IMPLICIT: AN AGENT-BASED RECOMMENDATION SYSTEM FOR WEB SEARCH.

Alexander Birukov, Enrico Blanzieri and Paolo Giorgini

December 2004

Technical Report # DIT-04-107

Implicit: An Agent-Based Recommendation System for Web Search

630

Abstract

The amount of information on Internet is increasing very fast and, as a result, search becomes more and more a harder task. A common solution is to use authority-based search engines. However, for a community of people with similar interests, quality of results can be improved exploiting also implicit knowledge. We propose an agent-based recommendation system for supporting communities of people in searching the web by means of a popular search engine. Agents use data mining techniques in order to learn and discover users behaviors, and they interact to share knowledge about the users. We also present a set of experimental results showing in terms of precision and recall how interaction increases the performance of the system.

1. Introduction

World Wide Web contains a huge amount of information. According to ISC Internet Domain Survey [14], in July 2004 there were 285,139,107 hosts on the Internet. This number has increased approximately by 22% since January 2004 (233,101,481). It means that the size of the Internet is constantly increasing and each year the number of pages becomes appreciably bigger. Therefore, complexity of search for required information is increasing. Actually, 56.3% of the Internet users perform search in this tremendous source of information at least once a day [13]. Analyzing user search behavior data [13], we can notice that one rarely (33% according to iProspect survey) goes further than the second page of results provided by a search engine. That is why web search tools should improve quality of the results on the first two pages.

Authority-based search engines [6] are one of the most powerful web search tools. However, the lack of personalization is one of their minuses. Gori and Witten [12] state that "[...]the need to protect minorities can only be addressed within new paradigms; new, personalized views of the web that supplement today's horizontal search services. Different users may merit different answers to the same query[...]". Recommendation systems is an example of such personalization. Typical recommendation system takes queries from user and exploits the knowledge about his/her needs, behavior patterns, search profiles and content information in order to make personalized recommendation on items of user interest. Focusing the attention on web search, there are two main classes of recommendation systems: some systems deal with the content of web pages [8, 22] and some use collaborative approach [15]. In both cases the obtained information is used to create suggestions for the user.

Applications of agents and multi-agent systems to web search area are reported in literature. The main idea is to use a software agent that assists its user during web search [8, 16, 23]. The agent can track user browsing or it can form user profiles in different areas in order to anticipate items of interest. Multi-agent systems aimed to help user during web search implement various approaches. It can be coalition of several agents providing user with results [19]. It is also possible to apply auction protocol and reward mechanism to agent collaboration [24]. Other authors [5, 7, 25] propose personal agents acting on behalf of their users, collaborating with each other and having the goal to improve their users' browsing. In some of the systems considered so far user is supposed to perform an extra work during search, e.g. he/she needs to specify the areas of his/her interest or to analyze a lot of results of searches similar to the current one. Sometimes there are also restrictions like ability to use only certain part of pre-defined knowledge or ontologies.

In this paper we present *Implicit*, a multi-agent recommendation system based on the concepts of Implicit Culture. Implicit Culture [3] is a generalization of Collaborative Filtering [20], that is a technique of producing personal recommendations by computing the similarity between one's ratings and the ratings of other people. Implicit Culture means that a new community member behaves in a certain sense like the other members without the need to express knowledge of the community explicitly. Our system is mainly intended to improve web search of a community of people with similar interests. When one of the community members needs a piece of information it might appear that someone already had the same problem. Our system exploits these past interactions in order to assist the user. Each user has his/her own personal agent assisting in searching the web. These agents interact with each other for the purpose of improving this assistance. The system implements a collaborative approach in order to provide users with suggestions from community members in addition to results from a search engine. Moreover, users are not forced to perform some extra work during search.

The paper is organized as follows. Section 2 gives description of what Implicit Culture is and describes system for Implicit Culture support (SICS) structure. In Section 3 we show the general architecture of the system. In Section 4 we present experimental simulation results illustrating the utility of *Implicit* in application to a working group. In Section 5 a review of the work related to our one is done and, finally, in Section 6 we give the conclusion.

