Search Your Memory ! - An Associative Memory Based Desktop Search System

Jidong Chen, Hang Guo
EMC Research China, Beijing, China
{chen_jidong,guo_hang}@emc.com

Wentao Wu, Chunxin Xie
Fudan University, Shanghai, China
{072021113,072021114}@fudan.edu.cn

1. INTRODUCTION

As most of the digital universe are created by individuals, personal information management (PIM) has become a hot topic. Searching desktop resources, including local files, emails, instance messages, cached web pages, etc., has become an important but time-consuming task. Personal search is different from web search. People generally have a vague picture of what is stored in their computers but they always forget the exact location and keyword of the resource content. Existing desktop search systems such as Google Desktop, Microsoft Windows Desktop Search, and Spotlight for Apple's OS only support keyword search that needs exact keyword matching to find resources.

Considering human memory recall, people appear to remember something through some associative memory fragments left in their brains as well as memory cues relating to the context of information capture or subsequent access. For example, a user wants to find a web page he had read while writing an important report, but he cannot remember the keywords about the web page. Interestingly, he does remember the connection and therefore finds the web page through the report. Psychology research has shown that people often remember things through chains of associations. Our idea is to exploit such associations and contexts to simulate human associative memory to enhance desktop search.

Current desktop file systems do not provide a means to link semantically related files. However, user activities in the desktop, to some extent, reflect his associative memory when he searches resources later. We developed the XSearcher, an associative memory based desktop search system. Desktop resources are connected in XSearcher with semantic links by analyzing user activities and the contexts in which the user works. Using these semantic links, associations among memory fragments can be built or rebuilt in a user's brain during a search. The link-based ranking scheme uses these links together with user's personal preferences to rank results by both relevance and importance. XSearcher provides a two-step search paradigm. The user first uses faceted search filters to quickly locate an easy-to-remember intermediate resource through the full-text keyword search results. Then the user can find the target resource by multiple navigation in the association graphs in XSearcher.

In some recent PIM systems, e.g. Stuff I've Seen [4], MyLifeBits [6], Haystack [7], Semex [5], Beagle++ [3] and Feldspar [2], contextual cues such as time, author and association information are exploited to search and present personal information. However, the above systems only support a few kinds of simple predefined associations. XSearcher enhances desktop search by building associative links of resources from implicit access patterns of user activity sequences. In XSearcher, there are three types of associations: Content-based Associations (CA), Explicit Activity-based Associations (EAA) and Implicit Activity-based Associations (IAA). CAs refer to associations that can be extracted from the content and attributes of resources. EAAs are explicit associations in the sense that they are bound with certain user activities such as jumping to another webpage. IAAs denote some implicit associations between resources, which can be discovered through user access pattern analysis and resource provenance analysis. These semantic associations are designed in accordance with a common mode of thinking in human minds so that associative memory can be exploited to help find the target resource.

2. ARCHITECTURE

The architecture of XSearcher is given in Figure 1. Besides the traditional desktop search components (left rectangle), XSearcher has several additional components (right rectangle) for tracking user activities and generating semantic links. XSearcher uses the activity event monitor to record desktop events and association analyzer to create semantic links from the recorded events. Then the resource associations and attributes are represented as RDF triples and stored in an RDF repository. The searcher and query processor module delegates keyword searches and RDF queries to the full-text index and repository respectively. RDF queries are used to retrieve the semantic links of a given resource. XSearcher provides two different ways to help users find their resources. Faceted search filters are used to refine the full-text search results through multi-dimensional classification. On the other hand, the association graph navigation is used to extend the searching results to associative resources via semantic links. The user interface visualizes search results and associative context in these two ways as well.

2.1 Desktop Association Analysis

XSearcher enhances desktop search by building associa-

ACM SIGMOD '09 Providence, RI, USA

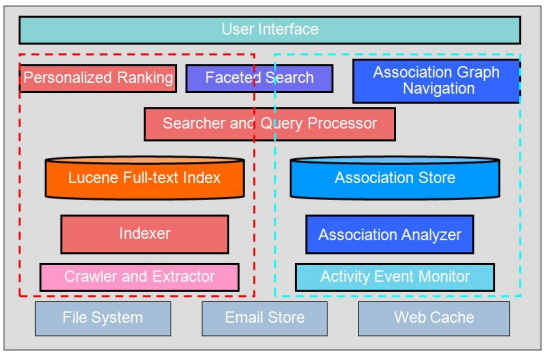

Figure 1: Architecture

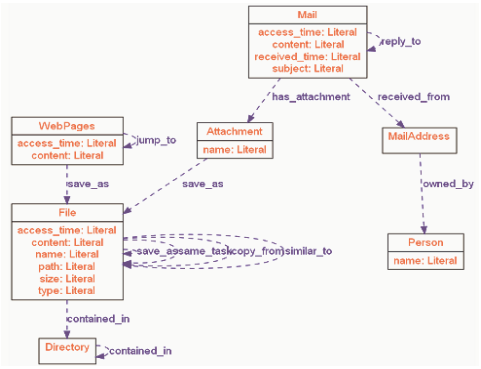

Figure 2: The Desktop Ontology

tive links of resources from user activity context. Figure 2 shows the desktop ontology in XSearcher, which represents desktop resource attributes and semantic associations among emails, folder and file hierarchies, cached web pages, etc. Through content analysis, user access pattern analysis and lineage analysis, we get the following three types of links.

