Distributed Multi-Agent Information Filtering—A Comparative Study

S. Mukhopadhyay, S. Peng, and R. Raje
Computer and Information Science, Indiana University Purdue University at Indianapolis, 723 West Michigan Street SL280, Indianapolis, IN 46202. E-mail: smukhopa@cs.iupui.edu, shengquan.peng@verizon.com, rraje@cs.iupui.edu

J. Mostafa
School of Library and Information Science and School of Informatics, Indiana University, Bloomington, IN 47402. E-mail: jm@indiana.edu

M. Palakal
Computer and Information Science, Indiana University Purdue University at Indianapolis, 723 West Michigan Street SL280, Indianapolis, IN 46202. E-mail: mpalakal@cs.iupui.edu

Information filtering is a technique to identify, in large collections, information that is relevant according to some criteria (e.g., a user's personal interests, or a research project objective). As such, it is a key technology for providing efficient user services in any large-scale information infrastructure, e.g., digital libraries. To provide large-scale information filtering services, both computational and knowledge management issues need to be addressed. A centralized (single-agent) approach to information filtering suffers from serious drawbacks in terms of speed, accuracy, and economic considerations, and becomes unrealistic even for medium-scale applications. In this article, we discuss two distributed (multi-agent) information filtering approaches, that are distributed with respect to knowledge or functionality, to overcome the limitations of single-agent centralized information filtering. Large-scale experimental studies involving the well-known TREC data set are also presented to illustrate the advantages of distributed filtering as well as to compare the different distributed approaches.

Introduction

Identification of relevant items in large information collections has been a problem of central interest in the information systems area. Its importance to efficient utilization of on-line information resources is rapidly increasing with the explosive growth of networked information in diverse domains. Information filtering refers to a set of techniques and tools developed to deal with this problem in a general way, and several academic and industrial information filtering systems currently exist (Mostafa, Mukhopadhyay, Lam & Palakal, 1997; Oard, 2001; Seth, 1994; Yan & Garcia-Molina, 1995). However, most of the reported work such as those cited above relate to a centralized approach where a single agent, incorporating all the knowledge (e.g., domain vocabulary) or functionalities (e.g., representation, classification, and user interest profile management), is used to filter information. In Mostafa et al. (1997), for example, the authors described a framework for information filtering by decomposing the problem into three main stages of information representation, information classification, and user interest profile learning, plus additional modules for document acquisition, and presentation to the user. Well-known techniques were used for representation (vector-space model; Salton, 1989) and classification (maximum clustering; Tou & Gonzalez, 1974), while a reinforcement learning algorithm (Thathachar & Sastry, 1985) was employed for user interest profiling. The work presented in Mostafa et al. (1997) deals with a single complete information filtering agent operating in a single user environment. In experimental studies on relatively small data sets involving human users, the method was found to perform well.

Such single-agent information filters face serious problems while dealing with relatively large-scale applications. Creating a single monolithic large filter serving over a large information domain leads to unacceptable computational complexity and poor fault tolerance. A collaborative society of agents involved in information filtering may overcome many of these limitations, while resulting in a filtering

Received March 9, 2004; revised May 25, 2004; accepted June 21, 2004

• Published online 6 April 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/asi.20176

performance that is very similar to that of a single large monolithic agent. Although a few research results have been reported on multi-agent information filtering or other information services (discussed below), a systematic methodology of developing such systems is still not clear. In particular, an experimental study of performance evaluation on a large-scale application, comparing different multi-agent approaches with each other and with a single-agent one, is lacking. In this article, we present two approaches to multi-agent information filtering that incorporate distribution with respect to knowledge (complete filtering agents equipped with different domain knowledge or thesaurus collaborating over a network), or with respect to functionality (atomic agents performing elementary sub-tasks of filtering). Two prototypical systems developed using the two approaches are described, i.e., D-SIFTER for a distributed knowledge approach and SIFTER-II for a distributed functionality approach. We also provide validation of the different distributed approaches with extensive large-scale information filtering experiments performed with the popular TREC data set. Such experiments highlight measurable advantages of a distributed approach over a centralized monolithic one, as well as provide benchmark comparisons between different distributed approaches.

Related Work

According to Lewis (1995) one of the key classification processes is determining topical labels for individual documents. More broadly, this means establishing for each document a topical area or an identifier that represents the semantic content of the document. This sense of classification is more conventional, as typically understood in library or indexing practices. *Pharos* (Dolin, Agrawal, El Abbadi, & Pearlman, 1998) is a prototypical example of topical classification system. The goal of this system, created by the Alexandria Digital Library team, was to allow users to identify diverse contents on the Web based on keywords they enter into the system. In a demonstration system, users could enter keywords that were matched with the Library of Congress Classification Scheme (LCC). The user then could select a list of newsgroups that were linked to the online LCC version. In this system, LC classes were represented as reduced LSI dimensions and standard similarity measures were used to match these vector representations with queries and newsgroup articles. Similar document classification approaches have also been explored by Larson (1992), employing the LCC and the LC subject headings and by Cheng and Wu (1995) using the Dewey Decimal Classification scheme. The Construe system (Hayes, 1992) supplemented a newswire database environment by automatically determining topical labels for Reuters news stories. In experiments conducted with 674 labels and 723 stories it was found that Construe could accurately place a document in its category at least 94 of the time (described by the authors as Recall). Construe was a rule-based system and as such it was strongly tied to the domain of documents. One of the few recent examples of a neural network application in topical classification of documents was presented by Lin (1997). In this research, a Kohonen feature map algorithm was used to partition document collections into topical "regions" for the purpose of visual display of the whole collection. The vector-space model proposed by Salton (1989) was used for document representation. A basic advantage of the Kohonen approach is that it requires little prior user intervention to successfully learn the distinct topical areas. Lin demonstrated the utility of this algorithm across different document collections. Multi-agent systems have grown out of the Distributed Artificial Intelligence community. Durfee and Montgomery (1989) define a multi-agent system (MAS) as a loosely-coupled network of problem solvers that work together to solve problems that are beyond their individual capabilities. These problem solvers, which are essentially autonomous, distributed, and maybe heterogeneous in nature, are called "agents" and usually have a single focus of control and/or intention. The issues encountered in implementing the multi-agent systems are communication, interaction, and coordination (Gasser, 1991). Multi-agent systems offer a way to relax the constraints of centralized, planned, and sequential control, to provide systems that are decentralized, emergent, and concurrent (Van Dyke Parunak, 1996). The advantages offered by multi-agent systems are fault-tolerance, modular software development, and flexibility (Baker, 1996).