2. Implicit Culture and SICS

This section describes the general idea of Implicit Culture. It also gives description of the System for Implicit Culture Support (SICS).

A group of agents working within a community exploits a great amount of knowledge and skills. Knowledge could be either explicit (when it is possible to describe and share it through documents and/or information bases) or implicit (when it is embodied in the capabilities and the abilities of the community members). When a new agent appears in the community it faces the problem of acquiring the necessary knowledge. This problem could be solved if the member actions were consistent with the knowledge and behaviors of the community, namely its culture. When the environment is under control it is possible to achieve this goal without requiring the agent to know about the group and its behavior. The relation between two groups of agents such that the agents belonging to a group behave consistently with the "culture" of the agents belonging to the other is called the notion of Implicit Culture [3].

For example, a new member of a lab would like to browse the papers submitted during the current year by the lab members. Let us suppose that the list of publications is located on the laboratory intranet and every laboratory member knows where it is, but the new one does not. If the personal agent of the new community member is able to provide him/her with this link and he/she access the material, then it is possible to say that new user behaves in accordance with community culture and that Implicit Culture relation is established.

A SICS is included into each personal agent and has the goal to establish Implicit Culture relation [3, 5], namely, transferring the knowledge about user actions to other agents. The basic architecture for the Systems for Implicit Culture Support consists of the following three basic components:

- *Observer*, the part of SICS that stores in database information about actions executed by the user;
- *Inductive module*, that analyzes the stored observations and implements data mining techniques to discover patterns in user behavior;
- *Composer*, that exploits the information collected by the observer and analyzed by the inductive module in order to produce better suggestions to its user or to other agents.

In the inductive module we use data mining techniques in order to extract the interesting patterns from the user behavior. There are alternative approaches that can be exploited. Clustering can be applied in order to get knowledge about the correlations in the observations. For instance, agents can be clustered by interests of their users, represented by the set of their past actions. Alternatively, we can apply association rules techniques, like apriori [1] for learning association rules between the actions. Clusters and rules are used by the composer module.

The goal of the composer is to propose links such that the agents would accept them. It consists of two modules. In general, the goal of the first one is to find prospective actions that satisfy the theory (namely the clusters or association rules). In our case it should simply find that a link is needed, so there is no need to learn the theory. The second module deals with selection of the only link among the ones in the data. More details on the structure and the implementation of the composer and SICS are given in work of Blanzieri et. al [5].

3. System structure

In this section we give a brief description of *Implicit*. *Implicit* is an agent-based web search system that implements the notion of Implicit Culture. The system is implemented using JADE (Java Agent Development Framework) [2], a FIPA-compliant [11] framework for multi-agent systems development. The architecture of the system is presented in Figure 1. Each user has his/her own personal agent assisting in web search. The personal agents incorporate the SICS module for

