The Personalized Index Service System in Digital Library

Jie Ou Shouxun Lin Jintao Li*
Digital Technology Laboratory
Institute of Computing Technology, Chinese Academy of Sciences
{jieou, sxlin, jtli}@ict.ac.cn

Abstract

China Digital Library is a federated digital library, which is composed of library components such as: local digital libraries, publishing companies, and so on. The objective of federated digital library is to share resources and services of its components, therefore users can access distributed information transparently. The immense distributed electronic information available in federated digital library raise two problems: how to find all information that user needed accurately, how to provide information that user is interested. The personalized index service system presented in this paper hold the promise of resolving these problems, which is composed of index service system and personalized service system. The former facilitates federated resource discovery. The latter learns model of user's interests and using the model to push interesting information and sort the search results, so that users are able to easily and quickly find interesting objects. The personalized service system learns the model of user's initial interests after user provides his/her interests, and the system updates the model according to user's access log and feedback to search results.

*The work reported in this paper has been funded in part by the China Digital Library Demonstration System, no. 863-306-ZD11-03.

1. Introduction

Digital library [1] is a managed collection of digital objects (content) and services (functionality) associated with the storage, discovery, retrieval, and preservation of those objects.

The demand for both services and information can be expected to grow so large that it is impossible for a single corporation, professional organization, or government to provide all that is necessary for digital library, the problem is exacerbated when digital library is considered on an international scale. Additionally, a large percentage of the information and services that will be made available through digital libraries will be privately held intellectual property, and it is unlikely that control over this property will be yield to third parties. Thus, a large number of groups will provide digital library content and services. Since users will demand content and services from everywhere in the world, federated digital library [2] become integral and critical infrastructure for China.

China Digital Library [3] is a federated digital library, which is composed of library components such as local digital libraries, publishing companies, and so on, it generates a single (virtual) view on many different library components without sacrificing autonomy.

There are immense distributed objects [4] in federated digital library. It is important to index these objects so that users can get content and services from

0-7695-1128-7/01 $10.00 © 2001 IEEE

anywhere in the world. Additionally, It is time consuming for users to look all the information in library for what they need, so pushing what user is interested is important. Because of the immense information, user always gets a great number of search results and spends a lot of time on selecting what he/she need. Generally, user searches for what he is interested, therefore, sorting the search results according to user's interests brings user what he needs quickly.

The personalized index service system presented in this paper can implement these functions by index service and using machine learning techniques. It is composed of index service system and personalized service system. The index service system is a core service in federated digital library that is necessary to provide basic digital library functionality. It provides the mechanisms for the discovery of digital objects, so that users can easily find and discover objects in the collection. The personalized service system can learn model of user's interests and using the model to push information and sort the search results, so users are able to easily and quickly find interesting objects in the federated digital library.

Personalized index service system in digital library is different from other personalized service systems such as WebWatcher, WebMate, and IDGS. It facilitates resource discovery as well as personalized service by using metadata and all information that user leaves in web server.

The paper is organized as follows. In Section 2, we present a summary of related work. In Section 3, the problems of personalized service systems are exhibited. In Section 4, we give an overview of the personalized index service system. In Section 5, the architecture of the index service system is provided. In Section 6, we describe the framework of personalized index service. Finally, we give a summary to our work in Section 7.

2. Related Work

WebWatcher [5] is a simple but operational tour guide, which has given over 5000 tours to people browsing CMU's School of Computer Science Web pages. WebWatcher accompanies users from page to page, suggests appropriate hyperlinks, and learns from experience to improve its advice-giving skills.

WebMate [6] is an agent that helps users to effectively browse and search the Web. The techniques that WebMate uses are: (1) multiple TF-IDF vectors to keep track of user interests in different domains that are automatically learned by WebMate; (2) the Trigger Pair Model to automatically extract keywords for refining document search. (3) During search, the user can provide multiple pages as similarity/relevance guidance for the search, accordingly the system extracts and combines relevant keywords from these relevant pages and uses them for keyword refinement. Therefore, WebMate provides effective browsing and searching help, it also compiles and sends to users personal newspaper by automatically collecting information from news sources.

IDGS (information discovering and gathering system) [7] is a prototype system aims at collecting China and English technology information on WWW for users automatically, which adopts vector space model and modified Robot technology. IDGS can collect technology information on WWW according to the documents that user provides.