Sykara and co-workers (Sycara & Zeng, 1996a, 1996b; Sycara, Decker, Pannu, Williamson, & Zeng, 1996) have investigated techniques for developing a distributed and adaptive collection of information agents that coordinate to retrieve, filter, and fuse information relevant to the user, task and situation, as well as anticipate a user's information needs. They presented a distributed system architecture, called RETSINA (Reusable Task Structure-based Intelligent Network Agents), which has three types of agents: interface agents, task agents, and information agents. In the system of agents, information gathering is seamlessly integrated with decision support. The MACRON multiagent system (Decker & Lesser, 1995) uses a centralized planner to generate subgoals that are pursued by a group of cooperating agents, using KQML, a standardized language for inter-agent communication and negotiation. Wondergem, van Bommel, Huibers, and van der Weide (1998), propose a formal framework for multi-agent systems in the context of information discovery, which is a synthesis of information retrieval (IR) and information filtering (IF). Different types of agents needed in information discovery applications were described in terms of the operations they support and the knowledge and information they use. The system webCobra (de Vel & Nesbitt, 1998) can automatically recommend high-quality Web documents to users with similar interests on arbitrarily narrow information domains. Collaborative filtering automatically retrieves and filters documents by considering the recommendation or feedback given by other users to the documents. SIGMA (System of Information

Gathering Market-based Agents) (Ferguson and Karakoulas, 1996; Karakoulas & Ferguson, 1995) decentralizes decision making for the task of information filtering in multidimensional spaces such as the Usenet netnews. Different learning and adaptation techniques are integrated with SIGMA for creating a robust network-based application, which adapts to both changes in the characteristics of the information available on the network as well as to changes in individual user's information interests. The Research Assistant Agent Project (RAAP; Delgado, Ishii, & Ura, 1998) is devoted to supporting collaborative research by classifying domain specific information retrieved from the Web, and recommending these "bookmarks" to other researcher with similar research interests.

Mukhopadhyay an colleagues (Mukhopadhyay, Peng, Raje, Palakal, & Mostafa, 2003) described a solution, based on machine learning, to the problem of identifying acquaintances (i.e., the most promising remote collaborators) in a multi-agent system engaged in collaborative automated information classification. They showed that high quality classification can be obtained with reduced communication cost and delay (in contrast to exhaustive agent interaction) by such an intelligent selective interaction mechanism.

All these works, while being interesting applications of multi-agent systems and information classification, do not present a systematic methodology for developing a multi-agent information filtering environment. In particular, at least two different approaches, i.e., distributed knowledge and distributed functionality, can be used to systematically construct general-purpose multi-agent information filtering systems that can be easily adapted to any domain of information. These two approaches need to be effectively compared. Further, a large-scale benchmark experimental study is also lacking, which compares the different distributed approaches with each other and with a centralized approach in terms of objective and quantitative performance measures. These issues are addressed in this article where we describe two different distributed multi-agent versions of an information filtering system, extending our earlier work on single-agent information filtering (Mostafa et al., 1997). These two systems, based on distributed knowledge (D-SIFTER) and distributed functionality (SIFTER-II), are different in their approaches and algorithms from other known multi-agent systems. We also report extensive experimental results with the distributed filtering systems on the well-known TREC-9 OHSUMED data set. We compare the results of the distributed systems in terms of both quantitative measures of filtering performance and processing time. These results, to our knowledge, represent the first time that a comprehensive experimental study has been carried out with benchmark document collections with multi-agent filtering systems, clearly pointing out the advantages of the multi-agent approaches. It is also worth noting that our multi-agent approaches are conceptually different from collaborative filtering approaches, since the former deal with collaboration between software agents while the latter make use of human collaboration in rating.

The Single-Agent and Multi-Agent Filtering Approaches Employed

In this section, we provide brief overviews of our single-agent filtering approach as well as our multi-agent approaches that are distributed with respect to knowledge and functionality, as mentioned in the introduction.

Review of Single-Agent Centralized Filtering (The SIFTER System)

The basic filtering model that was presented in a previously published article (Mostafa et al., 1997) for a single agent, consisted of three main modules: information representation, information classification, and user interest profile learning, apart from other optional peripheral modules such as information acquisition and information presentation.