CA CAs represent the content-based associations created by analyzing the content and attributes of desktop resources. For example, in Figure 2, the *similar_to* link connects two resources that are similar in terms of contents, file names and locations (e.g. same directory). A similarity function regarding these three factors are used to compute the distance between two resources. By analyzing the attributes of email messages and files, we can get the *has_attachment*, *reply_to*, *received_from*, *owned_by* and *contained_in* CA links.

EAA EAAs are explicit and deterministic activity based associations. EAA links are set when specific user events are observed. The event monitor is necessary to construct EAAs. We create three types of EAs in XSearcher: *jump_to*, *copy_from* and *save_as*. Each of them is bound with a type of user activity tracked by the event monitor. For instance when the user goes from one webpage to another, the two pages are connected by a *jump_to* link. When the attachment of an email (or a webpage) is saved as a local file, a *save_as* link will be created connecting the attachment (or webpage) and the file. Similarly the *copy_from* link is created based on "file copy" events.

IAA IAAs denote some implicit and nondeterministic associations between resources, which can be discovered by user access pattern analysis (e.g. the *same_task* link) and resource lineage analysis (e.g. the *provenance_of* link). We find that users tend to access and manipulate different resources to complete a task. For instance, when writing a paper, a user may look for related presentations, references, web pages and emails from colleagues. These resources are all somehow related to this task. So we define the *same_task* link to cluster resources by their related tasks. The *same_task* is usually very helpful for the user because we organize our memory in a similar way. A primitive algorithm is used to create *same_task* links. We have an observation that usually a task only has one *key resource* (the paper in the above example), *key* for short. In most cases the *key* has the longest *lifecycle* (the time interval between opening and closing). Therefore we cluster resources by their lifecycles. If the lifecycle of resource a is completely covered by that of resource b, then a and b go to one cluster. The resource that has the longest lifecycle is the *key* of the cluster. If the lifecycles of two keys are overlapped, we use simple rules to decide whether the two clusters should be merged or not. When no clusters can be merged, each cluster is output as one task. All resources in the cluster are connected by the *same_task* link. The *provenance_of* link can represent the origin of a resource. Users always create multiple versions of a document through several "file copy" and "save as" operations. By analyzing the lineage sequence of *save_as* and *copy_from* activities, we can create the *provenance_of* link between a document and its original copy. Actually, an IAA link is responding to a pattern of user activity sequence.

2.2 Event Monitor

In order to construct semantic associations, XSearcher needs to monitor user activities on their desktops. Currently, the event monitor can track both system-level and application-level activities. System-level activities include all file system operations and Window related activities such as Window creation, activation and closing. Application-level activities are restricted to specific applications. The monitor provides "plugins" for each of the applications to track user activities on MS-Office documents, Emails (Outlook), Adobe pdf files and cached web pages (Firefox). The plugins also extract application-specific metadata and associations between resources. When an email is received, for example, the Outlook plugin automatically extracts email metadata, e.g. "to", "from" and "subject" from corresponding email fields, and connects the attached documents with the email by the *has_attachment* link.

2.3 Personalized Ranking

Existing desktop search engines only employ content-based algorithms such as cosine similarity to rank personal desktop resources. However, the content-based ranking could not reflect the users personal preference. For desktop search, a top ranked document should be not only relevant to the query but also relatively important to the user. Link-based ranking algorithms, such as PageRank, are good ways to find authoritative results. With the established links from user activity, XSearcher can combine the two ranking schemes to generate personalized ranking results for every user.

The final score $R(d)$ of document d is the product of the relevance score $R(d)^q$ and the global authority score $R(d)^G$. Here q is the keyword vector. Similar to the ObjectRank [1] algorithm, different weights are manually assigned to different types of links since different links may have different

impact on $R(d)^G$. Therefore more important links, such as *same_task*, are given relatively higher weights. By this means, desktop resources are connected by weighted links. Then we can employ a link-based ranking algorithm to find important results. $R(d)^G$ is computed as following:

$$R(d)_n^G = d * A \times R(d)_{n+1}^G + (1-d) * e$$

where $R(d)_n^G$ denotes the global authority score after $n-th$ iteration. d is the dumping factor and it is set to 0.85. Matrix A denotes the weighted graph of indexed files. e is a uniform vector, which denotes the possibility distribution that a user moves to a random file when he is not following the existing links. However, in order to model personal preferences, we modify e by assigning different weights to different data sources according to their access frequency. In our algorithm, $e_i = count(i)/\sum_i count(i)$. Here $count(i)$ is the number of occurrences that file i appears in the user log. If file i does not appear in the log, $count(i)$ is set to 1.

