A New Approach of the Collaborative User Interface Agents

Tarek Helmy Satoshi Amamiya Tsunenori Mine Makoto Amamiya

Department of Intelligent Systems
Kyushu University
6-1 Kasuga koen, Kasuga-shi
Fukuoka 816-8580, Japan
Tel: 81-92-583-7615
Fax: 81-92-583-1338
E-mail: [helmy, roger, mine, amamiya]@al.is.kyushu-u.ac.jp

Abstract

Next generation of information systems will rely on cooperative intelligent agents for playing a fundamental role in actively searching and finding relevant information on behalf of their users in complex and open environments, such as the Internet. User Interface Agents (UIA) are semi-intelligent systems, which help the users to access, manage, share and exchange information. Recently, various researchers have proposed a learning approach towards building such agents and some working prototypes have been demonstrated. Such agents learn by watching over the shoulder of the user and detect patterns and regularities in the user's behavior. We present a new approach of the collaborative UIA that helps the user to retrieve information that is consistent to the user's need. The model provides tools and utilities for the user to manage his/her information repositories with dynamic organization and adaptation views. In order to investigate the performance of the UIA, we carried out several experiments. Through the experiments, the results ensure that the techniques of personalization, clustering the user's preferences, and making use of the preferences promise to achieve more relevant information to the user's queries.

1. Introduction

One of the biggest problems, we nowadays face in the information society is the information overload, which is boosted by the huge size of the WWW. The task of retrieving relevant information consistent with the useris information need has become increasingly difficult. Learning interface agents are computer programs that employ machine learning techniques in order to provide assistance to a user dealing with a particular application. Although they are successful in being able to learn their user's behavior and assist them, a major drawback of these systems is the fact that they require a sufficient amount of time before they can be of any use. A related problem is the fact that their competence is necessarily restricted to situations similar to those they have encountered in the past.

We present a collaborative framework of the UIAs to help alleviate these problems. In our model, the problem of information overload can be partly tackled by adding intelligent agents to the Web server [15] in what is called Web server agentification. Server agentification means creating software agents called Web Page Agents (WPA) on a Web server. Software agents could manifest various levels of intelligent behavior from simply reactive to adaptive and learning behavior, where agents actually learn what users like and dislike. The Server Agent (SA) is designed as a special server extension module, which learns to function in social environments and where necessary collaborates, completes or negotiates with other agents. The SA creates a WPA for each Web page in the Web server and is responsible for starting the search process and running of the registered WPAs. Once started, the WPAs, which can be seen as the local representatives of the Web pages, can access the local data of the Web pages and create the Interpretation Policies (IP). At the retrieval phase, the WPA uses the IP to decide whether or not the useris query belongs to the WPA, for more details see [8, 9, 17].

In this paper, we present a new model of how the UIA detects the User Preferences (UP) and calculates the relevancy of the retrieved Web pages to the UP and queries. We present the methodologies of clustering the UP into communities and make use of them. We introduce the collaborative filtering mechanism of the UIAs and the evaluation on the adaptability of the UIAs. Finally we present the experimental results, the related work and conclude by the future work.

2.1 The User Interface Agent

In our approach, the UIA communicates with the SAs to perform the search and maintains user's workspaces and preferences. The UIA is designed to learn the user's preference either explicitly or implicitly from his/her browsing behavior. We have developed and investigated some sensors in correlation with the elapsed time of visiting the Web page to let the UIA detects autonomously the actual user's implicit response. A UIA resides in a user's machine and is usually a running process that operates in parallel with the user, communicates with the WPAs via an SA to retrieve relevant information to the user's query. The UIA shows the results returned by the WPAs to the user after filtering and re-ranking them. It receives user's responses of his/her interest/not interest to the results and regards them as rewards to the results. The UIAs look over the shoulders of the users and record every action into the query history file. Followings are the UIAís job stream:

- The UIA analyzes the Natural Language (NL) query of the user by using a simple NL algorithm, it throws out non-relevant words and transforms the query to Q_{in}, where $Q_{in} = \langle k_1, k_2, \ldots, k_n \rangle$ stands for a vector of the keywords of the query.
- The UIA looks for relevant URLs to Q_{in} in the UP files by using the similarity equations (5) and (6) to be given in section 2.1.3.
- If the UIA finds relevant URLs in the UP files, then it shows them to the user and asks whether he/she is satisfied or wants to search the Web.
- If the UIA could not find in its UP files any URLs relevant to Q_{in} then the UIA submits Q_{in} to the router agent that routes Q_{in} to a relevant SA, which in turn forwards Q_{in} to its WPAs.
- The UIA receives the results returned by the SA via the router agent. The results consist of a set of Web pages and their similarity value to the given query.
- The UIA takes a set of queries from the UP files, whose similarity to Q_{in} is over the predefined threshold value, and creates a vector from the set of queries and Q_{in} in order to be used for filtering the retrieved results.

The user either explicitly marks the relevant documents using UIAís feedback menu or the UIA implicitly catches his/her response. The response is used to adapt the contents of the UP files, see section 4.

2.1.1 Implicit Response Implication by the UIA

For the UIA to be truly useful, user's interests must be inferred implicitly from actions and not obtained exclusively from explicit content ratings provided by the user, because having to stop to enter explicit ratings can alter normal patterns of browsing and reading. A more intelligent method is to use implicit ratings, where a rating is obtained by a method other than obtaining it directly from the user. Previous studies have shown that reading time to be a useful source of predicting UP implicitly [10, 13]. We investigated other sensors to make the UIA detects the actual user's implicit response. We developed the UIA's browser to record the user's implicit ratings and the explicit ratings to a Web page. Followings are the actions included in correlation with the elapsed time of visiting the Web page.

- The size of the clicked page and the number of links within the clicked page, as the time of visiting the navigational pages is different than the time of visiting content pages.
- Monitoring user's operations such as saving, printing, bookmarking, minimizing, or closing the page.
- Jumping to another link, where the UIA distinguishes between two types of links, if it is between pages with different or the same domain names.
- The clicked page's server response, as the user may spend some time and finally the server says it is unknown domain.
- Other heuristic factors like; type of the page (text, image, ..), and visit count to this page.

These implicit interest indicators have obvious advantages, including removing the cost of the user rating, and that every user interaction with his/her UIA can contribute to an implicit rating. In our experiments, a reward is given to the weights of keywords of a URL when the user does "*Print*", "*Save*" or "*Bookmark*" operation on a page. These operations have the same effect as the system modifies the weights of the keywords by using the "*Interesting*" explicit feedback. Conversely, if the user quickly jumps to another link or closes the browser without neither "*Print*" nor "*Bookmark*" a URL, this has the same effect as the user explicitly sends a "*Useless*" explicit response

2.1.2 User's Modeling by the UIA

Devising a way to constrain the search results using current user's interests allows the UIAs to retrieve higher proportion of relevant pages without requiring the users to refine the search. The UIA collects information about the user in two special files, named query history file and bookmark file. The query history file contains information about previously visited pages for specific queries, and the bookmark file contains a userís hot-list of Web links. A query history file records the URL that a user clicked, the number of occurrences that this URL is visited, both time and date of visiting, the time of leaving, and the query's keywords. The bookmark file records the URL, the number of occurrences that this URL is visited, time and date of bookmaking, and its title's keywords. The query and the title fields in the query history and the bookmark files are represented as a vector of keywords sorted in alphabetical order. A weight value is assigned to each keyword, so as to reflect the correlation between the keyword and the content of the URL, and is modified according to the user's responses. User's explicit/implicit responses ($\Re$) are *Useless, Not very useful, Mildly interesting, Neutral,* and *Interesting*, where $0 \le \Re \le 1$. Once a user has entered a query, the query will be compared against the contextual information available in the UP files. If the query is on a topic the user has previously seen, the UIA refines the user's query with similar terms and suggests results from prior searches using equations (5) and (6), to be given in section 2.1.3. The UIA also uses the UP to re-rank the retrieved search results, so that the pages promising to be more interesting to the user appear first. Several Web search systems now rely on user's profiles [1, 2, 5, 6], with promising results. However, those systems will only be effective if the user's profile reflects user's preference accurately. This involves not only determining the areas of information, which users are interested in, but also being able to adapt and to classify the user's interests as permanent or temporary interests. In our approach, the UIA distinguishes the user's temporary interest and long-term interest by monitoring the number of visiting, the time and the date of visiting the URLs. Although many studies have been conducted on incorporating user's profile technology into Web search, little has been done about studying the way profiles evolve over an extended period of time as the user's interests change. The UIA keeps track of the changes of user's interest with high degree of accuracy, this increases the relevance of the results obtained in a typical Web search and promoted the user's satisfaction with the system.