Information representation. The representation module converts a free-text document into a numerical or symbolic structure that permits further computer manipulation. The specific algorithm used is based on a thesaurus-based vector space model (i.e., the tf–idf or term frequency–inverse document frequency method; Salton, 1989). In this, a thesaurus containing a set of linear or structured representative terms are assumed to be available. A representative document set is also assumed to be available. A new document is then converted to a numerical vector of dimension equal to the cardinality of the thesaurus where the i^{th} element is given by

$$W_i = T_i \cdot \log(N/n_i)$$

where T_i is the frequency of the i^{th} term of the thesaurus in the new document, N is the total number of documents in the representative document set, and n_i is the number of documents in the representative document set containing the i^{th} term of the thesaurus. The tf–idf is a simple, powerful representation technique that is well-known in the Information Retrieval community.

It is worth noting that the thesaurus is an important component of the overall scheme representing the domain knowledge. Later, in the context of distributed multi-agent filtering, one approach will involve distributing or dividing up this knowledge.

Information classification. The objective of information classification is to organize the document vectors generated by the representation module into equivalence classes. This is accomplished in the present study by means of a similarity-based unsupervised clustering algorithm (the Maxi-Min clustering; Tou & Gonzalez, 1974) that does not require any feedback from the user. The details of the algorithm implementation can be found in Mostafa et al. (1997). Briefly, it consists of an offline cluster (centroid) discovery stage and an online classification stage. During the cluster discovery stage, a set of centroids or document vectors is computed using the representative document classification. During the classification stage, a new document vector is classified to the most similar centroid.

The time complexity of the implemented online classification algorithm increases as the product $N_T \cdot N_C$, where N_T is the number of terms in the thesaurus and N_C is the number of centroids (or clusters). Further, in empirical studies, N_C has been found to increase at least linearly with N_T. This makes the complexity of classification to be growing at least as N_T^2. Hence, for very large domains represented by correspondingly large thesauri, the online classification of documents may take unacceptably large time. This is one of the major motivations for using multiple agents equipped with disparate thesauri (knowledge) that may reduce the classification time significantly.

User interest profile learning. The objective of the user interest profile learning module is to model user's interest for each of the categories of information (as determined by the classification module) and use this model to present relevant information to the user in a rank-ordered fashion. This is accomplished by utilizing online relevance feedback and a reinforcement learning algorithm (Thathachar & Sastry, 1985). The information filtering task considered in Mostafa et al. (1997) is an online iterative presentation and refinement. This was realized by using two vectors, the profile vector (denoted by *d*) and the action probability vector (denoted by *p*) of dimensions equal to the number of clusters. The *d* vector represents the user model and is computed as the running average of relevance values given to documents belonging to each of the clusters. The *p* vector is used to probabilistically select one of the clusters as the most relevant one for presentation, as well as to make sure that all clusters receive adequate user feedback. Both *d* and *p* vectors are updated on the basis of user provided relevance feedback using learning rules that are described in Mostafa et al. (1997).

Multi-Agent Filtering: A Distributed Knowledge Approach (The D-SIFTER System)

In a multi-user centralized (single-agent) scenario, a single filtering agent is used to filter all documents pertaining to all domains of information for users. Assuming that the domains of interest for users may be different, this approach employs a thesaurus that is the union of the thesauri for the different domains. As the size of the thesaurus grows, so will the number of clusters that will represent all domains of information. Further, the document set to be processed will be the union of all documents in the domains of users' interest. Assuming that the centralized filtering server maintains separate profiles for each user, the online computation will involve representation (using the extended thesaurus) and classification (using the extended cluster space) for each of the documents in the extended document set, and routing the relevant documents to individual users based on the corresponding user profile. Clearly, such a centralized approach becomes computationally very expensive, fails to scale up to a large number of users, and suffers from poor fault-tolerance.

In the distributed knowledge approach, each user is assumed to have a complete filtering agent (with representation, classification, and user profiling modules). However, the thesaurus of each agent is only a (small) subset of the master centralized thesaurus. The disparate nature of the thesaurus represents the primary domain of interest for the corresponding user, and results in disparate classification space as well. When an agent is unable to represent or classify a given document using its own local thesaurus, it asks for collaboration from a remote agent with a different and adequate thesaurus. The small nature of the local thesaurus (and the corresponding small cluster space) results in fast processing of the documents. The collaboration between the agents, expected to occur for relatively few documents, helps to achieve a filtering performance comparable to that of a centralized monolithic single filtering agent, but in a considerably less time. Further, such a distributed approach offers several advantages such as fault tolerance and privacy over a single monolithic agent.

Distributed information classifier (DIC). In DIC, all agents are identical, except for the thesaurus. The communication among different agents takes place in an indirect manner through a common server. The server architecture is fairly simple—it has a waiting queue for storing the documents that need assistance from other agents. Each agent has its own result queue for storing the results of the classification. When an agent fails to classify a document, it puts that document into the waiting queue on the server. The agent periodically checks its result queue to see if there are any classification results, made available by other helping agents (Raje, Mukhopadhyay, Boyles, Papiez, & Mostafa, 1997).