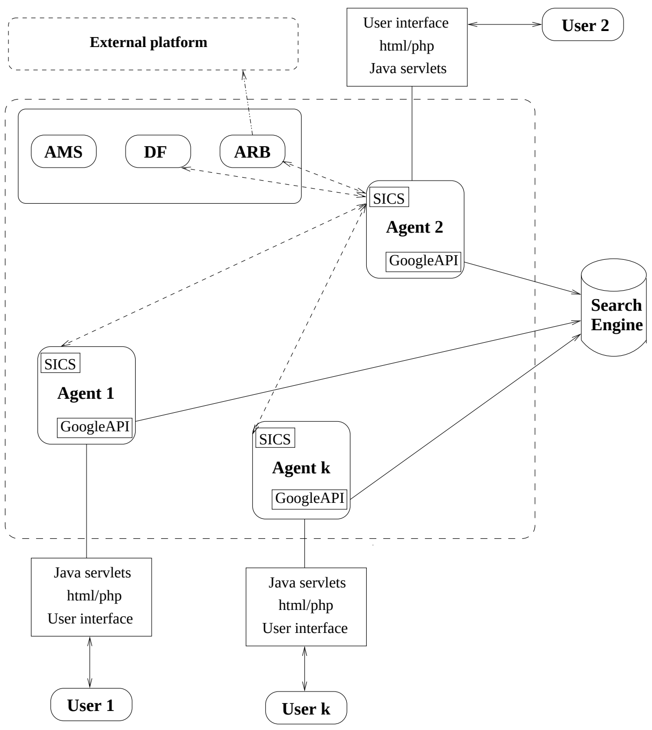

Figure 1. The architecture of the system. *Personal agents* **process queries from** *users* **and interact with each other to exchange links;** *Agent Management System (AMS)* **exerts supervisory control over the platform. It provides agent registration, search, deletion and other services;** *Directory Facilitator (DF)* **provides agents with other personal agents' IDs.** *Agent Resource Broker (ARB)* **deals with links to the services available on the other platforms.**

```
global result
for all message in INBOX do
 if (message.type == 'query') then
  result := nil
  if (query.sender == user) then
   google-search(query.sender,query.keyword,result.links)
   inform(self, user, result.links)
  end if
  SICS.internal-search(query.sender,query.keyword,result.links)
  SICS.external-search(query.sender,query.keyword,result.agents)
  if (query.sender == user) then
   if (result.agents == nil) then
    add(DF,result.agents)
   end if
   for all agent in result.agents do
    request(self,agent,query.keyword)
   end for
   inform(self, user, result.links)
  else
   inform(self, query.sender, result.links)
   inform(self, query.sender, result.agents)
  end if
 else if (message.type == reply) then
       if (message.content == resource-link) then
        add(resource-link, result.links)
       else if (message.content == agent-ID) then
             add(agent-ID, result.agents)
            end if
       end if
     end if
 else if (message.type == 'feedback') then
       add(feedback,observations)
     end if
 end if
end for
```

Figure 2. Scheme of personal agent's actions during search.

producing recommendations and capability of communication with an external source, e.g. Google [6]. The purpose of each agent is to propose its own user and the other agents links that will be probably accepted. For obtaining this goal agent uses different sources (internal and external). The personal agents are software agents running on the server side. On the client side there is html/php user interface that prompts the input search keyword, displays obtained results and collects feedback information.

Agents within the platform interact with each other and with their users exchanging messages. Each personal agent can process several types of messages. *Query* messages contain information inquiry from agent's user or from another agent. In such messages the main content is the search keyword. *Reply* messages are sent by another agent as an answer to a query from this one and contain link to web page or the ID of another agent. After the user search is finished *feedback* messages are sent to all agents that participated in the search. Feedback message contains an accepted link to web page or the ID of an agent that suggested an accepted link. Table 1 contains scheme and way of interactions between the system actors.

In the present implementation, the agent performs three types of search in the following order: first Google search (if query comes from the user), then Internal search and finally, External search. During Google search the agent queries Google search engine in order to obtain links for the keyword chosen by the user. In the Internal search the SICS module generate links to web pages, using past user actions. External search also uses SICS but in this case the goal of SICS is to propose agents to contact. The pseudocode describing personal agent's actions during the search is shown in Figure 2.

Implicit incorporates the capabilities of having some special agents in the platform. Although each agent encapsulates the ability of contacting the external search engine, e.g. Google, it is also possible to use wrapper agents for transferring the queries to other search engines like Yahoo! or Vivisimo. The Agent Resource Bro-