Our ranking algorithm is personalized because the user can manually adjust the weights of different types of associations, which explicitly show the user preferences. In addition, user activities can implicitly influence the ranking computation since some semantic links are created by analyzing user access patterns and some parameter in the algorithm are decided by the user access frequency as well.

2.4 Faceted Search

Faceted Search enables users to navigate in a multi-dimensional information space by combining text search with a progressive narrowing of choices in each dimension. In XSearcher, we apply faceted search to desktop search to refine results from keyword search. It combines the effectiveness and flexibility of full-text search with the ease-of-use of faceted navigation: the ability to find information based on attributes such as its location, file type, modification dates, file size, author, etc. Unlike web facets, desktop facets in XSearcher are well organized into a facet tree, which further classifies facets of the same taxonomy. In addition, users can select multiple facet terms from different taxonomies at the same time to quickly filter the result set. In XSearcher, there are two kinds of desktop facets, predefined desktop facets (e.g. file size, modified date, file type) and dynamically generated facets, such as the senders of email messages, in the sense that their values vary according to the result set.

3. DEMO SCENARIOS

In desktop search, users tend to associate resources in their minds and try to follow these associations when looking for specific resources. There are some examples here. "I remember that I have browsed several webpages weeks ago, but I forgot either its URL or its content. On the other hand, I remember that when I browsed the page, I was writing a technical report, and I remember some keywords of that document", "Several images were downloaded from attachments of an email several months ago. Now I want to find a paper with the same folder as these images. But I could not remember either its name or its directory. I only know that Peter sent the email and talked about desktop search topic". XSearcher manages to process such searches. Figure 3 shows the user interface of XSearcher. The association graph of XSearcher shows all directly related resources to a specific resource. A user can first locate an intermediate resource by keywords and faceted search, and then switch to the association graph to get its related resource. When the user selects one of the associative resources, say resource r, the graph will be updated and all related resources of r will be displayed. By this means, the user can navigate in the association graph, which simulates the associative memory in the mind of the user. We will demo how XSearcher deals with the associative search in the two examples.

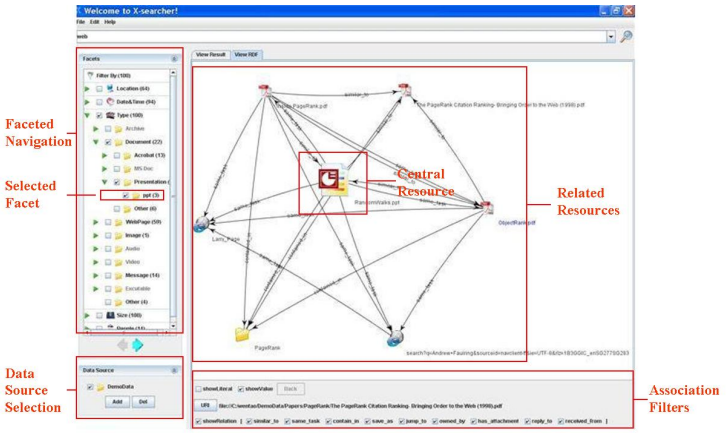

Figure 3: System User Interface

4. CONCLUSIONS

We developed an associative memory based desktop search system, XSearcher, which enhances conventional full-text search with semantic associations discovered from user activity context. In addition, the system provides the faceted search and association graph navigation to help users refine and associate search results generated by the full-text keyword search. XSearcher is superior to the traditional keyword based search engines because it is closer to the way that human associative memory works.

5. REFERENCES

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

[2] D. H. Chau, B. Myers and Faulring, A. What to Do When Search Fails: Finding Information by Association. In CHI 2008.

[3] P. A. Chirita, S. Ghita, W. Nejdl, and R. Paiu. Beagle++: Semantically enhanced searching and ranking on the desktop. In ESWC 2006.

[4] S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. Robbins. Stuff ive seen: a system for personal information retrieval and re-use. In SIGIR 2003.

[5] X. Dong and A. Halevy. A Platform for Personal Information Management and Integration. In CIDR 2005.

[6] J. Gemmell, G. Bell, R. Lueder, S. Drucker, and C. Wong. Mylifebits: fulfilling the memex vision. In ACM Conference on Multimedia 2002.

[7] D. R. Karger. Haystack: A Customizable General-Purpose Information Management Tool for End Users of Semistructured Data. In CIDR, 2005.