Distributed user profiler (DUP). The distributed classification is the first part of the distributed module of D-SIFTER. The distributed user profiler (Raje, Mukhopadhyay, Qiao, Palakal, & Mostafa, 2000) is the second part, which focuses on how to expand the user profile learning algorithm to the multi-agent collaboration scenario. In the following description, the term *local class* is used to refer to the classes before an agent collaborates with other agents, *remote classes* indicate the classes used by other agents while assisting this agent, and *total classes* is a combination of the local and remote classes of a particular agent under consideration.

Remote classes. Although, the user provides the feedback to each document, in D-SIFTER, the feedback actually goes to each class. During the cluster learning stage, each agent generates several local classes. Each agent also has a "NULL class," where any unclassified document is placed. Potentially, the documents in the NULL class can be further partitioned into several groups, depending on the similarity of the documents. As the thesaurus of each agent is limited, this further partitioning of the NULL class cannot be achieved by its owner agent. However, with the help of other agents, its decomposition can be discovered and the unclassified set can be made as small as possible. To achieve this, first we assign a universal identifier to each agent. When a document is classified by another agent, it attaches a tag consisting of its agent identifier and the class number to which the document is classified. The combination of the agent identifier and the class number is used to denote a remote class. For example, if an agent's identifier is "agent2," and the local class number is 3, the [agent2, 3], will uniquely identify this class in the system. This method has a significant advantage in that it provides a possibility for each agent to keep a map from a local class number to the unified class identifier. In addition, the distributed system is transparent to the user. The user does not need to know which classes are local and which classes are remote. Thus, in the user's view, the whole distributed multiple agents system behaves like a centralized system.

Creation of the class map and user profile. To implement the distributed user profile, we expanded the single agent user profiling concept by including a new module called *recorder*. Recorder is responsible for maintaining the map from the local class numbers to the unified class identifiers: $f: N \rightarrow \{C_{i,j}\}$ where i is the agent identifier and j is the local class number. This map is dynamically achieved by a constant collaboration among agents. Initially, the map only has information about the local classes. When an agent gets a result back from remote collaborator, it will check the identifier of the collaborator and class number. If the combination of the ID and class number is not in the local map, the agent will increase the total class number by one and add this relationship to the map. Eventually, the map will become stable and static, and then no new classes will be added to the map. If a new agent is added to the system or the knowledge base of an agent(s) is updated, then the process of establishing the class map will again resume. Each time the user logs into the system, the profile module will communicate with the recorder module to check the current number of total classes. If the total class number increases, the user profile will be expanded accordingly.

Multi-Agent Filtering: A Distributed Functionality Approach (The SIFTER-II System)

A further distribution of the filtering task can be achieved by means of collaboration between agents performing heterogeneous elementary subtasks. This is referred to as the distributed functionality approach, where the agents are not complete filters but perform subtasks such as document acquisition (*wrapper agents*), representation (*representer agents*), classification (*classifier agents*), and user profiling and interaction (*user agents*). Each user will have a specific personal user agent. However, the other agents will form a society that is shared. A complete information filtering task is accomplished by means of a collaboration thread between a number of elementary heterogeneous information agents. Such an approach is well suited to develop a large-scale, open information filtering environment incorporating possibly a multitude of algorithms for each subtask without redundancy and effort needed to introduce complete filtering agents.

Agents in SIFTER-II. A generic agent, in SIFTER-II, has a three-layered structure: a *communication layer*, a *controller layer*, and an *execution layer*. The following is a brief description of the different types of agents used in SIFTER-II.

The administrator agent provides the directory service to the SIFTER-II system. This agent provides all the information of the non-agent services, such as the training service, which will retrain the agent when the knowledge base is changed and the 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 knowledge base (thesaurus). When a domain agent starts, it will broadcast a message to find the administrator agent and register with it the domain name and the resources in this domain. If the domain is already registered, the administrator agent will ignore the registration. If not, the administrator agent will keep this information and broadcast a message to all user agents 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. Each user has a corresponding user agent, which has to be started by the user with a valid username and password. When this user agent is started, it first broadcasts a message to find the administrator agent and then requests the system information, such as the available domains and the location of the sifter server. After that, the user agent configures itself and joins the multi-agent community. The user agent keeps a user profile and updates it by using user's feedback. The user agent is also responsible for coordinating with the domain agent to get new documents and with the classification agent to classify the documents. The user can expand the default knowledge base 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 cannot 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 successful classification rate is improved dramatically, thus improving the filtering performance of the system.

The classification agent is in charge of classifying the documents. It has a representation and classification module, but does not have any knowledge base associated with it. When a user agent gets new documents, it will advertise the new task to classification agents and choose one agent from the responses. Then, the user agent will send 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.

Experimental Results

In this section, we present experimental results on the well-known TREC-9 OHSUMED document collection (TREC, 2001) to compare Distributed Knowledge Multi-agent (D-SIFTER), as well as Distributed Functionality Multi-agent (SIFTER-II) systems. These results compare the performances of the various systems in terms of well-known information retrieval criteria (precision and recall; Salton, 1989), as well as computational time. It is worth pointing out that, if computational time is not of concern, the precision and recall performance of a complex, monolithic, single-agent filter encapsulating all the knowledge and functionalities of multiple agents can never be surpassed by the corresponding multi-agent systems (D-SIFTER and SIFTER-II). However, processing time is an important consideration in practice, and the use of distributed approaches can make an impractical real-world information filtering scenario with a single-agent system (with a processing time of about 6 seconds per document) more practical and realistic with multi-agent systems (with a processing time of about 0.4 seconds per document), as reported in the results described below.

