SIFTER-II: A Heterogeneous Agent Society for Information Filtering

Rajeev R. Raje
Department of Computer and Information Science
Indiana University Purdue University Indianapolis
Indianapolis, IN 46202
rraje@cs.iupui.edu

Mingyong Qiao
Department of Computer and Information Science
Indiana University Purdue University Indianapolis
Indianapolis, IN 46202
mqiao@cs.iupui.edu

Snehasis Mukhopadhyay
Department of Computer and Information Science
Indiana University Purdue University Indianapolis
Indianapolis, IN 46202
smukhopa@cs.iupui.edu

ABSTRACT
A direct consequence of today's interconnected world is the phenomenal growth in the amount of available and accessible information. This information overload is going to increase as more machines will be connected to wide area networks. Thus, an effective and efficient information filtering system is needed, which will weed out unwanted information and present the relevant information in a rank-ordered fashion. This paper describes one such distributed information filtering system consisting of collaborating heterogeneous agents.

Keywords
Agents, Information Filtering, Collaboration

1. INTRODUCTION
These days it is rare to find isolated computing machines. The interconnected information world has presented its users with the challenging problem of effectively coping with the information overload. This overload is expected to increase in future. Hence, the role of an effective information filtering (IF) system can hardly be understated. IF systems use the concept of a user profile to learn user preferences and these profiles are stored/updated over longer periods of time. These characteristics make the software design of IF systems a challenge. In this paper, a distributed multi-agent information system, SIFTER-II, is presented. SIFTER-II has its roots in earlier versions, SIFTER – a stand-alone IF system and D-SIFTER – a homogeneous distributed IF system.

2. RELATED AND PREVIOUS WORKS
Pazzani and Billsus [6] described an agent-based system capable of recommending web sites to user's according to their interest profiles. The profiles were implemented as binary classifiers using different techniques. Chen, et al., [1] presented analysis of two types of web agents that scour the net with the objective of retrieving relevant documents. The baseline agent utilized a best-first search strategy and the other agent utilized a genetic algorithm. In another recent research, Menczer and Belew [3] showed that in a distributed environment a multi-agent community, in which each agent utilized user-feedback in reinforcement-learning mode for adaptation, can outperform conventional, non-adaptive information retrieval performance. They claimed that the superior performance is due to the fact that the multi-agent community is more suitable in adapting to a large, heterogeneous, and dynamic environment at different time and space scales. The Amalthaea system [5] employed genetic algorithms for two classes of agents: information filtering and discovery. In the Amalthaea ecosystem, the agents successfully competed, collaborated, and evolved to bring relevant information to the attention of the user.

SAC 2001, Las Vegas, NV

2.1 SIFTER
In SIFTER (Smart Information Filtering Technology for Electronic Resources) [4] an agent performs the task of filtering on behalf of a human user. The agent maintains a knowledge base called thesaurus. It consists of keywords and phrases from a specific domain of interest. The agent classifies incoming text documents using the thesaurus, maintains a user profile and updates it based on the user's feedback. An agent in SIFTER consists of three components: a document representation module, a document classification module and a user profile learning module.

SIFTER suffers from a limitation of non-classification. A document results into a NULL vector if it does not contain any terms from the thesaurus. Such documents are given a lower priority and thereby, providing a reduced filtering performance to the user, which is a direct consequence of an inadequate thesaurus. Although, the terms in the thesaurus are culled from authoritative sources, independent asynchronous discoveries of new terms would lead to an inadequate thesaurus. This drawback can be overcome by allowing multiple agents, with different thesauri, to collaborate. D-SIFTER (Distributed SIFTER) provides this enhancement to SIFTER.

2.2 D-SIFTER
D-SIFTER [7] allows an agent to either assist another agent or to request an assistance from other agents during the

process of document classification. In D-SIFTER, all the agents are homogeneous, and use the blackboard communication method. In addition to the SIFTER components, D-SIFTER has two new components: Distributed Classifier and Distributed User Profiler.

The distributed classifier permits the inter-agent collaboration during the document classification phase. When an agent fails to classify a document, it puts this document into a waiting queue on the shared server. If another agent classifies that document, it places the result back on the server. The original agent periodically checks the server for an arrival of the result. The distributed user profiler extends user profile learning algorithm to the multi-agent collaboration scenario [7]. Many experiments performed on D-SIFTER [7] indicate that the multi-agent scheme increases the success rate of document classification, which, in-turn, dramatically improves the information filtering performance.