2.1.3 Deducing the Category of Interest to the User

In order to look up relevant categories of interest to the user's query from the UP files, we define equations to calculate the similarity between a useris query and the contents of UP categories as follows. Assume we have a query history file or a bookmark file of n unique URLs gathered. $Q_{in} = \langle k_1, k_2, \dots, k_n \rangle$ stands for a vector of keywords of the query given by the user. $Q_j = \langle K^h_{j,1}, K^h_{j,2}, \dots, K^h_{j,m} \rangle$, ($1 \le j \le n$) stands for the vector of the query of *j-th* URL in the useris query history file, where $K^h_{j,i} = k^h_{j,i} \cdot w^h_{j,i}$, $k^h_{j,i}$ is the *i-th* keyword in the *j-th* URL and $0 \le w^h_{j,i} \le 1$ is its weight. Similarly, $T_j = \langle K^b_{j,1}, K^b_{j,2}, \dots, K^b_{j,l} \rangle$ and $K^b_{j,i} = k^b_{j,i} \cdot w^b_{j,i}$ are defined for the title of *j-th* URL in the useris bookmark file. The weight $w^h_{j,i}$ and $w^b_{j,i}$ are incrementally computed with the number t of visiting to URL_j using equation (1).

$$w_{j,i}(t+1) = \rho \cdot w_{j,i}(t) + (1-\rho) \cdot \Re \qquad (1)$$

Where $w_{j,i}$ means $w^h_{j,i}$ or $w^b_{j,i}$ and $\Re$ is a useris response described above. The initial value $w_{j,i}(1)$ is set by the useris first response. $0 < \rho \le 1$ is a weight of how much the useris response history should be accumulated. Note that $w_{j,i}$ means the accumulated useris preference of keyword K_{ji}. ρ is a function of t , i.e., $\rho(t)$, and $\rho(t)$ depends on how long useris response history upon the keyword will be involved in calculating and adapting the next weight $w(t+1)$. The value of ρ reflects how much the system trusts the useris current response. One way to automate this heuristic is to calculate for instance the variance of useris past responses and predicts the value of ρ . We calculate the similarity S^h_j between Q_{in} and the query field Q_j of j *-th* URL in the useris query history file, and similarity S^b_j between Q_{in} and the title field T_j of j *-th* URL in the useris bookmark file.

$$S^h_j = \sum_i w_{j,i} \cdot g(k_i) \qquad (2)$$

$$S^b_j = \sum_i w_{j,i} \cdot g'(k_i) \qquad (3)$$

Where $g(k_i) = 1$ if k_i exists in both Q_{in} and Q_j , otherwise $g(k_i) = 0$, and $g'(k_i) = 1$ if k_i exists in both Q_{in} and T_j, otherwise $g'(k_i) = 0$. Also, we calculate the similarity $S^{Q_{in}\text{-}URL}_j$ between Q_{in} and the keywords of the *j-th* URL.

$$S^{Q_{in}\text{-}URL}_j = \frac{S_{url}}{C_{in} + d_j - S_{url}} \qquad (4)$$

Where C_{in} is the number of words in Q_{in} , d_j is the number of words in the *j-th* URL and S_{url} is the number of words exist in both Q_{in} and URL_j .