Description of the TREC-9 OHSUMED Document Collection

TREC-9 Information Filtering Track for the year 2000 uses OHSUMED document collection. The OHSUMED training collection is a set of 54,710 references from MEDLINE, the online medical information database, consisting of titles and/or abstracts from 270 medical journals published during 1987. The OHSUMED test collection is a set of 293,856 references form MEDLINE, published over a 4-year period (1988–1991). The available fields are title, abstract, MeSH indexing terms, author, source, and publication type. William Hersh and colleagues (Hersh, Buck, Leone, & Hickam) developed the OHSUMED document collection for their information retrieval experiments. Some abstracts are truncated at 250 words and some references have no abstracts at all (titles only).

Description of the topic statements in TREC-9. There are two primary sources of filtering topics for the TREC-9 Filtering track: (a) a subset of the original query set developed by Hersh and co-workers (Hersh, Buck, Leone, & Hickam, 1994) for their IR experiments, and (b) a set of MeSH terms and their definitions (MESH, 2001). The topic statements are provided in the standard TREC format and consist of <title> and <desc> (= description) fields only. The meaning of these fields is slightly different for each query type. In the OHSUMED topics used in this study, <title> = patient description, and <desc> = information request. Physicians in a clinical setting built the test collection as part of a study assessing the use of MEDLINE (Hersh et al., 1994). Novice physicians using MEDLINE generated 106 queries. There are 63 OHSU topics available in the document collection.

Description of the relevance judgments in TREC-9. For the OHSUMED topics, the results were assessed for relevance by a different group of physicians, using a three-point scale: definitely, possibly, or not relevant. For evaluation of our filtering systems, all documents judged as either possibly or definitely relevant will be considered relevant. For example, in the OHSU1 topic, there are 6 relevant documents in the training set and 44 relevant documents in the test set.

Comparison Between Distributed Knowledge (D-SIFTER) and Distributed Functionality (SIFTER-II) Approaches

The master thesaurus consists of 9,241 terms that are chosen from MeSH term field of the training document set. For D-SIFTER, 3, 6, 9, and 12 agents were employed, corresponding to the division of the master thesaurus into 3, 6, 9, 12 disjoint parts. For SIFTER (represented in the figures as the case with one agent), 15,472 centroids were derived by clustering using the entire OHSUMED collection. In D-SIFTER, we adjusted the threshold parameter for clustering so that the total number of centroids in all agents together is nearly the same as that of SIFTER. Figure 1 shows the average time for classifying one document against the number of agents. Figures 2 and 3 show the filtering performance in terms of standard quantitative criteria of Precision

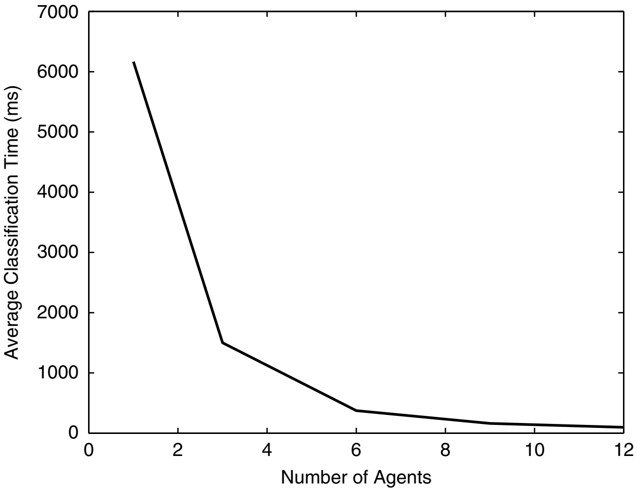

FIG. 1. Average classification time per document against the number of agents in D-SIFTER.

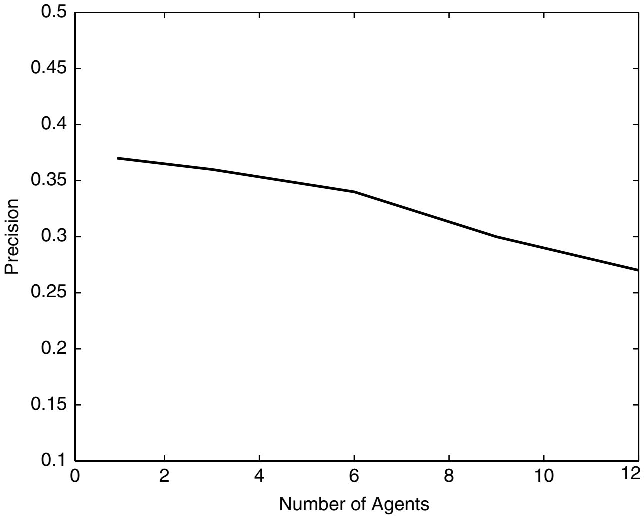

FIG. 2. Precision against the number of agents in D-SIFTER.