3. Problem Setting

Personalized service system helps users to effectively browse and search the Web. But these personalized service systems have the following deficiencies:

(1) All information filtering systems that automatically learn user interest profiles have a large time constant, which means it takes a long time before they start to make useful predictions. Because learning in natural language usually involves big search spaces, many examples are needed. Users will become disappointed with the initial bad performance, and might even stop using the system.

(2) User's interest model is modified only by user's feedback. Modifying interests model by web log

mining is needed.

The personalized service system learns the model of user's initial interests after user provides his/her interests, and the system updates the model according to user's access log and user's feedback to search results. User provides initial interests by answering a questionnaire, which can get user interests quickly. Modifying user's interests model according to user's access log and user's feedback to search results utilizes all information that user leave in web server, thereby it can adapt to the change of user's interests.

4. The Architecture of Personalized Index Service System

The personalized index service system is composed of index service system and personalized service system. The index service system is a core service in federated digital library that is necessary to provide basic digital library functionality. It provides the mechanisms for the discovery of distributed digital objects, so that users can easily find and discover objects in the collections. The personalized service system can learn model of user's interests and using the model to push information and sort the search results, so users are able to easily and quickly find objects in the federated digital library. The architecture of personalized index service system is shown in figure 1.

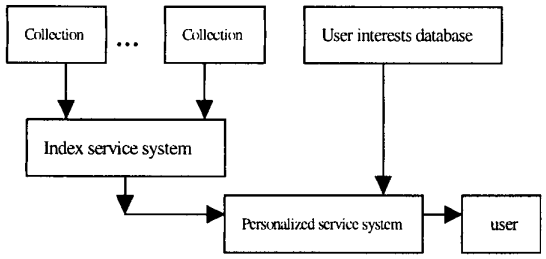

Fig. 1. **Architecture of personalized index service system**

5. Index Service System

China Digital Library is federated digital library, which is composed of library components such as local digital libraries, publishing companies, and so on. It generates a single (virtual) view on many different library components without sacrificing autonomy. The index service system is a core service in federated digital library as mentioned above. The index service system in China Digital Library is composed of a global index manager and several local index service systems. Local index service system provides index service to the objects in local resource. The architecture of the index service system is shown in figure 2. The local index service system is composed of a local index manager and several index services, the architecture of which is shown in figure 3. One example of an index service definition criterion is resource type. A video index service and book index service is designed in local index service system containing video and book sources. The resource in index service can be divided to one or several source(s). One example of a source definition criterion is subject, which may be determined by reading a controlled vocabulary metadata field of objects or derived via some natural language analysis. The architecture of index service is shown in figure 4.

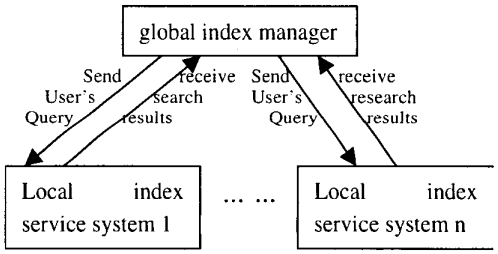

Fig. 2. **The architecture of index service system in China Digital Library**

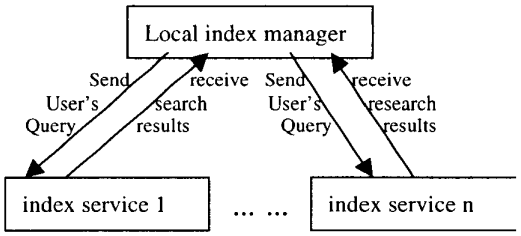

Fig. 3. **The architecture of local index service system**

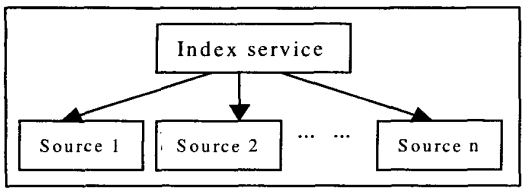

Fig. 4. **The architecture of index service**

5.1. Index Service

Index server collects information about digital objects. This information may be in the form of surrogates, such as the familiar cataloging records in the traditional library, or it may be full disseminations of the objects, such as that used by full-text search engines. The information that is collected is organized into structured indexes that allow search engines to respond to queries on the information with precision, recall, and efficiency. The response to a query is a "result set"-where each member of the set is generally a surrogate for a digital object that "matches" the query. The minimal form of surrogate is the unique identifier for the object that, when resolved by the name service, allows access to the digital object.[1]

We are intentionally informal about the nature of this information collection and indexing process. The method may vary across a broad spectrum with the following interesting data points:

(1) Automatic Indexing — An index server visits a set of repository servers and extracts and automatically indexes information from the digital objects contained in those repositories. This is similar in nature to the "web crawling" technique used by existing web search engines.

