From or Subject fields of the email. The user can look for all persons born before 1st January 1985 with the query *#combine(#datebefore(1/1/1985) #any.Person)*. Specific weights can be assigned as *#weight(Machine Learning.Title 20 Data Mining.Title 80)*.

The above predicates activate some nodes, which spread the activation using algorithms [1] that are sensitive to the structure and uncertainty in the PIN. SPIN continuously assimilates user edits and annotations into its ranking algorithms. SPIN also allows user to create queries where user can drag and drop PIN nodes which assists the user in query creation. The visual query creation is tightly coupled with PIN-Schema to show only valid options.

4.1 Search Results

Conventional IR engines display a list of search results, typically ordered by relevance. In SPIN, every query results in a list of graph fragments sorted by the decreasing order of relevance. The only restriction placed by the user is the number of nodes in each graph fragment, as large number of nodes tend to clutter the display and result in bad user experience. Taking this constraint into account, SPIN creates graph fragments, which in addition to the activated nodes, also show the nodes which resulted in making this fragment important. Eventually, the fragment helps user understand why this particular node is ranked higher.

A result graph fragment is typically focused around the activated nodes. A user can click on individual nodes to view details of the node. SPIN honors such user click as indicative of user's bias towards such nodes or concept by dynamically modifying the result fragment to match user's bias. Such clicks can be accidental and SPIN takes care of this to ensure good user experience. We employ several variants of algorithms in [1] to help user visually navigate the result graph.

References

[1] A. Balmin, V. Hristidis, and Y. Papakonstantinou. Objectrank: Authority-based keyword search in databases, 2004.

[2] Xin Dong, Alon Halevy, and Jayant Madhavan. Reference reconciliation in complex information spaces. In *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of*

data, pages 85–96, New York, NY, USA, 2005. ACM Press.

[3] Xin Dong and Alon Y. Halevy. A platform for personal information management and integration. In *CIDR*, pages 119–130, 2005.

[4] Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543, New York, NY, USA, 2002. ACM Press.

[5] David R. Karger and Dennis Quan. Haystack: a user interface for creating, browsing, and organizing arbitrary semistructured information. In *CHI '04: CHI '04 extended abstracts on Human factors in computing systems*, pages 777–778, New York, NY, USA, 2004. ACM Press.

[6] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

[7] Parag and Pedro Domingos. Multi relational record linkage. In *3rd Workshop on Multi-Relational Data Mining*, 2004.