Actor1	Actor2	Action	Target	Parameters	protocol/tools of communication
user	agent	request	resource-links	keyword	browser, servlets, FIPA Query Interaction Protocol
agent	Google	request	resource-links	keyword	GoogleAPI
Google	agent	inform		resource-links	GoogleAPI
agent	user	inform		resource-links	FIPA Query Interaction Protocol, servlets, browser
agent	agent's SICS	request	resource-links	keyword	java class method call
agent	agent's SICS	request	agent-IDs	keyword	java class method call
agent	DF	request	agent-IDs	—	java class method call
agent	agent2	request	resource-links	keyword	FIPA Iterated Contract Net Protocol
agent	agent2	request	agent-IDs	keyword	FIPA Iterated Contract Net Protocol
agent2	agent	inform		resource-links	FIPA Iterated Contract Net Protocol
agent2	agent	inform		agent-IDs	FIPA Iterated Contract Net Protocol
agent	user	inform		resource-links	FIPA Query Interaction Protocol
user	agent	inform		accepted-resource-links	browser, servlets, socket
agent	agent's SICS	inform		accepted-resource-links	java class method call
agent	agent's SICS	inform		accepted-agent-IDs	java class method call
agent	agent's SICS	inform		rejected-resource-links	java class method call
agent	agent's SICS	inform		rejected-agent-IDs	java class method call
agent	agent2	inform		accepted-resource-links	Feedback Protocol
agent	agent2	inform		accepted-agent-IDs	Feedback Protocol
agent	agent2	inform		rejected-resource-links	Feedback Protocol
agent	agent2	inform		rejected-agent-IDs	Feedback Protocol

Table 1. Scheme of interactions between the system actors within the search session. *Actor1* **communicates to** *Actor2* **performing the communication act** *Action*; **Actor1 would like to obtain** *Target* **as a result of communication; Actor1 provide** *Parameters* **to Actor2; the last column represents protocol or tool within the communication act.**

ker (ARB) is the special agent whose main purpose is to provide our agents with links to the services available on other platforms, e.g. wrappers.

The next section presents simulation results obtained using *Implicit*. In this experiment we do not use optional parts of *Implicit* such as wrappers and ARB. Thus, Google is the only external source of the resource links on the platform.

4. Experiment

In this section we present goal, materials, method and results of the experiment that was done using the platform. We also define the measures estimating quality of the suggestions produced by SICS.

The aim of the experiment is to understand how insertion of a new member into the community affects the relevance, in terms of precision and recall, of the links produced by SICS. We also want to check the hypothesis that after a certain number of interactions, personal agents will be able to propose links accepted in previous searches.

In our experiment, interaction between agents and users is replaced by interaction between agents and user models containing user profiles determining search keywords sequence and acceptance of the results. The results are among the first m links provided by Google for each keyword and the rank of the list is adopted as an identifier. Due to the fact that links provided by Google for certain keywords are reordered very quickly, before the experiment we store the links in a dataset. During simulation we used the dataset instead of contacting Google. User profile is a set of probabilities of choosing a specified link for a specified keyword. The profile is built using n keywords $k_1,\ k_2,\ \ldots,\ k_n$ and determining the probabilities $p(j|k_i)$ of choosing the j-th link, $j \in \{1, \ldots, m\}$ while searching with the i-th keyword. We assume that the user accepts one and only one link during search for the keyword k_i, so $\sum_{j=1}^{10} p(j|k_i) = 1$. The user profile can be seen as a set of association rules with a probability of acceptance of a certain link for a given keyword search. In our experiment the number of keywords n is equal to 10, the user profile is represented in Table 2.

We use the following performance-related notions in order to evaluate quality of the suggestions:

- Link is considered to be **relevant** to a particular keyword if probability of its acceptance, as specified in the user profile, is greater than some predefined relevance threshold.
- **Precision** is the ratio of the number of relevant links suggested to the total number of irrelevant and relevant links suggested.
- **Recall** is the ratio of the number of relevant links proposed to the total number of relevant links.