All agents in D-SIFTER are identical except for their thesauri. Also, agents communicate with each other via a shared server. Thus, D-SIFTER suffers from the problems associated with scalability, performance, and flexibility. Also, D-SIFTER does not allow the user to manually enhance the thesaurus. These limitations of D-SIFTER are eliminated in SIFTER-II.

3. SIFTER-II

The agents in SIFTER-II are heterogeneous, i.e., they have different functionalities. In particular, different agents perform the various information tasks, e.g., document collection, document classification, and user interaction. The user can enhance an agent's thesaurus and share with other user agents. Agents are allowed to join and leave freely. All these enhancements make SIFTER-II a new and improved system.

The design of SIFTER-II concentrates on resolving the significant issues of multi-agent system development: communication protocols, interaction between agents, coherence and coordination protocols in the multi-agent community [2], and special issues arising from the information filtering domain such as security, nature of agents and their roles to achieve information filtering.

Communication enables the agents to exchange information on the basis of which they coordinate their actions and cooperate with each other. SIFTER-II supports two communication methods, broadcasting and directed communication. Broadcasting provides an efficient way to communicate if an agent wants to post messages to all the agents or a group of agents. When an agent broadcasts a message, it includes its own identification, which other agents can use to find the broadcasting agent's physical address and establish a direct communication.

Interaction is a type of collective action wherein one agent takes an action or makes a decision that has been influenced by the presence or knowledge of another agent [2]. Recently, XML has become very popular, as a means of inter-agent interaction, as it provides a facility to define tags and structural relationships between them. SIFTER-II uses XML encoded messages to exchange information among agents. The types of messages include: *post, post-reply, sell* and *sell-reply*. The post and post-reply are used for advertising a new task and sell and sell-reply are used for creating a contract. The inter-agent coordination is achieved by the concept of conversations. A conversation is an agent's plan to achieve some goal, based on interactions with other agents.

Architecture of the common agent: A generic agent, in SIFTER-II, has three layers: *communication, control* and *execution*. In the communication layer, broadcasting and directed communication methods are defined. It also has a module that is responsible for converting between XML messages and an internal representation called message object. The advantage of such an approach is that control layer processes only the self-contained information object. The control layer defines all the rules of the conversations. The execution layer includes the definitions of the functions carried out by agents.

Types of Agents in SIFTER-II: SIFTER-II has many different types of agents and object services. These agents are classified into five categories: *administrator agent, domain agent, wrapper agent, user agent* and *classifier agent*. In addition, there is a *centroid generator service* and *a sifter server*.

The administrator agent provides the directory service to the SIFTER-II system. It provides all the information of the non-agent services, such as the centroid generation service, which retrains the agent when the thesaurus is changed and sifter server service, which lets the users communicate with their user agents, so that the agents can share the services in the system.

Each domain agent concentrates on a single domain, such as computer science or bioinformatics, of which it has a default thesaurus. When a domain agent starts, it broadcasts a message to find the administrator agent and registers with it the domain name and the resources in this domain. If the domain is already registered, the administrator agent ignores the registration. If not, this information is kept and a message is broadcasted to all the user agent about the new domain.

Each wrapper agent is responsible for retrieving documents from a specific source and transforming the information to a standard form. If there are new documents, the wrapper agent will notify the domain agents about these documents. The domain agents will broadcast the new documents to user agents.

The user agent is the proxy of the user. It has to be started by the user with a valid username and password. When a user agent is started, it first broadcasts a message to find the administrator agent, then requests the system information, such as what domains the system has right now and where the sifter server is located. After that, the user agent configures itself and joins the multi-agent community. The user agent keeps a user profile and updates it based on the user's feedback. The user agent is responsible for coordinating with the domain agent to get new documents and with classification agent to classify the documents. The user can expand the default thesaurus or create a new one, and share their own knowledge with other user agents. The knowledge sharing mechanism in SIFTER-II works as follows: when a user agent has a new document, which can not be classified by a classifier agent using the knowledge base of the user agent, the user agent will broadcast a *help-needed* message to other user agents. If other user agents, which received the help message, have the ability to assist, they will respond back to the originating user agent. This user agent will choose one agent from them and send the document to it. With this mechanism of the knowledge sharing, the classification and filtering performance is improved.