(2) Manual Indexing — In the style of traditional libraries, professional catalogers create ordered surrogates for digital objects and index that information.

(3) Human-Mediated Indexing — A hybrid of the two end points on the spectrum.

5.2. Local Index Manager

Given a query, local index manager has to choose the best index services to evaluate the query and merge the query results from these index services. For this, the local index manager must extract the source list, metadata, and content summaries from the sources periodically.

To select the right sources for a query and to query them we need information about their contents and capabilities. In the following we propose two pieces of metadata that every source is required to export: a list of metadata attributes, describing properties of the source, and a content summary of the source [8].

(1) Source metadata attributes

Each source exports information about itself by giving values to the metadata attributes below.

- FieldsSupported

  (A list of the fields that can be used for querying the source, such as title, author and so on.)
- ModifierSupported

(What Modifier are supported, such as *and, or* and so on)
- SourceLanguage

  (List of languages present at the source.)
- SourceName

  (The name of the source.)
- Linkage

  (URL where the source should be queried)
- ContentSummary linkage

(The URL of the source content summary, see below)
- Contact

(Contact information of the administrator of the source)

(2) Source content summary

This data can be used to decide whether the source is relevant to a given query, which includes:

① List of words that appear in the source, specifying the field corresponding to where in the documents they occurred (e.g. (title "databases"), (abstract "databases"), etc.)

② Statistics for each word listed, including the followings:

- Total number of posting for each word (i.e., the number of times that the word appears in the source)

- Document frequency for each word (i.e., the number of documents that contain the word)

③ Total number of document in source.

After receiving a query, the source reports the number of documents in the results. Since the source might modify the given query before processing it, the source also reports the query that it actually processed.

To merge the query results from multiple sources into a single, meaningful rank, a source should return the following information for each document in the query result:

- The id of the source (s) where the document appears
- The title and author of the document
- The sort id of the document

5.3. Global Index Manager

Given a query, the global index manager has to choose the best local index service systems to evaluate the query and merge the query results from these systems.

The global index manager sends user's query to the local index service systems containing information type that user needs. For this, the global index manager must store the type list of information in local index service system, and extract them from the systems periodically.

After receiving a query, the local index service system evaluates the query. After that system reports the number of documents in the results, also, since the system might modify the given query before processing it, the system reports the query that it actually processed.

To merged the query results from multiple sources into a single, meaningful rank, a source should return the following information for each document in the query result:

- The id of the source (s) where the document appears
- The title and author of the document
- The sort id of the document

6. Personalized Index Service

Personalized index service pushes interesting information and sorts the search results according to the user interests model.

6.1. Feature Extraction

Some features need to be extracted from documents and users' interests in order to provide a basis for computing the search heuristic. We use VSM (vector space model) to describe the documents and users' interests. Each document has a vector V, where element v_i is the weight of keyword d_i for that document. We calculate word weights using a TFIDF scheme, normalizing for document length. The weight v_i of a keyword d_i in a document T is given by [9]:

$$V_i = \frac{(0.5+0.5\frac{tf(i)}{tf_{\max}})(\log\frac{n}{df(i)})}{\sqrt{\sum_{T_j \in F}((0.5+0.5\frac{tf(j)}{tf_{\max}})^2(\log\frac{n}{df(j)})^2)}}$$

Where tf(i) is the number of times keyword d_i appears in a document T (the term frequency), df(i) is the number of documents in the source which contains d_i (the document frequency), n is the number of documents in the source and tf_{max} is the maximum term frequency over all words in T.