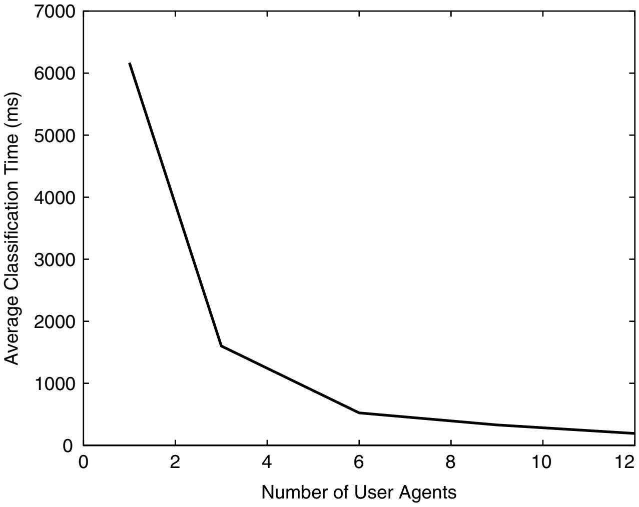

FIG. 4. Average classification time per document against number of user agents in SIFTER-II.

and Recall (Salton, 1989), respectively, against the number of agents. It can be seen that the average classification time for each document decreases drastically with the increasing number of agents in the systems. The filtering performance keeps almost constant with the increasing number of agents in the D-SIFTER, until the number of agents exceeds six. For example, with six agents, a speed-up of more than 20 was achieved (as indicated by the average time to classify a document), as compared to a single agent. This is due to the relatively smaller thesaurus and cluster space of each agent in D-SIFTER as compared to those in a single monolithic agent. It should be noted that document classification constitutes the most time-consuming step of the filtering process. It is also observed that as the number of agents in D-SIFTER increases, the precision somewhat goes down, while a corresponding increase in recall is observed. This can be attributed to the larger cluster space with increased number of agents and the fact that more clusters will be identified as relevant (meaning all documents in those clusters will also be termed relevant) as the number of agents increases. It appears from the results that up to six agents can be easily used in D-SIFTER instead of a single-agent SIFTER. This has the effect of drastically reducing the average classification time while retaining roughly the same filtering performance as that of SIFTER.

Same sets of thesauri and centroids as D-SIFTER are used by multiple user agents in SIFTER-II. Multiple user agents are run while the number of classification agents is kept always as one, for comparison purposes. Figure 4 shows the average classification time for classifying one document against the number of user agents in SIFTER-II. Figures 5 and 6 show the filtering performance in terms of precision and recall, respectively, against the number of user agents in SIFTER-II. It can be seen that the average classification time for each document decreases drastically with the increasing number of user agents in the systems.

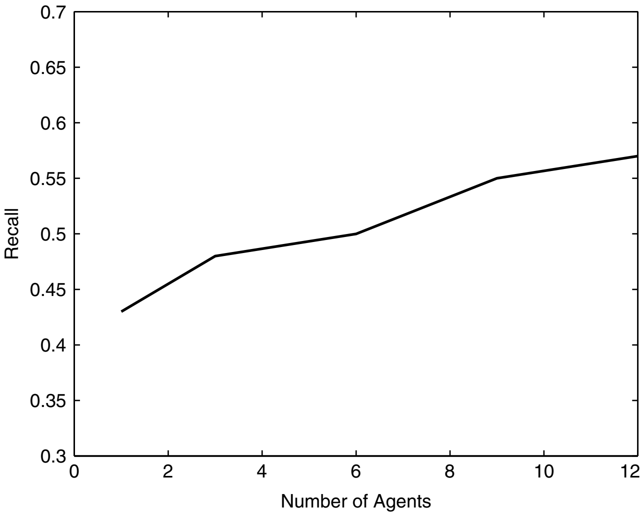

FIG. 3. Recall against the number of agents in D-SIFTER.

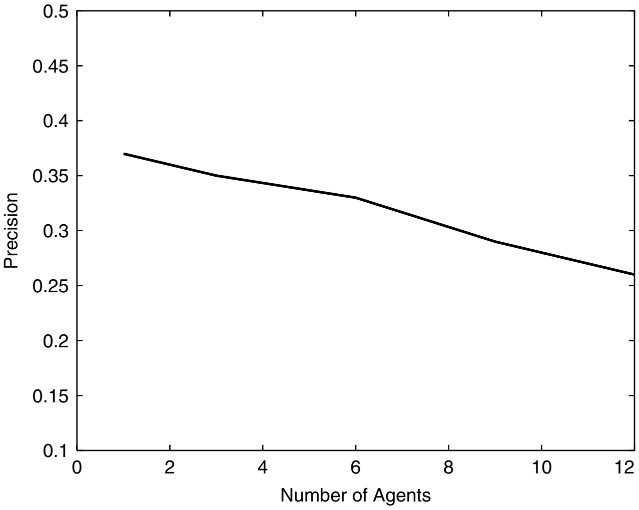

FIG. 5. Precision against number of user agents in SIFTER-II.

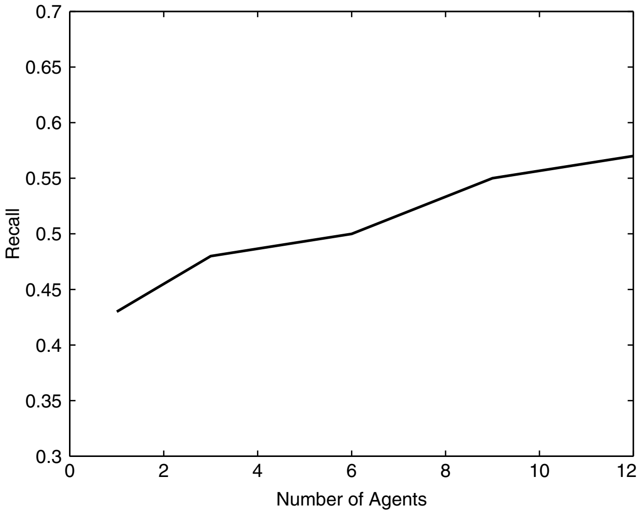

FIG. 6. Recall against number of user agents in SIFTER-II.