First: the total similarity $S^{Total}_{Q_{in}\text{-}H}$ between Q_{in} and the query history file is calculated by using equation 5, with a heuristic-weighting factor $0 \le \alpha \le 1$.

$$S^{Total}_{Q_{in}\text{-}H} = \underset{1 \le j \le n}{Max} \left[\alpha \cdot s^{Q_{in}\text{-}URL}_j + (1-\alpha) \cdot s^h_j \right] \qquad (5)$$

Second: by the same way, the total similarity $S^{Total}_{Q_{in}\text{-}B}$ between Q_{in} and the bookmark file is calculated using equation (6), with a heuristic-weighting factor $0 \le \beta \le 1$.

$$S^{Total}_{Q_{in}\text{-}B} = \underset{1 \le j \le n}{Max} \left[\beta \cdot s^{Q_{in}\text{-}URL}_j + (1-\beta) \cdot s^b_j \right] \qquad (6)$$

3. Making User's Browsing History Usable

Two major reasons why people make minimal use of their browsing history are 1) the rarity of tools that make saved history data both easy to understand and easy to use, and 2) the failure of existing tools to provide access at the right level of granularity. Both of these problems are rectified here, as the UIA is capable of tracking and segmenting the users' browsing history according to the changes in user's interest. The UIA records user's history automatically as the user browses, segments the history according to the userís changes in the interest, and provides the user with a set of tools to edit, annotate, and distribute the history segments to friends and colleagues having the same category of interest. The UIA detects a change in interest by comparing the current page to several pages visited in the recent past and looks for similarities using equation (5) and (6). The UIA monitors (in real time) the userís requests, calculates the changes in interest as the user browses from a page to another page, and displays the history information in skeins constructed according to those changes in interest. The UIA accumulates this information across browsing sessions so that the user has a long-term record of his browsing habits. First, the UIA monitors the HTTP stream through a programmable proxy server. Second, the UIA has a user's interest adaptation function that calculates and displays the user's changes in his/her interest over time.

3.1 WPV Representation by the UIA

A Web Page Vector (WPV) is used by the UIA to decide whether or not a Web page has a relevancy with the UP files. The WPV is a vector of important terms, which are extracted and weighted by analyzing the contents of the Web page. Since the terms are not all equally important for representating of the WPV of each Web page, an importance factor (λ) is assigned to each term and decided by the kind of HTML tags, in which the term is included in the Web page. This means that the UIA will emphasize/de-emphasize some keywords based on the value of λ. The UIA calculates the weight of a term and constructs the WPV of the Web page from the number of appearance (*tf*) and the HTML tags, which include the term within the Web page (e.g., in title, header, bold, italic), by using equation (7).

$$w_{ik} = \lambda_k \cdot tf_{ik} \qquad (7)$$

Where w_{ik} stands for the weight of term i in *k-th* HTML tag, and tf_{ik} stands for the number of occurrences that term i appears in *k-th* HTML tag. λ_k stands for the weight corresponding to HTML tag$_k$. The total weight of term i in the WPV is the sum of all its weights in the Web page and is calculated by using equation (8).

$$w_i = \sum_{k=1}^{n} w_{ik} \qquad (8)$$

Where n is the number of HTML tags within the Web page. The UIA calculates the relevancy between the keywords that assigned to a URL in the UP files and the WPV$_j$ by using equation (9).

$$S_j^{UP-WPV} = \frac{\sum_{i=1}^{m} h(t_i) \cdot w_i}{\sqrt{\sum_{i=1}^{m} h(t_i)^2} \cdot \sqrt{\sum_{i=1}^{m} w_i^2}} \qquad (9)$$

Where, $h(t_i) = 1$ if k_i exists in both the URL and WPV$_j$, otherwise $h(t_i) = 0$.