We compute recall in a slightly different way. The total number of relevant links is adjusted by adding a number of relevant links proposed by the agents to a number of relevant links presented in the user profile. We do it despite the fact that in reality the links from the agents already exist in user profile, because in such a way model of interactions becomes more similar to a real-life situation, where users (and their agents as

	Google rank of the link									
keyword	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
tourism	0	0	0.05	0.4	0.05	0.2	0.1	0.05	0.1	0.05
football	0.05	0	0.1	0.3	0.3	0.1	0.1	0.05	0	0
java	0.35	0.3	0.05	0.05	0.05	0.05	0.05	0.1	0	0
oracle	0.1	0.1	0.45	0.2	0	0.05	0.05	0	0	0.05
weather	0	0.3	0	0	0.5	0	0	0.1	0.1	0
cars	0	0	0.05	0.4	0.05	0.2	0.1	0.05	0.1	0.05
dogs	0.05	0	0.1	0.3	0.3	0.1	0.1	0.05	0	0
music	0.35	0.3	0.05	0.05	0.05	0.05	0.05	0.1	0	0
maps	0.1	0.1	0.45	0.2	0	0.05	0.05	0	0	0.05
games	0	0.3	0	0	0.5	0	0	0.1	0.1	0

Table 2. Basic profile. **Probabilities of acceptance links for a set of keywords. Links are numbered $1..10$.**

well) have different collections of links. However, with such interpretation of recall, the quality of system suggestions is underestimated.

Assuming that all users are members of the same community and have similar interests, the profile for each user is derived from the basic profile given in Table 2 by adding noise. We added noise uniformly distributed in [0.00,...,0.05] to each entry of the profile and then renormalized entries in order to keep the sum of each row equal to 1. Following this procedure we generated 5 different profiles.

From our set of 10 keywords for each agent we generate 25 sequences of 25 keywords by extraction with repetition. Each sequence is used for a search session modelling the query user behavior. We also need to model user acceptance behavior. Given a keyword in the sequence of keywords, accepted result is generated randomly according to the distribution specified in the profile. Other links obtained from the agents are marked as rejected.

In a simulation we run 25 search sessions for each agent in the platform. At the end of each session the observation data were deleted. We repeat the search sessions several times in order to control the effect of the order of the keywords and link acceptance. We run 5 simulations for 1,2,3,4,5 agents. With 1 agent in the platform, the agent acts alone without interactions with the others. With 5 agents we have a small community where agents interact with each other. We set the relevance threshold used to determine the relevance of links equal to 0.1.

We computed precision and recall of the links proposed by the agents. In Figure 3 line 1 represents precision of the links produced by the personal agent only. The SICS incorporated in the agent produces these links by analyzing stored observations. Line 2 represents precision of the links proposed by all the agents including the personal one. The agents were discovered at the External search stage or provided by the DF. In Figure 4 we have analogous curves for recall.

From these figures we can note that the increase of

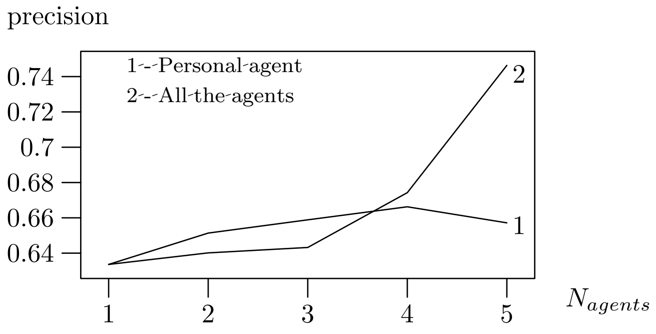

Figure 3. Average precision of 25 simulations with different number of agents.

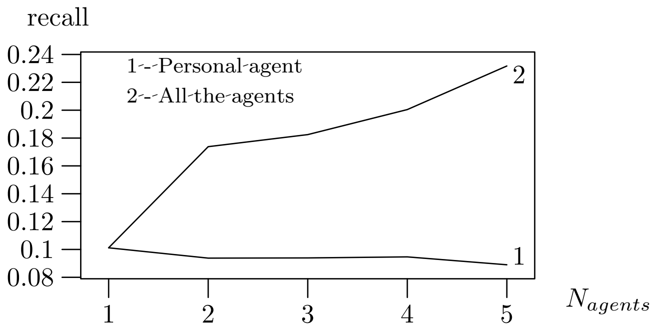

Figure 4. Average recall of 25 simulations with different number of agents.