The filtering performance keeps almost constant with the increasing number of user agents in SIFTER-II, until the number of user agents exceeds a certain number. Comparing the performance of SIFTER-II with that of D-SIFTER, the filtering performances are almost same in terms of precision and recall. However, as stated earlier, SIFTER-II is more open and flexible than D-SIFTER.

SIFTER-II allows multiple documents to be classified in parallel, thus, not over-loading any specific machine/agent. Therefore, if there are multiple classification agents, it may lead to a good performance in terms of classification time for a certain number of documents, although the filtering performance in terms of precision and recall is not related to the number of classification agents. Just like what have been done with 3, 6, 9, 12 user agents, an increasing number of classification agents (1, 3, 6) are used to run SIFTER-II system. The filtering performance is always the same with that of SIFTER-II system with one classification agent. Figure 7 shows the average classification time per document for SIFTER-II and D-SIFTER. It can be seen that the average classification time is decreasing with the increasing number of classification agents. Therefore, if machines are available so that multiple classification agents can be run, better performance in terms of time can be achieved in SIFTER-II. This makes SIFTER-II more flexible because it can dynamically change the number of classification agents according to the system's task load.

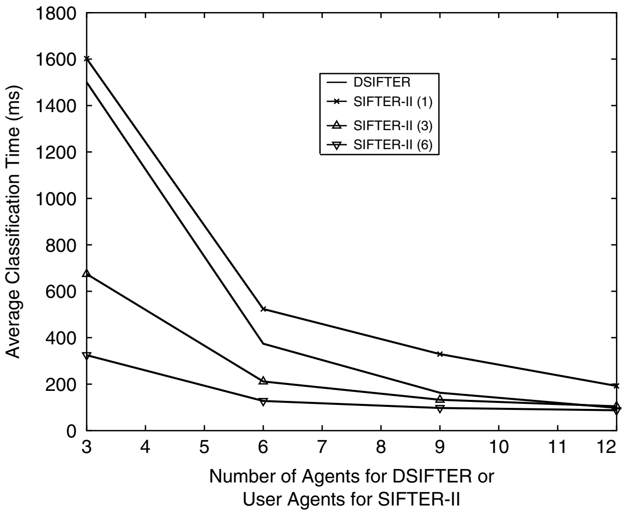

FIG. 7. Average classification time per document of D-SIFTER and SIFTER-II.

Comments and Conclusions

Fast processing of documents is critically important for information filtering services, particularly in online environments. The main advantage of multi-agent distributed information filtering is the possibility of achieving the same high filtering performance as a monolithic single-agent filter, but with a considerably reduced document processing time. Such fast processing, along with other well-known advantages of distributed computing systems (e.g., fault-tolerance, openness, and flexibility), makes multi-agent implementations an attractive alternative to single-agent ones. While the idea of a collaborative society of information agents offering information services to users has been recently introduced, a concrete case study documenting the advantages over a single-agent system on a bench-mark document collection has not been reported previously.

In this article, we described two approaches of developing distributed multi-agent information filtering systems, i.e., distributed knowledge approach and distributed functionality approach. These are extensions of our earlier work on single-agent information filtering. Complete prototype systems have been developed using the two approaches. We also compared the performance of the distributed systems with each other, with a centralized monolithic single-agent filter. The performances of SIFTER, D-SIFTER, and SIFTER-II systems have also been seen to compare favorably with other filtering systems on the bench-mark OHSUMED TREC-9 document collections (these results are not presented in this article to conserve space but will be presented elsewhere). Such comparisons are carried out in terms of quantitative measures of filtering performance, as well as document processing time whenever available. These studies clearly indicate that multi-agent filtering systems can be designed for real-world applications, offering high filtering performances with fast document processing.

Acknowledgments

This project was supported by a Digital Library Initiative Phase 2 grant (NSF grant no. 9817572). This is a joint program by multiple agencies: NSF, DARPA, NASA, NLM, LoC, NEH, FBI, NARA, SI, and IMLS. This work was also supported by an NSF ITR grant (NSF grant no. 0081944).

References

Baker, A.D. (1996). Metaphor or reality: A case study where agents bid with actual costs to schedule a factory. In S.H. Clearwater (Ed.), Market-based control (pp. 184–223). Hackensack, NJ: World Scientific.

Cheng, P.T.K., & Wu, A.K.W. (1995). ACS: An automatic classification system. Journal of Information Science, 21(4), pp. 289–299.

de Vel, O., & Nesbitt, S. (1998). A collaborative filtering agent system for dynamic virtual communities on the web. In J. Carbonell et al. (Eds.), Proceedings of the Conference on Automated Learning and Discovery, CONALD-98. Retrieved February 14, 2005, from http://citeseer.ist.psu.edu/de-collaborative.html

Decker, K., & Lesser, V. (1995, December). Macron: An architecture for multi-agent cooperative information gathering. Paper presented at the CIKM Conference, Workshop on Intelligent Information Agents, Baltimore, MD.