The UIA calculates the similarity between the contents of the UP and the Web pages based on the terms they have in both of the WPVs and the URLs of the Web pages. This similarity function is based on both UP-WPV and UP-URL similarities. It is a hybrid similarity function that includes two components. S_j^{UP-URL} component measures the similarity between the *URLs of the UP files* and the words of the URL of WPV$_j$ and is calculated by the same method as in equation (4). The S_j^{UP-WPV} component measures the similarity between the words attached to a URL in the UP files and the WPV of Web page$_j$ and is calculated by using equation (9). The whole similarity S_j^{total} is calculated by using equation (10), where $0 \le \partial \le 1$.

$$S_j^{total} = (\partial \cdot S_j^{UP-URL} + (1-\partial) S_j^{UP-WPV}) \qquad (10)$$

3.2 UP-Based Filtration by the UIA

Once a user has entered a query, the query will be compared against the contextual information in the UP files. If the query is on a topic the user has previously seen, the UIA refines the user's query Q_{in} with similar terms. The UIA creates a new list of keywords K_{filter} for the filtration, $K_{filter} = Q_{in} \cap K_c$. Where, $K_c = \{t_f \mid 1 \le f \le n\}$ is a list of the common keywords that exist in the vectors of the URLs in the UP files and have similarities with the keywords of Q_{in}. After receiving the results from the SA, the UIA re-ranks the retrieved search results, so that the pages promising to be more interesting to the user appear first. Moreover, While browsing specific Web portals, the UIA:

- Crawls the hyperlink structure staring from this portal and compares the existed links with the links that are exist in the UP files of the user; see section 3.2.
- Re-displays the links which are previously visited by the user within a separate window in an order and format reflects the number of times a link has been visited by the user.
- Displays the non-visited links in a separate window.

If all the links of this portal are new for the user, then the UIA shows them as they are and keeps track of the user's actions while browsing this portal.

3.3 Clustering the UP into Communities

The similarity between the keywords of the URLs in the UP files is calculated by the inclusive rate of a set of URLs where each of the keywords appears. This method is basically the same as in [14] and is used to calculate the similarity between the UP's keywords to create the base clusters. Let T_m be the set of UP's URLs where keyword m appears in these URLs, and $|T_m|$ be the number of URLs included in T_m. We set the similarity between two keywords in the URLs to 1 if one of the following conditions is satisfied and 0 otherwise.

$$|T_m \cap T_n| / |T_m| > v \quad (11)$$

$$|T_m \cap T_n| / |T_n| > v \quad (12)$$

The value v will be determined by considering the valance between the number of clusters and that of keywords holding one or less relative keywords. We set v as 0.5 based on preliminary experimental results [12].

4. Adaptability of the UIA

There are several approaches that can be used to learn and adapt a UP [5, 6, 7, 16]. The UIA interacts with the WPAs as follows. The UIA sends Q_{in} to the WPAs through the router and the SA. The WPAs choose an action to perform based on the relevancy between Q_{in} and their *IPs* and send the results back to the SA, which in turn forwards the results to the UIA. The UIA presents the results to the user. The user can click on each of those URLs and then, the UIA opens its browser to that URL. The user can check, evaluate this URLis contents and explicitly send feedback through the UIAis menu or the UIA implicitly gets the user's response. The UIA creates a new list of keywords K_f for the feedback, $K_f = Q_{in} \cap K_s$. Where, $K_s = \{t_j \mid 1 \le j \le m\}$ is a list of terms picked from the title and the headers of the selected URL, and $Q_{in} = \{q_j \mid 1 \le j \le l\}$ is the given query. According to the user's response $\Re$, the UIA does:

- If the selected URL from the retrieved URLs does not exist in the UP files then the UIA adds a new record for this URL into the UP files and initializes the weights of the query's terms to reflect the current user's response.
- If the URL selected is retrieved from the UP files, the UIA modifies the weights of the URL's keywords using the equation (1). Then, modifies the number of visiting of the selected URL.
- If one of the terms of the K_f does not exist in the query or the title fields of the URLs in the UP files, the UIA adds the new term with an initial weight reflects the current useris response.
- Refines the content of the UP files by deleting the keywords that have weights less than a predefined threshold value.

By this way, the content of the UP files will evolve over time to reflect the actual user's preference. Also, the keywords of the query and title fields continually move closer to or away from their URLs' contents.