community members causes the increase of the agents recall. It is probably conditioned by the fact that when we have more agents we also have more interactions between them. The agents provide each other only one link. So, having growth of the number of links provided by agents during the search, there is an increase of the percentage of relevant links proposed by agents and therefore increase of recall. Moreover, recall increase appears without decrease of precision and precision keeps on a rather high level — from 0.63 to 0.75. The value of recall is also rather good and changes from 0.09 to 0.23. We also studied the statistical significance of the difference between agents with the same profile and in different simulations. We performed t-Tests with Bonferroni correction, namely dividing p-value by the number of tests we have performed, in order to control type I error. These tests prove that the average recall for 4 and 5 agents is consistently better ($p < 0.01$) than the average recall of the simulations with smaller number of agents. The results also prove the hypothesis that after certain number of interactions, agents are able to propose links based on the past user actions.

In other words the obtained results prove that our way of complementing search engine with suggestions, produced as a result of collaboration, makes sense and allows to perform web search in a more qualitative way.

For the moment we have not run yet experiments for a number of agents bigger than five. However, we suppose that after a number of agents reaches a certain level, increase of the number of community members will cause only moderate increase of performance characteristics.

5. Related work

In this section we give a small survey of the papers related to *Implicit*.

Somlo and Howe [22] use the SurfAgent to assist the user while browsing the web. SurfAgent uses TF-IDF vectors [21] for representing its user profile in several topics of interest. Learned profiles are used for generation of queries to search engines in order to automatically find new pages which are interesting to the user.

Menczer [19] suggests complementing search engines with online web mining in order to take into account the dynamic structure of the web and to recommend recent web pages which are not yet known by common search engines. For obtaining this goal the adaptive population of web search agents united in the multi-agent system emulates user browsing behavior. The system consists of InfoSpiders — agents incorporating neural net inside and analyzing the hyperlinks (and context of the documents corresponding to them) on the currently browsing page in order to propose new documents to the user. So the main goal of this system is the discovery of new information, not yet presented in web search engines, in order to provide more up-to-date service to the user.

Goal-oriented search engine is considered by Liu et. al [18]. Authors suggest not searching by keywords, but by asking normal questions as in everyday life such as "What to do if my pet is sick". The described adaptive system uses a kind of artificially interpreted "common sense", which is stored in the database, in order to produce the answers like "Take it to the veterinarian". Moreover, the system searches the web pages corresponding to the answer — in this case result is homepages of the veterinarians that are the closest to user location.

Wei et. al [24] present a market-based recommendation system. It is a multi-agent system where agent acts on behalf of its user and sells the slidebar space where recommendations can be displayed. Other agents participate in this auction in order to show their links on this slidebar. When making an offer they also specify their price. The agent-initiator of the auction chooses the most profitable offers and displays them to the user. After the user accepts some results, his/her personal agent rewards the providers of the accepted links. Other agents receive no reward. Thus, agents try to make better suggestions in order to increase their profit.

A multi-agent referral system whose structure is very similar to ours is considered by Yu and Singh [25]. Each user has his/her own personal agent. The agents interact in order to provide the user with answers to his/her question. They are also able to give each other the links to other agents. There is a complex model of interactions in the system. From agent's point of view the other agents are classified as neighbours and acquaintances and their status in this classification determines the way of contacting them. The system uses ontologies to facilitate knowledge sharing among agents and the ontologies have to be predetermined and shared among all the agents, while we emphasize the implicit support of knowledge by managing documents, links and reference to people. Differently from our system, their agents do not answer all questions but only those are related to their own user interests. The paper is focused more on knowledge (in general) search rather than on web search. Finally, the system is mail-based while *Implicit* is a web based system that adopts FIPA standards and JADE platform.