Delgado, J., Ishii, N., & Ura, T. (1998). Intelligent collaborative information retrieval. In H. Coelho (Ed.), Progress in Artificial Intelligence—IBERAMIA'98, Lecture Notes in Artificial Intelligence Series No. 1484 (pp. 170–182). Berlin/Heidelberg/New York: Springer-Verlag.

Dolin, R., Agrawal, D., El Abbadi, A., & Pearlman, J. (1998, January). Using automated classification for summarizing and selecting heterogeneous information sources. DLib Magazine. Retrieved February 14, 2005, from http://www. dlib.org/dlib/january98/dolin/01dolin.html

Durfee, E.H., & Montgomery, T.A. (1989, September). Mice: A flexible testbed for intelligent coordination experiments. Paper presented at the 1989 Distributed Artificial Intelligence Workshop, Lake Quinhalt, WA.

Ferguson, I.A. & Karakoulas, G.J. (1996). Multiagent learning and adaptation in an information filtering market. In S. Sen (Ed.), Proceedings AAAI Spring Symposium on Adaptation, Co-evolution and Learning in Multiagent Systems (pp. 28–32). Menlo Park, CA: AAAI Press.

Gasser, L. (1991). Social conceptions of knowledge and action: DAI foundations and open systems. Artificial Intelligence, 47, 107–138.

Hayes, P.J. (1992). Intelligent high-volume text processing using shallow, domain-specific techniques. In P.S. Jacobs (Ed.), Text-based intelligent systems: Current research and practice in information extraction and retrieval (pp. 227–241). Hillsdale, NJ: Lawrence Erlbaum.

Hersh, W.R., Buck, C., Leone, T.J., & Hickam, D.H. (1994, July). Ohsumed: An interactive retrieval evaluation and new large test collection for research. Paper presented at the 17th Annual SIGIR Conference, Dublin, Ireland.

Karakoulas, G.J., & Ferguson, I.A. (1995). A computational market for information filtering in multi-dimensional spaces. In R. Burke (Ed.), Proceedings of the AAAI Fall Symposium on AI Applications in Knowledge Navigation and Retrieval (pp. 78–83. Menlo Park, CA: AAAI Press.

Larson, R.R. (1992). Experiments in automatic Library of Congress classification. Journal of the American Society for Information Science, 43(2), 130–148.

Lewis, D.D. (1995, July). Evaluating and optimizing autonomous text classification systems. Paper presented at the 18th ACM International Conference on Research and Development in Information Retrieval, Seattle, WA.

Lin, X. (1997). Map displays for information retrieval. Journal of the American Society for Information Science, 48(1), 40–54.

MESH. (2001). Retrieved February, 14, 2005, from http://www.nlm.nih.gov/mesh/

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

Mukhopadhyay, S., Peng, S., Raje, R., Palakal, M., & Mostafa, J. (2003). Multi-agent information classification using dynamic acquaintance lists. Journal of the American Society for Information Science and Technology, 54(10), 966–975.

Oard, D. (2001). Information filtering resources. Retrieved February 14, 2005, from http:// www.ee.umd.edu/medlab/filter

Raje, R., Mukhopadhyay, S., Boyles, M., Papiez, A., & Mostafa, J. (1997, November). An economic framework for a web-based collaborative information classifier. Paper presented at the International Association of Science and Technology for Development, SE'97 Conference, San Francisco, CA.

Raje, R., Mukhopadhyay, S., Qiao, M., Palakal, M., & Mostafa, J. (2000, July). Experiments with a distributed information filtering system. Paper presented at the 4th World Multiconference Systems, Cybernetics and Informatics, SCI-2000, Orlando, FL.

Salton, G. (1989). Automatic text processing. Reading, MA: Addison-Wesley.

Seth, B.D. (1994). A learning approach to personalized information filtering. Unpublished master's thesis, Massachusetts Institute of Technology, Cambridge, MA.

Sycara, K., & Zeng, D. (1996a, August). Multi-agent integration of information gathering and decision support. Paper presented at the European Conference on Artificial Intelligence, Budapest, Hungary.

Sycara, K., & Zeng, D. (1996b). Coordination of multiple intelligent software agents. International Journal of Cooperative Information Systems, 5(2), 181–211.

Sycara, K., Decker, K., Pannu, A., Williamson, M., & Zeng, D. (1996). Distributed intelligent agents. IEEE Expert, 11(6), 36–46.

Thathachar, M.A.L., & Sastry, P.S. (1985). A new approach to the design of reinforcement schemes for learning automata. IEEE Transactions on Systems, Man, and Cybernetics, 15, 168–175.

Tou, J.T., & Gonzalez, R.C. (1974). Pattern recognition principles. Reading, MA: Addison-Wesley.

TREC. (2001). Retrieved February 14, 2005, from http://trec.nist.gov/data.html

Van Dyke Parunak, H. (1996). Foundations of distributed artificial intelligence (pp. 139–163). New York: Wiley.

Wondergem, B.C.M., van Bommel, P., Huibers T.W.C., & van der Weide, T.P. (1998, March). Agents in cyberspace—Towards a framework for multi-agent systems in information discovery. Paper presented at the Electronic Workshop in Computing for the 20th BCS Colloquium on Information Retrieval, BCS-IRSG98, Grenoble, France.

Yan, T.Y., & Garcia-Molina, H. (1995, January). Sift—A tool for wide-area information dissemination. Paper presented at the 1995 USENIX Technical Conference, New Orleans, LA.