5. Experimental Results

We have performed several experiments to make a consistent evaluation of the UIA effectiveness. We mean here by the effectiveness, as it is purely a measure of, the ability of the UIA to satisfy the user in terms of the relevance of documents retrieved for his/her queries, and the ability of the UIA to adapt and learn over time.

Experiment 1: The focus is mainly for evaluating how much the query expansion and filtration of the retrieved results will enhance the precision. *In the experiment*, first, the user submits 20 queries to the UIA, which in turn forward the queries to the router agent directly without refining. The mean number of terms per query, including the noise words, is 6.2. Then, we calculated the precision of the retrieved URLs to the user's queries. Second, we calculated the precision for the same queries after allowing the UIA to expand the queries and to filter the retrieved results. The results depicted in Figure 1 show that verify the facts that the UIAs are helpful for the users in terms of retrieving relevant information consistent with the useris information need.

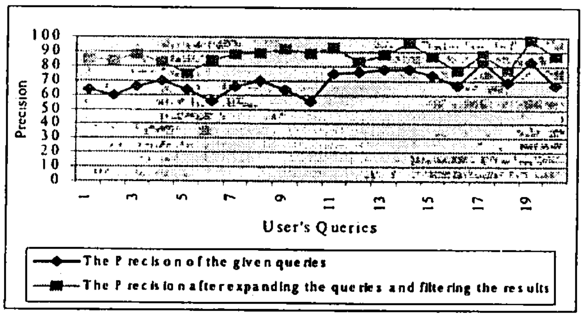

Fig. 1 Precision of the retrieved URLs

$$Precision = \frac{\textit{The number of relevant documents retrieved}}{\textit{Total number of documents retrieved}}$$

Experiment 2: In this experiment, we measured how the UIA is being able to adapt the contents of the UP over time and to get a good correlation between the keywords assigned to the URLs in the UP files and the context of the URLs. In order to understand the experiment, we define a Fitness value to show the correlation between the weights of the URL's keywords calculated automatically by the UIA (T) and the weights of the URL's keywords calculated by the user (S), as follows.

(1) User's actual interest: $S_j = \sum_{k=1}^{m} b_k \cdot W_k$, where W_k is the weight of attribute k, and $b_k = 1$ if the user judges the keyword k in the URL$_j$ as relevant for the context of the URL, else $b_k = 0$.

(2) Adaptation of the keywords by the UIA: $T_j = \sum_{k=1}^{m} W_k$. We define the Fitness value $F_j = S_j / T_j$, which reflects the correlation between the two adaptations for URL$_j$.

In the experiment, the user gave fifty different queries. After frequent interactions of retrieval, the user checked the correlation of each keyword with the context of the URLs in the UP files and calculated S and T values, then a Fitness value was calculated for the keywords of each URL in the UP files. The Fitness values calculated after 30 retrieval interactions are shown in Figure 2. Figure 2 shows the fitting values as they are converging over time to each other. This means that the UIA is being able to predict and adapt the URL's keywords to reflect well the context of the URLs in the UP files over time.

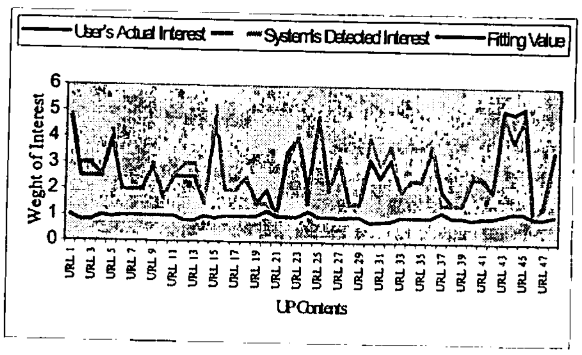

Fig. 2 Converging to the user's interest over time

Experiment 3: the focus here is to evaluate how well the UIA can cluster the contents of the UP files. The topic domains of the Web pages are various and sometimes one Web page contains several topics at the same time. Therefore, the relationship among clusters is not simple and some clusters could have close similarity. *In this experiment*, the UP files contain about 800 URLs. Equations (11) and (12) are used to measure the similarity among the URL's keywords to create the base clusters. After frequent interactions with the UIA where the users submitted 45 different queries, the clusters were assigned by the similarity and the threshold for the assignment was 0.5. All the clusters were assigned if the similarity was greater than or equal to the threshold value. The result depicted in Figure 3 shows that the UIA clustered of the UP's contents. Here are some samples of the clustered set of URLs in the UP files.