Since documents and user's interests are presented in VSM, Particular documents can be retrieved based on a similarity measure between the document vector and user interests model, often just the cosine of the angle between them. In our framework user interests model, or user profile, which is $U = (u_1, u_2 \cdots, u_n)$, The score for a page can be calculated by measuring how well it matches this profile, which is just a comparison between the vector of the document and U:

$$Sim(V,U) = \cos(V,U) = \frac{\sum_{k=1}^{n} u_k \cdot v_k}{\sqrt{\sum_{k=1}^{n} u_k^2} \cdot \sqrt{\sum_{k=1}^{n} v_k^2}}$$

6.2. Learning the model of user's interests

There are three methods to learn the model of user's interests:

(1) Learning the model of user's interests by user providing their interests.

(2) Learning the model of user's interests by the documents that user reads.

(3) Learning the model of user's interests by user's feedback to the search results and the pushed documents.

System can learn user interests model from sets of messages or other online documents that users have classified[10] or the answers that user provides to some questions. Learning user interests model from the answers that user provides to a questionnaire is presented in this paper. User is asked to fill in a simple questionnaire [11] of eight questions, these questions include: ① which academic field are you working? ② In which subjects within this field are you particularly interested? ③ Which projects are you currently working on? ④ Could you describe these projects in one or two sentences? ⑤ What other research do you like to read about? ⑥ Are you interested in music and if so in which kinds of music are you interested? ⑦ Are you interested in sports and if so in which sports are you interested? ⑧ Which other subjects do you like to read about? Descriptions of several sentences are preferred over single word descriptions, because many words are needed to use a vector space representation successfully. In our opinion, a user has several interests, so we sort the answers to various subclasses. Generately, these subclasses correspond to user's answer to different questions. Averaging the vectors from all answers could reduce too much the weights of words that are important for only one or a few of these subclasses.

Using this method, system doesn't require substatial training data before learning a successful user profile, so system can get user profile quickly. But the answers are still rather brief and user's interests may be changing.

Generally, user only accesses the documents that he/she is interested. So we assume the documents that user has accessed are interesting pages. We can learn the model of user's interests based on the documents that user reads in digital library, which is called web content mining.

Using this method, system can adapt to the change of user's interests, and doesn't require users to provide any information. But it requires substatial training data before learning a successful user profile.

If user gives feedback to the search results or objects that system pushes to him, system can update user interests model according to the feedback.

Updating the model of user's interests based on the documents that user reads in digital library is introduced in the following.

6.3. Transaction Model for Web Mining

User access information is stored in server log file. We can get visits from server log file. A visit is an ordered sequence of requests made by a user in a single session. Each request typically includes the time of the request, the URL requested, and the machine from which the request originated. A visit is described as following[12]:

$$t =< ip_t, uid_t, \{(l_1^t.url, l_1^t.time, l_1^t.length), \cdots, (l_m^t.url, l_m^t.time, l_m^t.length)\} >$$

Where ip_t is the IP address of user, uid_t is the id of user, $\forall 0 < k < m+1, l_k^t$ is the kth page that user accesses, $l_k^t.url$ is the URL of l_k^t, $l_k^t.time$ is the time that user accesses l_k^t, $l_k^t.length$ is the time length that user accesses l_k^t, $l_1^t.time < l_2^t.time < \cdots < l_m^t.time$。

6.4. Finding Content Pages

Content page is a page that contains the information that user needs. Navigation page is a page whose purpose is to provide link to guide users to content pages. In order to find out user's interests, system should select content pages from the pages that user accesses. There are two methods to find content pages: (1) finding content pages according to the access time length. If user access time length of a page

is longer than β, the page is considered to be a content page. Otherwise, the page is considered to be a navigation page. β is a value given by system. (2) Finding content pages by maximal forward reference. Assuming user will make a back reference after he accesses a content page, so the content page is the page that user access before user makes a back reference. For example, the pages that user accessed are: ABCDCEFECBG, the content pages is D, F, G. Experiment evaluates that the first method is better than the second. [12]

After getting the content pages, system extracts features from them based on VSM and uses the vectors of content pages to update the user interests model.

6.5. Updating User Interests Model

Let $M_1,\cdots,M_n$ be user interest model, $V_1,\cdots,V_m$ be the vectors of content pages that user has accessed. If V_i is similar to M_j, that is $V_i \cdot M_j > \delta$, δ is given by system, then system updates M_j by $M_j = M_j + \frac{m^2 * V_i * VI(i) * T(i)}{\sum_{k=1}^{m} VI(k) * \sum_{k=1}^{m} T(k)}$, where $VI(i)$ is the number of user accessing V_i, $T(i)$ is the time length of user accessing V_i. If V_i is not similar to any user interest model, then

$M_j = \frac{m^2 * V_i * VI(i) * T(i)}{\sum_{k=1}^{m} VI(k) * \sum_{k=1}^{m} T(k)}$ is a new interest vector of the user. Probably the user interests model will increase a lot, so if the value of a interest model is less than α, system do not use this model to push objects or sort search results.