The classification agent is in charge of classifying the documents. It has a representation and classification module, but does not have any knowledge base associate with it. When a user agent gets new documents, it advertises the task to classification agents and choose one agent from the responses. Then, the user agent sends the document and its knowledge base to the selected classification agent. This architecture lets the classification of multiple documents to work in parallel, not over-loading any machine/agent.

SIFTER-II also provides the user with a web-based access. It has a three tier architecture. The user interface enables the user to login to the system from the Web. The user can read relevant documents from the interface, give feedback and create a new thesaurus. The user interface is connected to the user agent through the middle tier, sifter server. When a user agent is started, it registers itself with the sifter server. The sifter server keeps a map of the user agents and users. When a user logs in, the sifter server first checks if that user's own agent is running or not. If it is not running, the login fails. After the user logs in, the sifter server creates a clientLink object which communicates with the user interface using sockets and relays user's requests to the user agent. This clientLink object is a session object, when the user logs out, it will be deleted. The user agent connects with database using JDBC. SIFTER-II is implemented in Java 1.2.

4. EXPERIMENTS

Many different experiments were carried out with SIFTER-II. Due to the space constraints, only a brief summary of these experiments is presented. In all the experiments the document source was made up of 5000 records from the technical articles in the Computer Science domain. The filtering performance was evaluated by using two related criteria: normalized recall and normalized precision.

The first category of experiments compared the real user with simulated user. As expected, this experiment demonstrated that there was a high degree of similarity in the performance of the real and simulated user. Hence, the simulated user was used in further experiments.

The second set of experiments evaluated the filtering performance as a result of a collaboration with other agents. One user agent was allowed to collaborate with others to overcome the inadequacies of the knowledgebase. Also, the user was allowed to increase, manually, the knowledgebase of the user agent. As expected, the collaboration improved the filtering performance.

The third set of experiments compared the D-SIFTER with SIFTER-II. Although, the nature of agents in D-SIFTER and SIFTER-II is different, the basic principles of collaboration are same. Hence, it was found that the filtering performance of D-SIFTER and SIFTER-II was comparable.

5. CONCLUSIONS AND FUTURE WORK

SIFTER-II is a flexible and efficient information filtering system. The experiments carried out with SIFTER-II clearly indicate that the classification and filtering performances improve substantially as a result of collaboration between many independent agents. As SIFTER-II is part of an ongoing effort, many exciting extensions are in works e.g., incorporation of appropriate economic models and selected collaboration using acquaintances are the most obvious ones. Addition of these features will make SIFTER-II a powerful and a flexible tool enabling the users to effectively overcome the information overload.

6. ACKNOWLEDGMENTS

This project is supported by the Digital Library Initiative Phase 2 of NSF.

7. ADDITIONAL AUTHORS

Additional authors: Mathew Palakal, IUPUI Indianapolis, IN and Javed Mostafa, IU, Bloomington, IN.

8. REFERENCES

[1] H. Chen, Y. Chung, M. Ramsey, and Y. Yang. A smart itsy bitsy spider for the web. *Journal of the American Society for Information Science*, 49(7):604–618, 1998.

[2] L. Gasser and A. H. Bond. *Readings in Distributed Artificial Intelligence*. Morgan Kaufmann, San Mateo, CA, 1988.

[3] F. Menczer and R. K. Belew. Adaptive information agents in distributed textual environments. In *Proceedings of the Autonomous Agents*, 1998.

[4] J. Mostafa, S. Mukhopadhyay, W. Lam, and M. Palakal. A multi-level approach to intelligent information filtering: Model, system, and evaluation. *ACM Transactions on Information Systems*, 15(4):368–399, 1997.

[5] A. Moukas. Amalthaea: Information discovery and filtering using a multi-agent evolving ecosystem. *Applied Artificial Intelligence*, 11:437 – 457, 1997.

[6] M. Pazzani and D. Billsus. Learning and revising user profiles: The identification of interesting web sites. *Machine Learning*, 27:313–331, 1997.

[7] R. Raje, S. Mukhopadhyay, M. Qiao, M. Palakal, and J. Mostafa. Experiments with a distributed information filtering system. In *Proceedings of the 4th World Multiconference Systems, Cybernetics and Informatics, SCI-2000*, 2000.