http://www.utoronto.ca/ian/software/software.html
http://agents.umbc.edu/kqml/
http://www.labs.bt.com/
http://www.denverpost.com/scene/classical0616.htm
http://www.ijcai.org/default.htm
http://www.ijcai.org/IJCAIfutureconf.htm
http://www.aaai.org/Conferences/IAAI/2001/iaai01.html
http://aaai.org/Press//Reports/Conferences/cf-99-01.html
http://aaai.org//Magazine/Dissertations/1998/eberlein.html
http://www.acm.org/reviews/Rev-info.html
http://www.acm.org/dl/subscribe.html
http://www.acm.org/pubs/citations/journals/ton/1996-4-6/p809-low/
http://www.acm.org/membership/library/corport_consort_pack.htm
http://www.acm.org/constitution/bylaw6.html
http://www.classical-artists.com/redviolin/default.htm
http://www.classical-artists.com/stephen-stirling/StvStirlg_Quotes.htm
http://www.classical-artists.com/
http://www.classical-artists.com/about.htm
http://www.fsz.bme.hu/opera/companies.html
http://www.fsz.bme.hu/opera/amato.html
http://www.grandopera.org/showscalendar.htm
http://www-al.is.kyushu-u.ac.jp/ninshiki_research_e.html
http://www-al.is.kyushu-u.ac.jp/ninshiki_members.html
http://www-al.is.kyushu-u.ac.jp/~amamiya/
http://www-al.is.kyushu-u.ac.jp/bib/amamiyal113.html
http://www.semanticweb.org/resources.html
http://www.semanticweb.org/SWWS/
http://www.semanticweb.org/news.html
http://www.semanticweb.org/knowmarkup.html
http://www.xml.org/xmlorg_resources/whitepapers.shtml
http://www.xml.org
http://www.fujitsu.com/
http://www.fujitsu.com/contact/

Fig. 3 Clustering of the user's preferences

Figure 3 shows that the URLs in the same cluster do share similar topic and contents under the general query topics.

6. Related Work

There are several research projects that deal with the implementation of agent-based Web search as intelligent interfaces, mediated searching, clustering, personalizing and recommending systems. For example, the Lira system [3] learns to browse the Internet on useris behalf. It searches the Web by taking a bounded of time, selecting the best pages and receiving an evaluation from the user. The CiteSeer system [4] helps users to find relevant Web based publications by getting keywords from the user and calling search engines to find relevant papers. Letiza system [11] is a user interface agent for assisting the browsing of the Web. Various learning approaches have been applied to discover user's preferences and to make personal recommendations. The WebWatcher system [16] is used to locate information on the Web by getting keywords from the users, suggesting hyperlinks and receiving evaluation. News Dude system [6] which performs a content-Based prediction to learn user's preferences. The differences between the UIAs and other systems are as follow. The user's profile in some other systems is often a subjective description of the users by the users themselves and the profile is static thus good for personalization for some time after it is collected, but its performance degrades over time as the profile ages. In this approach, the profile is automatically discovered and is dynamic one, which is adapting to the changes of the user's preferences over time thus making the personalization process both automatic and dynamic and hence up-to-date. We also use techniques to combine both of the explicit and implicit behaviors of the user to perform real-time personalization.