If user gives feedbacks to objects, we can update the user interests model based on the feedback. For example, an object V is viewed by the user and receives an evaluation e (an value in [+0.2,0,-0.2], +0.2 presents interesting, 0 presents relevant, -0.2 presents not interesting), $M_1,\cdots,M_n$ are vectors of user's interests model. V is vector of the object that user has evaluated. If V is similar to M_j, that is $V \cdot M_j > \delta$, δ is given by system, then system updates M_j by $M_j = M_j + e \cdot V$, e is the evaluation that user give to the object. If V is not similar to any vector of user interest model, then system adds a user interest vector which value is $e \cdot V$.

6.6. Personalized Services

Personalized service system learns model of user's interests and using the model to push interesting information and sort the search results, so that users are able to easily and quickly find interesting objects in the federated digital library. The personalized service system learns the model of user's initial interests after user provides his/her interests, and the system updates the model according to user's access log and user's feedback to search results. User provides initial interests by answering a questionnaire, which can get user interests quickly. Modifying user interest according to user's access log and user's feedback to search results use all information that user leaves in web server, thereby it can adapt to the change

of user's interests.

7. Summary

We have implemented the experimental index service system and most functions of personalized service system.

When users firstly register to use the personalized index service system, they are requested to input their interests by selecting their specialty, research interests, hobbies such as music, sports, and so on. After that, the system updates user's profile according to user's feedback to search results. Every time users register to use the system, the system pushes interesting information and sorts search results according to the their profile.

Updating user's profile according to user's access log will be implemented in the future.

References

[1] Carl Lagoze, Sandra Payette, An Infrastructure for Open-Architecture Digital Libraries, http://nestrl.cs.cornell.edu:80/Dienst/UI/1.0/Display/nestrl.cornell/TR98-1690 .

[2] Barry M.Leiner. The NCSTRL Approach to Open Architecture for the Confederated Digital Library. http://cs-tr.cs.cornell.edu:80/Dienst/UI/1.0/Display/cnri.dlib/december98-leiner.

[3] Gao Wen, Liu Feng, Huang Tie-Jun, Digital Library —— Theory and Technical Implementation, 2000, TsingHua publishing company.

[4] William Y. Arms, Christophe Blanchi, Edward A. Overly. An Architecture for Information in Digital Libraries, D-lib magazine, February 1997.

[5] Thorsten Joachims, Dayne Freitag and Tom Mitchell. Web Watcher: A Tour Guide for the WWW. http://WWW.cs.cmu.edu

[6] Liren Chen and Katia Sycara. WebMate: A Personal Agent for Browsing and Searching. http://www.cs.cmu.edu

[7] ZOU Tao, WANG Ji-Cheng, ZHU Hua-Yu, JIN Xiang-Yu, and ZHANG Fu-Yan, The Technology Implementation of Information Mining on WW, Journal of Computer Research & Development, 1999, 36(8); 1019~1024.

[8] Luis Gravano, Kevin Chang, Hector Garcia-Molina, Carl Lagoze, Andreas Paepcke. STARTS Stanford Protocol Proposal for Internet Retrieval and Search. January 19, 1997, http://www-db.stanford.edu/~gravano/starts_home.html.

[9] Marko Balabanovic, Yoav Shoham, Yeogirl Yun. An Adaptive Agent for Automated Web Browsing, http://www-diglib.stanford.edu/cgi-bin/get/SIDL-WP-1995/0023.

[10] Krulwich B, Burkoyc. The InfoFinder Agent: Learning user interests through heuristic phrase extraction. IEEE Expert, 1997 12(5): 22~27.

[11] H.C.M de Kroon, T.M. Mitchell and E.J.H. Kerckhoffs. Improving Learning Accuracy in Information Filtering. http://WWW.cs.cmu.edu

[12] Bamshad Mobasher, Namit Jain, Eui-Hong(Sam) Han and Jaideep Srivastava. Web Mining: Pattern Discovery from World Wide Web Transactions. http://www.cs.umn.edu/.