While we propose using tacit, implicit knowledge accumulated by the group of agents, Turner et. al [23] propose web search agent called FERRET that is able to use an explicit, pre-defined knowledge in order to improve its search capabilities. The presented agent uses such kind of knowledge while elaborating user query and adds context information to his/her query. Further the obtained information is used for the effective search of the scholarly information (only concerning the music) on the web. For this purpose agent interacts with various search engines and content sites. Very simple instance of the a priori knowledge is that the user is in hurry. In this case agent realizes that it is necessary to perform a fast search. One of the main ideas of the paper is specifying the context of the search a priori in order to improve it.

Degemmis et. al [9] present a recommendation system incorporating collaborative filtering and learning user profiles techniques. Thus, this system combine collaborative approach with analyzing web page content. The knowledge about users is represented in user profiles and used within the collaborative filtering algorithm to reduce the time of the recommendation generation.

The collaborative multi-agent web mining system "Collaborative Spiders" is given by Chau et. al [7]. It implements the post-retrieval analysis and implies across-user collaboration in web search. In order to provide the user with recommendations there is a special agent that performs profile matching to find the information potentially interesting to the user. Before the search user has to specify the area of the interest and privacy or publicity of the search. One of the sufficient differences between this system and *Implicit* is that the user should analyze excessive output because he/she has to browse a number of similar already finished search sessions.

Implicit Culture is also related to the notion of social navigation [10]. Both concepts emphasize the social aspect of the interaction between the users even when mediated by artifacts. However, the concept of Implicit Culture is formally defined and it emphasizes the implicit aspects of the interaction.

As it appears, there are research studies exploiting ideas similar to those presented in this paper but not dealing with web search. Also there is research that concern web search but do not use system structure similar to one used in *Implicit*.

6. Conclusion and future work

We described an agent-based recommendation system dealing with the extraction of implicit knowledge from user behavior during web search. The knowledge produced from observations is used in order to suggest links or agents to group of people and to their personal agents. The main idea is that we do not express this knowledge in explicit form but use it for improving quality of further search sessions, including searches performed by new users. Personal agents produce results by asking another personal agent about links and agent IDs. Each agent has the learning capabilities that help to produce results even without interaction. With interaction, when the user performs a search already done by one of the community members, he/she does not do it "from scratch" but exploits his/her and the others' experience. This feature increases the search quality.

The SICS architecture as well as Implicit Culture concepts allow *Implicit* to be a solution to the problem of finding necessary information on the web. One of the main advantages of our approach is represented by the use of both search engine results and suggestions produced by community members. The multi-agent system mimics natural user behavior of asking someone who probably knows the answer. Finally, the process of producing suggestions is completely hidden from the user and therefore does not force him/her to perform additional actions.

There are several directions of system modification. The composer could take into consideration also balancing between number of acceptances and rejections and it will exploit association rules techniques. The possibility of using association rules mining algorithms for solving recommendation tasks was presented in [17]. In our architecture it is possible to transfer part of the instance-based learning done in the composer module to a rule-based learning in the inductive module. This can be useful in order to improve efficiency and minimizing the storing of the data. Presently, we are conducting some experiments in this direction.

There are also possibilities of user profile improvements. Although for the present time it contains information only about acceptance and rejection of links obtained from Google, it is possible to have rules for acceptance of links from the other agents. Finally, one of the further directions is to analyze the social relations and interactions between the users.

References

[1] Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Databases, Santiago, Chile, September 1994. Available at: http://citeseer.ist.psu.edu/agrawal94fast.html

[2] Bellifemine, F., Poggi, A., Rimassa, G.: Developing multi-agent systems with a FIPA-compliant agent

framework. In: Software-Practice and Experience, Vol. 31 (2) (2001) 103-128

[3] Blanzieri, E., Giorgini, P.: From collaborative filtering to implicit culture. In: Proceedings of the Workshop on Agents and Recommender Systems, Barcelona (2000). Available at: http://dit.unitn.it/˜ implicit/

[4] Blanzieri, E., Giorgini, P., Giunchiglia, F., Zanoni, C.: Implicit culture-based personal agents for knowledge management. In: Lecture Notes in Artificial Intelligence 2926 (2004) 245-261. Available at: http://dit.unitn.it/˜ implicit/