7. Conclusions and Future Work

The paper introduced methods to learn useris preferences autonomously by monitoring the useris behavior while browsing on-line WWW. It also introduced both the mechanism of clustering the UP, and the collaborative mechanism to select the most relevant category of interest to the user's query. We carried out several experiments to investigate the performance of the UIAs. Through these experiments, we ensure that the UIAs learn and adapt over time. The UIA is being able to change the weights of the URL's keywords of the UP in a way that it reflects user's point of view in the correlation between these keywords and the correlated URLs. Moreover, we found that reforming the user's query and filtering the retrieved results based on the UP is quite effective in making the search process faster and easier. As future work, we want to investigate more implicit and explicit sensors to make accurate prediction of user's satisfaction as an attempt to minimize user's effort.

8. References

[1] Alexander Pretschner and Susan Gauch, "Ontology Based Personalized Search, *Proceedings of ICTAI*, 1999, pp. 391-398.

[2] Bamshad M., Robert C. and Jaideep S. "Automatic personalization based on Web usage mining", Communications of the *ACM Journal*, volume = 43, No. 8, pp. 142-151, 2000.

[3] Balabanovic M. and Y. Shoham, "Learning information Retrieval agents, Experiments with Automated Web Browsing", *Proceedings of AAAI Spring Symposium*, pp. 13-18, 1995.

[4] Ballacker K., S. Lawrence, and L. Giles, "CiteSeer: An Autonomous System for processing & organizing scientific literature on the Web", *Proceedings of Automated Learning and Discovery Conference (CONALD-98)*, Carnegie, Mellon University, Pittsburgh, 1998.

[5] Billsus, D. and Pazzani, M., "A hybrid user model for news Story classification", *Proceedings of the 7th International Conference on User Modeling*, Canada, pp. 99-108, 1999.

[6] Bonett Monica, "Personalization of Web services: Opportunities and Challenges", *Ariadna Issue 28*, June 2001, ISSN: 1361-3200.

[7] Budzik J. and Hammond K.," Watson: Anticipating and Contextualizing Information Needs", *Proceedings of Sixty-second annual meeting of the American Society for Information Science*, 1999.

[8] Helmy T., S. Amamiya and M. Amamiya, "Collaborative Kodama Agents with Automated Learning and Adapting for Personalized Web Searching", *Proceedings of the Thirteenth International Conference on Innovative Applications of Artificial Intelligence (IAAI/IJCAI-2001)*, pp. 65-72, August 7-9, 2001, Seattle, USA.

[9] Helmy T., S. Amamiya, T. Mine and M., "An Agent-oriented Personalized Web Searching System", *Proceedings of the Fourth International Bi-Conference on Agent Oriented Information System (AOIS02)*, 15-16 July 2002, pp. 113-116, Italy.

[10] Kim J., Oard D. and Romanik K., "Using Implicit Feedback for User Modeling in Internet and Intranet Searching", *Technical Report 2000*, collage of Library and Information service, University of Maryland.

[11] Lieberman H., "An Agent that assist Web Browsing", Proceedings of the 14th International Joint Conference of AI (IJCAI95), AAAI press, California, 1995, pp. 924-929.

[12] Mine T., S. Lu, M. Amamiya, "Discovering Relationship between Topics of Conferences by Filtering, Extracting and Clustering", *Proc. of the 3rd International Workshop on Natural Language and Information Systems* (NLIS2002), pp. 205-209, 2002.

[13] Morita M. and Shinoda Y., "Information filtering based on user behavior analysis and best match text retrieval", *Proceedings of the Seventeenth Int. ACM-SIGIR Conference* pp. 272-281.

[14] Oren Zamir and Oren Etzioni, "Web document clustering feasibility demonstration", *Proceedings of the 21th International ACM SIGIR Conference*, pp. 46-54, 1998.

[15] Stefan F., "How to Integrate Mobile Agents into Web Servers" http://citeseer.nj.nec.com/11122.html.

[16] Thorsten J., D. Freitage, and T. Mitchell, "WebWatcher: A tour guide for the World Wide Web", *Proceedings of International Joint Conference on Artificial Intelligence* (IJCAI97), pp. 770-775.

[17] Zhong G., S. Amamiya, K. Takahashi, T. Mine and M. Amamiya, "The Design and Application of KODAMA System", *IEICE Transactions INF.& SYST.*, Vol. E85-D, No. 04, pp. 637-646, April 2002.