[5] Blanzieri, E., Giorgini, P., Massa, P., Recla, S.: Implicit Culture for Multi-agent Interaction Support. Proceedings of Sixth International Conference on Cooperative Information Systems (CoopIS 2001), Trento - Italy (2001). Available at: http://dit.unitn.it/˜ implicit/

[6] Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Search Engine. In: Computer Networks and ISDN Systems 30 (1998) 107-117

[7] Chau, M., Zeng, D., Chen, H., Huang, M., Hendriawan, D.: Design and Evaluation of a Multi-agent Collaborative Web Mining System. In: Decision Support Systems 35 (2003) 167-183

[8] Chen, L., Sycara, K.: WebMate: A Personal Agent for Browsing and Search. In: Proceedings of the 2nd International Conference on Autonomous Agents, ACM Press, New York, NY (1998) 132-139

[9] Degemmis, M., Lops, P., Semeraro, G., Costabile, M.F., Lichelli, O., Guida, S.P.: A Hybrid Collaborative Recommender System Based on User Profiles. In: Proceedings of the Sixth International Conference on Enterprise Information Systems, Vol. 4. Porto (2004) 162-169

[10] Dieberger, A.: Supporting Social Navigation on the World Wide Web. In: Int. J. Human-Computer Studies 46 (1997) 805-825

[11] FIPA. Foundation for Intelligent Physical Agents. http://www.fipa.org/

[12] Gori, M., Witten, I.: The bubble of Web visibility. In: Communications of the ACM (2004 In Press)

[13] iProspect Search Engine User Attitudes Survey.: http://www.iprospect.com/premiumPDFs/ iProspectSurveyComplete.pdf

[14] Internet Systems Consortium, Inc. Internet Domain Survey.: http://www.isc.org/index.pl?/ops/ds/index.php

[15] Ishikawa, H., Ohta, M., Yokoyama, S., Watanabe, T., Katayama, K.: Active Knowledge Mining for Intelligent Web Page Management. In: Lecture Notes in Artificial Intelligence 2773 (2003) 975-983

[16] Lieberman, H.: Letizia: An Agent That Assists Web Browsing. In: International Joint Conference of Artificial Intelligent, Montreal (1995)

[17] Lin, W., Alvarez, S., Ruiz, C.: Efficient Adaptive-Support Association Rule Mining for Recommender Systems. In: Data Mining and Knowledge Discovery 6 (2002) 83-105

[18] Liu, H., Lieberman, H., Selker, T.: GOOSE: A Goal-Oriented Search Engine With Commonsense. In: Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Malaga (2002). Lecture Notes in Computer Science 2347 (2002) 253-263

[19] Menczer, F.: Complementing Search Engines With Online Web Mining Agents. In: Decision Support Systems 35 (2002) 195-212

[20] Resnick P, Iacovou N., Suchak M., Bergstrom, and Riedl J.: GroupLens: An open architecture for collaborative filtering of netnews. In: Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work, Chapel Hill, NC: ACM (1994), 175-186

[21] Salon, G., McGill, M.: Introduction to Modern Informational Retrieval. McGraw-Hill (1983)

[22] Somlo, G., Howe, A.: Using Web Helper Agent Profiles in Query Generation. In: Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, ACM Press, Melbourn (2003) 812-818

[23] Turner, R., Turner, E., Wagner, T., Wheeler, T., Ogle, N.: Using Explicit, A Priori Contextual Knowledge in an Intelligent Web Search Agent. In: Lecture Notes in Artificial Intelligence 2116 (2001) 343-352

[24] Wei, Y., Moreau, L., Jennings, N.: Recommender Systems: A Market-Based Design. In: Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, ACM Press, Melbourn (2003) 600-607

[25] Yu, Bin, Singh, M.: An Agent-Based Approach to Knowledge Management. In: Proceedings of Eleventh International Conference on Information and Knowledge Management (CIKM02), SAIC Headquarters, McLean, Virginia, USA (2002)